1. CSCI-605: Advanced OO Programming Concepts

Hans-Peter Bischof (mailto:hpb@cs.rit.edu)
Department of Computer Science (http://www.cs.rit.edu)
RIT (http://www.rit.edu)

July/2020

Note: I will update the notes on a regular basis.

2. General Information

Course Title: CSCI-605 Advanced Object-Oriented Programming Concepts

Office: 70-3005
(585) 475-5568
hpb@cs.rit.edu

Office Hours: [Tues|Thurs]day: 9:30am - 10:45, and by appointment https://rit.zoom.us/my/hxbics (https://rit.zoom.us/my/hxbics)

Course Home Page: https://www.cs.rit.edu/~hpb/Lectures/2201/605/index.html (https://www.cs.rit.edu/~hpb/Lectures/2201/605/index.html)

CS Home Page: http://www.cs.rit.edu (http://www.cs.rit.edu/)

2.1. Grading

You must earn a B or better to pass this bridge course.

Grade	Percentile				
A	100%	≥	x	>	93%
A-	93%	≥	x	>	90%
B+	90%	≥	x	>	87%
B	87%	≥	x	>	82%
B-	82%	≥	x	>	80%
C+	80%	≥	x	>	77%
C	77%	≥	x	>	73%
C-	72%	≥	x	>	70%
D	70%	≥	x	>	60%
F	60%	≥	x	≥	0%

2.2. What will You need?

• Java 12 or higher. You can download it here: Java download. (https://www.java.com/en/download/manual.jsp)

• An environment to develop, compile, and debug Java programs. Any modern IDE will do, but I can not help you with the IDE you have installed your computer. For example: Eclipse. (https://www.eclipse.org/eclipseide/)

• Zoom

• Time:

— Lecture: 2 * 75 minutes

— Preperation time for and after the lecture: 2 * 75 minutes

— Homework: between 10 and 20 hours

2.3. My Expectations

• Be ontime

• You must have studied the material from the previous lecture for the current lecture.

• Study the material covered during the lecture after the lecture

• Start the homework as soon as possible. I will present the homework every Wednesday, they are due the following Tuesday/midnight. You need to be in a positon to ask questions on Monday. You need to have thought about the homework, otherwise

— You will not be able to ask questions

— You will not be able to comprehend the questions and answers of others

2.4. Communication

• Lecture: In-class or via zoom

• Office hours: Posten on my home page (in person or zoom)

• By appointment

• Email

2.5. Recitation Sessions

Will be updated

• Friday: 9:30am - 10:45am

2.6. Tentative Schedule

Schedule (./CSCI-605_schedule.html)

I will use concepts from the Flipped Classroom. (http://cft.vanderbilt.edu/guides-sub-pages/flipping-the-classroom/)

I will use the concept active learning. (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4060654/)
One part of each homework will use concepts not covered in class. I will point where you can find these materials.

2.7. Syllabus

Syllabus (./CSCI-605_syllabus.html)

2.8. Academic Integrity

— This does not mean that someone else can do your homework for you. Any homework you submit must contain your significant intellectual contribution.

— The corollary is that you may not do someone else's work for them either. A willing supplier of the material is as guilty of academic dishonesty as the receiver.

— Any help you receive from someone must be acknowledged in the work submitted. Failure to acknowledge the source of a significant idea or approach is considered plagiarism and not allowed.

Academic integrety violations are dealt with severely. The consequences of the first incident:

— You will receive a grade of F for the course.

— A note describing the details of your case will become part of your academic record.

— Repeated offenses or more serious violations may result in your being suspended or

— Violations of the Academic Integrity can also result in suspension, expulsion and even criminal charges.

There is typically no third incident during your time at RIT.

2.9. Comment

The internet never forgets:

```
google CSCI 605 RIT
```

how good is your french? (https://github.com/guroy/CSCI-605)
Or you can find a old version on github Warning - the code is not well written! (https://github.com/paridhis-rivastava/CSCI-605)

Google: `CSCI 605 github`.

You know how to use a search engine, but you will not get fluent in Java.

2.10. Course Goals

- Intro into the language Java (12) JDK 12 description (https://www.oracle.com/technetwork/java/javase/12-relnote-issues-5211422.html)
  New in JDK 12 API specification, (http://cr.openjdk.java.net/~iris/se/12/latestSpec/api/index.html)
  new in JDK 12 (https://www.oracle.com/technetwork/java/javase/12-relnote-issues-5211422.html#NewFeature)
- Use of the major classes, like Collection Framework, RMI, Swing, etc.
- Introduction to design patterns.

This is not a ‘‘programming’’ course, per se. Programming is a means to an end, not an end in and of itself.

2.11. Coding Standard

Coding Standard (https://www.cs.rit.edu/~f2y-grd/java-coding-standard.html)

• Points will be taken of your hw solutions if you violate the coding standard.

2.12. Web Resources

JDK 12. (https://docs.oracle.com/javase/12/)
Java Api's (https://docs.oracle.com/en/java/javase/12/docs/api/index.html)
Java Language and Virtual Machine Specification (https://docs.oracle.com/javase/specs/index.html)
Java Puzzlers (http://www.javapuzzlers.com/java-puzzlers-sampler.pdf)

Example of a puzzle:

```
 1
 2      public class HelloGoodbye {
 3              public static void main(String[] args) {
 4                      try {
 5                              System.out.println("Hello world");
 6                              return;
 7                              // System.exit(0);
 8                      } finally {
 9                              System.out.println("Goodbye world");
10                      }
11              }
12      }
```

Source Code: Src/2/HelloGoodbye.java

What does it print? See also printf (https://docs.oracle.com/javase/tutorial/java/data/numberformat.html)

2.13. Texts

There is an almost infinite number of (not necessarily good) books about Java and even more about the World Wide Web.

2.14. Homeworks

• The solutions for the homeworkâs must be able to compile and run using Java installed on the lab machines.

• Your solution must compile and run on CS machines. You will get partial credit if not.

• The grader choses whom to ask.

• The homework is done in teams of two. Unless are noted differently in the hw.

• The team has to meet with a grader.

• Each team members must be able to explain the solution to her/him.

• The grade for each project is based on the correctness, your explanation, and the quality of the code.

• A solution must be submitted by each student for the first homework.

• You can submit as often as you like, but only the last submission counts.

• Signup sheets will be up at the graduate lab door by Wednesday. Be on time. **You will receive 0 points for the homework if your submission is late.**

https://docs.google.com/spreadsheets/d/1So-jVaBhfyG_IiGCWTDM4wgazpbG_nT-PVQGk61rdlf0/edit?usp=sharing (https://docs.google.com/spreadsheets/d/1So-jVaBhfyG_IiGCWTDM4wgazpbG_nTPVQGk61rdlf0/edit?usp=sharing)

2.15. Other Things

Course Organization

- cheating
- lectures - ask questions
- Recitation Sessions
- homework
    - — Grader
    - — Individual questions

2.16. Object-Oriented Ingredients

- Objects
- Class
- Methods
- Encapsulation
- Inheritance
- Polymorphism
- Reuse (to a given degree)

2.17. Object

Objects are the things you think about first in designing a program and they are also the units of code that are eventually derived from the process.

In between, each object is made into a generic class of object and even more generic classes are defined so that objects can share models and reuse the class definitions in their code.

Each object is an instance of a particular class or subclass with the class's own methods or procedures and data variables. An object is what actually runs in the computer.

2.18. Class

A class consists of all objects with like state which know exactly the same methods, i.e., a class knows the structure of its objects and contains the methods to manipulate the structure. An object is called an instance of a class.

Given a class, one normally creates objects. Objects are created dynamically with the prefix operator new which in turn calls a constructor method to initialize the instance variables. Uninitialized instance variables are zeroed.

Methods mostly access the instance variables of the receiver. If methods return their receiving objects, method calls (messages) can be cascaded.

The class is one of the defining ideas of object-oriented programming. Among the important ideas about classes are:

2.19. Encapsulation

Encapsulation is the inclusion within a program object of all the resources need for the object to function — basically, the methods and the data.

Other objects adhere to use the object without having to be concerned with how the object accomplishes it.

The idea is "don't tell me how you do it; just do it." An object can be thought of as a self-contained atom. The object interface consists of public methods and instance data.

2.20. Methods

A method is a programmed procedure that is defined as part of a class and included in any object of that class. A class (and thus an object) can have more than one method. A method in an object can only have access to the data known to that object, which ensures data integrity among the set of objects in an application. A method can be re-used in multiple objects.

- A class can have subclasses that can inherit all or some of the characteristics of the class. In relation to each subclass, the class becomes the superclass.
- Subclasses can also define their own methods and variables that are not part of their superclass. The structure of a class and its subclasses is called the class hierarchy.

Question: What is the difference between class and object

2.21. Inheritance

Inheritance is the concept that when a class of objects is defined, any subclass that is defined can inherit the definitions of one or more general classes.

This means for the programmer that an object in a subclass need not carry its own definition of data and methods that are generic to the class (or classes) of which it is a part.

This not only speeds up program development; it also ensures an inherent validity to the defined subclass object (what works and is consistent about the class will also work for the subclass)

2.22. Polymorphism

Polymorphism (from the Greek meaning "having multiple forms") is the characteristic of being able to assign a different meaning to a particular symbol or "operator" in different contexts.

2.23. Reuse

Do not reinvent the wheel, speed up the development, and reduces bugs. Existing classes have most likely no bugs.

3. What is Java?

Thanks to Axel T. Schreiner. (http://www.cs.rit.edu/~ats/)

The Java Language Specification Java SE 12 Edition (https://docs.oracle.com/javase/specs/jls/se12/html/index.html)

Java is a programming language developed at Sun under the direction of James Gosling. As far as possible it is based on concepts from C, Objective C and C++.

Java is interpreted and loads classes dynamically. There are CPU chips for Java; Sun showed a prototype Java computer early in 1996 and by now it is commercially available (if slow).

According to Hoff, Shaio and Starbuck, Java is ‘‘simple, object oriented, statically typed, compiled, architecture neutral, multi-threaded, garbage collected, robust, secure, extensible and well understood. Above all, however, Java is fun!’’

These terms are elaborated on below -- I do not quite accept all these claims, however ...

simple

Java is related to languages like C or C++, but to make the language small and easy to learn all inessentials were eliminated. This leads to more uniform programs and simpler maintenance.

Unfortunately, later versions, have introduced significant new, useful, but different concepts. There is a proliferation of libraries as various companies try to turn individuality into market dominance.

object-oriented

Modern programs are distributed and have graphical user interfaces. Both can be implemented more easily with object orientation. Unlike C++, Java is fully object oriented and thus furthers the right programming style for these problem areas.

statically typed

All data must be declared with types so that data and operations may be matched as far as possible during compilation. Methods are dynamically bound but overloaded signatures are decided during compilation. Like Objective C, Java permits type queries and object analysis, e.g., for the existence of methods, using the package java/lang/reflect at runtime.

compiled

Unlike TCL, Java avoids repeated source analysis. A program is compiled into byte codes, the machine language of the Java Virtual Machine (JVM). The JVM is interpreted or compiled just in time. Classes and methods are bound symbolically, i.e., classes can be recompiled individually.

architecture neutral

Byte codes are the same everywhere; only the JVM has to be implemented for a new platform. Java programs should run everywhere and can be distributed as binaries. Unlike C and C++, Java completely specifies the capacity and behavior of the primitive data types thus eliminating a serious portability problem.

Swing is implemented without any native code, relying only on the API defined in JDK 1.1.

multi-threaded

Graphical user interfaces provide the illusion of parallel execution. Threads offer an elegant implementation. Java has a thread system based on classes and the language contains simple synchronization mechanisms (monitors). Many class libraries are thread-safe.

garbage collected

Dynamic memory management as in C and C++, where the programmer attends to reusing resources, is efficient but error prone. Java only knows dynamic objects and vectors and completely frees the programmer from the problem of memory reuse. Garbage collection runs as a parallel thread and thus should not be a bottleneck in critical situations.

robust

Exceptions are an integral part of Java for error handling. The programmer is constantly forced to consider error possibilities in libraries and the compiler can check that exceptions are not hidden or overlooked.

secure

An interpreter can pretty much ensure that the interpreted program cannot crash it's platform. In connection with the Web, Java has additional security mechanisms that constrain foreign programs so that viruses are considered impossible — in spite of the fact that binary Java programs can run on arbitrary platforms.

Various groups have demonstrated, however, that security holes do exist. There is a move toward digitally signed programs and distributed trust.

extensible

Java methods can be implemented in other languages using the Java Native Interface (JNI). In principle, arbitrary libraries can be accessed as long as other security or portability aspects do not prevail.

well understood

While Java is a new language it's concepts are well known. The language definition is comprehensive and still short. Unlike C++, a programmer can certainly understand Java completely and use all of it. Java's relationship to C and it's dialects makes it easy to get into the language, although there are a number of subtle differences to be aware of.

51 reserved words

(list) (https://docs.oracle.com/javase/tutorial/java/nutsandbolts/_keywords.html)

• strictfp (1.2)

```
public strictfp class MyFPclass {
    // ... contents of class here ...
}
```

strictfp ensures that overflow and underflow occurs in the same places on all platforms

3.1. Compilation

- `javac X.java` creates `X.class` and classes with end with `...$number.class`
- `java X` executes the main method of ***`.class`*** if present.

3.2. Java Execution Basics

- Classpath:

  — The JVM is using a class loader to load classes used by an application on an as-needed basis.

  — The CLASSPATH environment variable tells the class loader where to find the classes.

  — Classpath entries can be directories that contain class files for classes, or archive files (such as .zip or .jar files) that contain classes.

  — Classpath entries are colon-separated on Unix-type systems and semicolon-separated on MS Windows systems

  — In a bash/sh environment:

```
% CLASSPATH=/home/hpb/Jrms/classes.jar:$CLASSPATH
% export CLASSPATH
```

  — For *javac/java/...* on a UNIX platform:

```
% jdkTool -classpath path1:path2...
```

  — Which JDK version is used?

```
% java -version
openjdk version "14.0.1" 2020-04-14
OpenJDK Runtime Environment AdoptOpenJDK (build 14.0.1+7)
OpenJDK 64-Bit Server VM AdoptOpenJDK (build 14.0.1+7, mixed mode, sharing)
```

  — Which classes are used?

```
% java -verbose  Hello.java
java -verbose Hello.java
[0.006s][info][class,load] opened: /Library/Java/JavaVirtualMachines/jdk-9.jdk
[0.015s][info][class,load] java.lang.Object source: jrt:/java.base
[0.016s][info][class,load] java.io.Serializable source: jrt:/java.base
[0.016s][info][class,load] java.lang.Comparable source: jrt:/java.base
[0.016s][info][class,load] java.lang.CharSequence source: jrt:/java.base
[0.016s][info][class,load] java.lang.String source: jrt:/java.base
[0.016s][info][class,load] java.lang.reflect.AnnotatedElement source: jrt:/jav
 ...
```

- Coding Standard: See Coding Standard (https://www.cs.rit.edu/~f2y-grd/java-coding-standard.html)

3.3. Comment

See also here. (http://java.sun.com/docs/books/jls/html/3.doc.html#48125)

Java defines three kinds of comments:

/* text */
: A traditional comment: all the text from the ASCII characters /* to the ASCII characters */ is

ignored (as in C and C++).

// text

A single-line comment: all the text from the ASCII characters // to the end of the line is ignored (as in C++).

/** documentation */

A documentation comment: the text enclosed by the ASCII characters /** and */ can be processed by a separate tool to prepare automatically generated documentation of the following class, interface, constructor, or member (method or field) declaration.

3.4. Different JVMs ...

Java Standard Edition (J2SE)

Java 2 Micro Edition (J2ME), Connected Device Configuration (CDC)

Java 2 Micro Edition (J2ME), Connected Limited Device Configuration (CLDC)

Edition	Configuration	RAM	Typical Use
J2MW	Java Card	small	Smart Card
J2ME	CLDC	32k	Cellphone, pagers
J2ME	CDC	512k	PDA's
J2SE	CDC	Lots	Desktop applications

3.5. J2SE Platform at a Glance

Copied from http://www.oracle.com/technetwork/java/javase/tech/index.html (http://www.oracle.com/technetwork/java/javase/tech/index.html)

4. Java Programming Basics

4.1. Identifier

An identifier (http://java.sun.com/docs/books/jls/html/3.doc.html)
is an unlimited-length sequence of Java letters and Java digits, the first of which must be a Java letter.
An identifier cannot have the same spelling as a keyword, Boolean literal, or the null literal.

1. Can be any length
2. First character must be a letter \ { 0, .. 9 }
3. Can not include '_' (JDK 10 modification)
4. Following characters can be letters or digits
5. Are case sensitive: TOM is NOT the same as Tom
6. A Java reserved word CANNOT be used as an identifier, ex. true
7. Must be declared to be of some type.

Identifier should/must be self explained. You might do the following exercises. (http://www.cs.rit.edu/~cs1/Exercises/_Var_names/index.html)

The words *i, k, lth, ...* have no real meaning for a human being.

lessThanTen	versus	*ltt*
thisIsSorted	versus	*t*
mph	versus	*speed*
maximum	versus	*m*

The basic syntax for declaring variables is:

```
typename identifier;
typename identifier = expression;
```

A variable has the following properties:

- memory location to store the value.
- type of data stored in the memory location.
- name used to refer to the memory location.

Note: You may find the definitive reference for the Java programming language here. (http://java.sun.com/docs/books/jls/html/index.html)

4.2. The first Program: Hello.java

```
 1      /**
 2       * Classical: Hello, World
 3       */
 4
 5      class Hello {
 6          public static void main (String args []) {
 7              System.out.printf("Hello World");
 8          }
 9      }
```

Source Code: Src/3/Hello.java

Output:

```
% javac Hello.java
% java Hello
Hello World
```

Format: (https://docs.oracle.com/javase/tutorial/java/data/numberformat.html)
format, arguments

Format rules: *%[flags][width][.precision]conversion-character*

Conversion Characters:

s formats strings
d formats decimal integers
f formats the floating-point numbers
t formats date/time values

Convertion Characters (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/Formatter.html)

```
 1      // https://docs.oracle.com/en/java/javase/12/docs/api/java.base/java/io/PrintS
 2      // https://docs.oracle.com/en/java/javase/12/docs/api/java.base/java/io/PrintS
 3      // https://docs.oracle.com/en/java/javase/12/docs/api/java.base/java/util/Form
 4      // %[argument_index$][flags][width]conversion
 5      // format and printf are equivalent to one another
 6      import java.util.Calendar;
 7      class UseOfFormat {
 8          public static void main (String args []) {
 9              double aDouble = 12.3456;
10              int aInt = 42;
11              String format = "";
12              System.out.format("%%d\n");
13              System.out.format("Number    : %%5.2f   -       %5.2f\n", aDouble);
14              format = "%tT";
15              System.out.format("Local time: %" + format + "  -       " + format + "
16              format = "%tc";
17              System.out.format("Local time: %" + format + "  -       -" + format +
18              format = "%+5.6f";
19              System.out.format("Padding: %+5.6f\n", -1 * aDouble);
```

```
20              System.out.format("Padding: %" + format + "     -       -       " + fo
21          }
22      }
23
```

Source Code: Src/3/UseOfFormat.java

Execution:

```
%d
Number    : %5.2f   -      12.35
Local time: %tT     -      13:33:15
Local time: %tc     -      -Thu Aug 20 13:33:15 EDT 2020
Padding: -12.345600
Padding: %+5.6f     -      -      +12.345600
```

An other example:

```
 1      import java.text.DecimalFormat;
 2      class UseOfDecimalPattern {
 3          public static void main (String args []) {
 4              double aDouble = 123456.789;
 5              String thePattern = "###,###.###";
 6              DecimalFormat myFormatter = new DecimalFormat(thePattern);
 7              String output = myFormatter.format(aDouble);
 8              System.out.println(aDouble + " " + thePattern + " " + output);
 9          }
10      }
11
```

Source Code: Src/3/UseOfDecimalPattern.java

Output:

```
123456.789 ###,###.### 123,456.789
```

Please take a look at the https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/text/Format.html (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/text/Format.html)
and https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/Formatter.html (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/Formatter.html)

You should use a source code control system (https://en.wikipedia.org/wiki/List_of_version_control_software)
or github. (https://github.com/)
Do not make your source code publicly available.

4.3. Documentation — javadoc

The various JDK packages are documented on HTML pages in a standardized format that contains very many references to other classes or replaced methods etc.

See here: https://docs.oracle.com/en/java/javase/14/javadoc/javadoc.html#GUID-7A344353-3BBF-45C4-8B28-15025DDCC643 (https://docs.oracle.com/en/java/javase/14/javadoc/javadoc.html#GUID-7A344353-3BBF-45C4-8B28-15025DDCC643)

javadoc (https://docs.oracle.com/javase/8/docs/technotes/guides/javadoc/index.html)
creates the documentation directly from the program sources. Special comments /** ... */ before class, variable and method declarations are transferred to the documentation. The comments may contain significant tags

See coding standard

4.4. Declaration and Creation

```
 1      class DeclarationAndCreation
 2      {
 3         public static void main(String args[])
 4         {
 5            String aString;
 6            aString = new String("1234");
 7            aString.length();
 8         }
 9      }
```

Source Code: Src/4/DeclarationAndCreation.java

4.5. Declaration versus Creation

A declaration declares and connects a variable with a type.

```
int index;
String aString;
```

A creation creates an object.

The following programs have problems:

```
 1      /**
 2       * The program will not work.
 3       * What is the problem?
 4       *
 5       */
 6
 7      class WillNotCompile
 8      {
 9         public static void main(String args[])
10         {
11            String aString;
12            aString.length();
13         }
14      }
```

Source Code: Src/4/WillNotCompile.java

```
yps 4 62 javac WillNotCompile.java
WillNotCompile.java:18: Variable aString may not have been initialized.
      aString.length();
      ^
1 error
```

This program will terminate:

```
 1      /**
 2       * The program will die.
 3       * What is the problem?
 4       *
 5       * @version     $Id$
 6       *
 7       * @author      hp bischof
 8       *
 9       * Revisions:
10       *            $Log$
11       */
12
13      class WillDie
14      {
15         public static void main(String args[])
16         {
17            String aString = new String();
18            aString = null;
19            aString.length();
20         }
21      }
22
```

Source Code: Src/4/WillDie.java

Execution:

```
% javac WillDie.java
% java WillDie
java/lang/NullPointerException
        at WillDie.main(WillDie.java:19)
```

4.6. String Object/Class - Literals

• https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/lang/String.html (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/lang/String.html)

• https://docs.oracle.com/javase/tutorial/essential/regex/literals.html (https://docs.oracle.com/javase/tutorial/essential/regex/literals.html)

• Strings are constants

• All String literals are instances of the String class and exist once in a JVM.

```
 1      class StringLiteral {
 2
 3        public static void main( String args[] ) {
 4              String aString = "you";
 5              String bString = "yo" + "u";    // compiler
 6              String cString = "you";
 7
 8              if ( cString == aString )
 9                      System.out.println("1. equal");
10              if ( cString.equals(aString) )
11                      System.out.println("2. equal");
12              if ( "you".equals(aString) )
13                      System.out.println("3. equal");
14              if ( bString.equals(aString) )
15                      System.out.println("4. equal");
16              if ( "yo" + "u" == aString )
17                      System.out.println("5. ==");
18              if ( bString == aString )
19                      System.out.println("6. ==");
20              if ( bString == new String("you") )
21                      System.out.println("7. ==");
22              else
23                      System.out.println("8. !=");
24
25        }
26      }
```

Source Code: Src/4/StringLiteral.java

Result:

```
1. equal
2. equal
3. equal
4. equal
5. ==
6. ==
8. !=
```

The values of of variable of type String are inmutable.

Why? (https://www.programcreek.com/2013/04/why-string-is-immutable-in-java/)

4.7. Playing with Strings

```
 1      /**
 2       * Deal with Strings objects.
 3       *
 4       * @version     $Id$
 5       *
 6       * @author      hp bischof
 7       *
 8       * Revisions:
 9       *            $Log$
10       */
11
12      class StringThing
13      {
14         public static void main(String args[])
15         {
16            String aString;
17
18            aString = new String("Last Stop Wonderland! ");
19            System.out.println( aString.length() );
20            System.out.println( aString.toUpperCase() );
21            System.out.println( aString.toUpperCase() + ".");
22            System.out.println( aString.length() + 1 );
23            System.out.println( 1 + aString.length()  );
24            System.out.println( 1 + aString + 1 );
25            System.out.println( aString + ( 1 + 1 )  );
26         }
27      }
28
```

Source Code: Src/4/StringThing.java

Result:

```
22
LAST STOP WONDERLAND!
LAST STOP WONDERLAND! .
23
23
1Last Stop Wonderland! 1
Last Stop Wonderland! 2
```

Other Example

```
 1      /**
 2        * "abc" versus new String("abc")`
 3        */
 4
 5      class StringL {
 6
 7        public static void method(String id, String literal, String aNewString)
 8              System.out.println(id + " in method");
 9              System.out.print("\tliteral= aNewString\n       ");
```

```
10              System.out.println( literal == aNewString);
11        }
12        public static void main( String args[] ) {
13              String aString = "abc";
14              System.out.print("abc == aString\n      ");
15              System.out.println("abc" == aString);
16
17              String newString = new String("abc");
18              System.out.print("abc == new String(abc)\n      ");
19              System.out.println("abc" == newString);
20
21              method("1", "abc", "abc");
22              method("2", "abc", new String("abc") );
23              method("3", "abc", "ab" + "c");
24              method("4", "abc", "" + "abc");
25        }
26      }
```

Source Code: Src/4/StringL.java

Output:

```
abc == aString
       true
abc == new String(abc)
       false
1 in method
       literal= aNewString
       true
2 in method
       literal= aNewString
       false
3 in method
       literal= aNewString
       true
4 in method
       literal= aNewString
       true
```

4.8. From a Previous Bridge Exam = Now availbale = the Solution Up to You

```
 1      public class X_s1 {
 2
 3         public static void method_a()        {
 4              String one  = "1";
 5              String ten  = "10";
 6              do      {
 7                      System.out.println("a_1: " + ( one == ten )  );
 8                      one = one + "0";
 9              } while ( one == ten );
10              System.out.println("a_2: " + ( one == ten )  );
11         }
12         public static void method_b()        {
13              int one  = 1;
```

```
14              int ten  = 10;
15
16              System.out.println("b_1: " + 3 * one + 0 * ten );
17          }
18          public static void method_c()        {
19              String right = new String(10 * 1 - 10 + "");
20              String middle  = "00".substring(0, 1);
21              String left  = "0";
22              String leftLeft  = "0";
23              int    aInt  = 0;
24
25              System.out.println("c_1: " +     ( "0" == left )   );
26              System.out.println("c_2: " +     ( "0" == middle   ) );
27              System.out.println("c_3: " +     ( left  +  aInt + aInt ) );
28              System.out.println("c_4: " +     ( ( left  =  left + aInt + aInt ) ) +
29              System.out.println("c_5: " +     ( ( leftLeft = left = ( left  =  left
30              System.out.println("c_6: " +     ( leftLeft ) );
31                      // c_3: 000123
32                      // c_4: 0000001234
33                      // c_5: 0000000000
34                      // c_6: 0000012345
35              System.out.println("c_7: " +     ( ( "1" + ""  ) == "1" ) );
36              System.out.println("c_8: " +     ( ( aInt  + "1" ) == ( left + "1" ) )
37          }
38          public static void main(String argv[]) {
39              method_c();
40              method_a();
41              method_b();
42          }
43      }
44      /*      c_1: true
45              _2: false
46              _3: 000
47              _4: 000000
48              _5: 0000000000
49              _6: 00000
50              _7: true
51              _8: false
52              _1: false
53              _2: false
54              _1: 30
55      */
```

Source Code: Src/4/X_s1.java

Execution:

```
% java X_s1
c_1: true
c_2: false
c_3: 000
c_4: 000000
c_5: 0000000000
c_6: 00000
c_7: true
c_8: false
```

```
a_1: false
a_2: false
b_1: 30
```

4.9. Use of the StringThing Class

Example:

```
 1      /**
 2        * Play with the String class
 3        *
 4        * @version    $Id$
 5        *
 6        * @author     Hpb
 7        *
 8        *     $Log$
 9        */
10
11      class String_1 {
12
13        public static void main( String args[] ) {
14              String aString = "David";
15              String bString = "David Bowie";
16
17              if ( "hello".equals("hello") )
18                      System.out.println("equal");
19              if ( "David" == aString )
20                      System.out.println("David == aString ");
21              System.out.println(aString.length());
22              System.out.println(aString.charAt(0));
23
24              System.out.println(aString.indexOf("vid"));
25
26              System.out.println(aString.substring(2,3));
27              System.out.println(aString.substring(
28                                      aString.indexOf("a"),
29                                      aString.indexOf("i")
30                                                ));
31
32              System.out.println(aString.concat(" Bowie").length());
33
34              String help = bString.substring(0, aString.length());
35              System.out.println("-->" + help + "<--" );
36              if ( "David" == help )
37                      System.out.println("David == help ");
38              if ( "David" == aString )
39                      System.out.println("David == bString ");
40        }
41      }
```

Source Code: Src/4/String_1.java

Result:

```
equal
ava String_1
equal
David == aString
5
D
2
v
av
11
-->David<--
David == bString
```

A question was asked:

```
 1      class Sq {
 2
 3         public static void main(String args[]) {
 4              String aString = "0";
 5              String bString = "0" + "1";
 6              System.out.println(aString + "1" == "01" );
 7              System.out.println(bString  == "01" );
 8              System.out.println("0" + "1" == "01");
 9
10        }
11      }
```

Source Code: Src/4/Sq.java

Output:

```
false
true
true
```

Why is it not not true in all cases?

4.10. Strings and Numbers

```
 1
 2      class StringAndInteger
 3      {
 4         public static void main(String args[])
 5         {
 6            System.out.println("Well, 3 + 4 = " + 7 );
 7            System.out.println("Well, 3 + 4 = " + 3 + 4 );
 8            System.out.println("Well, 3 + 4 = " + ( 3 + 4 ) );
 9         }
10      }
11
```

Source Code: Src/4/StringAndInteger.java

```
 1
 2      class StringAndInteger2
 3      {
 4         public static void main(String args[])
 5         {
 6            System.out.println(3 + 7);
 7            System.out.println(3 + 7 + "abc");
 8            System.out.println(3 + 7 + "abc" + 1);
 9            System.out.println(3 + 7 + "abc" + 1 + 2);
10            System.out.println("" + 3 + 7 + "abc" + 1 + 2);
11            System.out.println("" + ( 3 + 7 ) + "abc" + ( 1 + 2 ));
12         }
13      }
14
```

Source Code: Src/4/StringAndInteger2.java

4.11. More on Strings

```
 1      /**
 2        * Play with the String class
 3        *
 4        * @version    $Id$
 5        *
 6        * @author     Hpb
 7        *
 8        *     $Log$
 9        */
10
11      class StringUse {
12
13        public static void compare(String aString, String bString)    {
14              if ( aString.equals(bString) )
15                      System.out.println("\tequal");
16              else
17                      System.out.println("\t! equal");
18              if ( aString == bString)
19                      System.out.println("\t== ");
20              else
21                      System.out.println("\t! ==");
22
23        }
24        public static void main( String args[] ) {
25              String aString = "David";
26              String bString = "David";
27              compare(aString, bString);
28
29              System.out.println("Using New");
30              aString = new String("David");
31              bString = new String("David");
32              compare(aString, bString);
33
34              System.out.println("Concatenation 1");
35              aString = "Da" + "vid";
36              bString = "" + "David";
37              compare(aString, bString);
38
39              System.out.println("Concatenation 2");
40              aString = "Da" + "vid";
41              bString = "D" + "a" + "vid";
42              compare(aString, bString);
43
44        }
45      }
```

Source Code: Src/4/StringUse.java

Execution:

```
% java StringUse
equal
       ==
Using New
       equal
       ! ==
Concatenation 1
       equal
       ==
Concatenation 2
       equal
       ==
```

4.12. Confusion about this

```
 1      /**
 2       * Use of this!
 3       */
 4
 5      class UseOfThis
 6      {
 7         int id;
 8         UseOfThis(int id) {
 9              this.id = id;
10         }
11         private void method_2()
12         {
13              System.out.println("method_2: " + this);
14         }
15
16         private void method_1()
17         {
18              System.out.println("method_1: " + this);
19              this.method_2();
20              method_2();
21         }
22         public String toString()     {
23              return "" + id;
24         }
25
26         public static void main(String args[])
27         {
28            UseOfThis aUseOfThis = new UseOfThis(1);
29            UseOfThis bUseOfThis = new UseOfThis(2);
30
31            System.out.println(aUseOfThis);
32            System.out.println(bUseOfThis);
33
34            aUseOfThis.method_1();
35            bUseOfThis.method_1();
36         }
37      }
38
```

Source Code: Src/4/UseOfThis.java

4.13. Primitive/Basic Types and Values

A primitive type is predefined by the Java language and named by a reserved keyword. Please see also here. (http://java.sun.com/docs/books/jls/html/4.doc.html)

Remember:

A variable has the following properties:

- memory location to store the value.
- type of data stored in the memory location.
- name used to refer to the memory location.

Java knows the following types:

type	#bits	def. v.	minimum value	maximum value
byte	8 bits	0	-128	127
char	16 bits	0	0	65535
short	16 bits	0	-32768	32767
int	32 bits	0	-2147483648	2147483647
long	64 bits	0	-9223372036854775808	9223372036854775807
float	32 bits	0.0	-3.40282347E38	3.40282347E38
double	64 bits	0.0	-1.79E+308	1.79E308

Constants as in ANSI-C, constants consist of a decimal mantissa with a decimal point and an exponent with prefix e or E and optional sign. Many parts can be omitted, but one of a decimal point, an exponent, or a suffix must be present.

Constants are float only with the suffix f or F. With or without the suffix d or D they are double.

With the methods *Float.intBitsToFloat()* and *Double.longBitsToDouble()* one can change bitwise representations into floating point values.

Examples:

```
int     index;
int     milesPerHour, maximumSpeed;
float   pressure, sizeOfme;
double  starTrekSpeed;
int     picturesTakenSofar = 20;
double  probability = 0.789;
```

Conditions can only be expressed with boolean values.

boolean has the predefined constants

- true
- false.

The names are not reserved; however, they are only recognized as boolean constants.

4.14. Enum, EnumSet and EnumMap

Bitmap Solution for the problem to identify a few different cases:

```
int one   = 1;      // 0001
int three = 2;      // 0011
int bit   = 2// 0010
if ( one == ( one & bit ) )
             //     0001 &&
             //     0010
             // ==  0000 != 0001
      ...
if ( three == ( three & bit ) )
             //     0011 &&
             //     0001
             // ==  0000 == 0010
```

Enum (https://docs.oracle.com/javase/8/docs/api/java/util/EnumSet.html)
Solution for the problem to identify a few different cases

```
 1      import java.lang.Enum;
 2
 3      public class EnumExample        {
 4      enum TheColors {
 5                RED, GREEN, BLUE
 6        };
 7        public static void whatTheColors(TheColors theTheColors)      {
 8              if ( theTheColors == TheColors.RED )
 9                      System.out.println("Roses are red");
10              else    // can this happen?
11                      System.out.println("Unkonw color: " + theTheColors);
12        }
13        public static void main( String args[] ) {
14              whatTheColors(TheColors.RED);
15        }
16      }
```

Source Code: Src/4/EnumExample.java

From the documentation: "The space and time performance of this class should be good enough to allow its use as a high-quality, type safe alternative to traditional int-based "bit flags." Even bulk operations (such as containsAll and retainAll) should run very quickly if their argument is also an enum set." More later: Collection framework.

```
1      import java.util.EnumSet;
2      import java.util.Set;
3
4      public class EnumSetExample     {
5      enum Color {
6                RED, GREEN, BLUE
7        };
8        public static void main( String args[] ) {
```

```
 9              EnumSet<Color> redGreen = EnumSet.of(Color.RED, Color.GREEN);
10              EnumSet<Color> complementOf = EnumSet.complementOf(redGreen);
11              System.out.println("redGreen       = " + redGreen );
12              System.out.println("complementOf   = " + complementOf );
13        }
14      }
```

Source Code: Src/4/EnumSetExample.java

Result:

```
redGreen       = [RED, GREEN]
complementOf   = [BLUE]
```

4.15. Unicode

Skip

See also here. (http://www.unicode.org/)

and here: http://www.unicode.org/charts/ (http://www.unicode.org/charts/)

It is necessary for a modern environment to handle, uniformly and comfortably, the textual representation of all the major written languages.

Unicode Standard defines a uniform 16-bit code based on the principle of unification:

two characters are the same if they look the same even though they are from different languages.

This principle, called Han unification, allows the large Japanese, Chinese, and Korean character sets to be packed comfortably into a 16-bit representation.

The UTF encoding of the Unicode Standard is backward compatible with ASCII.

Letting numbers be binary, a rune c is converted to a multibyte UTF sequence as follows: (See also UTF-8 (https://www.w3schools.com/charsets/ref_html_utf8.asp)

1. c in [00000000.0bbbbbbb] 0bbbbbbb
2. c in [00000bbb.bbbbbbbb] 110bbbbb, 10bbbbbb
3. c in [bbbbbbbb.bbbbbbbb] 1110bbbb, 10bbbbbb, 10bbbbbb

A byte in the ASCII range 0-127 represents itself in UTF.

Thus UTF is backward compatible with ASCII.

4.16. String to int

```
 1      /**
 2       * Converts a String to an int - this is one way out of many
 3       *
 4       */
 5
 6      class  StringToInt
 7      {
 8         public static void main(String args[])
 9         {
10            int i;
11            // Integer aInt = new Integer("4");
12            Integer aInt = Integer.parseInt("4");
13            i = aInt.intValue();
14            i = Integer.parseInt("4");
15
16         }
17      }
18
```

Source Code: Src/4/StringToInt.java

See also: here. (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/lang/Integer.html)

4.17. Arithmetic Expressions

http://www.cs.rit.edu/~cs1/Exercises/_Expressions (http://www.cs.rit.edu/~cs1/Exercises/_Expressions)

4.18. Arithmetic Operators

- the table below shows some of the arithmetic operators that Java provides, in the order of their precedence.
- parentheses can be used to change the order of evaluation
- an arithmetic expression returns (calculates) a value when executed.

binary Operator	Description
+	addition
-	subtraction
*	multiplication
/	integer division, if both operands are integer; real division otherwise
%	remainder
<<	bit shift left
>>	bit shift right
>>>	unsigned right shift
&	bitwise and
^	bitwise xor
|	bitwise or

Bit operation example:

```
 1      class BitPrint {
 2              private void printBytes (String what, int value) {
 3                      System.out.print(what + "\t=\t" + value + "\t=\t");
 4                      for ( int index = 31; index >= 0 ; index --)    {
 5                              if ( ( ( 1 << index ) & value ) == ( 1 << index ) )
 6                                      System.out.print("1");
 7                              else
 8                                      System.out.print("0");
 9
10                      }
11                      System.out.println();
12              }
13
14              public static void main (String args []) {
15                      BitPrint aBitPrint = new BitPrint();
16
17                      aBitPrint.printBytes("3       ", 3);
18                      aBitPrint.printBytes("-3      ", -3);
19                      aBitPrint.printBytes("5       ", 5);
20                      aBitPrint.printBytes("5 >> 1  ", (5 >> 1));
21                      aBitPrint.printBytes("5 >> 2  ", (5 >> 1));
22                      aBitPrint.printBytes("-5      ", -5);
23                      aBitPrint.printBytes("-5 >> 1 ", (-5 >> 1));
24                      aBitPrint.printBytes("-5 >>> 1", (-5 >>> 1));
25              }
26      }
```

Source Code: Src/4/BitPrint.java

Result:

```
% javac BitPrint.java && java BitPrint
3                   =           3           =           00000000000000000000000000000011
-3                  =           -3          =           11111111111111111111111111111101
5                   =           5           =           00000000000000000000000000000101
5 >> 1              =           2           =           00000000000000000000000000000010
5 >> 2              =           2           =           00000000000000000000000000000010
-5                  =           -5          =           11111111111111111111111111111011
-5 >> 1             =           -3          =           11111111111111111111111111111101
-5 >>> 1            =           2147483645          =           01111111111111111111111111111101
```

4.19. Unary/Binary Operator

unary Operator	Description
++	increment by 1
--	decrement by 1
!	not

Example:

```
int left;
int right;
int result;

     left  = 4;
     right = 3;
     result = 4 * 3;
     result = 4 + 3;
     result = 4 / 3;
     result = 4 % 3;

     left ++;
     result = left++;
     result = ++left;

     right ++;
     result = right++;
     result = ++right;
```

4.20. Mixed Mode Arithmetic and Casting

When an expression contains more than one arithmetic type all are converted to the heaviest.

byte → char → short → int → long → float → double

For example, 2 + 3.3 is interpreted as 2.0 + 3.3.

Java is strongly typed. However, the type of the results of evaluating an expression may be changed using casting. To cast, place the target type in parentheses before the operand or expression to be converted.

For example, if we really wanted the results of 2 + 3.3 to be integer, we could use:

```
double aDouble;
int aInt;

        aInt = 2 + (int) 3.3;
        aInt = (int) (2 + 3.3);
        aDouble = 2 + 3.3;
```

Example:

```
 1      /**
 2       * The program deals with operators.
 3       * Comment not included.
 4       *
 5       * @version   $Id$
 6       *
 7       * @author    hp bischof
 8       *
 9       * Revisions:
10       *      $Log$
11       */
12
13      class OpEx
14      {
15         public static void main(String args[])
16         {
17         char aChar           = 'b';
18         byte aByte           = 2;
19
20         int  intVar_1        = 1;
21         int  intVar_2        = 2;
22         int  intRes          = 3;
23         double       doubleVar_1     = 3.8;
24         double       doubleVar_2     = 4.8;
25         double       doubleRes       = doubleVar_1 - doubleVar_2;
26
27         System.out.println("1. " + aChar);           // man ascii decimal set
28         System.out.println("2. " + aByte);
29         System.out.println("3. " + aByte+aChar);
30         System.out.println("4. " + aByte+0);
31         System.out.println("5. " + aChar+0);
32
33         intRes = 5 / 3;        System.out.println("6. " + intRes);
34         intRes = 5 % 3;        System.out.println("7. " + intRes);
35      // intRes = 5 / doubleVar_2;    // Doesn't work, why?
36         intRes = (int)(5 / doubleVar_2); System.out.println("8. " + intRes);
37
38         doubleRes = 5   / doubleVar_2; System.out.println("9. " + doubleRes);
39         doubleRes = 5.0 / doubleVar_2; System.out.println("10. " + doubleRes);
40         }
41      }
```

Source Code: Src/4/OpEx.java

Execution:

```
% javac OpEx.java
% java OpEx
1. b
2. 2
3. 2b
```

4. 20
5. b0
6. 1
7. 2
8. 1
9. 1.0416666666666667
10. 1.0416666666666667

4.21. Assignment Operators

There are 12 assignment operators; all are syntactically right-associative (they group right-to-left). See also here. (http://java.sun.com/docs/books/jls/html/15.doc.html#5281)

= *= /= %= += -=
<<= >>= >>>= &= ^= |=

<<=	bit shift left
>>=	bit shift right
>>>=	unsigned right shift
&=	and
^=	xor
|=	or

The syntax for an assignment is:

Assignment:
LeftHandSide AssignmentOperator AssignmentExpression

LeftHandSide	must be a variable
AssignmentOperator	must be one of = *= /= %= += -= <<= >>= >>>= &= ^= |=
AssignmentExpression	must be ConditionalExpression or Assignment

Note: A variable that is declared final cannot be assigned to.

4.22. Playing with the Basic Types

```
1       /**
2        * Ranges demonstrates integer value manipulation,
3        * bit operations, and conversions.
4        * Comment not included.
5        * @version   $Id$
6        *
7        * @author    Axel T. Schreiner
8        * @author    hp bischof
9        *
10       * Revisions:
11       *      $Log$
12       */
13
14      class Ranges {
15              /** uses complement operations */
16              short x;
17              void intRange () {
18                      System.out.println("int\t" + (~0 >>> 1) +
19                              "\t" + (~ (~0 >>> 1)));
20              }
21              /** maximum and minimum long value */
22              static final long maxLong = ~0L >>> 1;
23              static final long minLong = ~ (~0L >>> 1);
24
25              void longRange () {
26                      System.out.println("long\t" + maxLong + "\t" + minLong);
27              }
28
29              /** uses casts and literals */
30              void shortRange () {
31                      System.out.println("short\t" + Short.MIN_VALUE + "\t" +
32                      Short.MAX_VALUE);
33                      System.out.println("short\t" + (short)077777 + "\t" +
34                      (short)0x8000);
35              }
36              /** shifts ones until no further changes occur */
37              void byteRange () {
38                      byte i, j = 1;
39
40                      do {
41                              i = j; j = (byte)(i << 1 | 1);
42                      } while (j > i);
43                      System.out.print("byte\t" + i);
44
45                      do {
46                              i = j; j = (byte)(i << 1);
47                      } while (j < i);
48                      System.out.println("\t" + i);
49              }
50
51
```

```
52              public static void main (String args []) {
53                      Ranges aRange = new Ranges();
54
55                      aRange.byteRange();
56                      aRange.shortRange();
57                      aRange.intRange();
58                      aRange.longRange();
59              }
60
61      }
```

Source Code: Src/4/Ranges.java

Result:

```
% java Ranges
byte          127           -128
short         -32768        32767
short         32767         -32768
int           2147483647          -2147483648
long          9223372036854775807         -9223372036854775808
```

intRange()
: produces a maximal bit pattern by complementing 0 and shifting without propagating the sign, and a minimal bit pattern by complementing once more.

\+
: concatenates strings; integer operands are converted to short strings. java/gtext has facilities for explicit formatting.

static
: defines *maxLong* and *minLong* as class variables,

final
: permits exactly one assignment. Initialization is done with a long literal and thus long arithmetic.

shortRange()
: uses int literals which must be explicitly cast to short. The following expression produces a minimal value in C but not in Java (why?):

```
(short) ~ ((unsigned short) ~0 >> 1)
```

byteRange()
: uses do-while loops. byte can be initialized with int, but for assignment an explicit cast is required.

4.23. Control Flow

4.24. Conditions

• Conditions can only be expressed with boolean values; unlike C, an implicit comparison to zero is not acceptable.

• Boolean has the predefined constants true and false. The names are not reserved; however, they are only recognized as boolean constants.

4.25. Relational Operators

• Simple boolean expressions consist of comparing things using relational operators. There two types of relational operators: equality and comparison.

•

Equality operators are defined for all objects and primitive types.

== equal
!= not equal

All comparisons produce true or false.

4.26. Logical Operators

These operators take boolean arguments and return boolean results.

• They are used to construct complex boolean expressions from simple ones consisting of boolean values and relational expressions.

&	bitwise and * - not Logical Operators
&&	conditional and (short circuits)
|	bitwise or * - not Logical Operators
||	conditional or (short circuits)
^	bitwise xor * - not Logical Operators
!	not (unary operator)/boolean complement

x && y	y will be evaluated only if x is true
x || y	y will be evaluated only if x if false

• Executed from left to right

```
 1      class LeftToRight {
 2
 3         public static int f(String whichTest, int x) {
 4              System.out.println("f(" + x + ")");
 5              return 1 / (3 - x );
 6
 7         }
 8         public static void test_1()  {
 9              boolean y;
10              for (int x = 0; x < 5; x ++ )           {
11                      y = ( x != 3 ) && ( f("1", x) > 0.5);
12              }
13         }
14         public static void test_2()  {
15              boolean y;
16              for (int x = 0; x < 5; x ++ )           {
17                      y = f("2", x) > 0.5 && ( x != 3 );
18              }
19         }
20         public static void main(String args[]) {
21              test_1();
22              test_2();
23         }
24
25      }
```

Source Code: Src/4/LeftToRight.java

Note:

```
 1
 2      class ShortCut
 3      {
 4         private boolean testIt(double n) {
 5              return n != 0.0;
 6         }
 7         private double oneDn(double n)      {
 8              return 1.0 / n;
 9         }
10
11         public static void main(String args[])
12         {  double n;
13            ShortCut aS = new ShortCut();
14
15            n = 0.5;
16            if ( aS.testIt(n) && ( aS.oneDn(n) > 1 ) )
17                    System.out.println("1:  1 / n " + 1 / n );
18
19            n = 0;
20            if ( aS.testIt(n) && ( aS.oneDn(n) > 1 ) )
21                    System.out.println("2:  1 / n " + 1 / n );
22
23            System.out.println("3: 1.0 / 0 = " + 1.0 / 0 );
24
25            if ( ( n == 0 ) || ( n == 1 ) )
26                    System.out.println("4: ( n == 0 ) || ( n == 1 ) )");
27
28             System.out.println("5. true || false && true: " + ( true || false && tr
29
30             if ( 4 == ( n = 4  ) )
31                     System.out.println("6. 4 == ( n = 4  )");
32         }
33      }
34
```

Source Code: Src/4/ShortCut.java

Result:

```
% javac ShortCut.java && java ShortCut
1:  1 / n 2.0
3: 1.0 / 0 = Infinity
4: ( n == 0 ) || ( n == 1 ) )
5. true || false && true: true
6. 4 == ( n = 4  )
```

The precedence for the arithmetic, relational, boolean operators, and assignment from highest to lowest is:

Operation	Symbol	Precedence	Association
		(from highest, 1, to lowest, 11)	
grouping	( )	1	left to right
unary	+ -	2	right to left
multiplication	*	3	left to right
division	/	3	left to right
remainder	%	3	left to right
addition	+	4	left to right
subtraction	-	4	left to right
less than	<	5	left to right
less than or equal	<=	5	left to right
greater than	>	5	left to right
greater than or equal	>=	5	left to right
equal	==	6	left to right
not equal	!=	6	left to right
bit and	&	7	left to right
xor	^	8	left to right
bit or	|	9	left to right
conditional and	&&	10	left to right
conditional or	||	11	left to right
assignment	=, +=, *= ...	12	N.A.

4.27. if Statement

See also http://docs.oracle.com/javase/specs/jls/se8/html/jls-14.html#jls-14.9 (http://docs.oracle.com/javase/specs/jls/se8/html/jls-14.html#jls-14.9)

The if statement allows conditional execution of a statement or a conditional choice of two statements, executing one or the other but not both.

IfThenStatement:

```
if ( Expression )
        Statement
```

Example:

```
x = 3;
y = 4;
if ( x > y )
    z = x;
```

IfThenElseStatement:

```
if ( Expression )
        Statement
else
        Statement
```

Example:

```
x = 3;
y = 4;
if ( x > y )
    z = x;
else
    z = y;
```

How often and why is index incremented?

```
 1      class If
 2      {
 3         public static void main(String args[]) {
 4              int index = 2;
 5                      System.out.println("1. " + index );
 6                      if ( ++index == 2 )
 7                              System.out.println("2. " + index );
 8                      else if ( index++  == 3 )
 9                              System.out.println("3. " + index );
10                      else
11                              System.out.println("4. " + index );
12         }
13
14      }
```

Source Code: Src/4/If.java

Execution:

```
% java If
1. 2
3. 4
```

4.28. Find the Maximum of two Numbers I

```
 1      /**
 2        *  Find the maximum of two numbers.
 3        *
 4        * @version    $Id$
 5        *
 6        * @author     Hpb
 7        *
 8        * Revisions:
 9        *     $Log$
10        */
11
12      class Maximum_2 {
13        public static double maximum(double _first, double _second ) {
14              double max;
15                                      // find maximum
16              if ( _first > _second )
17                      max = _first;
18              else
19                      max = _second;
20              return max;
21        }
22
23        private double minimum(double _first, double _second ) {
24              double minimum;
25                                      // find minimum
26              if ( _first < _second )
27                      minimum = _first;
28              else
29                      minimum = _second;
30              return minimum;
31        }
32
33        public static void main( String args[] ) {
34          Maximum_2 aMaximum_2 = new Maximum_2();
35          double firstN    = 42.0;
36          double secondN   = 666.0;
37
38          System.out.println("Maximum(" + firstN + ", " + secondN + ")  =  " +
39                             aMaximum_2.maximum( firstN, secondN) );
40
41          System.out.println("Minimum(" + firstN + ", " + secondN + ")  =  " +
42                             aMaximum_2.minimum( firstN, secondN) );
43        }
44      }
```

Source Code: Src/4/Maximum_2.java

As a Nassi Shneidermann diagram:

public static double maximum(double _first, double _second)

if if (_first > _second) then Yes : return _first else No
return _second }

public static double minimum(double _first, double _second)

if if (_first < _second) then Yes : return _first else No
return _second }

4.29. The Conditional Operator

?:

uses the boolean value of one expression to decide which of two other expressions should be evaluated.

The conditional operator is syntactically right-associative (it groups right-to-left), so that

```
a ? b : c ? d : e ? f : g
```

means the same as:

```
a ? b : (c ? d : (e ? f : g)).
```

The conditional operator may be used to choose between second and third operands of numeric type, or second and third operands of type boolean, or second and third operands that are each of either reference type or the null type. All other cases result in a compile-time error.

See also: http://java.sun.com/docs/books/jls/html/15.doc.html#5257 (http://docs.oracle.com/javase/specs/jls/se8/html/jls-15.html#jls-15.2)

```
 1
 2      class QuestionM
 3      {
 4         public static void main(String args[]) {
 5              int value       =        2 > 3 ? 2 : 3;
 6              String aString  = "      2 > 3 ? 2 : 3";
 7              System.out.println(aString + " = " + value);
 8
 9              value           =        ( 1 > 2 ? 3 :( 4 < 5 ? 6 : 7));
10              aString         = "      ( 1 > 2 ? 3 :( 4 < 5 ? 6 : 7))";
11              System.out.println(aString + " = " + value);
12
13              value           =        1 > 2 ? 3 : 4 > 5 ? 8 : 7;
14              aString         = "      1 > 2 ? 3 : 4 > 5 ? 8 : 7";
15              System.out.println(aString + " = " + value);
16         }
17      }
```

Source Code: Src/4/QuestionM.java

Execution:

```
% java QuestionM
2 > 3 ? 2 : 3 = 3
( 1 > 2 ? 3 :( 4 < 5 ? 6 : 7)) = 6
1 > 2 ? 3 : 4 > 5 ? 8 : 7 = 7
```

Example:

```
 1      class Maximum_3 {
 2        private double maximum(double _first, double _second ) {
 3              return _first > _second ? _first : _second;
 4        }
 5
 6        public static double minimum(double _first, double _second ) {
 7              return _first < _second ? _first : _second;
 8        }
 9
10        public static void main( String args[] ) {
11          Maximum_3 aMax = new Maximum_3();
12          double firstN    = 42.0;
13          double secondN   = 7.0;
14
15
16          System.out.println("Maximum(" + firstN +
17              ", " + secondN + ") =  " +
18              aMax.maximum( firstN, secondN) );
19
20          System.out.println("Minimum(" + firstN +
21              ", " + secondN + ") =  "
22              + aMax.minimum( firstN, secondN) );
23        }
24      }
```

Source Code: Src/4/Maximum_3.java

4.30. while Statement

See also http://docs.oracle.com/javase/specs/jls/se8/html/jls-16.html#jls-16.2.10 (http://docs.oracle.com/javase/specs/jls/se8/html/jls-16.html#jls-16.2.10)

The while statement executes an Expression and a Statement repeatedly until the value of the Expression is false.

WhileStatement:

while (Expression)
    Statement

Example:

```
x = 1;
while ( x < 10 )    {
    print x
    x += 2;
}
```

```
 1
 2
 3      class While
 4      {
 5         public static void main(String args[]) {
 6              int index = 1;
 7              while ( index > 0 ?  ( index == 7 ) : (index == 8 ) ) {
 8                      System.out.println("index = " + index );
 9              }
10              System.out.println("index = " + index );
11         }
12      }
```

Source Code: Src/4/While.java

Execution:

```
% java While
index = 1
```

An example:

```
 1
 2
 3      class While_1
 4      {
 5         public static void main(String args[]) {
 6              int index = 1;
 7              while ( ++index++ < 4 ) {
 8                      System.out.println("index = " + index );
 9              }
10              System.out.println("index = " + index );
11         }
```

```
12      }
```

Source Code: Src/4/While_1.java

An example:

```
 1
 2
 3
 4      class While_2
 5      {
 6         public static void main(String args[]) {
 7              int index = 1;
 8              while ( ++index < 4 ) {
 9                      System.out.println("index = " + index );
10              }
11              System.out.println("index = " + index );
12         }
13      }
```

Source Code: Src/4/While_2.java

Execution:

```
% java While_2
index = 2
index = 3
index = 4
```

The Expression must have type boolean, or a compile-time error occurs.

A while statement is executed by first evaluating the Expression. If evaluation of the Expression completes abruptly for some reason, the while statement completes abruptly for the same reason. Otherwise, execution continues by making a choice based on the resulting value:

If the value is true, then the contained Statement is executed. Then there is a choice:

If execution of the Statement completes normally, then the entire while statement is executed again, beginning by re-evaluating the Expression.

If the value of the Expression is false, no further action is taken and the while statement completes normally.

If the value of the Expression is false the first time it is evaluated, then the Statement is not executed.

4.31. Calculate Sqrt(2) without the MathClass

Algorithm: What is your idea?

```
 1      /**
 2        *  Calculate Sqrt(2) without the MathClass
 3        *
 4        * @version    $Id$
 5        * @author     Hpb
 6        * Revisions:
 7        *     $Log$
 8        */
 9
10      class Sqrt {
11
12        static double epsilon = 0.000001;
13
14        private double calculateSqrt_2() {
15              double leftOfSqrtOf2 = 1.0;
16              double rightOfSqrtOf2 = 2.0;
17              while ( (rightOfSqrtOf2 * rightOfSqrtOf2 - leftOfSqrtOf2 * leftOfSqrtO
18                      double pointInbetween = ( rightOfSqrtOf2 + leftOfSqrtOf2 ) * 0
19                      if ( pointInbetween * pointInbetween > 2.0 )
20                              rightOfSqrtOf2 = pointInbetween;
21                      else
22                              leftOfSqrtOf2 = pointInbetween;
23              }
24              return leftOfSqrtOf2;
25        }
26
27        public static void main( String args[] ) {
28
29          System.out.println("sqrt(2)  = " +
30                             new Sqrt().calculateSqrt_2() + " +-" + epsilon );
31        }
32      }
```

Source Code: Src/4/Sqrt.java

4.32. Continue

• continue statement may occur only in a while, do, or for statement;

• continue statement with no label attempts to transfer control to the innermost enclosing while, do, or for statement;

• this statement, which is called the continue target, then immediately ends the current iteration and begins a new one.

• If no while, do, or for statement encloses the continue statement, a compile-time error occurs.

http://docs.oracle.com/javase/specs/jls/se8/html/jls-14.html#jls-14.16 (http://docs.oracle.com/javase/specs/jls/se8/html/jls-14.html#jls-14.16)

Example 1:

```
 1      class Continue_1 {
 2
 3        public static void main( String args[] ) {
 4
 5                int n = 0;
 6
 7      label1:
 8                 while ( n < 6 )  {
 9                      System.out.println("1. n == " + n);
10
11                      while ( n < 4 )  {
12                              n++;
13                              System.out.println("    2.  n == " + n);
14                              if ( n > 2 )
15                                      continue label1;
16                              System.out.println("    3.  n == " + n + "--------");
17                      }
18                      n++;
19              }
20        }
21      }
```

Source Code: Src/4/Continue_1.java

Execution:

```
% java Continue_1
1. n == 0
       2.  n == 1
       3.  n == 1--------
       2.  n == 2
       3.  n == 2--------
       2.  n == 3
1. n == 3
       2.  n == 4
1. n == 4
1. n == 5
```

4.33. Break

• A break statement transfers control out of an enclosing iteration or switch statement.

• Labeled breaks are used to jump out of nested loops

• break statement with no label attempts to transfer control to the innermost enclosing switch, while, do, or for statement;

• If no switch, while, do, or for statement encloses the break statement, a compile-time error occurs.

http://docs.oracle.com/javase/specs/jls/se8/html/jls-14.html#jls-14.15 (http://docs.oracle.com/javase/specs/jls/se8/html/jls-14.html#jls-14.15)

Example 1:

```
 1      class Break_1 {
 2
 3        public static void main( String args[] ) {
 4
 5              int n = 0;
 6
 7         here:        {
 8              while ( true )  {
 9                      System.out.println("1. n == " + n);
10
11                      while ( n < 100 )  {    // while ( true ) --> which problem
12                              n++;
13                              System.out.println("    2. n == " + n);
14                              if ( n > 2 )
15                                      break here;
16                      }
17                      System.out.println("3.  n == " + n);
18                 }
19              }
20        }
21      }
```

Source Code: Src/4/Break_1.java

Example 2:

```
 1      class Break_2 {
 2
 3        public static void main( String args[] ) {
 4
 5              int n = 0;
 6
 7              System.out.println("start");
 8              while ( n < 100 )  {
 9                      if ( n > 4 )
10                              System.exit(1);
11
12                      while ( n < 100 )  {    // while ( true ) --> which problem
13                              n++;
14                              System.out.println("    inner while here n == " + n);
15                              if ( n > 2 )
```

```
16                                      break;
17                      }
18                      System.out.println("outer while here n == " + n);
19                  }
20              System.out.println("after here }");
21        }
22      }
```

Source Code: Src/4/Break_2.java

4.34. Return

A return statement returns control to the invoker of a method or constructor.

```
1       /* How can we set the exit code?        */
2
3       class Return {
4
5         public static void main( String args[] ) {
6               int x = 0;
7               return x;
8         }
9       }
```

Source Code: Src/4/Return.java

An other example:

```
1       /* How can we set the exit code?        */
2
3       class Return_1 {
4
5         public static int method() {
6               System.exit(2);
7               return 0;
8         }
9         public static void main( String args[] ) {
10              method() ;
11              System.out.println("xxx");
12        }
13      }
```

Source Code: Src/4/Return_1.java

4.35. Return vs. Continue vs. Break

• What are the differences?

4.36. Abrupt Completion

Abrupt completion of the contained Statement is handled in the following manner:

An abrupt completion always has an associated reason, which is one of the following: (from here) (https://books.google.com/books?id=mh-KAwAAQBAJ&pg=PA406&lpg=PA406&dq=Abrupt+Completion+java&source=bl&ots=qg-CYqVIz61&sig=NF0eCzgBNtcPKe3zJ0bd14RPtXQ&hl=en&sa=X&ved=0CEQQ6AEwBmoVCh-MIgP_2kKPUxwIVxm0-Ch2i6gid#v=onepage&q=Abrupt%20Completion%20java&f=false)

• A break with no label

• A break with a given label

• A continue with no label

• A continue with a given label

• A return with no value

• A return with a given value

• A throw with a given value, including exceptions thrown by the Java virtual machine

```
1       class Break {
2
3         public static void main( String args[] ) {
4
5               int n = 0;
6
7          here: {
8               while ( true )  {
9                       System.out.println("a: outer while here n == " + n);
10
11                      if ( n > 4 )
12                              System.exit(1);
13
14                      while ( true )  {
15                              System.out.println("    inner while here n == " + n);
16                              if ( ++n == 0 )
17                                      System.out.println("n == 0");
18                              else if ( n++ == 1 )    {
19                                      System.out.println("    n == 1");
20                                      System.out.println("    break");
21                                      break;
22                              } else if ( n++ == 2 )
23                                      System.out.println("    n == 2");
24                              else
25                                      System.out.println("    n == 3");
26
27                              System.out.println("    executing break here");
28                              System.out.println("    n is " + n );
29
30                              break here;
31
32                      }
33                      System.out.println("b: outer while here n == " + n);
34                 }
35                 // unreachable statement ...System.out.println("here }");
36              }
37        }
38      }
```

Source Code: Src/4/Break.java

4.37. do Statement

See also here (http://docs.oracle.com/javase/specs/jls/se8/html/jls-16.html#jls-16.2.11)

The do statement executes a Statement and an Expression repeatedly until the value of the Expression is false.

```
do Statement while ( Expression ) ;
```

4.38. for Statement

See also (http://docs.oracle.com/javase/specs/jls/se8/html/jls-14.html#jls-14.14)

The *for statement* executes some initialization code, then executes an Expression, a Statement, and some update code repeatedly until the value of the Expression is false.

ForStatement:

```
f.or ( ForInit; Expression; ForUpdate)
            Statement
```

Example:

```
 1      class For_1 {
 2
 3        public static void main( String args[] ) {
 4              int index = 0;
 5              for ( index = 0 ; index < 1; index ++  )        {
 6                      System.out.println("1. index = " + index );
 7              }
 8              System.out.println("2. index = " + index );
 9        }
10      }
```

Source Code: Src/4/For_1.java

An other example:

An other example:

```
 1      class For_2 {
 2
 3        public static void main( String args[] ) {
 4              int index = 0;
 5              for ( index = 0 ; index < 1; index ++  )        {
 6                      index = -1;
 7                      System.out.println("1. index = " + index );
 8                      break;
 9              }
10              System.out.println("2. index = " + index );
11        }
12      }
```

Source Code: Src/4/For_2.java

An other example:

```
 1      class For_3 {
 2
 3        public static void main( String args[] ) {
 4              for ( int index = 0 ; index < 1; index ++  )    {
 5                      System.out.println("1. index = " + index );
 6                      break;
 7              }
```

```
 8                System.out.println("2. index = " + index );
 9          }
10        }
```

Source Code: Src/4/For_3.java

4.39. Find all Prime Numbers in [2 ... 100]

isPrime(n): do for index = 2 to n - 1 : if (index % n == 0) : return false } : return true *findAllPrimeN()*: do for index = 1 to 100 : if (isPrime(n)) : print index }

```
 1      /**
 2        *  Find all prime numbers in the range
 3        *  between 1 and 10
 4        *
 5        * @version    $Id$
 6        *
 7        * @author     Hpb
 8        *
 9        * Revisions:
10        *     $Log$
11        */
12
13      class Prime_1 {
14
15        private boolean isPrime(int n) {
16
17              for ( int index = 2; index < n; index ++ ) {
18                      if ( n % index  == 0 )
19                              return false;
20              }
21
22              return true;
23        }
24        public static void main( String args[] ) {
25          Prime_1 aPrime = new Prime_1();
26
27          for ( int index = 2; index <= 10; index ++ )
28              if ( aPrime.isPrime(index) )
29                      System.out.println(index + " " );
30        }
31      }
```

Source Code: Src/4/Prime_1.java

4.40. Switch Statement

See also here (http://docs.oracle.com/javase/specs/jls/se8/html/jls-14.html#jls-14.11)

The switch statement transfers control to one of several statements depending on the value of an expression. The type of the switch expression can be

- char
- byte
- short
- int
- strings
- enum type See here (http://docs.oracle.com/javase/specs/jls/se8/html/jls-14.html#jls-14.11)

Switch Statement:

```
switch ( Expression )  {
    case ConstantExpression_1 : action_1;
    case ConstantExpression_2 : action_2;
    ...
    default: action_d
```

Example:

```
 1      class Switch {
 2
 3      static void method(int k) {
 4              switch (k) {
 5              case 1:  System.out.println("with break: 1 ");
 6                       break;
 7              case 2:  System.out.println("with break: 2 ");
 8                       break;
 9              default: System.out.println("with break: default");
10              }
11      }
12
13      static void methodWithoutDefault(int k) {
14              switch (k) {
15              case 1:  System.out.println("   without break: 1 ");
16                       break;
17              case 2:  System.out.println("   without break: 2 ");
18                       break;
19              }
20      }
21
22      public static void main(String[] args) {
23              new Switch().method(3);
24              new Switch().methodWithOutDefault(2);
25              new Switch().methodWithOutDefault(3);
26
27        }
28      }
```

Source Code: Src/4/Switch.java

4.41. Partial Lowercase → Uppercase

```
 1      /**
 2        *  Test of the switch statement.
 3        *
 4        * @version    $Id$
 5        *
 6        * @author     hpb
 7        *
 8        * Revisions:
 9        *     $Log$
10        */
11
12
13      class Switch_1 {
14        private String itWasA(char c) {
15              switch( c )     {
16              case 'a':       return("A");    // break?
17              case 'b':       return("B");    // break?
```

```
18              case 'c':       return("C");    // break?
19              case 100:       return("D");    // break?
20              case 101:       return("E");    // break?
21              default:        return("no clue, but not an [a-e]");
22                                              // What happens if
23                                              // we delete this line?
24              }
25        }
26
27
28        public static void main( String args[] ) {
29            char theChar;
30
31            theChar = 'd';
32            System.out.println("You typed in an '" +
33                      new Switch_1().itWasA(theChar) + "'");
34
35            System.exit(0);   // beware of ...
36        }
37      }
```

Source Code: Src/4/Switch_1.java

Characters can be safely converted to a integers (Unicode), but should be avoided at all times.

4.42. Questions

• Which variable names are valid:

```
1       class X_1
2       {
3          public static void main(String args[])
4          {
5             int aInt;
6             int countUpTo5;
7             int 5IsA_niceNumber;
8             int ooo\";
9             int notToMany:areAllowd;
10         }
11      }
12
```

Source Code: Src/4/X_1.java

• What is the output of the following program:

```
1       class X_2
2       {
3          public static void main(String args[])
4          {
5             System.out.println("I like to play " + 6  + 2 );
6             System.out.println("I like to play " + 6  * 2 );
```

```
 7            System.out.println("I like to play " + ( 6  + 2 ) );
 8         }
 9      }
10
```

Source Code: Src/4/X_2.java

• Will the following program compile?

```
 1      class X_3
 2      {
 3         public static void main(String args[])
 4         {
 5            i += 64;
 6            System.out.println("1. " + ( i << 2 ) );
 7         }
 8      }
 9
```

Source Code: Src/4/X_3.java

• What is the output of the following program:

```
 1      class X_4
 2      {
 3         public static void main(String args[])
 4         {
 5            int i = 0;
 6            i += 63;
 7            System.out.println("1. " + ( i++ >> 2 ) );
 8            System.out.println("2. " + ( 1 > 2 ? 3 : 6 ));
 9
10
11              /*
12               *      a ? b : c ? d : e ? f : g
13               *      means the same as
14               *      a ? b : (c ? d : (e ? f : g)).
15               */
16            System.out.println("3. " +
17               ( 1 > 2 ? 3 : 4 < 5 ? 6 : 9 < 10 ? 7 : 8));
18            System.out.println("4. " +
19               (1 > 2 ? 3 : (4 < 5 ? 6 : (9 < 10 ? 7 : 8))));
20         }
21      }
22
```

Source Code: Src/4/X_4.java

• What is the output of the following program:

```
 1      class X_5 {
 2
 3        public static void main( String args[] ) {
 4
 5              int n = 0;
 6
 7              while ( true )  {
 8                      System.out.println("xx");
 9                      if ( n++ == 0 ) {
10                              System.out.println("n == 0");
11                      } else if ( n++ == 1 )  {
12                              System.out.println("n == 1");
13                      } else if ( n-- == 2 )
14                              System.out.println("n == 2");
15
16              }
17        }
18      }
```

Source Code: Src/4/X_5.java

5. Scanner: Overview

• Introduced to satisfy faculty, students, and anyone who wants to write a quick-and-dirty Java program that uses console I/O.

• Works like StreamTokenizer

• Implements Iterator<String>

• See here: https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/Scanner.html (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/Scanner.html)

• Perl-like pattern matching available

• See here: https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/regex/Pattern.html (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/regex/Pattern.html)

5.1. Scanner: Constructors

• Scanner(File source)

• Scanner(InputStream source)

• Scanner(String source)

• System.in is an InputStream

• There are also constructors to work with:
alternate character sets; input objects from the java.nio library.

5.2. Scanner: Subset of Reading Methods

• String next()

• String next(Pattern pattern)

• boolean nextBoolean()

• double nextDouble()

• int nextInt()

• int nextInt(int radix)

• String nextLine()

5.3. Scanner: Subset of Testing Methods

• boolean hasNext()

• boolean hasNext(Pattern ptrn)

• boolean hasNextBoolean()

• boolean hasNextDouble()

• boolean hasNextInt()

• boolean hasNextInt(int radix)

• boolean hasNextLine()

5.4. Scanner: Example 1

```
1       import java.util.Scanner;
2
3       public class Scanner1 {
4           public static void main( String[] args ) {
5               Scanner sc  = new Scanner( System.in);
```

```
 6              System.out.printf("> ");
 7              while ( sc.hasNext() )  {
 8                      String line = sc.nextLine();
 9                      System.out.printf("-%s-%n", line );
10                      System.out.printf("> ");
11              }
12              sc.close();
13          }
14      }
15
16
```

Source Code: Src/6_jdk15/Scanner1.java

Exection:

```
% java Scanner1
> 1 2 3 4 hello
-1 2 3 4 hello-
> ups
-ups-
> # ^D here ....
```

5.5. Scanner: Example 2

```
 1
 2      import java.util.Scanner;
 3
 4      public class Scanner2 {
 5          public static void main( String[] args ) {
 6              Scanner sc  = new Scanner( System.in);
 7              System.out.printf("> ");
 8              while ( sc.hasNext() )  {
 9                      Integer aInteger = sc.nextInt();
10                      System.out.printf("-%d-%n", aInteger );
11                      System.out.printf("> ");
12              }
13              sc.close();
14          }
15      }
16
17
```

Source Code: Src/6_jdk15/Scanner2.java

Execution:

```
% java Scanner2
> 1 2 3 4 1.0
-1-
> -2-
> -3-
> -4-
```

```
> Exception in thread "main" java.util.InputMismatchException
        at java.util.Scanner.throwFor(Scanner.java:819)
        at java.util.Scanner.next(Scanner.java:1431)
        at java.util.Scanner.nextInt(Scanner.java:2040)
        at java.util.Scanner.nextInt(Scanner.java:2000)
        at Scanner2.main(Scanner2.java:9)
```

Reading from a file

5.6. Scanner: Example 3

```
 1
 2      import java.util.Scanner;
 3
 4      public class Scanner3 {
 5          public static void main( String[] args ) {
 6              Scanner sc  = new Scanner("1blobblob2blob3").useDelimiter("blob");
 7              System.out.printf("> ");
 8              while ( sc.hasNext() )  {
 9                      String line = sc.next();
10                      System.out.printf("-%s-%n", line );
11                      System.out.printf("> ");
12              }
13              sc.close();
14          }
15      }
16
17
```

Source Code: Src/6_jdk15/Scanner3.java

Execution:

```
% java Scanner3
> -1-
> --
> -2-
> -3-
```

5.7. Scanner: Example 4

```
 1      /*
 2       * example is from: http://www.cs.rit.edu/~hpb/Jdk5/api/java/util/Scanner.html
 3       */
 4      import java.util.Scanner;
 5      import java.util.regex.MatchResult;
 6
 7      public class Scanner4 {
 8          public static void printIt(String input)    {
 9              Scanner sc = new Scanner(input);
10
11              System.out.println("sc.findInLine: " +
```

```
12                      sc.findInLine("(\\d+) fish (\\d+) fish (\\w+) fish (\\w+)"));
13              MatchResult result = sc.match();
14
15              for (int i=1; i<=result.groupCount(); i++) {
16                      System.out.println(i + ": " + result.group(i));
17              }
18          }
19
20          public static void main( String[] args ) {
21              String input = "1 fish 2 fish red fish blue fish";
22              printIt(input);
23
24          }
25      }
26
27
```

Source Code: Src/6_jdk15/Scanner4.java

Execution:

```
% java Scanner4
1: 1
2: 2
3: red
4: blue
```

5.8. Pattern

See https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/regex/Pattern.html (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/regex/Pattern.html)

https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/regex/Matcher.html (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/regex/Matcher.html)

5.9. Pattern - Example

Example:

```
Pattern p = Pattern.compile("a*b");
Matcher m = p.matcher("ab"); // would it match b?
boolean b = m.matches();
```

```
1       import java.util.regex.Pattern;
2       import java.util.regex.Matcher;
3
4       public class P {
5
6           public static void main(String[] args) {
7               String patternDefinition = "a+b*";
8               String text              = "aa";
9               System.out.println(text + " is matched by " + patternDefinition +  ":
10          }
```

```
11      }
```

Source Code: Src/6_jdk15/P.java

An other one:

```
 1      // read: https://docs.oracle.com/javase/tutorial/essential/regex/quant.html
 2      import java.util.regex.Pattern;
 3      import java.util.regex.Matcher;
 4
 5      public class PatternQuestions {
 6
 7          public static void main(String[] args) {
 8      // Find a text to match the pattern
 9              String patternDefinition;
10              String text;
11
12              text              = "aa";
13              patternDefinition = "a+b*";
14              System.out.println(text + " is matched by " + patternDefinition +  ":
15
16              patternDefinition = "x[a-z]+b";
17              System.out.println(text + " is matched by " + patternDefinition +  ":
18
19              patternDefinition = "^a[a-z][b-z]i";
20              System.out.println(text + " is matched by " + patternDefinition +  ":
21
22              patternDefinition = "\\s\\S\\w";
23              System.out.println(text + " is matched by " + patternDefinition +  ":
24
25              patternDefinition = "a*b*c*.";
26              System.out.println(text + " is matched by " + patternDefinition +  ":
27
28              patternDefinition = "[0-9]{3}t[0-9]{2}";
29              System.out.println(text + " is matched by " + patternDefinition +  ":
30
31              patternDefinition = "[0-9]{3}\\.[0-9]{2}";
32              System.out.println(text + " is matched by " + patternDefinition +  ":
33
34          }
35      }
```

Source Code: Src/6_jdk15/PatternQuestions.java

the solution:

```
 1      // read: https://docs.oracle.com/javase/tutorial/essential/regex/quant.html
 2      import java.util.regex.Pattern;
 3      import java.util.regex.Matcher;
 4
 5      public class PatternQuestions_sol {
 6
 7          public static void main(String[] args) {
```

```
 8      // Find a text to match the pattern
 9              String patternDefinition;
10              String text;
11
12              text              = "aa";
13              patternDefinition = "a+b*";
14              System.out.println(text + " is matched by " + patternDefinition +  ":
15
16              text              = "xaab";
17              patternDefinition = "x[a-z]+b";
18              System.out.println(text + " is matched by " + patternDefinition +  ":
19
20              text              = "aabi";
21              patternDefinition = "^a[a-z][b-z]i";
22              System.out.println(text + " is matched by " + patternDefinition +  ":
23
24              text              = " 1w";
25              patternDefinition = "\\s\\S\\w";
26              System.out.println(text + " is matched by " + patternDefinition +  ":
27
28              text              = "a";
29              patternDefinition = "a*b*c*.";
30              System.out.println(text + " is matched by " + patternDefinition +  ":
31
32              text              = "123.12";
33              patternDefinition = "[0-9]{3}t[0-9]{2}";
34              System.out.println(text + " is matched by " + patternDefinition +  ":
35
36              text              = "123.12";
37              patternDefinition = "[0-9]{3}\\.[0-9]{2}";
38              System.out.println(text + " is matched by " + patternDefinition +  ":
39
40          }
41      }
```

Source Code: Src/6_jdk15/PatternQuestions_sol.java

An other one:

```
 1      import java.util.regex.Pattern;
 2      import java.util.regex.Matcher;
 3
 4      public class PatternExample {
 5
 6          public static void main(String[] args) {
 7              long maxCount = 3;
 8              String text[] = { "dog", "Fog", "Dog", "dad", "a mug"  };
 9              String patternDefinition = "([Da]?\\s[um]+g)|[^D]og";
10              Pattern pattern = Pattern.compile(patternDefinition);
11
12                      // Pattern:
13                      //      Max one of the set {D, a}, followed by a white space c
14                      //      followed by one or more of the set {g, u, m}.
15                      //      The \\ are required for one escape
```

```
16                      //              matches
17                      //                      a mug
18                      //  or
19                      //      not a D followed by og
20                      //              matches
21                      //                      Fog, dog, but not Dog
22              for ( int index = 0; index < text.length; index ++ )    {
23                      Matcher aMatcher = pattern.matcher(text[index]);
24                      System.out.println(text[index] + " is matched by " + patternDe
25                              aMatcher.matches() );
26              }
27          }
28      }
```

Source Code: Src/6_jdk15/PatternExample.java

Execution:

```
% java PatternExample
dog is matched by ([Da]?um]+g)|[^D]og: true
Fog is matched by ([Da]?um]+g)|[^D]og: true
Dog is matched by ([Da]?um]+g)|[^D]og: false
dad is matched by ([Da]?um]+g)|[^D]og: false
a mug is matched by ([Da]?um]+g)|[^D]og: true
```

5.10. Scanner: Example 7

```
 1
 2      import java.util.Scanner;       // what is this good for?
 3      import java.io.File;            // what is this good for?
 4
 5      public class Scanner7 {
 6          public static void asIs() {
 7              Scanner sc  = new Scanner(System.in);
 8              System.out.print(": ");
 9              while ( sc.hasNext() )
10                      System.out.print("-" + sc.next() + "+");
11              sc.close();
12              System.out.println();
13          }
14          public static void whiteSpace(String description, String theDelimiter) {
15              Scanner sc  = null;
16              try {
17                      sc  = new Scanner(new File( "words.txt") );
18              } catch ( Exception e ) {}
19              sc.reset();
20              sc.useDelimiter(theDelimiter);  // A whitespace character: [ \t\n\x0B\
21              System.out.println(description);
22              System.out.println("\tdelimiter: " + theDelimiter);
23              while ( sc.hasNext() )
24                      System.out.println("\t-" + sc.next() + "+");
25              sc.close();
26              System.out.println();
```

```
27          }
28          public static void main( String[] args ) {
29              whiteSpace("white space* comma white spice*", "\\s*,\\s*");
30              whiteSpace("white space+ comma white spice*", "\\s+,\\s*");
31              whiteSpace("white space* comma or semicolom white space*", "\\s*(,|;)\
32          }
33      }
34
35
```

Source Code: Src/6_jdk15/Scanner7.java

Execution:

```
% cat words.txt
1, 2,3,    ,5;7
% cat words.txt | od -t a
0000000    1   ,  sp   2   ,   3   ,  sp  sp  sp  sp   ,   5   ;   7  nl
0000020
% java Scanner7 < words.txt
white space* comma white spice*
       delimiter: ,
       -1+
       -2+
       -3+
       -+
       -5;7
+

white space+ comma white spice*
       delimiter:   -1, 2,3,+
       -5;7
+

white space* comma or semicolom white space*
       delimiter: (,|;)
       -1+
       -2+
       -3+
       -+
       -5+
       -7
+
```

6. Class Relationships

See also: http://java.sun.com/docs/books/jls/second_edition/html/classes.doc.html#228205 (http://java.sun.com/docs/books/jls/second_edition/html/classes.doc.html#228205)

- Class declarations define new reference types and describe how they are implemented.
- Constructors are similar to methods, but cannot be invoked directly by a method call; they are used to initialize new class instances.
- Method signatures include return type, and argument types.
- Static initializers are blocks of executable code hat may be used to help initialize a class when it is first loaded.
- The body of a class declares members, static initializers, and constructors.

• Static import feature allows to access classes of a package without package qualification (*Math.PI or PI*).

• The scope of the name of a member is the entire declaration of the class to which the member belongs. Field, method, and constructor declarations may include the access modifiers public, protected, or private. The members of a class include both declared and inherited members.

• Newly declared fields can hide fields declared in a superclass or super interface. Newly declared methods can hide, implement, or override methods declared in a superclass.

- visibility modifier: public/protected/private

Modifier	Class	Package	Subclass	World
public	Y	Y	Y	Y
protected	Y	Y	Y	N
no modifier	Y	Y	N	N
private	Y	N	N	N

- Return type: void/primitive type/reference to a object
- Class methods/Class variables are declared with static.
- Static declaration inside a method change the lifetime of a variable.

6.1. Example

```
 1      import static java.lang.Math.*;
 2      public class  CmToInch {
 3               static final double centimeterToInchMulitplier = 2.54;
 4               static CmToInch aCmToInch = null;      // use before define
 5                                                      // assign object before use
 6               double inCentimeter = 0;
 7               double inInch;                         // = 0;
 8               static double totalCentimeter;
 9               static double totalInch;
10
11
12               public CmToInch()    {
13               }
14               public CmToInch(int inInch)    {
15                      this.inInch = inInch;
16               }
17               public void cm(double soLong){
18                      inCentimeter = soLong;
19                      inInch       = inCentimeter * centimeterToInchMulitplier;
20                      totalCentimeter += inCentimeter;
21                      totalInch += inInch;
22
23               }
24               public String toString(){
25                      return "centimeter/inch = "  + inCentimeter + "/" + inInch;
26               }
27               public static void total(){
28                      System.out.println("    totalCentimeter " + totalCentimeter);
29                      System.out.println("    totalInch "       + totalInch);
30               }
31
32               public static void main(String args[] )        {
33                      System.out.println("pi  = " + PI);      // from where
34                      System.out.println("e   = " + Math.E);
35                      CmToInch aCmToInch = new CmToInch();    // which one
36                      CmToInch aaCmToInch = new CmToInch();
37
38                      aCmToInch.cm(1);
39                      aaCmToInch.cm(10);
40                      System.out.println(aaCmToInch);
41                      total();
42                      new CmToInch().total();
43               }
44      }
```

Source Code: Src/5/CmToInch.java

Execution:

```
% java CmToInch
pi     = 3.141592653589793
e      = 2.718281828459045
centimeter/inch = 10.0/25.4
    totalCentimeter 11.0
    totalInch 27.939999999999998
    totalCentimeter 11.0
    totalInch 27.939999999999998
```

6.2. Class Details

6.3. Static in Classes/Methods Lifetime

• Class Variables: If a variable is declared static, there exists exactly one incarnation of the field,

• Static Methods: A method that is declared static is called a class method. A class method is always invoked without reference to a particular object.

• Non Static Methods: A method that is not declared static is called an instance method, and sometimes called a non-static method. An instance method is always invoked with respect to an object, which becomes the current object to which the keywords this and super refer during execution of the method body.

• Variables can be declared:

static: class variable

— final: can be assigned once or zero time

— transient: not stored or saved via the standard serialization process

— volatile: A variable may be modified by multiple threads. This gives a hint to the compiler to fetch the value each time, rather store a locale copy. This also prohibits same optimization procedures.

• See also: http://java.sun.com/docs/books/jls/second_edition/html/classes.doc.html#78119 (http://java.sun.com/docs/books/jls/second_edition/html/classes.doc.html#78119)

```
 1
 2      public class  Overview  {
 3        int                   instanceVariable;
 4        static    int         classVariable;
 5        final     int         finalVariable; // static?
 6        volatile  int         volatileVariable;
 7        transient int         transientVariable;
 8
 9
10        public Overview()     {
11              finalVariable = 42;
12        }
13        public Overview(int aLocalVariable)   {
14              finalVariable = 43;
15        }
16        void instanceMethod() {
17              finalVariable = 43;
18              instanceVariable = 22;
19              classVariable = 33;
20        }
```

```
21        static void classMethod()     {
22              classVariable = 3;
23        }
24
25        public static void main(String args[] )       {
26              Overview aOverview = new Overview();
27              Overview bOverview = new Overview();
28              Overview cOverview = new Overview(1);
29              cOverview = bOverview;
30              aOverview.instanceMethod();
31              instanceMethod();
32              bOverview.classMethod();
33              // values of aOverview.? bOverview.?
34              // aOverview.finalVariable??
35
36        }
37      }
```

Source Code: Src/5/Overview.java

- how many instances of the variables to exist?
- How many objects do exist?

6.4. Parameter Passing

- The formal parameters of a method, if any, are specified by a list of comma-separated parameter specifiers.
- Each parameter specifier consists of a type and an identifier (optionally followed by brackets) that specifies the name of the parameter.
- If a method has no parameters, only an empty pair of parentheses appears in the method's declaration.
- If two formal parameters are declared to have the same name (that is, their declarations mention the same identifier), then a compile-time error occurs.
- When the method is invoked, the values of the actual argument expressions initialize newly created parameter variables, each of the declared type, before execution of the body of the method.
- The scope of formal parameter names is the entire body of the method. These parameter names may not be redeclared as local variables or exception parameters within the method; that is, hiding the name of a parameter is not permitted.
- call by value

6.5. Example I

```
 1      public class  ExampleClass      {
 2        static int aLocalVariable = 3;
 3
 4        public ExampleClass() {
 5              aLocalVariable = 2;
 6        }
 7        public ExampleClass(int aLocalVariable)       {
 8              this.aLocalVariable = aLocalVariable;
 9              aLocalVariable = 6;     // not the instance variable
10        }
11
12        public static void main(String args[] )       {
13              ExampleClass aExampleClass = new ExampleClass();
14              ExampleClass aExampleClass2 = new ExampleClass(1);
15              ExampleClass aExampleClass3 = new ExampleClass(2);
16              int x = 3;
17              aExampleClass = new ExampleClass(x);
18              System.out.println(x);
19              System.out.println("the value is: " + aExampleClass.aLocalVariable);
20              // System.out.println(aLocalVariable);
21        }
22      }
```

Source Code: Src/5/ExampleClass.java

Questions:

• How does the the JVM find the main method, when you execute *java ExampleClass*?

• Describe the execution order of the constructors.

• Which variables will be modified when?

6.6. Example II

```
 1      import java.util.Vector;
 2
 3      public class  Args {
 4
 5              int    anInt   = -1;
 6              String aString = "a";
 7              int[]  anArray = { 4, 2 };
 8
 9              public void testString(String arg )      {
10                      arg = "b";
11              }
12              public void testArray(int[] arg )      {
13                      arg[1] = 3;
14              }
15              public void testInt(int arg )      {
16                      arg = 42;
17              }
18              public void testString()        {
```

```
19                      System.out.println("1.  " + aString);
20                      testString(aString);
21                      System.out.println("2.  " + aString);
22              }
23              public void testArray() {
24                      System.out.println("3.  " + anArray[0] + ", " + anArray[1]);
25                      testArray(anArray);
26                      System.out.println("4.  " + anArray[0] + ", " + anArray[1]);
27              }
28              public void testInt()   {
29                      System.out.println("5.  " + anInt );
30                      testInt(anInt);
31                      System.out.println("6.  " + anInt );
32              }
33              public static void main(String args[] )       {
34                      new Args().testString();
35                      new Args().testArray();
36                      new Args().testInt();
37              }
38      }
```

Source Code: Src/5/Args.java

Execution:

```
% java Args
1.     a
2.     a
3.     4, 2
4.     4, 3
5.     -1
6.     -1
```

An example with our own class:

```
 1      public class  Car {
 2               int speed = 0;
 3               public void setSpeed(int speed){
 4                       this.speed = speed;
 5               }
 6               public String toString(){
 7                       return "speed = "  + speed;
 8               }
 9               static public void changeStateOf(Car thisCar ) {
10                      thisCar.setSpeed(42);
11              }
12              public static void main(String args[] )        {
13                      Car aCar = new Car();
14                      System.out.println("1:  " + aCar);
15                      changeStateOf(aCar);
16                      System.out.println("2:  " + aCar);
17              }
18      }
```

Source Code: Src/5/Car.java

6.7. Example III

```
 1      public class  ExampleClassIII   {
 2
 3        String aString = null;
 4
 5        public void method(String a)  {
 6              a = new String("set in method");
 7              System.out.println("2. method:a:" + a );
 8        }
 9        public void test()    {
10              String aString = new String("set in test");
11
12              System.out.println("1. test:aString:" + aString );
13              method(aString);
14              System.out.println("3. test:aString:" + aString );
15        }
16        public static void main(String args[] )       {
17              new ExampleClassIII().test();
18        }
19      }
```

Source Code: Src/5/ExampleClassIII.java

6.8. Example IV

```
 1      public class  AnOtherExample    {
 2        int instanceV = 1;
 3        static AnOtherExample staticAnOtherExample;
 4        AnOtherExample instanceAnOtherExample;
 5
 6        public AnOtherExample()       {
 7              this.instanceAnOtherExample = this;
 8        }
 9        public void create() {
10              AnOtherExample aAnOtherExample;
11              for ( int index = 0; index < 10; index ++ )     {
12                      aAnOtherExample = new AnOtherExample();
13              }
14        }
15        public String toString() {
16              return "this/instanceAnOtherExample " + this + "/" + instanceAnOtherEx
17              // return "" + instanceV;
18        }
19
20        public static void main(String args[] )       {
21              staticAnOtherExample = new AnOtherExample();
22              System.out.println(staticAnOtherExample);
23              staticAnOtherExample.create();
24        }
25      }
```

Source Code: Src/5/AnOtherExample.java

• does it compile? ja

• How many different instanceV exist? maximal 2

• What will happen if we comment the return/toString in? Das Program wird abstuerzem weil toString() rekursiv immer wieder aufgerufen wird

Example VI

```
 1
 2      public class  NotCorrect        {
 3
 4        static int counter;
 5        NotCorrect aNotCorrect =  new NotCorrect(); // hint
 6
 7        public NotCorrect() {
 8               System.out.println("so Many Calls: " + counter++ );
 9        }
10        public static void main(String args[] )       {
11              NotCorrect aNotCorrect = new NotCorrect();
12        }
13      }
```

Source Code: Src/5/NotCorrect.java

• does it compile? Das ist der Momnet wo Kenntinse der deutschen Sprache helpfen wuerden. Aver ich kann auch schwaebisch. Des programmle witd sich wohle uebersetza lassa.
Draw the memory pic during execution

• Does it execute? Joa, aber esch wird not ueberleba , das objektlw word jedesmoal nei gmoachet.
Output?

6.9. A Point Class

Use of a *Point* Class:

```
 1      /**
 2       * This class implements a point test program.
 3       *
 4       * @version   $Id$
 5       *
 6       * @author    hp bischof
 7       *
 8       * Revisions:
 9       *      $Log$
10       */
11
12
13
14      public class TestPoint {
15        private static Point aPoint;
16
17
18      /**
```

```
19        * The main program.
20        *
21        * @param    args    command line arguments (ignored)
22        */
23        public static void main(String args[])
24        {
25              System.out.println("Point.soManyPoints = " + Point.soManyPoints );
26              aPoint = new Point(2, 3);
27              System.out.println("x = " + aPoint.getX() );
28              System.out.println("y = " + aPoint.getY() );
29
30              aPoint = new Point();
31              aPoint.initPoint(4, 5);
32              System.out.println("x = " + aPoint.getX() );
33              System.out.println("y = " + aPoint.getY() );
34
35              aPoint.move(6, 7);
36              System.out.println("x = " + aPoint.getX() );
37              System.out.println("y = " + aPoint.getY() );
38
39              System.out.println("aPoint.soManyPoints = " + aPoint.soManyPoints );
40              System.out.println("Point.soManyPoints  = " + Point.soManyPoints );
41        }
42      }
43
44
45
46
```

Source Code: Src/5/TestPoint.java

The *Point* Class:

```
1       /**
2        * This class implements a point in a two dimensional
3        * area.
4        * All method print when they are called.
5        *
6        * @version   $Id$
7        *
8        * @author    hp bischof
9        *
10       * Revisions:
11       *      $Log$
12       */
13
14      public class Point {
15                                      // class variable
16        static int soManyPoints;      // so many points were created.
17
18        private int x;                // x coordinate of the point
19        private int y;                // y cooridnate of the point
20
21      /**
```

```
22       * Default Constructor.
23       * Increases the counter soManyPoints by 1.
24       *
25       * @return            Point a Point object
26       */
27        public Point(){
28              super();
29              System.out.println("    in Point() constructor");
30              soManyPoints ++;
31        }
32
33      /**
34       * Constructor.
35       * initialize x and y values of a point.
36       *
37       * @param       x       x coordinate
38       * @param       y       y coordinate
39       *
40       * @return            Point a Point object
41       */
42        public Point(int x, int y){
43              super();
44              this.x = x;
45              this.y = y;
46              System.out.println("    in Point(int, int) constructor");
47        }
48
49      /**
50       * So many points have been created.
51       *
52       * @return int So many points have been created
53       */
54        public static int soManyPoints(){
55              return soManyPoints;
56        }
57
58      /**
59       * initialzes x and y of a point.
60       *
61       * @param       x       int x coordinate
62       * @param       y       int y coordinate
63       *
64       * @return            Point a Point object
65       */
66        public Point initPoint(int x, int y){
67              System.out.println("    in initPoint(int, int)");
68
69              this.x = x;
70              this.y = y;
71
72              return this;
73        }
74
75      /**
```

```
76       * move a point
77       *
78       * @param       x       int delta x value
79       * @param       y       int delta y value
80       *
81       * @return            Point a Point object
82       */
83        public Point move(int x, int y){
84              System.out.println("    in move(int, int)");
85
86              this.x += x;
87              this.y += y;
88
89              return this;
90        }
91
92      /**
93       * Returns the x coordinate of a point
94       *
95       * @return            int x value
96       */
97        public int getX(){
98              System.out.println("    in getX()");
99              return this.x;
00        }
01
02      /**
03       * Returns the y coordinate of a point
04       *
05       * @return            int x value
06       */
07        public int getY(){
08              System.out.println("    in getY()");
09              return this.y;
10        }
11      }
```

Source Code: Src/5/Point.java

II. Execution of the test program:

```
Point.soManyPoints = 0
        in Point() constructor
        in Point(int, int) constructor
        in getX()
x = 2
        in getY()
y = 3
        in Point() constructor
        in initPoint(int, int)
        in getX()
x = 4
        in getY()
y = 5
```

```
        in move(int, int)
        in getX()
x = 10
        in getY()
y = 12
        in getNPoints()
nPoints = 2
aPoint.soManyPoints = 2
```

You may find the javadoc pages here. (Src/5/javadoc/index.html)

6.10. Additional Examples

See http://docs.oracle.com/javase/specs/jls/se7/html/index.html (http://docs.oracle.com/javase/specs/jls/se7/html/index.html)

```
 1      public class  Scope_1   {
 2
 3        String aString = null;
 4
 5        public void method(String aString)    {
 6              this.aString = aString;
 7        }
 8        public Scope_1 test() {
 9              String aString = new String("set in test");
10
11              method(aString);
12              return this;
13        }
14        public static void main(String args[] )       {
15              System.out.println((new Scope_1().test()).aString);
16
17        }
18      }
```

Source Code: Src/5/Scope_1.java

Question arrived 22:44

• Some confusions regarding scopes.

```
void func(){
    int index=2;
    for (int index=0; index<5; index++) {//some code}
}
```

It doesn't compile saying that the variable "index" is already defined.

```
void func1(){
    int index;
    for(int index=0; index<5; index++){//some code}

    System.out.println(index); // Does not compile saying that the variable "index" ca
}
```

• How does the auto-boxing actually proceeds in the following program from this link:-

```
 1      public class SimpleBoxingTypes {
 2          public static void main( String[] args ) {
 3              Float float42Object     = Float.valueOf(42F);
 4              Integer integer42Object = Integer.valueOf(42);
 5              Double double42Object   = Double.valueOf(42.0);
 6
 7              double floatObject  = float42Object + (double)integer42Object + (doubl
 8              float ff  = floatObject + (float)integer42Object + ((float)double42Obj
 9
```

```
10                System.out.println(floatObject);
11            }
12        }
13
14
```

Source Code: Src/6_jdk15/SimpleBoxingTypes.java

• http://www.cs.rit.edu/~hpb/Lectures/20121/Java_707/all-6.15.html (http://www.cs.rit.edu/~hpb/Lectures/20121/Java_707/all-6.15.html)

Boxing - primitive to Object type
Unboxing - Object to primitive type
What is the concept I am missing in this example? I see only type casting.
Secondly, why does the commented float line doesn't compile? Why does it start compiling when I use the primitive types instead of object types? Why do the boxing and unboxing things don't help? Does this demonstrate some type of a difference between the two types?

• I see a break statement at the end of the default case everywhere? Does anything significantly change if I don't provide it.

7. Inheritance

See also here. (http://java.sun.com/docs/books/jls/html/8.doc.html#30229)
Java has simple inheritance, i.e., a class can only extend one superclass.

Class Members

The members of a class type are all of the following:

- Members inherited from its direct superclass, except in class Object, which has no direct superclass
- Members inherited from any direct super interfaces
- Members declared in the body of the class.
- Is a relationship

Members of a class that are declared private are not inherited by subclasses of that class. Only members of a class that are declared protected or public are inherited by subclasses declared in a package other than the one in which the class is declared.

Constructors and static initializers are not members and therefore are not inherited.

7.1. Syntax

The Java language specification can be found here: https://docs.oracle.com/javase/specs/jls/se14/html/jls-8.html#jls-8.8 (https://docs.oracle.com/javase/specs/jls/se14/html/jls-8.html#jls-8.8)

```
class subClassName [ extends superClassName ]
{
  ...
}
```

Super class is object, if *extends superClassName* is missing.

See https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/lang/Object.html (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/lang/Object.html)

7.2. Constructor Sequence

```
 1      public class B {
 2
 3        int aBint;
 4
 5        public B()    {
 6              System.out.println("public B()" );
 7        }
 8        public B(int aBint)   {
 9              this.aBint = aBint;
10              System.out.println("public B(int aBint)" );
11        }
12        public String toString()      {
13              return ""  + getClass();
14        }
15        public static void main(String args[])        {
16              System.out.println("1: " + new B());
17              System.out.println("2: " + new B(42));
18        }
19      }
20
```

Source Code: Src/6/B.java

Next

```
 1      class BB extends B {
 2
 3        int aBBint;
 4        public int x;
 5
 6        public BB()   {
 7              System.out.println("public BB()" );
 8        }
 9        public BB(int aBBint) {
10              this.aBBint = aBBint;
11              System.out.println("public BB(int x)" );
12        }
13        public String toString()      {
14              return "" + getClass();
15        }
16        public static void main(String args[])        {
17              System.out.println("1: " + new BB());
18              System.out.println("2: " + new BB(42));
19        }
20      }
```

Source Code: Src/6/BB.java

Next

```
% java BB
public B()
public BB()
1: class BB
public B()
public BB(int x)
2: class BB
```

Example:

The Flock class:

```
 1      public class Flock {
 2
 3        private static int soManyBirds = 0;
 4        static final int   maxNumberOfBirds = 3;
 5
 6        public Flock()        {
 7              soManyBirds ++;
 8        }
 9        public int soManyBirds()      {
10              return soManyBirds;
11        }
12        public String toString()      {
13              return ""  + soManyBirds;
14        }
15      }
```

Source Code: Src/6/Flock.java

The pigeon class

```
 1      public class Pigeon extends Flock {
 2
 3        private String name;
 4
 5        public Pigeon()       {
 6        }
 7        public Pigeon(String name)    {
 8              this.name = name;
 9        }
10        private static void test()    {
11              int soManyBirdsCreated = 0;
12              String name = "a";
13              Pigeon lastOne = null;  // why?
14              while (  soManyBirdsCreated++ < maxNumberOfBirds )      {
15                      lastOne = new Pigeon(name += "a");
16              }
17              System.out.println("so many are in the flock: " + lastOne.soManyBirds(
18              System.out.println("last name used = " + name );
19        }
20        public static void main(String[] args )       {
21              test();
22        }
```

```
23      }
```

Source Code: Src/6/Pigeon.java

class C extends class B extends class A

A picture:

7.3. How to get access to super class methods/variables?

• Super class:

```
 1      public class S {
 2
 3        public int intS;              // what is the value of intS?
 4
 5        public S ()   {
 6              System.out.println("S()");
 7        }
 8        public S method(int x)        {
 9              intS = x;
10              System.out.println("S method(int x)");
11              return this;
12        }
13        public String toString()      {
14              return "S: " + intS;
15        }
16        public static void main(String args[])        {
17              System.out.println("new S()     " +  new S());
18        }
19      }
```

Source Code: Src/6/S.java

• Sub class:

```
 1      public class SubclassOfS extends S {
 2
 3        public int intS;
 4
 5        public SubclassOfS()  {
 6              System.out.println("SubclassOfS ()");
 7        }
 8
 9        public S method(int x)        {
10              intS = x;
11                      System.out.println("S method(int x)");
12              super.method(9);
13                      System.out.println("4. super: " + super.toString() );
14              super.intS = 4;
15                      System.out.println("5. super: " + super.toString() );
16              return this;
17        }
18        public String toString()      {
```

```
19              return "SSubclassOfS: " + intS;
20        }
21
22
23        public static void main(String args[])        {
24              SubclassOfS aSubclassOfS = new SubclassOfS();
25              S  aS =  aSubclassOfS.method(42);
26              System.out.println(aS);
27              System.out.println(aSubclassOfS);
28              System.out.println("1. SubclassOfS!intS      = " + aSubclassOfS.intS);
29              System.out.println("2. ((S)SubclassOfS)!intS = " + ((S)aSubclassOfS).i
30      //      method(3);              // <--- what is the problem here ...
31
32        }
33      }
```

Source Code: Src/6/SubclassOfS.java

Execution:

```
% java SubclassOfS
S()
SubclassOfS ()
S method(int x)
S method(int x)
4. super: S: 9
5. super: S: 4
SSubclassOfS: 42
SSubclassOfS: 42
1. SubclassOfS!intS      = 42
2. ((S)SubclassOfS)!intS = 4
```

7.4. Default Constructor

• Constructors create objects from the class blueprint

• The default constructor has no arguments

• Constructors can invoke, as the first statement, other constructors

• Constructor declarations look like method declarations, except that they use the name of the class and have no return type.

• The compiler will provide a default constructor if none is defined. The compiler will then verify that the super class does have provided default constructor

• The compiler will give an error, if a parameterized constructor is defined, but no default constructor is defined, and you use the provided constructor. This makes sense because you never wanted to allow to create an 'empty' object. See example.

```
1       public class DefaultConstructor {
2
3         final int MINIMUM = 4;
4         int thisNumbershouldAlwyasBeInitializedWithAvalueGreaterThan4;
5       /*
6         DefaultConstructor()  {
7         }
8       */
```

```
 9        DefaultConstructor(int aInt)  {
10              aInt = ( aInt <= MINIMUM ) ? MINIMUM : aInt;
11              thisNumbershouldAlwyasBeInitializedWithAvalueGreaterThan4 = aInt;
12        }
13        public static void main(String args[]) {
14              // DefaultConstructor aDefaultConstructor = new DefaultConstructor();
15              DefaultConstructor aDefaultConstructor = new DefaultConstructor(3);
16              System.out.println(aDefaultConstructor.thisNumbershouldAlwyasBeInitial
17        }
18      }
19      class SubClass extends DefaultConstructor {
20        SubClass()    {
21              System.out.println(" SubClass()" );
22        }
23        public static void main(String args[]) {
24              SubClass aSubClass = new SubClass();
25        }
26      }
```

Source Code: Src/6/DefaultConstructor.java

An example - a player can only be created with a name:

```
 1      public class Player {
 2
 3        private String name;
 4
 5        Player(String name)   {
 6              this.name = name;
 7        }
 8        public String getName()    {
 9              return name;
10        }
11        public static void main(String args[]) {
12              Player aPlayer = new Player("John");
13              // Player bPlayer = new Player();
14              System.out.println("Name of the PLayer is: " + aPlayer.getName() );
15        }
16      }
17
```

Source Code: Src/6/Player.java

and a game can only be played with player who have a name:

```
 1      public class Game {
 2
 3        private String name;
 4
 5        Game(String name)     {
 6              this.name = name;
 7        }
 8        public String nameOfTheGame()    {
 9              return name;
```

```
10        }
11        public static void main(String args[]) {
12              Game aGame = new Game("BackGammon");
13              Player playerOne = new Player("Rose");
14              Player playerTwo = new Player("John");
15
16              System.out.println(playerOne.getName() + " and " + playerTwo.getName()
17                                  " are playing a game of " + aGame.nameOfTheGame() +
18        }
19      }
20
```

Source Code: Src/6/Game.java

7.5. Private, Protected and Final I

Access control can be specified in a class, interface, method, or field declaration to control when access to a member is allowed. Access is a different concept from scope; access specifies the part of the Java program text within which the declared entity can be referred to by a qualified name, a field access expression, or a method invocation expression in which the method is not specified by a simple name. The default access is that a member can be accessed anywhere within the package that contains its declaration; other possibilities are public, protected, and private.

7.6. Determining Accessibility

Whether a package is accessible is determined by the host system. If a class or interface type is declared public, then it may be accessed by any Java code that can access the package in which it is declared. If a class or interface type is not declared public, then it may be accessed only from within the package in which it is declared.

A member type or a constructor of a class type is accessible only if the type is accessible and the member or constructor is declared to permit access:

- If the member or constructor is declared public, then access is permitted.
- Protected methods and variables are accessible to subclasses and provide a way of opening up the encapsulation.
- Private methods and variables are not accessible to subclasses.
- A final class can not be sub classed.

7.7. Packages

• A package is a set of related classes

• Classes in the same package can access each other's protected members.

• This makes it easier to avaid name and to controll access

• How to create:

```
package name;
```

Copied from: https://www.javatpoint.com/package (https://www.javatpoint.com/package)

Example:

Package:

```
 1      package myPackage;
 2
 3      public class MyHashCode {
 4        private String aString;
 5
 6        public MyHashCode(String aString)     {
 7              this.aString = aString;
 8        }
 9        public int hashCode() {
10              return aString.length();
11        }
12      }
```

Source Code: Src/6/MyHashCode.java

Use:

```
 1      import myPackage.MyHashCode;
 2
 3      public class UseOfMyHashCode    {
 4
 5              public static void main(String args[]) {
 6                      MyHashCode aMyHashCode = new MyHashCode("abc");
 7                      System.out.println("a. " +  aMyHashCode.hashCode() );
 8                      System.out.println("b. " +  (new MyHashCode("a")).hashCode() )
 9              }
10      }
```

Source Code: Src/6/UseOfMyHashCode.java

Compulation and execution:

```
% javac -d . MyHashCode.java  # creates myPackage
% ls myPackage
% MyHashCode.class
% javac UseOfMyHashCode.java
% java UseOfMyHashCode
a. 3
b. 1
```

7.8. Polymorphism

A subclass can override an inherited method by providing a new, identical method declaration.

```
 1      public class Vehicle {
 2
 3        String honkSound = "vehicle honk";
 4        int wheels       = 4;
 5
 6        public void setSound (String honkSound)    {
 7              this.honkSound = honkSound;
 8        }
 9        public String toString ()    {
10              return "v";
11        }
12        public void setSoManyWheels (int wheels)    {
13              this.wheels = wheels;
14        }
15        public int soManyWheels ()    {
16              return wheels;
17        }
18        public void honk()    {
19              System.out.println(honkSound);
20        }
21      }
22
```

Source Code: Src/6/Vehicle.java

```
 1      public class Train extends Vehicle {
 2
 3        String honkSound = "choo-choo";
 4        int wheels       = 32;
 5
 6        public void setSound (String honkSound)    {
 7              this.honkSound = honkSound;
 8        }
 9        public int soManyWheels ()    {
10              return wheels;
11        }
12        public void onlyAtrainCanDoThis()     {
13        }
14        public void honk()    {
15              System.out.println(honkSound);
16        }
17        public void de()      {
18              System.out.println("de");
19        }
20        public static void main(String[] args )       {
21              new Vehicle().honk();
22              new Train().honk();
23              new Train().de();
24        }
25      }
26
```

Source Code: Src/6/Train.java

Train example:

```
 1      public class TrainEx {
 2
 3        String honkSound = "choo-choo";
 4
 5        public void honk()    {
 6              System.out.println(honkSound);
 7        }
 8        public static void main(String[] args )       {
 9              new Vehicle().honk();
10              new Train().honk();
11
12              Train   aTrain   = new Train();
13              Vehicle aVehicle = aTrain; // new Vehicle();
14              Vehicle aVehicleOtherReference = aVehicle;
15                      System.out.println(aVehicle + " " +aTrain);
16                      System.out.println(aVehicle + " " +aVehicle);
17              aVehicle.honk();
18              aTrain.honk();
19              System.out.println("--------------------");
20              aVehicle.setSound("ringADing");
21              aVehicle.honk();
22              aTrain.honk();
```

```
23              System.out.println("aVehicle.honkSound: " + aVehicle.honkSound);
24              System.out.println("aTrain.honkSound: " + aTrain.honkSound);
25
26        }
27      }
28
```

Source Code: Src/6/TrainEx.java

Output:

```
% java Train
vehicle honk
choo-choo
Vehicle@36baf30c    Train@7a81197d
Vehicle@36baf30c    Vehicle@36baf30c
vehicle honk
choo-choo
--------------------
ringADing
Tuk-Tuk
```

7.9. Inner Classes

Since Java version 1.1 inner classes are possible. Inner classes cannot be declared as native, synchronized, transient or volatile.

Example:

```
 1      /**
 2       * This class implements a inner class.
 3       *
 4       * @version   $Id$
 5       *
 6       * @author    Axel T. Schreiner
 7       * @author    hp bischof
 8       *
 9       * Revisions:
10       *      $Log$
11       */
12      class InnerClass {
13          static class A {
14              static void hi () {
15                      System.err.println("4.hi");
16              }
17          }
18
19          class B {
20              void hi () {
21                      System.err.println("3.hi");
22              }
23          }
24
25          void hi () {
```

```
26                   class C {
27                      void hi () {
28                              System.err.println("2.hi");
29                      }
30                   }
31                   Object o = new C() {
32                      void hi () {
33                              System.err.println("1.hi");
34                      }
35                   };
36                   ((C)o).hi(); new C().hi(); new B().hi();
37           }
38           static public void main (String args []) {
39                    new InnerClass().hi();
40                    A.hi();
41           }
42       }
```

Source Code: Src/5/InnerClass.java

Result:

```
% java InnerClass
D.hi
C.hi
B.hi
A.hi
```

Hi.A is a nested top-level class and could be an interface. A has the same access to things as other classes in a package; more importantly, A can reach all static things in Hi.

Hi.B is a member class and cannot be an interface or contain static methods such as newInstance(). An instance of B can access all members of that instance of Hi which created it; therefore, an instance of B cannot be created in a class method of Hi. If necessary, the prefix Hi.this is used to access a member of Hi from B.

Hi.C is a local class and cannot be an interface. Methods of Hi.C can access all final local variables and parameters of it's surrounding method or block and additionally all members (or static members, depending on context) of Hi.

Hi.o is an object of an anonymous class and has the same access as an object of a local class. The class has no names and consequently no constructor; it can be subclassed from an interface.

All inner classes can have their own instance variables.

We will discuss inner classes more at the end of chapter 6.

7.10. Class Cast - 0

Vehicle:

```
 1      public class Vehicle {
 2
 3        String honkSound = "vehicle honk";
 4        int wheels       = 4;
 5
 6        public void setSound (String honkSound)    {
 7              this.honkSound = honkSound;
 8        }
 9        public String toString ()    {
10              return "v";
11        }
12        public void setSoManyWheels (int wheels)    {
13              this.wheels = wheels;
14        }
15        public int soManyWheels ()    {
16              return wheels;
17        }
18        public void honk()    {
19              System.out.println(honkSound);
20        }
21      }
22
```

Source Code: Src/6/Vehicle.java

Car:

```
 1
 2      public class Car extends Vehicle {
 3
 4        String honkSound = "honk";
 5        int wheels       = 4;
 6
 7        public void setSound (String honkSound)    {
 8              this.honkSound = honkSound;
 9        }
10        public int soManyWheels ()    {
11              return wheels;
12        }
13        public void onlyACarCanDoThis()       {
14              System.out.println("onlyACarCanDoThis");
15        }
16        public void honk()    {
17              System.out.println(honkSound);
18        }
19        public static void main(String[] args )       {
20              new Vehicle().honk();
21              new Train().honk();
22        }
23      }
24
```

Source Code: Src/6/Car.java

PoliceCar:

```
 1
 2      public class PoliceCar extends Car {  // Car extends Vehicle
 3
 4        String honkSound = "tatue tata";
 5        int wheels       = 4;
 6
 7        public void setSound (String honkSound)    {
 8              this.honkSound = honkSound;
 9        }
10        public int soManyWheels ()    {
11              return wheels;
12        }
13        public void onlyACarPoliceCanDoThis() {
14              System.out.println("onlyACarPoliceCanDoThis");
15        }
16        public void honk()    {
17              System.out.println(honkSound);
18        }
19        public static void main(String[] args )       {
20
21              PoliceCar aPoliceCar = new PoliceCar();
22              Car aCar = aPoliceCar;
```

```
23              Vehicle aVehicle = aCar;
24              aCar.honk();
25              aPoliceCar.onlyACarPoliceCanDoThis();
26              aCar.onlyACarCanDoThis();
27              // aVehicle.onlyACarCanDoThis();
28        }
29      }
30
```

Source Code: Src/6/PoliceCar.java

```
% java
tatue tata
onlyACarPoliceCanDoThis
onlyACarCanDoThis
```

7.11. Class Cast - 1

Vehicle:

```
 1      public class Vehicle {
 2
 3        String honkSound = "vehicle honk";
 4        int wheels       = 4;
 5
 6        public void setSound (String honkSound)    {
 7              this.honkSound = honkSound;
 8        }
 9        public String toString ()    {
10              return "v";
11        }
12        public void setSoManyWheels (int wheels)    {
13              this.wheels = wheels;
14        }
15        public int soManyWheels ()    {
16              return wheels;
17        }
18        public void honk()    {
19              System.out.println(honkSound);
20        }
21      }
22
```

Source Code: Src/6/Vehicle.java

Train:

```
 1      public class Train extends Vehicle {
 2
 3        String honkSound = "choo-choo";
 4        int wheels       = 32;
 5
```

```
 6        public void setSound (String honkSound)    {
 7              this.honkSound = honkSound;
 8        }
 9        public int soManyWheels ()    {
10              return wheels;
11        }
12        public void onlyAtrainCanDoThis()     {
13        }
14        public void honk()    {
15              System.out.println(honkSound);
16        }
17        public void de()      {
18              System.out.println("de");
19        }
20        public static void main(String[] args )       {
21              new Vehicle().honk();
22              new Train().honk();
23              new Train().de();
24        }
25      }
26
```

Source Code: Src/6/Train.java

Casting:

```
 1      public class Casting {
 2
 3        String honkSound = "choo-choo";
 4
 5        public void honk()    {
 6              System.out.println(honkSound);
 7        }
 8        public static void main(String[] args )       {
 9              new Vehicle().honk();
10              new Train().honk();
11
12              Train   aTrain   = new Train();
13              Vehicle aVehicle = (Train)aTrain;
14              Vehicle aVehicleOtherReference = aVehicle;
15                      System.out.println(aTrain + "   " +aTrain);
16                      System.out.println(aVehicle + " " +aVehicle);
17              aVehicle.honk();
18              System.out.println("aVehicle.soManyWheels() = " + aVehicle.soManyWheel
19              aVehicle.honk();
20              aVehicle.setSound("ring ring");
21              aTrain.setSound("ring ring");
22              aVehicle.honk();
23              aTrain.honk();
24
25              // aVehicle.onlyAtrainCanDoThis();
26              aVehicle.wheels = 0;
27                      System.out.println("aVehicle.soManyWheels() = " + aVehicle.soM
28                      System.out.println("aTrain.soManyWheels() = " + aTrain.soManyW
```

```
29                      System.out.println("aVehicleOtherReference.soManyWheels() = "
30                      System.out.println("aVehicle.wheels = " + aVehicle.wheels);
31        }
32      }
```

Source Code: Src/6/Casting.java

Output:

```
vehicle honk
choo-choo
Train@36baf30c      Train@36baf30c
Train@36baf30c      Train@36baf30c
choo-choo
aVehicle.soManyWheels() = 32
choo-choo
ring ring
ring ring
aVehicle.soManyWheels() = 32
aTrain.soManyWheels() = 32
aVehicleOtherReference.soManyWheels() = 32
aVehicle.wheels = 0
```

7.12. Class Cast - II

```
1       public class S5 {
2
3         public int inS6andS5 = 1;
4         public int inS5Only = 1;
5
6         public String toString()      {
7               return "S5: " + "inS6andS5 =  " + inS6andS5 + " inS5Only =  " + inS5On
8         }
9         public void both()    {
10              inS6andS5 = 11;
11        }
12        public static void main(String args[]) {
13              System.out.println(new S5());
14        }
15      }
16
```

Source Code: Src/6/S5.java

Next

```
1       public class S6 extends S5  {
2
3         public int inS6andS5 = 2;
4
5         public void both()    {
```

```
 6              inS6andS5 = 22;
 7        }
 8        public String toStringSuper()      {
 9              return super.toString();
10        }
11        public String toString()      {
12              return "S6: " + "inS6andS5 =  " + inS6andS5 + " inS5Only =  " + inS5On
13        }
14        public int superA()      {
15                return super.inS6andS5;
16        }
17
18        public static void main(String args[]) {
19              S6 aS6 = new S6();
20              S5 aS5 = (S5)aS6;
21
22              aS6.both();
23                      System.out.println("1. aS6 = " + aS6 );
24                      System.out.println("1. aS5 = " + aS5 + "\n" );
25              aS5.both();
26                      System.out.println("2. aS6 = " + aS6 );
27                      System.out.println("2. aS5 = " + aS5 );
28                      System.out.println("2. aS6.toStringSuper() = " + aS6.toStringS
29
30              aS6.inS6andS5 = 3;
31              System.out.println("aS6.inS6andS5 = 3;");
32                      System.out.println("3. aS6 = " + aS6 );
33                      System.out.println("3. aS5 = " + aS5 + "\n" );
34
35              aS6.inS5Only = 4;
36              System.out.println("aS6.inS5Only = 4;");
37              System.out.println("4. aS6 " + aS6);
38              System.out.println("4. aS5 " + aS5 + "\n");
39
40              aS5.inS5Only = 5;
41              System.out.println("aS5.inS5Only = 5;");
42              System.out.println("5. aS6 " + aS6);
43              System.out.println("5. aS5 " + aS5);
44
45        }
46      }
47
```

Source Code: Src/6/S6.java

Output of:

```
1. aS6 = S6: inS6andS5 =  22 inS5Only =  1
1. aS5 = S6: inS6andS5 =  22 inS5Only =  1

2. aS6 = S6: inS6andS5 =  22 inS5Only =  1
2. aS5 = S6: inS6andS5 =  22 inS5Only =  1
2. aS6.toStringSuper() = S5: inS6andS5 =  1 inS5Only =  1

aS6.inS6andS5 = 3;
3. aS6 = S6: inS6andS5 =  3 inS5Only =  1
```

```
3. aS5 = S6: inS6andS5 =  3 inS5Only =  1

aS6.inS5Only = 4;
4. aS6 S6: inS6andS5 =  3 inS5Only =  4
4. aS5 S6: inS6andS5 =  3 inS5Only =  4

aS5.inS5Only = 5;
5. aS6 S6: inS6andS5 =  3 inS5Only =  5
5. aS5 S6: inS6andS5 =  3 inS5Only =  55. aS5.testIt 5
```

7.13. Abstract Classes

An abstract class

• specifies a public method interface which can be inherited by direct or indirect subclasses.

• may declare methods, but not implement them.

• can not be instantiated.

Classes who extend an abstract class share the same, possibly extended, interface.

• Is a relationship

Storing objects in a container and it is guaranteed that each object can execute a particular method.

```
Area  allTwoDThings[] = new Area [MAXIMUM];
```

```
 1
 2
 3      public class TestAbstract_2 {
 4
 5        static final int MAXIMUM = 4;
 6
 7        public static void main(String args[])
 8        {
 9              Area  allTwoDThings[] = new Area [MAXIMUM];
10              for ( int i = 0; i < MAXIMUM; i++ )     {
11                      if ( i % 2 == 0 )
12                              allTwoDThings[i] = new Square(2 * ( i + 24 ));
13                      else
14                              allTwoDThings[i] = new Circle(2 * ( i + 24 ));
15              }
16              int  sumOfAllAreas = 0;
17
18              for ( int i = 0; i < MAXIMUM; i++ )
19                      sumOfAllAreas += allTwoDThings[i].area();
20
21              System.out.println("sumOfAllAreas = " + sumOfAllAreas );
22        }
23      }
```

Source Code: Src/6b/TestAbstract_2.java

Next

```
 1      /**
 2       * Abstract class
 3       * @version   $Id$
 4       *
 5       * @author    hp bischof
 6       *
 7       * Revisions:
 8       *      $Log$
 9       */
10
11      abstract class Area extends Object {
12
13        String type;
14
15        public String getType()       {
16              return type;
17        }
18
19        public abstract int area();
20        public abstract int perimeter();
21      }
```

Source Code: Src/6b/Area.java

Next

```
 1      /**
 2       * This class implements a Circle class.
 3       *
 4       * @version   $Id$
 5       *
 6       * @author    hp bischof
 7       *
 8       * Revisions:
 9       *      $Log$
10       */
11
12      public class Circle extends Area {
13        private int radius;
14        public Circle(int _radius)    {
15              type = "Circle";
16              radius = _radius;
17        }
18
19        public int area()     {
20              return (int)(Math.PI * radius * radius);
21        }
22      //   /*
23        public int perimeter()        {
24              return (int)(Math.PI * radius * radius);
25        }
26      //   */
27      }
```

Source Code: Src/6b/Circle.java

You will get a compiler error, if a class doesn't implement all methods.

```
% javac C*a
Circle.java:12: class Circle must be declared abstract.
It does not define int perimeter() from class Area.
public class Circle extends Area {
             ^
1 error
```

```
 1      /**
 2       * This class implements a Square class.
 3       *
 4       * @version   $Id$
 5       *
 6       * @author    hp bischof
 7       *
 8       * Revisions:
 9       *      $Log$
10       */
11
12      public class Square extends Area {
13
14        private int length;
15
16        public Square(int _length)    {
17              type = "Square";
18              length = _length;
19        }
20
21        public int area()     {
22              return length * length;
23        }
24
25        public int perimeter()        {
26              return 4 * length;
27        }
28
29      }
```

Source Code: Src/6b/Square.java

7.14. Class Cast and Abstract Classes

```
 1      abstract class A {
 2
 3        public int x;
 4
 5        abstract public A a(int x);
 6
 7        public A aa(int x)    {
 8              System.out.print("- in A!aa");
 9              return this;
10        }
11
12      }
```

Source Code: Src/6/A.java

Next

```
 1      class AX extends A {
 2
 3        public int x;
 4
 5        public A a(int x)     {
 6              System.out.print("= in AX!a");
 7              return this;
 8        }
 9
10        public static void main(String args[])        {
11              AX aAX = new AX();
12              A  aA  = (A)aAX;
13
14              System.out.println("aAX.a(42)   " + aAX.a(42) );
15              System.out.println("aAX.a(43)   " + aAX.aa(43) );
16
17              System.out.println("aA.aa(44)   " + aA.aa(44) );
18              System.out.println("aA.a(45)    " + aA.a(45) ); // <--
19        }
20      }
```

Source Code: Src/6/AX.java

Next

```
% java AX
= in AX!aaAX.a(42)  AX@e76cbf7
- in A!aaaAX.a(43)  AX@e76cbf7
- in A!aaaA.aa(44)  AX@e76cbf7
= in AX!aaA.a(45)   AX@e76cbf7
```

Next

```
 1      class AXX extends A {
 2
 3        public int x;
 4
 5        public A a(int x)     {
 6              System.out.println("    in AX!a");
 7              return this;
 8        }
 9
10        public static void main(String args[])        {
11              AX  aAX   = new AX();
12              AXX aAXX  = new AXX();
13
14              System.out.println("aAX.a(42)   " + aAX.a(42) );
15              System.out.println("aAXX.a(43)  " + aAXX.a(43) );
16        }
17      }
```

```
Source Code: Src/6/AXX.java
```

7.15. Site Note: Documentation — javadoc

The various JDK packages are documented on HTML pages in a standardized format that contains many references to other classes or replaced methods etc.

javadoc (http://java.sun.com/products/jdk/javadoc/index.html)
creates the documentation directly from the program sources. Special comments /** ... */ before class, variable and method declarations are transferred to the documentation. The comments may contain significant elements:

@see class#method	creates a reference.
@param name text	describes a method parameter.
@return text	describes the result value of a method.
@exception class text	describes an exception.

The documentation is useful even without special comments because it describes the embedding in the class hierarchy and the redefinition of methods. References to existing documentation and graphics would have to be post processed, however.

Example:

```
% javadoc ----
javadoc: invalid flag: ----
usage: javadoc [options] [packagenames] [sourcefiles] [@files]
-overview <file>          Read overview documentation from HTML file
-public                   Show only public classes and members
-protected                Show protected/public classes and members (default)
-package                  Show package/protected/public classes and members
-private                  Show all classes and members
-help                     Display command line options
-doclet <class>           Generate output via alternate doclet
-docletpath <path>        Specify where to find doclet class files
-1.1                      Generate output using JDK 1.1 emulating doclet
-sourcepath <pathlist>    Specify where to find source files
-classpath <pathlist>     Specify where to find user class files
-bootclasspath <pathlist> Override location of class files loaded
                          by the bootstrap class loader
-extdirs <dirlist>        Override location of installed extensions
-verbose                  Output messages about what Javadoc is doing
-locale <name>            Locale to be used, e.g. en_US or en_US_WIN
-encoding <name>          Source file encoding name
-J<flag>                  Pass <flag> directly to the runtime system

Provided by Standard doclet:
-d <directory>            Destination directory for output files
-use                      Create class and package usage pages
-version                  Include @version paragraphs
-author                   Include @author paragraphs
-splitindex               Split index into one file per letter
-windowtitle <text>       Browser window title for the documenation
-doctitle <html-code>     Include title for the package index(first) page
-header <html-code>       Include header text for each page
-footer <html-code>       Include footer text for each page
-bottom <html-code>       Include bottom text for each page
-link <url>               Create links to javadoc output at <url>
-linkoffline <url> <url2> Link to docs at <url> using package list at <url2>
-group <name> <p1>:<p2>.. Group specified packages together in overview page
-nodeprecated             Do not include @deprecated information
-nodeprecatedlist         Do not generate deprecated list
-notree                   Do not generate class hierarchy
-noindex                  Do not generate index
-nohelp                   Do not generate help link
-nonavbar                 Do not generate navigation bar
-helpfile <file>          Include file that help link links to
-stylesheetfile <path>    File to change style of the generated documentation
-docencoding <name>       Output encoding name
1 error
```

```
% javadoc -d Html  TestTwoDThings.java Square.java Circle.java \
  Cube.java  TwoDThings.java
Loading source file TestTwoDThings.java...
Loading source file Square.java...
Loading source file Circle.java...
Loading source file Cube.java...
Loading source file TwoDThings.java...
Constructing Javadoc information...
Building tree for all the packages and classes...
Building index for all the packages and classes...
Generating Html/overview-tree.html...
Generating Html/index-all.html...
Generating Html/deprecated-list.html...
Building index for all classes...
Generating Html/allclasses-frame.html...
Generating Html/index.html...
Generating Html/packages.html...
Generating Html/Circle.html...
Generating Html/Cube.html...
Generating Html/Square.html...
Generating Html/TestTwoDThings.html...
Generating Html/TwoDThings.html...
Generating Html/serialized-form.html...
Generating Html/package-list...
Generating Html/help-doc.html...
Generating Html/stylesheet.css...
```

You may find the documentation here:

The makefile which I used to create the javadoc for chapter 5/Testpoint:

```
 1
 2      CLASSPATH = .#                          default explicit classpath
 3      C       = .class#                       class files
 4      J       = .java#                        Java source files
 5      JAR     = jar
 6      JAVA    = CLASSPATH=$(CLASSPATH) java
 7      JAVAC   = CLASSPATH=$(CLASSPATH) javac
 8      JAVADOC = CLASSPATH=$(CLASSPATH) javadoc
 9      JDOC    = javadoc
10
11
12      all::   $c
13
14      1:      $c
15              $(JAVA) Expression
16      2:      $c
17              @ $(JAVA) Expression -c
18      3:      $c
19              $(JAVA) Go
20
21      jdoc:
22              if [ ! -d $(JDOC) ]; then mkdir $(JDOC); fi
23              $(JAVADOC)                                      \
```

```
24                -d $(JDOC)                                                    \
25                -use                                                          \
26                -splitIndex                                                   \
27                -windowtitle 'Expression '                      \
28                -doctitle 'LP<sup><font size="-2">TM</font></sup> Expression' \
29                -header '<b>LP </b><br><font size="-1">v1.0</font>'   \
30                -bottom 'Copyright hpb.'      \
31                -version                                                      \
32                -author Point.java TestPoint.java
```

Source Code: Src/5/makefile

7.16. Additional Examples

Given are the following class hierarchy:

```
1
2       abstract class A {
3
4         public abstract int isAbstract();
5
6         public A concrete()   {
7               System.out.println("A!concrete()");
8               return this;
9         }
10
11      }
12
```

Source Code: Src/6_a/A.java

Next

```
1
2       class B extends A {
3
4         public B()    {
5               System.out.println("    B()");
6         }
7
8         public int isAbstract()       {
9               System.out.println("    B!isAbstract()");
10              return 1;
11        }
12
13        public A concrete()   {
14              System.out.println("B!concrete()");
15              return this;
16        }
17
18      }
19
```

Source Code: Src/6_a/B.java

Next

```
 1
 2      class C extends A {
 3
 4        public C()    {
 5              System.out.println("    C()");
 6        }
 7
 8        public int isAbstract()       {
 9              System.out.println("    C!isAbstract()");
10              return 2;
11        }
12
13        public static void main(String args[] )       {
14              B aB = new B();
15              C aC = new C();
16
17              aB.isAbstract();
18              aC.isAbstract();
19
20              (aB.concrete()).isAbstract();
21              (aC.concrete()).isAbstract();
22
23        }
24
25      }
26
```

Source Code: Src/6_a/C.java

Next

Draw the class hierarchy.

```
% java C
    B()
    C()
        B!isAbstract()
        C!isAbstract()
B!concrete()
        B!isAbstract()
A!concrete()
        C!isAbstract()
```

• Why what is happening in the marked lines. Will this program compile? Will this program run?

```
 1
 2      class UseBandC {
 3
 4        public static void main(String args[] )       {
 5              int sum = 0;
 6              final int MAX;          // or final int MAX = 6;
 7              MAX = 3;
 8              Object[] aArray = new Object[MAX];      // ***
 9              // A[] aArray = new A[MAX];             // ***
10
11              for ( int i = 0; i < aArray.length; i ++ )
12                      if ( i % 2 == 0 )
13                              aArray[i] = new B();    // ***
14                      else
15                              aArray[i] = new C();    // ***
16
17              for ( int i = 0; i < aArray.length; i ++ )      {
18                      sum += aArray[i].isAbstract();  // ***
19                      sum += ( (A)aArray[i]).isAbstract();  // ***
20              }
21
22        }
23
24      }
25
```

Source Code: Src/6_a/UseBandC.java

Next

```
% java UseBandC
    B()
    C()
    B()
        B!isAbstract()
        C!isAbstract()
        B!isAbstract()
```

• Why what is happening in the marked lines. Will this program compile? Will this program run?

```
 1
 2      import java.util.Vector;
 3      class UseBandCandV {
 4
 5        public static void main(String args[] )       {
 6              int sum = 0;
 7              final int MAX = 4;
 8              Vector aVector = new Vector();
 9
10              for ( int i = 0; i < MAX; i++ )
11                      if ( i % 2 == 0 )
12                              aVector.add( new B());
```

```
13                      else
14                              aVector.add( new C());
15
16              for ( int i = 0; i < MAX; i ++ )
17              //      sum += aVector.elementAt(i).isAbstract();  ///////      Why do
18                      sum += ((A)aVector.elementAt(i)).isAbstract();  ///////
19
20        }
21
22      }
23
```

Source Code: Src/6_a/UseBandCandV.java

7.17. Interfaces

• An interface specifies which methods must be implement.

• An interface defines an public API.

• This means, we can make sure, that unrelated classes share the same part of the interface.

• An interface defines constants. They are public, static and final regardless of wether these modifiers have been specified.

• Methods implementations have been added to be able to extend the funtionality by modifying the interface and not the implementations.

• Interface methods can't be native, static, synchronized, final, private, or protected

• Abstract and native methods can't have a body.

• Fields in a field a static and final.

```
 1      public interface X {
 2
 3              static double MIMUM_INCREASE = 1.6 ;    // %  final
 4
 5      /*
 6       * Interface methods can't be native,
 7       * static, synchronized, final, private, or protected
 8         Abstract and native methods can't have a body.
 9       */
10              public void volume()
11              {
12                      System.out.println("xxxx");
13              }
14              public void setPrice(int x);
15      }
```

Source Code: Src/6c/X.java

Next

```
 1      public interface InCommon {
 2              static double INCREASE = 0.1;
 3              static double MINIMUM_VOLUME = 0;
 4              static double MAXIMUM_VOLUME = 10;
 5              static double DEFAULt_VOLUME = 3;
 6
 7              public double   getVolume();
 8              public boolean  setVolume(double volumeValue);
 9              public boolean  increaseVolumeBy(double deltaVolume);
10      }
```

Source Code: Src/6c/InCommon.java

Next

```
1       public class Phone implements InCommon {
2
3               double currentVolume = 0;
4               public boolean setVolume(double deltaVolume) {
5                       boolean rValue;
6                       if ( rValue = ( currentVolume + deltaVolume < MAXIMUM_VOLUME )
7                               currentVolume += deltaVolume;
8                       return rValue;
9               }
10              public double getVolume() {
11                      return currentVolume;
12              }
13              public boolean increaseVolumeBy(double deltaVolume) {
14                      return setVolume(INCREASE);
15              }
16      /*
17      Phone.java:1: error: Phone is not abstract and does not override abstract meth
18      public class Phone implements InCommon {
19      */
20      }
21
```

Source Code: Src/6c/Phone.java

Next

```
1       public class Headset implements InCommon {
2
3               double currentVolume = 0;
4               public boolean setVolume(double deltaVolume) {
5                       return setPhoneVolumeForHeadSet(deltaVolume);
6               }
7               public double getVolume() {
8                       return getPhoneVolumeForHeadSet();;
9               }
10              public boolean increaseVolumeBy(double deltaVolume) {
11                      return increasePhoneVolumeForHeadSet(deltaVolume);
12              }
13      }
14
```

Source Code: Src/6c/HeadSet.java

• An interface can extend or inherit from more than one interface, and can provide implementations:

```
1       public interface C extends A,B {
2
3               int AB = 1;
4               // Attempt to reference field AB in a int.
5
6               public void c();
7       }
```

Source Code: Src/6c/C.java

Next

```
1       public interface B {
2
3               static int B  = 2;
4               int AB = 2;
5
6               public void b();
7       }
```

Source Code: Src/6c/B.java

Next

```
1       public interface A {
2
3               static int A  = 1;
4               int AB = 1;
5
6               public void a();
7
8               default void b(){};
9
10              default double c(){};
11      }
```

Source Code: Src/6c/A.java

Next

```
1       public class Cuse implements C {
2
3           public void a() {
4               System.out.println("CUse!a");
5               // System.out.println("B  = " + A.AB);
6           }
7
8           public void b() {
9               System.out.println("CUse!b");
10          }
```

```
11          public void c() {
12              System.out.println("CUse!c");
13          }
14
15          public static void main(String argv[])      {
16              new Cuse().a();
17              System.out.println("A  = " + A);
18              System.out.println("B  = " + B);
19          }
20      }
21
```

Source Code: Src/6c/Cuse.java

Implementation of methods defined in an interface and class:

```
 1      public interface AA {
 2
 3              public void a();
 4
 5      /*
 6              default void b(){
 7                      System.out.println("A.b()");
 8              }
 9      AAandBBuse.java:1: error: class AAandBBuse inherits unrelated defaults for b()
10      public class AAandBBuse implements AA, BB {
11      */
12      }
```

Source Code: Src/6c/AA.java

Next

```
 1      public interface BB {
 2
 3              public void a();
 4
 5              default void b(){
 6                      System.out.println("A.b()");
 7              }
 8      }
```

Source Code: Src/6c/BB.java

Next

```
 1      public class AAandBBuse implements AA, BB {
 2
 3          public void a() {           // interface
 4              System.out.println("AAandBBuse!a");
 5          }
 6
 7          public void b(){
```

```
 8              System.out.println("AAandBBuse.b()");
 9              BB.super.b();
10          }
11
12          public static void main(String argv[])      {
13              new AAandBBuse().a();
14              new AAandBBuse().b();
15          }
16      }
17
```

Source Code: Src/6c/AAandBBuse.java

Output:

```
AAandBBuse!a
AAandBBuse.b()
A.b()
```

7.18. Differnces between Abstract Classes and Interfaces

From: https://www.tutorialspoint.com/differences-between-abstract-class-and-interface-in-java

Supported Methods

- Abstract class: can have both an abstract as well as concrete methods.
- Interface: can have only abstract methods. Java 8 onwards, it can have default as well as static methods.

Multiple Inheritance

- Abstract class: Multiple Inheritance is not supported.
- Interface: supports multiple inheritance.

Supported Variables

- Abstract class: final, non-final, static and non-static variables supported.
- Interface: Only static and final variables are permitted.

Implementation

- Abstract class can implement an interface.
- Interface: can not implement an interface, it can extend an interface.

Inheritance

- Abstract class: can inherit another class using extends keyword and implement an interface.
- Interface: can inherit only an inteface.

Inheritance

- Abstract class: can be inherited using extends keyword.
- Interface: can only be implemented using implements keyword.

Access to Members

- Abstract class: can have any type of members like private, public.
- Interface: can only have public members.

7.19. Aggregation

• Aggregation is a design term, which means that you create an new object by composing it out of the others.

• Aggregation relationships are specified by classes and reflected by their instance objects.

• For example: A Cylinder class can be defined as:

```
 1      /**
 2       * This class implements a Cylinder Class
 3       * NOT COMPLETE
 4       * @version   $Id$
 5       *
 6       * @author    hp bischof
 7       *
 8       * Revisions:
 9       *      $Log$
10       */
11
12      public class Cylinder {
13        private aCircle;
14        private aRect;
15
16        public Cylinder(int _radius, _height) {
17              aCircle = new Circle(radius);
18              aRect = new Rectangle(aCircle.perimeter(), height);
```

```
19        }
20        public int area()     {
21              return aCircle.area * 2 + aRect.area();
22        }
23        ....
24      }
```

Source Code: Src/6b/Cylinder.java

7.20. Short Examples for Clarification

Default Constructor Sequence

Which constructor is called when?

```
 1      public class X_1 {
 2
 3        public X_1()    {
 4              System.out.println("    in X_1!X_1()");
 5        }
 6
 7        public X_1(int x)    {
 8              System.out.println("    in X_1!X_1(int x)");
 9        }
10
11        public X_1(int x, int y)    {
12              System.out.println("    in X_1!X_1(int x, int y)");
13        }
14
15      }
```

Source Code: Src/6g/X_1.java

```
 1      class X_2 extends X_1 {
 2
 3        public X_2()    {
 4              // super();     // default
 5              System.out.println("    in X_2!X_2()");
 6        }
 7        public X_2(int x)    {
 8              // super();     // default
 9              super(x);
10              System.out.println("    in X_2!X_2(int x)");
11        }
12
13        public X_2(int x, int y)    {
14              // super();     // default
15              System.out.println("    in X_2!X_2(int x, int y)");
16        }
17
18        public static void main(String args[])
19        {
20              X_2 aX_2 = new X_2();
21              X_2 aaX_2 = new X_2(3);
22              X_2 aaaX_2 = new X_2(3, 3);
23        }
24      }
25
```

Source Code: Src/6g/X_2.java

Result:

```
in X_1!X_1()
    in X_2!X_2()
in X_1!X_1(int x)
    in X_2!X_2(int x)
in X_1!X_1()
    in X_2!X_2(int x, int y)
```

Constructor must match

Superclass has no *default()* constructor!

```
1       public class X_1 extends Object {
2
3         public X_1(int x)     {
4               System.out.println("    in X_1!X_1(int x)");
5         }
6
7       }
```

Source Code: Src/6h/X_1.java

```
 1      class X_2 extends X_1 {
 2
 3        public X_2()    {
 4              System.out.println("    in X_2!X_2()");
 5        }
 6
 7        public static void main(String args[])
 8        {
 9              X_2 aX_2 = new X_2();
10        }
11      }
12
```

Source Code: Src/6h/X_2.java

Result:

```
% javac X*a
X_2.java:3: No constructor matching X_1() found in class X_1.
  public X_2()    {
         ^
1 error
```

• Overloading of constructors is identical in behavior to overloading of methods. The overloading is resolved at compile time by each class instance creation expression.

• If a class contains no constructor declarations, then a default constructor that takes no parameters is automatically provided:

• If the class being declared is the primordial class Object, then the default constructor has an empty body.

• Otherwise, the default constructor takes no parameters and simply invokes the superclass constructor with no arguments.

• A compile-time error occurs if a default constructor is provided by the compiler but the superclass does not have a constructor that takes no arguments.

Methods

Access of methods and *super* methods.

```
 1      public class X_1 {
 2
 3        public X_1()    {
 4              System.out.println("    in X_1!X_1()");
 5        }
 6
 7        public X_1(int x)    {
 8              System.out.println("    in X_1!X_1(int x)");
 9        }
10
11        public void a()    {
12              System.out.println("    in X_1!a()");
13        }
14
15      }
```

Source Code: Src/6f/X_1.java

```
 1      class X_2 extends X_1 {
 2
 3        public X_2()    {
 4              // super();     // default
 5              super.a();
 6              System.out.println("    in X_2!X_2()");
 7        }
 8        public X_2(int x)    {
 9              // super();     // default
10              super(x);
11              System.out.println("    in X_2!X_2(int x)");
12        }
13
14        public void a()    {
15              super.a();
16              System.out.println("    in X_2!a()");
17        }
18
19
20        public static void main(String args[])
21        {
22              X_2 aX_2 = new X_2();
23              aX_2.a();
24              X_2 anOtherX_2 = new X_2(3);
25        }
26      }
27
```

Source Code: Src/6f/X_2.java

Result:

```
% java X_2
    in X_1!X_1()
    in X_1!a()
        in X_2!X_2()
    in X_1!a()
    in X_2!a()
```

Instance Variables

Which one do I get?

```
1       public class X_1 extends Object {
2
3         int x1 = 1;
4         int x2 = 11;
5
6       }
```

Source Code: Src/6i/X_1.java

```
 1      class X_2 extends X_1 {
 2
 3              int x1 = 2;
 4
 5        public void a()    {
 6              System.out.println("    in X_1!a()");
 7              System.out.println("    in X_1!a()!x1 = " + x1 );
 8              System.out.println("    in X_1!a()(X_1)this.x1 = " + this.x1 );
 9              System.out.println("    in X_1!a()!super.x1 = "
10                                              + super.x1 );
11        }
12
13        public static void main(String args[])
14        {
15              X_2 aX_2 = new X_2();
16              System.out.println("    main!x1 = " + aX_2.x1 );
17              aX_2.a();
18        }
19      }
20
```

Source Code: Src/6i/X_2.java

Result:

```
% java X_2
    main!x1 = 2
    in X_1!a()
    in X_1!a()!x1 = 2
    in X_1!a()(X_1)this.x1 = 2
    in X_1!a()!super.x1 = 1
```

Private or Protected?

Can I?

```
1       public class X_1 extends Object {
2
3         static int x1 = 1;
4         private int x2 = 11;
5
6       }
```

Source Code: Src/6k/X_1.java

```
1       class X_2 extends X_1 {
2
3               static int x1 = 2;
4               int x2 = 22;
5
6         public void a()    {
7               System.out.println("    in X_1!a()");
8               System.out.println("    in X_1!a()!x1 = " + x1 );
9               System.out.println("    in X_1!a()!super.x1 = "
10                                              + super.x1 );
11              System.out.println("    in X_1!a()!X_1.x1 = "
12                                              + X_1.x1 );
13
14              /*   System.out.println("    in X_1!a()!super.x2 = "
15               *                              + super.x2);
16               *  X_2.java:13: Variable x2 in class
17               *               X_1 not accessible from class X_2.
18               *                                          ^
19               *
20               */
21        }
22
23        public static void main(String args[])
24        {
25              X_2 aX_2 = new X_2();
26              System.out.println("    main!x1 = " + aX_2.x1 );
27              System.out.println("    main!x1 = " + aX_2.x2 );
28              aX_2.a();
29        }
30      }
```

Source Code: Src/6k/X_2.java

Result:

```
% java X_2
    main!x1 = 2
    main!x1 = 22
    in X_1!a()
    in X_1!a()!x1 = 2
    in X_1!a()!super.x1 = 1
    in X_1!a()!X_1.x1 = 1
```

7.21. A Binary Search Tree

Interface:

```
1       public interface Node {
2
3           public boolean isLess(Node aNode);
4           public boolean isEqual(Node aNode);
5           public boolean isGreater(Node aNode);
6
7       }
```

Source Code: Src/6t/Node.java

Possible mplementation:

```
 1
 2      public class StringNode implements Node {
 3
 4          private String info;
 5
 6          public StringNode( String info ) {
 7              this.info = info;
 8          }
 9
10          public boolean isLess( Node aNode ) {
11              return ( info.compareTo( aSNode.info ) < 0 );// wrong
12          }
13          public boolean isGreater( Node aNode ) {
14              StringNode aStringNode = ( StringNode )aNode;
15
16              return ( info.compareTo( aStringNode.info ) > 0 );
17          }
18          public boolean isEqual( Node aNode ) {
19              StringNode aStringNode = ( StringNode )aNode;
20
21              return ( info.compareTo( aStringNode.info ) == 0 );
22          }
23          public String toString() {
24              return ( info );
25          }
26      }
```

Source Code: Src/6t/StringNode.java

This code will not compile? Why? (des is not so schwer, a node who is a textle, is not a nodle of the interface.

A binary search tree implementation:

```
 1      public class BinarySearchTree {
 2
 3          private Node data;
 4          private BinarySearchTree left;
 5          private BinarySearchTree right;
 6
 7          public BinarySearchTree() {
 8              this( null );
 9          }
10
11          public BinarySearchTree( Node data ) {
12              this.data = data;
13              this.left = this.right = null;
14          }
15
16          public void insert( Node aNode ) {
17              if ( aNode.isLess( data ) )       {
18                  if ( left == null )
```

```
19                  left = new BinarySearchTree( aNode );
20              else
21                  left.insert( aNode );
22          }
23          else {
24              if ( right == null )
25                  right = new BinarySearchTree( aNode );
26              else
27                  right.insert( aNode );
28          }
29      }
30
31      public BinarySearchTree find( Node aNode ) {
32          if ( aNode.isEqual( data ) ) return this;
33
34          if ( this.left != null && aNode.isLess( data ) )
35              return left.find( aNode );
36
37          if ( this.right != null && aNode.isGreater( data ) )
38              return right.find( aNode );
39
40          return null;
41      }
42
43      public void printInorder() {
44          if ( left != null ) left.printInorder();
45          System.out.print( data.toString() + " " );
46          if ( right != null ) right.printInorder();
47      }
48
49      public void printPostOrder() {
50          if ( left != null ) left.printPostOrder();
51          if ( right != null ) right.printPostOrder();
52          System.out.print( data.toString() + " " );
53      }
54
55      public void printPreOrder() {
56          System.out.print( data.toString() + " " );
57          if ( left != null ) left.printPreOrder();
58          if ( right != null ) right.printPreOrder();
59      }
60
61      public static void main( String args[] )      {
62          BinarySearchTree aBS =
63              new BinarySearchTree( new StringNode( "d" ) );
64
65          aBS.insert( new StringNode( "b" ) );  //            d
66          aBS.insert( new StringNode( "a" ) );  //      b          f
67          aBS.insert( new StringNode( "c" ) );  //    a   c      e   g
68          aBS.insert( new StringNode( "f" ) );  //
69          aBS.insert( new StringNode( "e" ) );  //
70          aBS.insert( new StringNode( "g" ) );  //
71
72          // Try the traversals
```

```
73
74              // Inorder output should be:  a b c d e f g
75
76              System.out.print( "Inorder:\t" );
77              aBS.printInorder();
78              System.out.println();
79
80              // Postorder output should be:  a c b e g f d
81
82              System.out.print( "Postorder:\t" );
83              aBS.printPostOrder();
84              System.out.println();
85
86              // Preorder output should be:  d b a c f e g
87
88              System.out.print( "Preorder:\t" );
89              aBS.printPreOrder();
90              System.out.println();
91
92              // Try looking for some stuff
93
94              if ( aBS.find( new StringNode( "d" ) ) != null )
95                      System.out.println( "found d" );
96              if ( aBS.find( new StringNode( "a" ) ) != null )
97                      System.out.println( "found a" );
98              if ( aBS.find( new StringNode( "g" ) ) != null )
99                      System.out.println( "found g" );
00              if ( aBS.find( new StringNode( "x" ) ) != null )
01                      System.out.println( "found x" );
02        }
03
04      } // BinaryTree
05
06
```

Source Code: Src/6t/BinarySearchTree.java

Get the name of a Class:

```
 1      import java.lang.reflect.Method;
 2
 3      public class PrintClassName {
 4
 5        public static void printClassName(Object o)   {
 6              System.out.println("o.getClass().getName() = " +
 7                                 o.getClass().getName());
 8        }
 9
10        public static void printMethods(Object o)     {
11              Method[] m = o.getClass().getMethods();
12              for (int i = 0; i < m.length; i ++ )
13                      System.out.println("m[" + i + "]\t=\t" +
14                              m[i].getName());
15
```

```
16        }
17
18        public static void main(String args[])        {
19              String aString = "aaa";
20              Integer aInteger = new Integer("0");
21
22              printClassName((Object)aString);
23              printClassName((Object)aInteger);
24
25              printMethods((Object)aInteger);
26        }
27      }
```

Source Code: Src/6/PrintClassName.java

7.22. Nested Classes

• Nested classes can be: non-static or static nested.

• Non-static nested classes are refered to as inner classes.

• A nested class is a class that is a member of another class.

```
class Outer{
   . . .
  class AnestedClass {
          . . .
  }
}
```

• A nested class can be declared static or not.

• A static nested class is called a static nested class.

• A non static nested class is called an inner class.

• As with static methods and variables, a static nested class is associated with its enclosing class.

• And like class methods, a static nested class cannot refer directly to instance variables or methods defined in its enclosing class-it can use them only through an object reference.

• As with instance methods and variables, an inner class is associated with an instance of its enclosing class and has direct access to that object's instance variables and methods.

• Because an inner class is associated with an instance, it cannot define any static members itself.

• Example:

```
 1      public class NestedClassEx {
 2
 3        public int inNestedClass;
 4
 5        void inNestedClass()  {
 6              System.out.println("NestedClass!inNestedClass");
 7              (new AinnerClass()).aInnerClassM2();
 8        }
 9
10        static class AstaticClass     {
11              static void aStaticClassM1()    {
12                      System.out.println("AstaticClass!aStaticClassM1");
13              }
```

```
14              void aStaticClassM2()   {
15                      System.out.println("AstaticClass!aStaticClassM2");
16              }
17        }
18
19        class AinnerClass     {
20      /*
21              static void aInnerClassM1()     {
22                      System.out.println("AinnerClass!aInnerClassM1");
23              }
24         NestedClassEx.java:15: inner classes cannot have static declarations
25              static void aInnerClassM1()     {
26       */
27
28              void aInnerClassM2()    {
29                      System.out.println("AinnerClass!aInnerClassM2");
30              }
31        }
32
33        public static void main(String args[])        {
34
35              AstaticClass.aStaticClassM1();
36              (new AstaticClass()).aStaticClassM2();
37
38              (new NestedClassEx()).inNestedClass();
39              // (new AinnerClass()).aInnerClassM2();
40        }
41      }
```

Source Code: Src/6/NestedClassEx.java

7.23. Anonymous Classes

• Inner class can be declared without naming it.

• Anonymous classes can make code difficult to read. Their use should be limit to those classes that are very small (no more than a method or two) and whose use is well-understood (like the AWT event-handling adapter classes).

• Example:

```
public static void main(String[] args) {
    JFrame f = new JFrame("RectangleDemo");
    f.addWindowListener(new WindowAdapter() {
        public void windowClosing(WindowEvent e) {
            System.exit(0);
        }
    });
    RectangleDemo controller = new RectangleDemo();
    controller.buildUI(f.getContentPane());
    f.pack();
    f.setVisible(true);
}
```

7.24. Static Initializer Blocks

• Static initializer blocks are primarily used for initialization.

• The code in a static initializer block is executed when the class is initialized/

• A class can have more than one static initializer block.

• S.java:

```
 1      public class S {
 2
 3        static public int intS;
 4
 5        public S ()   {
 6              System.out.println("in S constructor");
 7        }
 8
 9        static {
10              System.out.println("S:Static 1");
11        }
12
13        static {
14              System.out.println("S: Static 2");
15        }
16
17        public static void main(String args[])        {
18              System.out.println("new S()     " +  new S());
19        }
20      }
```

Source Code: Src/6_AddOn/S.java

• SubclassOfS.java:

```
 1      public class SubclassOfS extends S {
 2
 3        public int intS;
 4
 5        static {
 6              System.out.println("SubclassOfS: Static 1");
 7        }
 8
 9        public SubclassOfS () {
10              System.out.println("in SubclassOfS constructor");
11        }
12
13        public static void main(String args[])        {
14              System.out.println("In SubClass of S");
15              SubclassOfS aSubclassOfS = new SubclassOfS();
16        }
17      }
```

Source Code: Src/6_AddOn/SubclassOfS.java

7.25. Questions

• Will it compile:

```
1       public class Yellow {
2         private String yellowPrivate = "yellowPrivate";
3
4        public static void main(String args[]) {
5               System.out.println(yellowPrivate);
6        }
7       }
```

Source Code: Src/6_q/Yellow.java

• Will it compile and if yes, what is the output: (eine neue variable wird erzeugt).

```
1       public class Coke {
2         private        String cokePrivate  = "cokePrivate";
3         private        String s;
4         private static String cokePrivateS = "cokePrivateS";
5
6         public void m()       {
7               cokePrivate = "java";
8         }
9
10        public void change(String cokePrivate)        {
11              cokePrivate = "hello";
12         }
13        public void print()   {
14              System.out.println("1. cokePrivate  = " + cokePrivate  );
15              System.out.println("2. cokePrivateS = " + cokePrivateS );
16              System.out.println("-----------------------------------");
17        }
18        public static void main(String args[])        {
19              Coke aCoke = new Coke();
20              aCoke.m();
21              aCoke.print();
22              aCoke.change("t");
23              aCoke.print();
24       }
25      }
```

Source Code: Src/6_q/Coke.java

• Will it compile: (die antwort zu der frage kann nur verneint werden)

```
1       public class Red {
2         private String redPrivate = "redPrivate";
3       }
```

Source Code: Src/6_q/Red.java

An other example:

```
1       public class Blue {
2         private String bluePrivate = "bluePrivate";
3
4         public boolean isLess(Red aRed)       {
5               return bluePrivate == aRed.redPrivate;
6         }
7
8        public static void main(String args[]) {
9               Red  aRed = new Red();
10              Blue aBlue = new Blue();
11              aBlue.isLess(aRed);
12       }
13      }
```

Source Code: Src/6_q/Blue.java

• Will it compile and if yes, what is the output:

```
1       public class H {
2         private        String hPrivate  = "hPrivate";
3         private static String hPrivateS = "hPrivateS";
4
5         public H(String hPrivate)     {
6               this.hPrivate = hPrivate;
7         }
8
9         public void knete()   {
10              this = this("RIT");
11        }
12
13        public void print(String tag) {
14              System.out.println(tag + "hPrivate  = " + hPrivate  );
15        }
16
17        public static void main(String args[])        {
18              H aH = new H();
19              aH.print("1.");
20              aH.knete();
21              aH.print("2.");
22       }
23      }
```

Source Code: Src/6_q/H.java

Execution:

```
H.java:10: Invalid left hand side of assignment.
        this = this("RIT");
        ^
H.java:10: Only constructors can invoke constructors.
        this = this("RIT");
                    ^
H.java:18: No constructor matching H() found in class H.
        H aH = new H();
```

• Will it compile and if yes, what is the output:

```
 1      public class Bauer {
 2        private        String bauerPrivate  = "bauerPrivate";
 3        private static String bauerPrivateS = "bauerPrivateS";
 4
 5        public Bauer()        {
 6        }
 7
 8        public Bauer(String bauerPrivate)     {
 9              this.bauerPrivate = bauerPrivate;
10        }
11
12        public void knete()   {
13              this = new Bauer("RIT");
14        }
15
16        public static void main(String args[])        {
17              Bauer aBauer = new Bauer();
18       }
19      }
```

Source Code: Src/6_q/Bauer.java

• Will it compile and if yes, what is the output: (tja, das sollte nicht so schwierig sein)

```
 1      class Oh {
 2        public static void intMethod(int intArg) {
 3              intArg = 22;
 4        }
 5        public static void intArrayMethod(int[] intArg) {
 6              intArg[0] = 22;
 7        }
 8        public static void main(String[] args) {
 9              int intVariable = 2;
10              int intArray[]  = { 2, 2, 2 };
11              System.out.println("1. " + intVariable );
12              intMethod(intVariable);
13              System.out.println("2. " + intVariable );
14
15              System.out.println("3. " + intArray[0] );
16              intArrayMethod(intArray);
```

```
17              System.out.println("4. " + intArray[0] );
18           }
19      }
```

Source Code: Src/6_q/Oh.java

• Will it compile:

```
1       public interface I {
2               static int iStatic       = 1 ;
3               public int iPublic       = 6 ;
4               private int iPrivate     = 6 ;
5               protected int iProtected = 6 ;
6       }
```

Source Code: Src/6_q/I.java

• Will it compile:

```
1       public interface Ic {
2               static int iStatic       = 1 ;
3               public int iPublic       = 6 ;
4
5               public void furyo();
6       }
```

Source Code: Src/6_q/Ic.java

An other interface:

```
1       public class Ic1 implements Ic {
2
3        public void furyo ()   {
4               iPublic = 4;
5               iStatic = 2;
6        }
7
8       }
```

Source Code: Src/6_q/Ic1.java

• Will it compile: (eins ist genug)

```
1       public class Bier {
2         private int bier;
3
4         public Bier() {
5               bier ++;
6         }
```

```
 7
 8        public void print()   {
 9              System.out.println("bier  = " + bier  );
10        }
11
12        public static void main(String args[])        {
13              Bier aBier = new Bier();
14              for ( int i = 0; i < 1000; i ++ )
15                      aBier = new Bier();
16              aBier.print();
17       }
18      }
```

Source Code: Src/6_q/Bier.java

• Will it compile and what is the output: (ein Object koennte nicht zugwiesen sein)

```
 1      public class Wein {
 2        private int wein;
 3
 4        public Wein() {
 5              wein ++;
 6        }
 7
 8        public void print()   {
 9              System.out.println("wein  = " + wein  );
10        }
11
12        public static void main(String args[])        {
13              Wein aWein;
14              for ( int i = 0; i < 1000; i ++ )
15                      aWein = new Wein();
16              aWein.print();
17       }
18      }
```

Source Code: Src/6_q/Wein.java

• Will it compile and what is the output:

```
 1      public class ApfelSaft {
 2        private int gut;
 3
 4        public ApfelSaft()    {
 5              gut ++;
 6        }
 7
 8        public void print()   {
 9              System.out.println("gut  = " + gut  );
10        }
11
```

```
12        public static void main(String args[])        {
13              ApfelSaft aApfelSaft = null;
14              for ( int i = 0; i < 1000; i ++ )
15                      aApfelSaft = new ApfelSaft();
16              aApfelSaft.print();
17       }
18      }
```

Source Code: Src/6_q/ApfelSaft.java

• Will it compile and what is the output: (das dritte ist falsch)

```
 1
 2      public class SEqual {
 3
 4        public static void main(String args[])        {
 5              String s = "a";
 6              String b = "a";
 7
 8              if ( s == b )
 9                      System.out.println("1. s == b ");
10              if ( s.equals(b) )
11                      System.out.println("1. s.equals(b) ");
12
13              if ( "Furyo" == "Furyo" )
14                      System.out.println("2. \"Furyo\" == \"Furyo\" ");
15
16              s = new String("a");
17              b = new String("a");
18
19              if ( s == b )
20                      System.out.println("3. s == b ");
21              if ( s.equals(b) )
22                      System.out.println("4. s.equals(b) ");
23       }
24
25      }
```

Source Code: Src/6_q/SEqual.java

Execution:

```
1. s == b
1. s.equals(b)
2. "Furyo" == "Furyo"
4. s.equals(b)
```

7.26. What is the following Example doing?

• Number Api (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/lang/Number.html)

• Example 1:

```
 1
 2      public abstract class Node extends Number {
 3
 4        public byte byteValue () {
 5          return (byte)longValue();
 6        }
 7
 8        public short shortValue () {
 9          return (short)longValue();
10        }
11
12        public int intValue () {
13          return (int)longValue();
14        }
15
16        public float floatValue () {
17          return (float)doubleValue();
18        }
19
20        protected abstract static class Binary extends Node {
21
22          protected Number left;
23
24          protected Number right;
25
26          protected Binary () {}
27
28          protected Binary (Number left, Number right) {
29            this.left = left; this.right = right;
30          }
31        }
32
33        protected abstract static class Unary extends Node {
34
35          protected Number tree;
36
37          protected Unary (){}
38
39          protected Unary (Number tree) {
40            this.tree = tree;
41          }
42        }
43
44       public static class Add extends Binary {
45
46          public Add (Number left, Number right) {
47            super(left, right);
48          }
49
```

```
50          public long longValue () {
51            return left.longValue() + right.longValue();
52          }
53
54          public double doubleValue () {
55            return left.doubleValue() + right.doubleValue();
56          }
57        }
58
59        public static class Mul extends Binary {
60          public Mul (Number left, Number right) {
61            super(left, right);
62          }
63
64          public long longValue () {
65            return left.longValue() * right.longValue();
66          }
67
68          public double doubleValue () {
69            return left.doubleValue() * right.doubleValue();
70          }
71        }
72
73        public static class Minus extends Unary {
74          public Minus (Number tree) {
75            super(tree);
76          }
77
78          public long longValue () {
79            return - tree.longValue();
80          }
81
82          public double doubleValue () {
83            return - tree.doubleValue();
84          }
85        }
86
87        public static void main(String args[])        {
88              Node aNode = new Node.Add(new Double(1), new Double(2));
89              System.out.println("i) aNode = " + aNode.floatValue() );
90
91              aNode = new Node.Add(
92                        new Node.Mul( new Double(2), new Double(3)),
93                        new Node.Mul( new Double(4), new Double(5))
94                                  );
95              System.out.println("ii) aNode = " + aNode.floatValue() );
96              aNode = new Node.Add(
97                        new Double(1),
98                        new Node.Minus(new Double(2))
99                                  );
00              System.out.println("iii) aNode = " + aNode.floatValue() );
01        }
02      }
```

Source Code: Src/6_AddOn/Node.java

8. Generics

See also: http://docs.oracle.com/javase/specs/jls/se8/html/jls-4.html#jls-4.4 (http://docs.oracle.com/javase/specs/jls/se8/html/jls-4.html#jls-4.4)
and https://docs.oracle.com/javase/tutorial/java/generics/why.html (https://docs.oracle.com/javase/tutorial/java/generics/why.html)

and stolen from their:

• Generics enable types (classes and interfaces) to be parameters when defining classes, interfaces and methods. Much like the more familiar formal parameters used in method declarations, type parameters provide a way for you to re-use the same code with different inputs. The difference is that the inputs to formal parameters are values, while the inputs to type parameters are types.

8.1. Generic Class Definition

```
class name<T1, T2, ..., Tn> { .... }
```

Type parameter names are, by convention, single, uppercase letters. This makes it easy to differentiate between a type variable and a class/interface.

Type Parameter Naming Conventions

E - Element
K - Key
N - Number
T - Type
V - Value

8.2. Old versus New

• Old:

```
List myIntList = new LinkedList();
myIntList.add(new Integer(0));
Integer x = (Integer) myIntList.iterator().next();
```

• New:

```
List<Integer> myIntList = new LinkedList<Integer>();
myIntList.add(new Integer(0));
Integer x = myIntList.iterator().next();
```

8.3. Defining Simple Generics I

•

```
1
2       public interface List<E> {
3               void add(E x);
4               Iterator<E> iterator();
5       }
6       public interface Iterator<E> {
7               E next();
8               boolean hasNext();
9       }
```

Source Code: Src/11_jdk15/List.java

8.4. Defining Simple Generics II

• Node:

```
1       public class Node<F> {
2           public Node( F value, Node<F> next ) {
3               this.value = value;
4               this.next = next;
5           }
6           public F value;
7           public Node<F> next;
8       }
```

Source Code: Src/11_jdk15/Node.java

• The Generic Stack class:

```
1       class RStack<E> {
2           public void push( E x ) {
3               head = new Node<E>(
4                           x, head );
5           }
6           public E pop() {
7               E result = head.value;
8               head = head.next;
9               return result;
10          }
11          private Node<E> head = null;
12
13          public static void main(String args[] )     {
14              RStack<String> s;
15              s = new RStack<String>();
16              s.push("hello");
17              String w = s.pop();
18          }
19      }
```

Source Code: Src/11_jdk15/RStack.java

• How about legacy code:

```
1       class Stack {
2
3           public static void main(String args[] )     {
4               RStack s;
5               s = new RStack();
6               s.push("hello");
7               String w = (String)s.pop();
8           }
9       }
```

Source Code: Src/11_jdk15/Stack.java

Compilation:

```
% javac  Stack.java
Note: Stack.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details
```

A Vector use case:

```
 1      import java.util.Vector;
 2
 3      class VectorUse {
 4
 5
 6          static void doTheWork()     {
 7              Vector<String> aStringVector    = new Vector<String>();
 8              Vector<Integer> aIntegerVector  = new Vector<Integer>();
 9              Vector<Object> aObjectVector    = new Vector<Object>();
10
11              aStringVector.add("a");
12              aIntegerVector.add(1);
13              aObjectVector.add(new Object());
14
15              // aStringVector.add( aObjectVector.firstElement() );
16              aObjectVector.add( aStringVector.firstElement() );
17          }
18          public static void main(String args[] )     {
19              new  VectorUse().doTheWork();
20          }
21      }
```

Source Code: Src/11_jdk15/VectorUse.java

A stack:

```
 1      import java.util.Vector;
 2      class MyStack<E> {
 3          Vector<E> myStack = new Vector<E>();
 4
 5          public void push( E anElement ) {
 6              myStack.add(anElement);
 7          }
 8          public E pop() {
 9
10              if ( ! myStack.isEmpty() )      {
11                      E anElement = myStack.lastElement();
12                      myStack.remove(myStack.size() - 1) ;
13                      return anElement;
14              } else  {
15                      return null;
16              }
17          }
```

```
18          public boolean isEmpty()    {
19              return myStack.isEmpty();
20          }
21      }
```

Source Code: Src/11_jdk15/MyStack.java

A use case:

```
 1      import java.util.Vector;
 2
 3      public class MyStackUse {
 4
 5          public static void testString()     {
 6              MyStack<String> aMyStack = new MyStack<String>();
 7              aMyStack.push("a");
 8              aMyStack.push("b");
 9              while ( ! aMyStack.isEmpty() )  {
10                      System.out.println(aMyStack.pop());
11              }
12          }
13          public static void testInteger()    {
14              MyStack<Integer> aMyStack = new MyStack<Integer>();
15              aMyStack.push(1);
16              aMyStack.push(2);
17              while ( ! aMyStack.isEmpty() )  {
18                      System.out.println(aMyStack.pop());
19              }
20          }
21          public static void testMyStack()    {
22              MyStack<MyStack> aMyStack = new MyStack<MyStack>();
23              aMyStack.push(new MyStack<Integer>());
24              aMyStack.push(new MyStack<Integer>());
25              aMyStack.push(new MyStack<String>());
26              // aMyStack.push(new Object() ); // wil not compile
27
28              while ( ! aMyStack.isEmpty() )  {
29                      System.out.println(aMyStack.pop());
30              }
31          }
32          public static void main(String args[] )     {
33              // testString();
34              // testInteger();
35              testMyStack();
36          }
37      }
```

Source Code: Src/11_jdk15/MyStackUse.java

8.5. Remarks

• A compile-time only language extension.

• Parameterized types are NOT macro expanded. i.e., no real instantiation

• Compiler forgets type parameters after checking. Some usage limitations (arrays, instanceof) due to lack of guaranteed run-time type knowledge

• Type safety was primary concern. *Readability is usually improved* [Doug Lea]

• JDK 7 or later: type arguments can be empty, if the compuler can determine the type from the content

```
Box<Integer> integerBox = new Box<>();
```

8.6. An other Example

```
static void cleanUp( Collection <String>  c ) {
    for (
          Iterator <String> i = c.iterator();
          i.hasNext();
    )
        if (<String> i.next().length() == 4)
            i.remove();
}
```

8.7. Multiple Parameter Types

```
 1      import java.util.*;
 2
 3      public class MultipleTypes<E, V> {
 4              List<E>     data = new ArrayList<E>();
 5              Vector<V> volume = new Vector<V>();
 6
 7              public V getV() {
 8                      return volume.elementAt(0);
 9              }
10      }
```

Source Code: Src/11_W/MultipleTypes.java

Next

```
 1      import java.util.*;
 2
 3      public class Sorting<E, V> {
 4              List<E>     data = new ArrayList<E>();
 5              Vector<V> volume = new Vector<V>();
 6
 7              public void add(E element) {
 8                      data.add(element);
 9              }
10              public void addOne(V element) {
11                      volume.add(element);
12              }
13              public V getV(int position) {
14                      return volume.elementAt(position);
15              }
```

```
16
17              public static void main(String args[])  {
18                      Sorting<String, Integer> aSortedString = new Sorting<String, I
19                      aSortedString.add("hello");
20                      aSortedString.addOne(Integer.valueOf("1"));
21
22                      Sorting<Integer, String> aSortedInteger = new Sorting<Integer,
23                      aSortedInteger.add(Integer.valueOf("1"));
24                      aSortedInteger.addOne("hello");
25              }
26      }
```

Source Code: Src/11_W/Sorting.java

8.8. Remarks II

• No matter to what the type parameters are bound, the code for the generic class is the same.

— Objects* are assumed at compile time.

— No basic types allowed (see autoboxing)

The concept of instantiating a generic class is misleading in Java's case Is this legal?

•

```
List<String> ls = new ArrayList<String>();   // 1
List<Object> lo = ls;                        // 2
```

• Is an ArrayList of String objects

• It is not legal, because:

```
lo.add(new Object());
String s = ls.get(0); // De zwoate isch des problemle. A s isch net on o'le.
```

```
 1
 2      import java.util.*;
 3
 4      public class ObjectsStringLists {
 5
 6              public static void main(String args[])  {
 7                      List<String> ls = new ArrayList<String>();   // 1
 8                      List<Object> lo = ls;
 9              }
10      }
```

Source Code: Src/11_W/ObjectsStringLists.java

Compiler Error:

```
ObjectsStringLists.java:8: error: incompatible types: List<String> cannot be converted
            List<Object> lo = ls;
```

8.9. Generic Methods

From here (https://docs.oracle.com/javase/tutorial/java/generics/methods.html)

Generic methods are methods that introduce their own type parameters. This is similar to declaring a generic type, but the type parameter's scope is limited to the method where it is declared. Static and non-static generic methods are allowed, as well as generic class constructors.

The syntax for a generic method includes a list of type parameters, inside angle brackets, which appears before the method's return type. For static generic methods, the type parameter section must appear before the method's return type.

The clases: Brick extends Cube extends Square extends Area

```
1       public abstract class Area {
2          abstract double area();
3       }
```

Source Code: Src/11_W/Area.java

Implementation 1:

```
1       import java.lang.Math;
2
3       public class Equilateral extends Area {
4
5               double length;
6
7               Equilateral(double length)      {
8                       this.length = length;
9               }
10              public double area()    {
11                      return length * length * 0.25 * Math.sqrt(3);
12              }
13      }
```

Source Code: Src/11_W/Equilateral.java

Implementation 2:

```
1       public class Square extends Area {
2
3               double length;
4
5               Square()        {
6
7               }
8               Square(double length)   {
9                       this.length = length;
10              }
11              public double area()    {
12                      return length * length;
13              }
14      }
```

Source Code: Src/11_W/Square.java

Implementation 3:

```
1       public class Cube extends Square {
2
3               double length;
4
5               Cube(double length)     {
6                       this.length = length;
7               }
8               public double area()    {
9                       return 6 * length * length;
10              }
11              public double volume()  {
12                      return length * length * length;
13              }
14      }
```

Source Code: Src/11_W/Cube.java

Implementation 4:

```
1
2       public class Brick extends Cube {
3
4               double height;
5               double width;
6
7               Brick(double length, double height, double width)       {
8                       super(length);
9                       this.height = height;
10                      this.width = width;
11              }
12              public double area()    {
13                      return 2 * length * width + 2 * width * height + 2 * length *
14              }
15              public double volume()  {
16                      return length * width * height;
17              }
18      }
```

Source Code: Src/11_W/Brick.java

Use case:

```
1       import java.util.*;
2       import java.io.Serializable;
3
4       class GenericMethod {
5
6               static <T> void fromArrayToCollection(T[] from, Collection<T> to) {
7                       for (T anElement : from) {
```

```
                                to.add(anElement);
                        }
                }
                static <T extends Cube> void print(T aTobject)  {
                        System.out.println(aTobject);
                }
                static <T> T pick(T a1, T a2) { return a2; }

                public static void main(String[] args)  {
                        Object[] objectArray = new Object[100];
                        Collection<Object> aCollection = new ArrayList<Object>();

                        Serializable s = pick("d", new ArrayList<String>());

                        // T inferred to be Object
                        fromArrayToCollection(objectArray, aCollection);

                        // T inferred to be String
                        String[] stringArray = new String[100];
                        Collection<String> aStringCollection = new ArrayList<String>()
                        fromArrayToCollection(stringArray, aStringCollection);


                        print(new Brick(1, 2, 3) );
                        print(new Cube(1) );
                        print(new Square(1) );
        /*
        GenericMethod.java:28: error: method print in class GenericMethod cannot be ap
                        print(new Square(1) );
                        ^
          required: T
          found:    Square
          reason: inference variable T has incompatible bounds
            lower bounds: Cube
            lower bounds: Square
          where T is a type-variable:
            T extends Cube declared in method <T>print(T)
        1 error
        */
                }
        }
```

Source Code: Src/11_W/GenericMethod.java

Generic Methods Example 2:

```
        class Compare {
            public static <K, V> boolean compare(Fruit<K, V> p1, Fruit<K, V> p2) {
                return p1.getName().equals(p2.getName()) &&
                       p1.getPrice().equals(p2.getPrice());
            }
        }
        class Compare2<K, V> {
            public static void staticMethod()   { }
```

```
 9          public boolean compare(Fruit<K, V> p1, Fruit<K, V> p2) {
10              return p1.getName().equals(p2.getName()) &&
11                     p1.getPrice().equals(p2.getPrice());
12          }
13      }
14
15      class Fruit<K, V> {
16
17          private K fruitName;
18          private V price;
19
20          public Fruit(K fruitName, V price) {
21              this.fruitName = fruitName;
22              this.price = price;
23          }
24
25          public void setKey(K fruitName) { this.fruitName = fruitName; }
26          public void setValue(V price) { this.price = price; }
27          public K getName()   { return fruitName; }
28          public V getPrice() { return price; }
29      }
30
31      public class GenericMethod_2 {
32              public static void main(String[] args)  {
33                      Fruit<Double, String> p1 = new Fruit<Double, String>(1.23, "ap
34                      Fruit<Double, String> p2 = new Fruit<Double, String>(3.21, "pe
35                      boolean same1 = Compare.<Double, String>compare(p1, p2);
36                      boolean same2 = Compare.compare(p1, p2);
37                      boolean same3 = new Compare2<Double, String>().compare(p1, p2)
38              }
39      }
```

Source Code: Src/11_W/GenericMethod_2.java

8.10. Upper Bounded Type Parameters

• Upper Bound Type Parameters allow to restrict the types that can be used as type arguments

• `<T extends Class>`

A class that uses bound type parameters: Brick extends Cube extends Square extends Area

```
 1      import java.util.LinkedList;
 2      import java.util.List;
 3      public class UpperB<T extends Cube> {
 4
 5          T theThing;
 6
 7          public UpperB(T theThing)  {
 8              this.theThing = theThing;
 9          }
10          double area()       {
11              return theThing.area();
12          }
```

```
13          public static void main(String[] args)      {
14
15
16                  // UpperB<Square> aSquare = new UpperB<Square>(new Square(12));
17                  UpperB<Cube>   aCube   = new UpperB<Cube>(new Cube(12));
18                  UpperB<Brick>  aBrick  = new UpperB<Brick>(new Brick(12, 24, 36));
19                              // linked list op UpperB->Brick
20                  List<UpperB<Brick>> aList = new LinkedList<UpperB<Brick>>();
21                              // linked list op UpperB->Square
22                  List<UpperB<Cube>> bList = new LinkedList<UpperB<Cube>>();
23                  List<UpperB<? extends Cube>> cList = new LinkedList<UpperB<? exten
24                  aList.add( new UpperB<Brick>(new Brick(12, 24, 36) ) );
25                  bList.add( new UpperB<Cube>(new Cube(12) ) );
26                  // cList.add( new UpperB<Square>(new Square(12) ) );
27                  cList.add( new UpperB<Brick>(new Brick(12, 24, 36) ) );
28      /*
29                  cList.add( new UpperB<Square>(new Suare(12) ) );
30      UpperB.java:26: error: incompatible types: UpperB<Square> cannot be converted
31                  cList.add( new UpperB<Square>(new Suare(12) ) );
32                                 ^
33      */
34
35          }
36      }
```

Source Code: Src/11_W/UpperB.java

Compilation:

```
LowerB.java:14: error: type argument Square is not within bounds of type-variable T
          LowerB<Square> aSquare = new LowerB<Square>(new Square(12));
                   ^
  where T is a type-variable:
    T extends Cube declared in class LowerB
LowerB.java:14: error: type argument Square is not within bounds of type-variable T
          LowerB<Square> aSquare = new LowerB<Square>(new Square(12));
                                               ^
  where T is a type-variable:
    T extends Cube declared in class LowerB
2 errors

shell returned 1
```

8.11. Multiple Bounds

Q type parameter can have multiple bounds:

```
<T extends A & B & C>
```

A type variable with multiple bounds is a subtype of all the types listed in the bound. If one of the bounds is a class, it must be specified first.

Example:

```
class A { /* ... */ }
interface B { /* ... */ }
interface C { /* ... */ }

class D <T extends A & B & C> { /* ... */ }
```

If bound A is not specified first, you get a compile-time error:

class D <T extends B & A & C> { /* ... */ } // compile-time error

```
<T extends B & C & D>
```

```
 1      // adapted from http://docs.oracle.com/javase/specs/jls/se8/html/jls-4.html#jl
 2
 3      class A { }
 4      interface B { }
 5      interface C { }
 6
 7      class D<T extends A & B & C> {}
```

Source Code: Src/11_W/D.java

An other example:

```
 1      // adapted from http://docs.oracle.com/javase/specs/jls/se8/html/jls-4.html#jl
 2
 3      class C {
 4          public    void c1()    {
 5                      System.out.println("C:c1");
 6          }
 7          private   void c2()   {}
 8      }
 9      interface I {
10          void i1();
11      }
12      class B extends C implements I {
13          public    void i1()    {
14                      System.out.println("B:i1");
15          }
16          private   void c2()   {}
17      }
18
19
20      class MultipleB2<T extends C & I> {
21          void test(T t) {
22              t.c1();         // OK
23              t.i1();         // OK
24
25
26      /*
27              t.c2();         // Compile-time error
```

```
28              t.c2();    // Compile-time error
29               ^
30        symbol:   method c2()
31        location: variable t of type T
32        where T is a type-variable:
33          T extends C,I declared in class MultipleB2
34      1 error
35      */
36          }
37
38          public static void main(String[] args )     {
39              new MultipleB2<B>().test(new B() );
40          }
41      }
```

Source Code: Src/11_W/MultipleB2.java

An other example:

```
 1      // adapted from http://docs.oracle.com/javase/specs/jls/se8/html/jls-4.html#jl
 2
 3      class C {
 4          public    void c1()    {
 5                      System.out.println("C:c1");
 6          }
 7          private   void c2()   {}
 8      }
 9      interface I {
10          void i1();
11      }
12      class B extends C implements I {
13          public    void i1()    {
14                      System.out.println("B:i1");
15          }
16          private   void c2()   {}
17      }
18
19
20      class MultipleB<T extends C & I> {
21      /*
22          static void testOne(T t) {
23              t.c1();           // OK
24              t.i1();     // OK
25              // t.mCPrivate();    // Compile-time error
26          }
27      error: non-static type variable T cannot be referenced from a static context
28          static void testOne(T t) {
29      */
30          static <T extends C & I> void test(T t) {
31              t.c1();           // OK
32              t.i1();     // OK
33              // t.mCPrivate();    // Compile-time error
34          }
35
```

```
36          public static void main(String[] args )     {
37                      test(new B() );
38          }
39      }
```

Source Code: Src/11_W/MultipleB.java

8.12. Generic Arrays

Java's arrays contain, at runtime, information about its component type. Since you don't know what T is at runtime, you can't create the array.

There for you can not create arrays of parameterized types

```
 1      import java.lang.reflect.Array;
 2
 3      public class MyVector1<T> {
 4              T[] data = null;
 5              public MyVector1( int size ) {
 6                      // data = (T[])new Object[size];
 7                      data = (T[])getArray( new Object().getClass(), size );
 8
 9              }
10              public T get( int i ) {
11                      return data[i];
12              }
13              public void set( int i, T val ) {
14                      data[i] = val;
15              }
16              public <T> T[] getArray(Class<T> aClass, int size) {
17                      @SuppressWarnings("unchecked")
18                              T[] arr = (T[]) Array.newInstance(aClass, size);
19
20                  return arr;
21              }
22              public <T> T[] getArray(int size) {
23                      @SuppressWarnings("unchecked")
24                              T[] arr = (T[]) Array.newInstance(new Object().getClas
25                  return arr;
26              }
27              public static void main(String args[]) {
28                      MyVector1<String> aMyVector1 = new MyVector1<String>(11);
29                      aMyVector1.set(0, "a");
30                      System.out.println("aMyVector1.get(0): " + aMyVector1.get(0));
31
32              }
33      }
34
35      /*
36      MyVector1.java:7: warning: [unchecked] unchecked cast
37                      data = (T[])getArray( new Object().getClass(), size );
38                                              ^
39        required: T[]
40        found:    CAP#1[]
```

```
41        where T is a type-variable:
42          T extends Object declared in class MyVector1
43        where CAP#1 is a fresh type-variable:
44          CAP#1 extends Object from capture of ? extends Object
45      1 warning
46
47      */
```

Source Code: Src/11_W/MyVector1.java

Next

```
% javac MyVector.java && java `echo MyVector.java > .x; sed -e 's/.java//' .x`
MyVector.java:4: generic array creation
                data = new T[size];
                       ^
1 error



% javac -Xlint MyVector1.java
yVector1.java:6: warning: [unchecked] unchecked cast
found   : java.lang.Object[]
required: E[]
             data = (E[])new Object[size];
                         ^
MyVector1.java:7: warning: [unchecked] unchecked cast
found   : java.lang.Object[]
required: E[]
             data = (E[])getArray( new Object().getClass(), size );
                                 ^
MyVector1.java:24: warning: [unchecked] unchecked call to set(int,E) as a member of th
             aMyVector1.set(0, "a");
                           ^
3 warnings
% java MyVector1
aMyVector1.get(0): a
```

8.13. Generic Arrays II

Corrected Version 2:

```
 1
 2      import java.util.Arrays;
 3
 4      class MyArray<E> {
 5          private final Object[] objArray;   //object array
 6          public final int length;
 7
 8          public MyArray(int length)    {
 9              objArray = new Object [length];
10              this.length = length;
11          }
```

```
12          E get(int i) {
13              @SuppressWarnings("unchecked")
14              final E e = (E)objArray[i];
15              return e;
16          }
17          void set(int i, E e) {
18              objArray[i] = e;
19          }
20          public String toString() {
21              return Arrays.toString(objArray);
22          }
23          public static void main(String[] args){
24              final int length = 5;
25              MyArray<Integer>intArray = new MyArray<Integer>(length);
26              for (int i = 0; i < length; i++)
27                  intArray.set(i, i * 2);
28              System.out.println(intArray);
29          }
30      }
31
```

Source Code: Src/11_W/MyArray.java

Compilation and Output:

```
% javac -Xlint MyArray.java
% java MyArray
```

8.14. Restrictions on Generics

Stolen from: https://docs.oracle.com/javase/tutorial/java/generics/restrictions.html (https://docs.oracle.com/javase/tutorial/java/generics/restrictions.html)

• Cannot Instantiate Generic Types with Primitive Types

• Cannot Create Instances of Type Parameters

• Cannot Declare Static Fields Whose Types are Type Parameters

• Cannot Use Casts or instanceof With Parameterized Types

• Cannot Create Arrays of Parameterized Types

• Cannot Create, Catch, or Throw Objects of Parameterized Types

• Cannot Overload a Method Where the Formal Parameter Types of Each Overload Erase to the Same Raw Type

8.15. Wildcards

https://docs.oracle.com/javase/tutorial/java/generics/wildcardGuidelines.html (https://docs.oracle.com/javase/tutorial/java/generics/wildcardGuidelines.html)

• In generic code, the question mark (?), called the wildcard, represents an unknown type.

• The wildcard can be used in a variety of situations:

— as the type of a parameter, field, or local variable;

— sometimes as a return type (though it is better programming practice to be more specific).

— The wildcard is never used as a type argument for a generic method invocation, a generic class instance creation, or a supertype.

8.16. Upper Bounded Wildcards

Stolen from above: https://docs.oracle.com/javase/tutorial/java/generics/wildcardGuidelines.html (https://docs.oracle.com/javase/tutorial/java/generics/wildcardGuidelines.html)

Example: public static void process(List<? extends Number>)

All objects of classes which extend Number

• List<Integer>, List<Double>, and List<Number>;

• Integer|Double extends Number

Upper Bound Wildcard: "? extends E": Denotes a family of subtypes of type Type. This is the most useful wildcard
(bounded by its super class E) A method that works on lists of Vector and the subtype of Vector, such as Stack, *List<extends Vector>*
A method that works on lists of Integer and the super types of Integer, such as Integer, Number, and Object: *List<? super Integer>*

```
 1      import java.util.*;
 2
 3      public class UpperBound {
 4          public static void main(String[] args) {
 5              List<Double>  listOfDoubles   = new LinkedList<Double>();
 6              List<Integer> listOfIntegers  = new LinkedList<Integer>();
 7
 8              listOfDoubles.add( Double.valueOf(1.0) );
 9              listOfIntegers.add( Integer.valueOf(2) );
10
11              System.out.println("sum of integer's is: " + sum(listOfIntegers));
12              System.out.println("sum of double's is:  "  + sum(listOfDoubles));
13          }
14
15          // instead of double?
16          private static double sum(List<? extends Number> list)  {
17              double sum=0.0;
18              for (Number i: list)
19                  sum += i.doubleValue();
20              return sum;
21          }
22      }
```

Source Code: Src/11_W/UpperBound.java

Next

```
% java UpperBound
sum of integer's is: 2.0
sum of double's is:  1.0
```

8.17. Unbounded Wildcards

Unbound: "?": Denotes the set of all types or any

Example: public static void printList(List<?> list)

Can print a list of any type

Useful approach:

- a method that can be implemented using functionality provided in the Object class.
- code is using methods in the generic class that don't depend on the type parameter.
- for ex: List.size() or List.clear().
- Class<?> is so often used because most of the methods in Class<T> do not depend on T.

```
 1      import java.util.*;
 2
 3      public class Unbound {
 4
 5              public static void printList(List<?> list) {
 6                      for (Object elem: list)
 7                              System.out.print(elem + " ");
 8                      System.out.println();
 9              }
10
11              public static void main(String args[]) {
12                      List<Integer> listOfIntegers = Arrays.asList(1, 2, 3);
13                      List<String>  listOfStrings = Arrays.asList("a", "b", "c");
14                      printList(listOfIntegers);
15                      printList(listOfStrings);
16              }
17      }
18
```

Source Code: Src/11_W/Unbound.java

Output:

```
% java Unbound
1 2 3
a b c
```

8.18. Lower Bounded Wildcards

A lower bounded wildcard is expressed using the wildcard character ('?'), following by the super keyword, followed by its lower bound:

Brick extends Cube extends Square extends Area

a method which can only pring cubes and brick, but not squares

```
 1      import java.util.LinkedList;
 2      import java.util.List;
 3      import java.util.Date;
 4      import java.sql.Time;
 5
 6      class LowerBound {
 7          public static void main(String[] args) {
 8              List<Cube>  listOfCube = new LinkedList<Cube>();
 9              List<Brick>   listOfBrick  = new LinkedList<Brick>();
10              List<Square>   listOfSquare  = new LinkedList<Square>();
11
12              listOfCube.add( new Cube(1) );
13              listOfBrick.add( new Brick(1, 2, 3) );
14              listOfSquare.add( new Square(1) );
15      /*
16              printOnlyUpToIntegerClass(listOfBrick);
17      LowerBound.java:16: error: incompatible types: List<Brick> cannot be converted
18              printOnlyUpToIntegerClass(listOfBrick);
19                                        ^
20      Note: Some messages have been simplified; recompile with -Xdiags:verbose to ge
21      1 error
22      */
23              printOnlyUpToIntegerClass(listOfCube);
24              printOnlyUpToIntegerClass(listOfSquare);
25          }
26
27          public static void printOnlyUpToIntegerClass(List<? super Cube> list) {
28              System.out.println(list);
29          }
30      }
```

Source Code: Src/11_W/LowerBound.java

Date() (https://docs.oracle.com/javase/8/docs/api/java/util/Date.html)

```
% java LowerBound
[05:17:36]
[Sun Sep 20 08:51:17 EDT 2020]
```

8.19. Summary

- Upper bound: public static void process(List<? extends Number>)
  Number class and sub classes of number, Integer, Double
- Unbound: public static void printList(List<?> list) -
  List of all types
- Lower Bounded public static void addNumbers(List<? super Integer> list)
  Integer and super classes of Integer, Integer, Numbers, Object

8.20. Bound Types vs Wildcards

• Bounded type parameters can have multiple bounds

• Wildcards can have a lower or upper bound. There is no lower bound for type parameters

• Bounded type parameters allow to express dependencies among the types of one or more arguments in a method and/or the return type

8.21. Wildcard Example

```
 1      import java.util.*;
 2
 3      public class WildCard {
 4
 5          public static void printCollection_2(Collection<?> c) {
 6              for (Object e : c)
 7                      System.out.println("2: " + e);
 8          }
 9          public static void printCollection(Collection c) {
10              Iterator i = c.iterator();
11              while ( i.hasNext() ) {
12                      System.out.println("1: "  + i.next());
13              }
14          }
15          public static void main(String args[]) {
16              String anArray[] = {"echoes", "Shine", "Tiger" } ;
17              List l = Arrays.asList(anArray);
18              printCollection(l);
19              printCollection_2(l);
20          }
21      }
22
```

Source Code: Src/11_W/WildCard.java

8.22. Bound Wildcards

•

```
public abstract class Shape {
        public abstract void draw(Canvas c);
}
public class Circle extends Shape {
        private int x, y, radius;
        public void draw(Canvas c) { ... } }
public class Rectangle extends Shape {
        private int x, y, width, height;
        public void draw(Canvas c) { ... } }
```

• These classes can be drawn on a canvas:

```
public class Canvas { public void draw(Shape s) {
        s.draw(this);
   }
}
```

• Any drawing will typically contain a number of shapes.

• Assuming that they are represented as a list, it would be convenient to have a method in Canvas that draws them all:

```
public void drawAll(List<Shape> shapes) {
        for (Shape s: shapes) {
                s.draw(this);
}
```

• What we really want is:

```
public void drawAll(List<? extends Shape> shapes) { ... }
```

• *class C<T extends Figure> {..}*

— T must descend from Figure.

— Allows Figure's features to be referenced.

— Keyword extends is even used for interfaces!

• Example:

```
public void drawAll(List<? extends Shape> shapes) { ... }
```

• List<? extends Shape> is an example of a bounded wildcard

• However, in this case, we know that this unknown type is in fact a subtype of Shape 1. Shape is the upper bound of the wildcard.

• This defines upper bounds

• The type Collection<? super String> is a super type of any Collection where T is a super type of String

• This collection can store Strings and Objects

8.23. Model View Controller

• The MVC paradigm is a way of breaking an application, into three parts:

— the model,

— the view, and the

— controller.

• Input → Processing → Output

• Controller → Model → View

• Picture:

• Model: encapsulate the logic

• View: I/O, view

• Controller: command center

How does the communication work?

• method calls

• events (move: model, operations, view, events)

Persistance

Controller

+ easy to test
+ makes it easier to re-use the mode
- typically more complex

Model - Active Record Pattern

+ easy
- harder to test
- reusing may be negatively impacted
- obviously more complex

8.24. View

View of the system

multiple views are possible

8.25. Observer - Observable Model

We will come back later - java.util.concurrent

• Simply, the Observer pattern allows one object (the observer) to watch another (the subject).

• The Java programming language provides support for the Model/View/Controller architecture with two classes:

• Observer (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/Observer.html) — object that wishes to be notified when the state of another object changes

• Observable (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/Observable.html) — any object whose state may be of interest, and in whom another object may register an interest

8.26. Observer

• Interface

`void update(Observable o, Object arg)`

This method is called whenever the observed object is changed. An application calls an Observable object's notifyObservers method to have all the object's observers notified of the change.

8.27. Observable

From api spec:

• This class represents an observable object, or "data" in the model-view paradigm. It can be subclassed to represent an object that the application wants to have observed.

An observable object can have one or more observers. An observer may be any object that implements interface Observer. After an observable instance changes, an application calling the Observable's notifyObservers method causes all of its observers to be notified of the change by a call to their update method.

The order in which notifications will be delivered is unspecified. The default implementation provided in the Observable class will notify Observers in the order in which they registered interest, but subclasses may change this order, use no guaranteed order, deliver notifications on separate threads, or may guarantee that their subclass follows this order, as they choose.

Note that this notification mechanism is has nothing to do with threads and is completely separate from the wait and notify mechanism of class Object.

Idea:

8.28. Example

```
1
2
3
4       import java.util.Observer;
5       import java.util.Observable;
6
7       class TheObserver implements Observer {
8
9               public TheObserver()    {
```

```
10              }
11
12              public void update( Observable aObservable, Object o ) {
13                      System.out.println("    " + this + ":  TheObserver!update - o
14              }
15      }
16
17      class TheObservable extends Observable {
18
19              String aString;
20
21              public TheObservable()  {
22                      System.out.println("    " + this + ": TheObserver!TheObservabl
23              }
24
25              public void exectueSetChanged() {
26                      setChanged();
27              }
28              public void setName(String aString) {
29                      this.aString = aString;
30                      notifyObservers();
31              }
32      }
33
34      public class Test {
35
36              public static void main(String args[]) {
37                      TheObservable aObservable = new TheObservable();
38                      TheObserver  aTheObserver = new TheObserver();
39
40                      System.out.println("aObservable = " + aObservable );
41                      aObservable.addObserver(new TheObserver());
42                      aObservable.addObserver(new TheObserver());
43                      aObservable.addObserver(new TheObserver());
44                      System.out.println("setName(you)");
45                      aObservable.setName("you");
46                      System.out.println("aObservable.exectueSetChanged()");
47                      System.out.println("setName(me)");
48                      aObservable.exectueSetChanged();
49                      aObservable.setName("me");
50              }
51      }
```

Source Code: Src/13/Test.java

Next

```
       TheObservable@2ac1fdc4: TheObserver!TheObservable
setName(you)
s.exectueSetChanged()
setName(me)
       TheObserver@50cbc42f:  TheObserver!update
       TheObserver@75412c2f:  TheObserver!update
       TheObserver@282ba1e:  TheObserver!update
```

Other example

Observer:

```
 1
 2      import java.util.Observer;
 3      import java.util.Observable;
 4
 5      public class TheObserver implements Observer {
 6
 7              int id = 0;
 8              static int counter = 0;
 9
10              public TheObserver()    {
11                      id = counter ++;
12              }
13
14              public void update( Observable aObservable, Object o ) {
15                      if ( o instanceof String )      {
16                              System.out.println("TheObserver:update: a String objec
17                              System.out.println("TheObserver:update:o " + o );
18                      } else {
19                              System.out.println("TheObserver:update: ! a String obj
20                              System.out.println("TheObserver:update:o " + o );
21                      }
22              }
23      }
24
```

Source Code: Src/13/TheObserver.java

Observable:

```
 1
 2      import java.util.Observer;
 3      import java.util.Observable;
 4
 5      public class TheObservable extends Observable {
 6
 7              String name = null;
 8              Integer value = null;
 9
10              public TheObservable(String name)       {
11                      this.name = name;
12              }
13
14              public void setName(String name) {
15                      this.name = name;
16                      setChanged();
17                      notifyObservers(name);
18              }
19              public void go(Integer value) {
20                      this.value = value;
21                      setChanged();
```

```
22                      notifyObservers(value);
23              }
24      }
```

Source Code: Src/13/TheObservable.java

Main method

```
 1      public class Main {
 2
 3              public static void main(String args[]) {
 4                      TheObservable s = new TheObservable("Dr Hook and the Medicine
 5                      TheObserver  aTheObserver = new TheObserver();
 6
 7                      s.addObserver(aTheObserver);
 8                      s.setName("you");
 9                      s.go(1234);
10              }
11      }
```

Source Code: Src/13/Main.java

Output:

```
% java Main
TheObserver:update: a String object came in
TheObserver:update:o you
TheObserver:update: ! a String object came in
TheObserver:update:o 1234
```

9. Exceptions and Assertions

9.1. Exceptions

See also: Exceptions (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/lang/Exception.html)

When a Java program violates the semantic constraints of the Java language, a Java Virtual Machine signals this error to the program as an exception. An example of such a violation is an attempt to index outside the bounds of an array.

Java specifies that an exception will be thrown when semantic constraints are violated and will cause a non-local transfer of control from the point where the exception occurred to a point that can be specified by the programmer. An exception is said to be thrown from the point where it occurred and is said to be caught at the point to which control is transferred.

Java programs can also throw exceptions explicitly, using throw statement.

The Java language checks, at compile time, that a Java program contains handlers for checked exceptions, by analyzing which checked exceptions can result from execution of a method or constructor.

An Error indicates are serious, most likely not recoverable, situation.

9.2. Runtime Exceptions

• Runtime Exceptions (https://docs.oracle.com/javase/7/docs/api/java/lang/RuntimeException.html)

• unchecked exceptions classes are the class RuntimeException and its subclasses

• class Error and its subclasses

Example:

```
 1      public class RunTime {
 2
 3        private void divide(int a, int b)     {
 4              int result = a / b;
 5        }
 6        private void callDivider(int a, int b)        {
 7              divide(a, b);
 8        }
 9
10        public static void main(String[] args) {
11              int a = (int) Math.random();
12              int b = 1;
13              new RunTime().callDivider(a, b);
14              new RunTime().callDivider(b, a);
15        }
16      }
```

Source Code: Src/7/RunTime.java

9.3. Compile Time Exceptions

• All others

• The list is long

Example:

```
 1      import java.io.FileNotFoundException;
 2      import java.util.Scanner;
 3      import java.io.File;
 4
 5      public class CompileTime {
 6
 7        private void openFile (String filenName ) throws FileNotFoundException {
 8              new Scanner(new File(filenName) );
 9        }
10        private void callOpenFile()   {
11              try {
12                      openFile("thisFileDoesNotExist.txt");
13              } catch ( FileNotFoundException e  )    {
14                      System.out.println("getCasue: " + e.getCause());
15                      e.printStackTrace();
16              }
17        }
18
19        public static void main(String[] args) {
```

```
20              new CompileTime().callOpenFile();
21          }
22      }
```

Source Code: Src/7/CompileTime.java

9.4. Runtime Exceptions are Not Checked

The runtime exception classes (RuntimeException and its subclasses) are exempted from compile-time checking because, in the judgment of the designers of Java, having to declare such exceptions would not aid significantly in establishing the correctness of Java programs. Many of the operations and constructs of the Java language can result in runtime exceptions. The information available to a Java compiler, and the level of analysis the compiler performs, are usually not sufficient to establish that such runtime exceptions cannot occur, even though this may be obvious to the Java programmer. Requiring such exception classes to be declared would simply be an irritation to Java programmers.

For example, certain code might implement a circular data structure that, by construction, can never involve null references; the programmer can then be certain that a NullPointerException cannot occur, but it would be difficult for a compiler to prove it. The theorem-proving technology that is needed to establish such global properties of data structures is beyond the scope of this Java Language Specification.

9.5. Runtime Exceptions--The Controversy

Copied from: read here ... (https://docs.oracle.com/javase/tutorial/essential/exceptions/runtime.html)

• Although Java requires that methods catch or specify checked exceptions, they do not have to catch or specify runtime exceptions, that is, exceptions that occur within the Java runtime system.

• Because catching or specifying an exception is extra work, programmers may be tempted to write code that throws only runtime exceptions and therefore doesn't have to catch or specify them.

• This is "exception abuse" and is not recommended.

9.6. Throwable and Error

Please see: http://www.cs.rit.edu/usr/local/jdk/docs/api/index.html (http://www.cs.rit.edu/usr/local/jdk/docs/api/index.html)

• An Error is a subclass of Throwable that indicates serious problems that a reasonable application should not try to catch.

• Most such errors are abnormal conditions.

• An Error indicates typically it is not recoverable

• The ThreadDeath error, though a "normal" condition, is also a subclass of Error because most applications should not try to catch it.

```
 1      public class ErrorE {
 2
 3        private void thisMethodThrowsAnE(int index) throws Exception, Error {
 4
 5              if ( index == 0 )       {
 6                      System.out.println("thisMethodThrowsAnException() ---> " );
 7                      throw new Exception("in thisMethodThrowsAnException");
 8              } else {
 9                      System.out.println("thisMethodThrowsAnError() ---> " );
10                      throw new Error("in thisMethodThrowsAnException");
11              }
12
13        }
14
15        private void caller() {
16              for ( int index = 0; index < 2; index ++ )      {
17                      try {
18                              thisMethodThrowsAnE(index);
19                      } catch (Exception e)   {
20                              e.printStackTrace();
21                      } catch (Error e)       {
22                              e.printStackTrace();
23                      } finally       {
24                              System.out.println("Finally");
25                              System.out.println("Ok, a few things to clean up" );
26                      }
27              }
28        }
29
30        public static void main(String[] args) {
31              new ErrorE().caller();
32        }
```

```
33      }
```

Source Code: Src/7/ErrorE.java

Output:

```
java ErrorE
thisMethodThrowsAnException() --->
java.lang.Exception: in thisMethodThrowsAnException
        at ErrorE.thisMethodThrowsAnE(ErrorE.java:7)
        at ErrorE.caller(ErrorE.java:18)
        at ErrorE.main(ErrorE.java:31)
Finally
Ok, a few things to clean up
thisMethodThrowsAnError() --->
java.lang.Error: in thisMethodThrowsAnException
        at ErrorE.thisMethodThrowsAnE(ErrorE.java:10)
        at ErrorE.caller(ErrorE.java:18)
        at ErrorE.main(ErrorE.java:31)
Finally
Ok, a few things to clean up
```

9.7. Try

When an exception is thrown, control is transferred from the code that caused the exception to the nearest dynamically-enclosing catch clause of a try statement that handles the exception.

Syntax:

```
try
{
    statement sequence
}
```

The exception can be thrown and caught in the same try block if necessary.

```
try
{
    f();
}
catch (ExceptionType1 e1 )
       throw e1;
}
catch (ExceptionType1 e1 )
       throw e2;
}
```

9.8. Catch

It can be followed by zero or more catch blocks:

```
catch ( parameter )
{
    statement sequence
}
```

Example:

```
try {
    anObject.f();
    anObject.g();
} catch (SomeExeption_1 e) {
    // do something to recover
}
catch (SomeExeption_2 e) {
    // do something to recover
}
```

9.9. Finally

The finally block will be always executed, regardless of what happens in the try block. (with the exception of system exit)

This provides a place where you can put statements which will be always executed.

```
finally ( )
{
    statement sequence
}
```

```
 1
 2      public class Finally_1 {
 3
 4        private int noSystemExit()     {
 5                       try {
 6                               throw new Exception("1");
 7                       } catch (Exception e)   {
 8                               System.out.println("2");
 9                               return 0;
10                       } finally       {
11                               System.out.println("3 finally");
12                               // return 1;
13                       }
14              System.out.println("xx");
15              return 4;
16        }
17        private void withSystemExit()  {
18                       try {
19                               throw new Exception("4");
20                       } catch (Exception e)   {
21                               System.out.println("5");
22                               System.exit(0);
23                       } finally       {
24                               System.out.println("6 finally");
25                       }
26                       System.out.println("exit(): you will not see this line");
27        }
28
29        public static void main(String[] args) {
30              new Finally_1().noSystemExit();
31              new Finally_1().withSystemExit();
32        }
33      }
```

Source Code: Src/7/Finally_1.java

9.10. Finally

What is the output of the follwing program?

```
 1
 2
 3      public class Finally_4 {
 4
 5       public int tryCatchFinally() {
 6              int i = 0;
 7              try {
 8                      i = 1;
 9                      System.out.println("a: " + i );
10                      try {
11                              i = 2;
12                              System.out.println("b: " + i );
13                              return i;               // this will be the return val
14                      } catch (Exception e) {
15                              e.printStackTrace();
16                      } finally {
17                              System.out.println("c: " + i );
18                              ++i;
19                      }
20                      System.out.println("d: " + i );
21                      return i;
22              } catch (Exception e) {
23                      e.printStackTrace();
24              } finally {
25                      System.out.println("e: " + i );
26                      ++i;
27                      System.out.println("f: " + i );
28              }
29              return i;
30              }
31        public static void main(String[] args) {
32              System.out.println("Calling new Finally_4().tryCatchFinally() " + new
33         }
34      }
```

Source Code: Src/7/Finally_4.java

What is the output of the follwing program?

```
 1
 2
 3
 4      public class Finally_5 {
 5
 6       public void tryCatchFinally() {
 7              int[] intArray = {1};
 8              try {
 9                      try {
10                              try {
11                                      System.out.println("" + intArray[1]);
12                                      System.out.println("Exception 0 ");
13                              } catch(Exception e) {
14                                              System.out.println("Exception 1 ");
15                              }
```

```
16                      } catch(Exception e){
17                              System.out.println("" + intArray[1]);
18                              System.out.println("Exception 2 ");
19                      }
20              } catch(Exception e) {
21                      System.out.println("Exception 3 ");
22              } finally {
23                      System.out.println("finally 2 ");
24              }
25
26
27
28        }
29        public static void main(String[] args) {
30              new Finally_5().tryCatchFinally();
31        }
32      }
```

Source Code: Src/7/Finally_5.java

9.11. Try-with-resources Statement

Problem:

```
 // http://docs.oracle.com/javase/tutorial/essential/exceptions/tryResourceClose.html
static String readFirstLineFromFileWithFinallyBlock(String path) throws IOException {
    BufferedReader br = new BufferedReader(new FileReader(path));
    try {
        return br.readLine();
    } finally {
        if (br != null) br.close();
    }
}
```

• Resources like streams must be closed under all circumstances.

• Try-finaly can be used to do so.

• Try-with-resources statement is the way to do it.

• For example - BufferedReader, in Java SE 7 and later, implements the interface java.lang.Auto-Closeable.

Example:

```
 1      // first line
 2      import java.io.*;
 3      public class TryWithResource {
 4
 5
 6      static void readAndPrint(String inF ) throws IOException  {
 7
 8              try (
 9                      BufferedReader in = new BufferedReader( new FileReader(inF) );
10                  ) {
11                              System.out.println(in.readLine() );
```

```
12              } catch (Exception e) {
13                      System.out.println("Could not open file");
14                      e.printStackTrace();
15              }
16        }
17
18        public static void main(String args[]) {
19              if ( args.length != 1 )  {
20                      System.out.println("Usage: java FileIO file");
21              } else  {
22                      System.out.println("Inputfile: " + args[0]);
23                      try {
24                              readAndPrint(args[0]);
25                      } catch (Exception e) {
26                              e.printStackTrace();
27                      }
28              }
29        }
30      }
```

Source Code: Src/7/TryWithResource.java

Example with finally:

```
 1      // first line
 2      import java.io.*;
 3      public class TryWithOutResourceAndFinally {
 4
 5
 6      static void readAndPrint(String inF ) throws IOException  {
 7              BufferedReader in = null;
 8
 9              try     {
10                      in = new BufferedReader( new FileReader(inF) );
11                      System.out.println(in.readLine() );
12              } catch (Exception e) {
13                      System.out.println("Could not open file");
14                      e.printStackTrace();
15              } finally {
16                      if ( in != null )
17                              in.close();
18              }
19        }
20
21        public static void main(String args[]) {
22              if ( args.length != 1 )  {
23                      System.out.println("Usage: java FileIO file");
24              } else  {
25                      System.out.println("Inputfile: " + args[0]);
26                      try {
27                              readAndPrint(args[0]);
28                      } catch (Exception e) {
29                              e.printStackTrace();
30                      }
```

```
31                }
32          }
33       }
```

Source Code: Src/7/TryWithOutResourceAndFinally.java

9.12. Throw

A throw statement allows an exception to be thrown.

Syntax:

```
throw typeThrowableException;
```

Example:

```
throw new Exception("Nope, this was not too good!");
```

9.13. Exceptions are Precise

Exceptions in Java are precise: when the transfer of control takes place, all effects of the statements executed and expressions evaluated before the point from which the exception is thrown must appear to have taken place.

No expressions, statements, or parts thereof that occur after the point from which the exception is thrown may appear to have been evaluated.

If optimized code has speculatively executed some of the expressions or statements which follow the point at which the exception occurs, such code must be prepared to hide this speculative execution from the user-visible state of the Java program.

9.14. Handling Asynchronous Exceptions

• Most exceptions occur synchronously as a result of an action by the thread in which they occur

• An asynchronous exception is an exception that can potentially occur at any point in the execution of a program.

• Asynchronous exceptions are rare. They occur only as a result of:

— An invocation of the *stop()* (Thread or ThreadGroup)

— An internal error in the Java virtual machine

9.15. Example 1

```
 1      /**
 2       * This class plays with exceptions
 3       *
 4       * @version   $Id$
 5       *
 6       * @author    hp bischof
 7       *
 8       * Revisions:
 9       *      $Log$
10       */
11
12      public class Excep_1 {
13
14        private int convert(String s) {
15              int result = 0;
16
17              try {
18                      result = Integer.parseInt(s);
19              } catch ( NumberFormatException e ) {
20                      System.out.println("Haeh? " + e );
21                      e.printStackTrace();
22              }
23              return result;
24        }
25
26        public static void main(String[] args) {
27              new Excep_1().convert("42");
28              new Excep_1().convert("opa");
29        }
30      }
```

Source Code: Src/7/Excep_1.java

Result:

```
% java Excep_1
Haeh? java/lang/NumberFormatException: opa
java/lang/NumberFormatException: opa
        at java/lang/Integer.parseInt(Integer.java)
        at java/lang/Integer.parseInt(Integer.java)
        at Excep_1.convert(Excep_1.java:18)
        at Excep_1.main(Excep_1.java:28)
```

9.16. Example 2

```
 1      /**
 2       * This class plays with exceptions
 3       *
 4       * @version   $Id$
 5       *
 6       * @author    hp bischof
 7       *
 8       * Revisions:
 9       *      $Log$
10       */
11
12      public class Excep_2 {
13
14        private void f(int n) throws NullPointerException,
15                                     InterruptedException {
16              System.out.println("f(" + n + ")" );
17              switch (n)      {
18                      case 1:  throw new NullPointerException("1");
19                               // break; why is it not required
20                      default: throw new InterruptedException("default");
21              }
22        }
23
24        public static void main(String[] args) {
25              for (int index = 1; index < 3; index ++ )       {
26                      try {
27                              new Excep_2().f(index);
28                      } catch (NullPointerException e)        {
29                              e.printStackTrace();
30                      }
31
32                      catch (Exception e)     {
33                              System.out.println(e.getMessage() );
34                      }
35
36              }
37        }
38      }
```

Source Code: Src/7/Excep_2.java

Result:

```
% java Excep_2
f(1)
java/lang/NullPointerException: 1
        at Excep_2.f(Excep_2.java:17)
        at Excep_2.main(Excep_2.java:25)
f(2)
default
```

Typical compiler errors:

```
Exception java/lang/Exception must be caught,
or it must be declared in the throws clause of this method.
                        new Excep_2().f(3);
                                       ^
1 error
```

9.17. Example 3

```
 1      public class Excep_3 {
 2
 3        private void thisMethodThrowsAnException() throws Exception {
 4              throw new Exception("in thisMethodThrowsAnException");
 5              // System.out.println("thisMethodThrowsAnException() ---> " );
 6
 7              // System.out.println("thisMethodThrowsAnException() <--- " );
 8              //  ^
 9              // 1 error
10              // System.out.println("thisMethodThrowsAnException() <--- " );
11        }
12
13        private int  caller() {
14              try {
15                      new Excep_3().thisMethodThrowsAnException();
16                      return 0;
17              } catch (Exception e)   {
18                      e.printStackTrace();
19                      return 1;
20              } finally       {
21                      System.out.println("Finally");
22                      System.out.println("Ok, a few things to clean up" );
23                      return 2;
24              }
25        }
26
27        public static void main(String[] args) {
28              System.out.println(new Excep_3().caller());
29        }
30      }
```

Source Code: Src/7/Excep_3.java

Result:

```
% java Excep_3
thisMethodThrowsAnException() --->
java.lang.Exception: in thisMethodThrowsAnException
        at Excep_3.thisMethodThrowsAnException(Excep_3.java:5)
        at Excep_3.caller(Excep_3.java:18)
        at Excep_3.main(Excep_3.java:30)
Finally
Ok, a few things to clean up
2
```

9.18. Example 4

```
 1
 2      public class Finally_3 {
 3
 4        static int rValue = 0;
 5        private int caller()   {
 6                        try {
 7                                throw new ArithmeticException("in thisMethodThrowsAnEx
 8                        } catch (ArithmeticException e) {       // order is relevant
 9                                rValue = 2;
10                                System.out.println("caller: before return! rValue =  "
11                                return rValue;
12                                // rValue = 11;
13                        } catch (Exception e)   {
14                                rValue = 1;
15                                System.out.println("caller: before return! rValue =  "
16                                return rValue;
17                                // rValue = 11;
18                        } finally       {
19                                System.out.println("finaly: before return! rValue =  "
20                                rValue = 111;
21                                return rValue;
22                                // rValue = 1111;
23                        }
24        }
25        public static void main(String[] args) {
26              System.out.println("Calling new Finally_3().caller() " + new Finally_3
27        }
28      }
```

Source Code: Src/7/Finally_3.java

Result:

```
% java Finally_3
caller: before return! rValue =  2
finaly: before return! rValue =  2
Calling new Finally_3().caller() 111
```

9.19. Typical Compiler Error

```
Exception java/lang/Exception must be caught,
or it must be declared in the throws clause of this method.
                        new Excep_2().f(3);
                                       ^
1 error
```

9.20. Try Example

```
 1      /**
 2       * This class plays with exceptions
 3       *
 4       * @version   $Id$
 5       *
 6       * @author    hp bischof
 7       *
 8       * Revisions:
 9       *      $Log$
10       */
11
12      public class Try {
13
14        private void f(int n) throws Exception {
15              System.out.println("f(" + n + ")" );
16              switch (n)      {
17                      case 1:  throw new NullPointerException("1");
18                      default: throw new Exception("default");
19              }
20        }
21
22        public static void main(String[] args) {
23              int countExceptions = 0;
24              for (int index = 0; index < 3; index ++ )       {
25                      try {
26                              new Try().f(index);
27                      } catch (Exception e)   {
28                              e.printStackTrace();
29                      }
30
31                      finally  {
32                              countExceptions ++;
33                      }
34              }
35              System.out.println("Caught " + countExceptions +
36                                      " exceptions.");
37        }
38      }
```

Source Code: Src/7/Try.java

Result:

```
% java Try
f(0)
java/lang/Exception: default
        at Try.f(Try.java:18)
        at Try.main(Try.java:26)
f(1)
java/lang/NullPointerException: 1
        at Try.f(Try.java:17)
        at Try.main(Try.java:26)
f(2)
```

```
java/lang/Exception: default
        at Try.f(Try.java:18)
        at Try.main(Try.java:26)
Caught 3 exceptions.
```

9.21. Exceptions and Inheritance

```
1       // What is the exceution order?
2
3       public class ExceptionsAndInheritance1 {
4
5         public void importantFunction() throws InterruptedException {
6               System.out.println("ExceptionsAndInheritance1:importantFunction -->");
7               throw new InterruptedException("ExceptionsAndInheritance1.java");
8         }
9
10        public static void main(String[] args) {
11              try {
12                      new ExceptionsAndInheritance1().importantFunction();
13              } catch (Exception e)   {
14                      System.out.println("Main ");
15                      e.printStackTrace();
16              }
17        }
18      }
```

Source Code: Src/7/ExceptionsAndInheritance1.java

Next:

```
1       // What is the execution order?
2
3       public class ExceptionsAndInheritance2 extends
4               ExceptionsAndInheritance1 {
5
6         public void importantFunction() {
7               System.out.println("ExceptionsAndInheritance2:importantFunction -->");
8         }
9
10        public static void main(String[] args) {
11              ExceptionsAndInheritance2 e2 = new ExceptionsAndInheritance2();
12              ExceptionsAndInheritance1 e1Cast = (ExceptionsAndInheritance2)e2;
13              ExceptionsAndInheritance1 e1 = new ExceptionsAndInheritance1();
14
15              System.out.println("------------------------------");
16              e2.importantFunction();
17              System.out.println("------------------------------");
18              // e1Cast.importantFunction(); // will this line compile?
19              try {
20                      e1Cast.importantFunction();
21              } catch (Exception e)   {
22                      System.out.println("Main ");
23                      e.printStackTrace();
24              }
25              System.out.println("------------------------------");
26              try {
27                      e1.importantFunction();
28              } catch (Exception e)   {
```

```
29                      System.out.println("Main ");
30                      e.printStackTrace();
31              }
32              System.out.println("------------------------------");
33        }
34      }
```

Source Code: Src/7/ExceptionsAndInheritance2.java

Will it compile? explain your answer.

9.22. Reading From Files

```
 1      import java.io.*;
 2      public class FileIO {
 3
 4        static void cp(String inF, String outF )      {
 5              BufferedReader in  = null;
 6              BufferedWriter out = null;;
 7
 8              try {
 9                      in = new BufferedReader(
10                              new FileReader(inF) );
11                      out = new BufferedWriter(
12                              new FileWriter(outF) );
13                      int oneInt;
14                      while ( ( oneInt = in.read() ) >= 0 )   {
15                              out.write(oneInt);
16                      }
17              }
18              catch ( FileNotFoundException e )       {
19                      e.printStackTrace();
20                      System.out.println("Can't find the file!");
21              }
22              catch ( IOException e ) {  // Throws: IOException !!!
23                      e.printStackTrace();
24                      System.out.println("Could not be opened for writing!");
25              }
26              catch ( Exception e )   {
27                      System.out.println("Can't find the file!");
28              }
29              finally {
30                      try {
31                              if ( in != null )
32                                      in.close();
33                              if ( out != null )
34                                      out.close();
35                      } catch ( IOException e ) {}
36              }
37        }
38
39        public static void main(String args[]) {
40              if ( args.length != 2 )
41                      System.out.println("Usage: java FileIO f1 f2");
42              else    {
43                      System.out.println(args[0] + "  " + args[1] );
44                      cp(args[0], args[1]);
45              }
46        }
47      }
```

Source Code: Src/7/FileIO.java

With try-with-resources

```
 1      import java.io.*;
 2      public class FileIO_withTR {
 3
 4        static void cp(String inF, String outF )      {
 5
 6              try (
 7                      BufferedReader in = new BufferedReader(
 8                              new FileReader(inF) );
 9                      BufferedWriter out = new BufferedWriter(
10                              new FileWriter(outF) );
11              )       {
12                      int oneInt;
13                      while ( ( oneInt = in.read() ) >= 0 )   {
14                              out.write(oneInt);
15                      }
16              }
17              catch ( FileNotFoundException e )       {
18                      e.printStackTrace();
19                      System.out.println("Can't find the file!");
20              }
21              catch ( IOException e ) {  // Throws: IOException !!!
22                      e.printStackTrace();
23                      System.out.println("Could not be opened for writing!");
24              }
25              catch ( Exception e )   {
26                      System.out.println("Can't find the file!");
27              }
28        }
29
30        public static void main(String args[]) {
31              if ( args.length != 2 )
32                      System.out.println("Usage: java FileIO_withTR f1 f2");
33              else    {
34                      System.out.println(args[0] + "  " + args[1] );
35                      cp(args[0], args[1]);
36              }
37        }
38      }
```

Source Code: Src/7/FileIO_withTR.java

See also AutoCloseable. (https://docs.oracle.com/javase/8/docs/api/java/lang/AutoCloseable.html)

9.23. Incomplete Exception Index

- ArithmeticException
- ArrayIndexOutOfBoundsException
- ArrayStoreException
- ClassCastException
- ClassNotFoundException
- CloneNotSupportedException

- Exception
- IllegalAccessException
- IllegalArgumentException
- IllegalMonitorStateException
- IllegalStateException
- IllegalThreadStateException
- IndexOutOfBoundsException
- InstantiationException
- InterruptedException
- NegativeArraySizeException
- NoSuchFieldException
- NoSuchMethodException
- NullPointerException
- NumberFormatException
- RuntimeException
- SecurityException
- StringIndexOutOfBoundsException
- and many more

9.24. Exception Class

See also http://www.cs.rit.edu/usr/local/jdk/docs/api/java.lang/Exception.html. (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/lang/Exception.html)

Exception()

Constructs an Exception with no specified detail message.

Exception(String)

Constructs an Exception with the specified detail message.

9.25. Throwable

Exception Class is a subclass of
http://www.cs.rit.edu/usr/local/jdk/docs/api/java.lang/Throwable.html. (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/lang/Throwable.htm)

fillInStackTrace()
: Fills in the execution stack trace.

getLocalizedMessage()
: Creates a localized description of this Throwable.

getMessage()
: Returns the detail message of this throwable object.

printStackTrace()
: Prints this Throwable and its backtrace to the standard error stream.

printStackTrace(PrintStream)
: Prints this Throwable and its backtrace to the specified print stream.

printStackTrace(PrintWriter)
: Prints this Throwable and its backtrace to the specified print writer.

toString()
: Returns a short description of this throwable object.

9.26. Create a new Exception class

Use standard inheritance techniques. The superclass should be https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/lang/Throwable.html (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/lang/Throwable.html) or a sub class.

```
1       /**
2        * Thrown to indicate that a method has been passed
3        * an illegal or inappropriate ssid.
4        */
5
6       class  NumberException extends Exception
7       {
8               /**
9                * Constructs an NumberException with no detail message
10               */
11              public NumberException()        {
12                      super();
13              }
14
15              /**
16               * Constructs an NumberException with Number.Exception detail message
17               *
18               ** @param    s    the detail message.
19               */
20              public NumberException(String s)        {
21                      super(s);
22              }
23      }
```

Source Code: Src/7/NumberException.java

9.27. Exception and Inheritance

Is the 'throws' part of the method signature?

Example:

```
Class A: private void f(int n) throws Exception {

Class B extends A: private void f(int n) throws Exception {
```

```
 1
 2      public class A {
 3
 4        private void f(int n) throws Exception {
 5              System.out.println("f(" + n + ")" );
 6              switch (n)      {
 7                      case 1:  throw new NullPointerException("1");
 8                               // break;      unreachable
 9                      default: throw new Exception("default");
10              }
11        }
12
13        public static void main(String[] args) {
14              try {
15                      new A().f(1);
16              } catch (Exception e)   {
17                      e.printStackTrace();
18              }
19        }
20      }
```

Source Code: Src/7/A.java

Extends A

```
 1
 2      public class B extends A {
 3
 4        private void f(int n) {
 5              System.out.println("f(" + n + ")" );
 6        }
 7
 8        public static void main(String[] args) {
 9              new B().f(1);
10        }
11      }
```

Source Code: Src/7/B.java

9.28. Student Question

```
1      public class Test       {
2
3      /*
4      http://download.oracle.com/javase/tutorial/essential/exceptions/finally.html
5      https://docs.oracle.com/javase/specs/jls/se8/html/jls-14.html#jls-14.20.2
6      */
7              public int tryCatchFinally() {
8                   try {
9                      try {
10                         System.out.println("Inner TRY");
11                         int i = 1/0;
12                         System.out.println("Never Seen: Inner TRY after 1/0");
13                         return 1;
14                     } catch (Exception e) {
15                         System.out.println("Inner CATCH");
16                         System.out.println(e);
17                         int i = 1/0;
18                         System.out.println("Never Seen: Inner CATCH after 1/0");
19                         return 2;
20                     }
21                     finally {
22                         System.out.println("Inner FINALLY!");
23                         // int i = 1/0;
24                         // System.exit(1);
25                         return 3;   // what will happen if we comment this line ou
26                     }
27                  } catch (Exception e) {
28                         System.out.println("Outer Catch");
29                         return 4;
30                  } finally {
31                         System.out.println("Outer FINALLY");
32                         // System.exit(1);  // hat will happen if we comment this
33                         return 6;   // what will happen if we comment this line ou
34                  }
35             }
36
37
38             public static void main(String[] args ) {
39                     // return value is?
40                     System.out.println("new Test().tryCatchFinally(); = " +
41                                     new Test().tryCatchFinally() );
42             }
43     }
```

Source Code: Src/7/Test.java

- Will it compile?
- Will it execute?

9.29. Student Question II

```
 1      public class Test_2     {
 2      /*
 3      TRY
 4      CATCH
 5      nested catch
 6      CATCH after 1/0
 7      FINALLY
 8      new Test_2().tryCatchFinally(); = 3
 9      */
10      /*
11      http://download.oracle.com/javase/tutorial/essential/exceptions/finally.html
12      */
13              public int tryCatchFinally() {
14                      try {
15                          System.out.println("TRY");
16                          int i = 1/0;
17                          System.out.println("TRY after 1/0");
18                          return 1;
19                      } catch (Exception e) {
20                          System.out.println("CATCH");
21                          try {
22                                  int i = 1/0;
23
24                          } catch (Exception ee) {
25                                  System.out.println("nested catch");
26                          }
27                          System.out.println("CATCH after 1/0");
28                          return 2;
29                      } finally {
30                          System.out.println("FINALLY");
31                          return 3;   // same question
32                      }
33              }
34
35
36              public static void main(String[] args ) {
37                      // is the return value 2?
38                      System.out.println("new Test_2().tryCatchFinally(); = " +
39                                      new Test_2().tryCatchFinally() );
40              }
41      }
```

Source Code: Src/7/Test_2.java

• Will it compile?

• Will it execute?

9.30. Assertions

• assertions are used for pre/post conditions.

• An assertion is a boolean expression that a programmer specifically proclaims to be true during program runtime execution

• Declaration:

```
    assert expression1;
    assert expression1 : errorMessageExpr
```

9.31. Assertions: Example

```
 1      /*
 2       * Execution: java -ea Assertion_1
 3       */
 4      public class Assertion_1 {
 5              public void method( int value ) {
 6                      // assert 0 <= value : -1 ;
 7                      assert 0 <= value : "this did not work out" ;
 8                      System.out.println("asertM ---->");
 9                      System.out.println("\tvalue = " + value );
10                      System.out.println("asertM <----");
11              }
12
13              public static void main( String[] args ) {
14                      Assertion_1 asertM = new Assertion_1();
15                      asertM.method( 1 );
16                      asertM.method( -1 );
17              }
18      }
```

Source Code: Src/7/Assertion_1.java

Execution:

```
% javac Assertion_1.java
% java Assertion_1
asertM ---->
        value = 1
asertM <----
asertM ---->
        value = -1
asertM <----
% java -ea Assertion_1
asertM ---->
        value = 1
asertM <----
// Exception in thread "main" java.lang.AssertionError: -1
Exception in thread "main" java.lang.AssertionError: this did not work out
       at Assertion_1.method(Assertion_1.java:6)
       at Assertion_1.main(Assertion_1.java:15)
make: *** [run] Error 1
```

9.32. Assertions: Processed

```
if if true
then yes
       : No further  action
else no
       if if expression2 exists
       then yes
              : Evaluate expression2 and use the  result in a single-parameter form of
       else np
 use the default AssertionError constructor
}
}
```

9.33. Assertions: Enable

• Java command line argument `-ea`

```
    % java -verbose
        ...
        -ea[:<packagename>...|:<classname>]
        -enableassertions[:<packagename>...|:<classname>]
                       enable assertions
        -da[:<packagename>...|:<classname>]
        -disableassertions[:<packagename>...|:<classname>]
                       disable assertions
        -esa | -enablesystemassertions
                       enable system assertions
        -dsa | -disablesystemassertions
                       disable system assertions
        ...
```

9.34. Assertions: Throwable

See also: http://www.cs.rit.edu/usr/local/j2sdk1.5.0-beta1/docs/api/index.html (http://www.cs.rit.edu/usr/local/j2sdk1.5.0-beta1/docs/api/index.html)

•

```
 1      /*
 2       * Execution: java -ea:
 3       */
 4      public class Assertion_2 {
 5              public void method( int value ) {
 6                      assert 0 <= value: "Value must be postive =" + value + "=";
 7                      System.out.println("asertM ---->");
 8                      System.out.println("\tvalue = " + value );
 9                      System.out.println("asertM <----");
10              }
11
12              public static void printAssertion( AssertionError ae ) {
13                      StackTraceElement[] stackTraceElements = ae.getStackTrace();
14                      StackTraceElement stackTraceElement = stackTraceElements[ 0 ];
15
16                      System.err.println( "AssertionError" );
17                      System.err.println( "   class=   " + stackTraceElement.getClas
```

```
18                      System.err.println( "   method=  " + stackTraceElement.getMeth
19                      System.err.println( "   message= " + ae.getMessage() );
20              }
21
22              public static void main( String[] args ) {
23                      Assertion_2 asertM = new Assertion_2();
24                      try {
25                              asertM.method( 1 );
26                              asertM.method( -1 );
27                      } catch( AssertionError ae )    {
28                              printAssertion(ae);
29                      }
30              }
31      }
```

Source Code: Src/7/Assertion_2.java

•

```
% java -ea Assertion_2
asertM ---->
        value = 1
asertM <----
AssertionError
   class=   Assertion_2
   method=  method
   message= Value must be positive =-1=
```

```
 1      /*
 2       * Execution: java -ea:
 3       */
 4      public class Assertion_3 {
 5              public int method( int i ) {
 6
 7                      if (i % 3 == 0) {
 8                              System.out.println("i % 3 == 0");
 9                      } else if (i % 3 == 1) {
10                              System.out.println("i % 3 == 1");
11                      } else {
12                              System.out.println("else");
13                              assert false : i;
14                      }
15                      return 99;
16              }
17
18              public static void main( String[] args ) {
19                      Assertion_3 asertM = new Assertion_3();
20                      try {
21                              System.out.println( asertM.method( 3 ));
22                              System.out.println( asertM.method( 5 ));
23                      } catch( AssertionError ae )    {
24                              ae.printStackTrace();
25                      }
26              }
```

```
27      }
```

Source Code: Src/7/Assertion_3.java

Execution:

% java -ea Assertion_3 i % 3 == 0 99 else java.lang.AssertionError: 5 at Assertion_3.method(Assertion_3.java:13) at Assertion_3.main(Assertion_3.java:22)

9.35. Assertions: Disabling

• Assertions can also be disabled down to the class level

• the command line argument `-disableassertions` (`-da` parallels the syntax of the assertion-enabling switch.

• a command line can contain as many enable- and disable-assertion switches as desired.

10. I/O: Files and Streams

See also: http://www.cs.rit.edu/usr/local/jdk/docs/api/java/io/package-summary.html (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/io/package-summary.html)

File:

- Files can store persistent information.
- Objects can be stored in a file.
- Files offer random access.
- Files can be created, removed, overwritten and appended.
- Files must have a name — the name depends on the operating system. They don't depend on Java.

In Java, file I/O as well as keyboard/screen I/O is handled by streams.

10.1. Overview

- java.io (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/io/package-summary.html)
  package has two inheritance chains that can be used for dealing with files. One chain starts with the abstract classes InputStream (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/io/InputStream.html)
  and OutputStream, (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/io/OutputStream.html)
  and the second chain starts with Reader (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/io/Reader.html)
  and Writer (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/io/Writer.html)

- InputStream and OutputStream and their subclasses are used for input and output of byte values.

  InputStreamReader and OutputStreamWriter combine an encoding and a stream and thus are the preferred classes for text input and output. This means they are byte based.

- The Reader and Writer classes are defined for reading from character files. They provide facilities specific to character files, such as, reading a line of input.

  Reader and Writer classes are abstract — they can have abstract methods (declaration without a body) and no objects may be instantiated. Abstract classes define basic behavior for related subclasses including instance variables and some method implementations.

  Text files normally contain byte values that represent a selection of char values by way of an encoding like 8859_1 (ISO Latin-1), or a code page like Cp850 (PC Latin-1), or with the unicode transfer format UTF8.

10.2. I/O Streams Classifictations

• Byte Streams handle I/O of raw binary data.

• Character Streams handle I/O of character data, automatically handling translation to and from the local character set.

• Buffered Streams optimize input and output by reducing the number of calls to the native API.

• Scanning and Formatting allows a program to read and write formatted text.

• I/O from the Command Line describes the Standard Streams and the Console object.

• Data Streams handle binary I/O of primitive data type and String values.

• Object Streams handle binary I/O of objects.

10.3. Byte Streams vs. Chracter Streams

• Byte Streams perform I/O on 8 bits of day

• Reader (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/io/Reader.html)

• Writer (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/io/Writer.html)

• Character stream I/O automatically translates this internal format to and from the local character set. In Western locales, the local character set is usually an 8-bit superset of ASCII.

• InputStream (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/io/InputStream.html)

• OutputStram (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/io/OutputStream.html)

10.4.

• Input Stream

For an input stream, the source of data might be a file, a String, an array of bytes, or bytes written to an output stream (typically by another thread). There are also "filter input streams" that take data from another input stream and transform or augment the data before delivering it as input. For example, a DataNumberInputStream (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/io/DataInputStream.html)
passes bytes through verbatim but counts line terminators as they are read.

The drawings have been created by Charles L. Perkins. (http://rendezvous.com/java/)

10.5. Output Stream

For an output stream, the sink of data might be a file, an array of bytes, or a buffer to be read as an input stream (typically by another thread). There are also "filter output streams" that transform or augment data before writing it to some other output stream.

An instance of class File represents a path name (a String) that might identify a particular file within a file system. Certain operations on the file system, such as renaming and deleting files, are done by this class rather than through streams.

10.6. Using Streams

No matter where the information is coming from or going to the algorithm for reading/writing is pretty much always the same:

Reading:

```
open a stream for reading
while more information
     read
     process

close stream
```

Writing:

```
open a stream for writing
while more information
     process
     write

close stream
```

10.7. Stream Wrappers

From here. (http://doc.novsu.ac.ru/oreilly/java/exp/ch08_01.htm)

10.8. Data Processing Streams

Processing streams perform some sort of operation, such as buffering or character encoding, as they read and write. Like the data sink streams, java/io often contains pairs of streams: one that performs a particular operation during reading and another that performs the same operation (or reverses it) during writing. This table gives java/io's processing streams

Process	Character Stream	Byte Stream
Buffering	BufferedReader, BufferedWriter	BufferedInputStream, BufferedOutputStream
Filtering	FilterReader, FilterWriter	FilterInputStream, FilterOutputStream
Converting between Bytes and Characters	InputStreamReader, OutputStreamWriter	
Concatenation		SequenceInputStream
Object Serialization		ObjectInputStream, ObjectOutputStream
Data Conversion		DataInputStream, DataOutputStream
Counting	LineNumberReader	LineNumberInputStream
Peeking Ahead	PushbackReader	PushbackInputStream
Printing	PrintWriter	PrintStream

10.9. Data Sink Streams

Data sink streams read from or write to specialized data sinks such as strings, files, or pipes. Typically, for each reader or input stream intended to read from a specific kind of input source, java/io contains a parallel writer or output stream that can create it. The following table gives java/io's data sink streams.

Sink Type	Character Streams	Byte Streams
Memory	CharArrayReader,	ByteArrayInputStream,
	CharArrayWriter	ByteArrayOutputStream
	StringReader,	StringBufferInputStream
	StringWriter	StringBufferInputStream
Pipe	PipedReader,	PipedInputStream,
	PipedWriter	PipedOutputStream
File	FileReader,	FileInputStream,
	FileWriter	FileOutputStream

CharArrayReader and CharArrayWriter
ByteArrayInputStream and ByteArrayOutputStream

Use these streams to read from and write to memory. You create these streams on an existing array and then use the read and write methods to read from or write to the array.

FileReader and FileWriter
FileInputStream and FileOutputStream

Collectively called file streams, these streams are used to read from or write to a file on the native file system.

PipedReader and PipedWriter
PipedInputStream and PipedOutputStream

Implement the input and output components of a pipe. Pipes are used to channel the output from one program (or thread) into the input of another.

StringReader and StringWriter
StringBufferInputStream

Use StringReader to read characters from a String as it lives in memory. Use StringWriter to write to a String. StringWriter collects the characters written to it in a StringBuffer, which can then be converted to a String. StringBufferInputStream is similar to StringReader, except that it reads bytes from a StringBuffer.

java/io

The hierarchy of classes defined in package https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/lang/package-tree.html (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/lang/package-tree.html)

10.10. A Copy Program

• try-with-resources statement is a try statement that declares one or more resources.

• the resource is an object that must be closed after the program is finished with it.

• Any object of a class that implements java.lang.AutoCloseable (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/lang/AutoCloseable.html)

•

```
 1      import java.io.*;
 2
 3      public class InOut_1 {
 4        public static void main( String args[] ) {
 5          byte[]  buffer = new byte[1024];
 6          int     n;
 7
 8          if ( args.length < 2 )      {
 9              System.err.println(
10                   "Usage: java InOut_1 from to");
11              System.exit(1);
12          }
13
14          try (
15              DataInputStream in = new DataInputStream(
16                                       new FileInputStream(args[0]) );
17              DataOutputStream out = new DataOutputStream(
18                                       new FileOutputStream(args[1]) );
19              ) {
20
21                      while ( (n = in.read(buffer) ) != -1 ) {
22                              out.write(buffer, 0, n);
23                      }
24
25          }
26          catch ( FileNotFoundException ef)   {
27              System.out.println("File not found: " + args[1]);
28          }
29          catch ( IOException ef)     {
30              System.out.println("File not found: " + args[1]);
31          }
32          catch ( Exception e)        {
33              System.out.println("ExceptionType occurred: " +
34                      e.getMessage() );
35          }
36
37        }
38      }
```

Source Code: Src/9_was/InOut_1.java

Result:

```
% java InOut_1
% java InOut_1 InOut_2.class x
% diff InOut_1.class x
%
```

You can skip forward ...

```
 1      import java.io.*;
 2
 3      public class Skip {
 4        public static void main( String args[] ) {
 5          byte[]  buffer = new byte[1024];
 6          int     n;
 7
 8          if ( args.length < 2 )      {
 9              System.err.println(
10                   "Usage: java Skip from to");
11              System.exit(1);
12          }
13
14          try (
15              DataInputStream in = new DataInputStream(
16                                       new FileInputStream(args[0]) );
17              DataOutputStream out = new DataOutputStream(
18                                       new FileOutputStream(args[1]) );
19
20              ) {
21                      in.skipBytes(10);
22                      while ( (n = in.read(buffer) ) != -1 ) {
23                              out.write(buffer, 0, n);
24
25                  }
26          } catch ( FileNotFoundException ef) {
27              System.out.println("File not found: " + args[1]);
28          }
29          catch ( IOException ef)     {
30              System.out.println("File not found: " + args[1]);
31          }
32          catch ( Exception e)        {
33              System.out.println("ExceptionType occurred: " +
34                      e.getMessage() );
35          }
36
37        }
38      }
```

Source Code: Src/9_was/Skip.java

10.11. Size Matters: The second Copy Program

Does the size of the read and write blocks matter?

```
 1      import java.io.*;
 2
 3
 4      public class InOut_2 {
 5        static final int BUFSIZE = 1024;
 6
 7        public static void copy( String inF , String outF, int bufSize ) {
 8          DataInputStream in;
 9          DataOutputStream out;
10          byte[]  buffer = new byte[bufSize];
11          int     n;
12
13
14          try (
15              DataInputStream in = new DataInputStream(
16                                       new FileInputStream(inF) );
17              DataOutputStream out = new DataOutputStream(
18                                       new FileOutputStream(outF) );
19
20              ) {
21                      while ( (n = in.read(buffer) ) != -1 ) {
22                              out.write(buffer, 0, n);
23                      }
24
25          }
26          catch ( FileNotFoundException ef)   {
27              System.out.println(ef.getMessage() );
28          }
29          catch ( IOException ef)     {
30              System.out.println(ef.getMessage() );
31          }
32          catch ( Exception e)        {
33              System.out.println("ExceptionType occurred: " +
34                      e.getMessage() );
35          }
36        }
37
38
39        public static void main( String args[] ) {
40          int     bufSize = BUFSIZE;
41
42          if ( args.length < 2 )      {
43              System.err.println(
44                   "Usage: java InOut_1 from to [size]");
45              System.exit(1);
46          }
47
48          if ( args.length  == 3 )    {
49              try {
50                      bufSize = Integer.parseInt(args[2]);
```

```
51              }
52              catch ( NumberFormatException e )       {
53                      System.out.println("Can't convert " + args[2]
54                              + " to an integer.");
55              }
56
57          }
58          System.out.println("BufferSize = " + bufSize);
59          copy(args[0], args[1], bufSize);
60        }
61      }
```

Source Code: Src/9_was/InOut_2.java

Result:

```
% for i in 1 2 512 1024 10240
> do
> /usr/bin/time java InOut_2 from to $i && diff from to
> done
BufferSize = 1

real       49.8
user       21.3
sys        24.4
BufferSize = 2

real       27.0
user       10.8
sys        12.3
BufferSize = 512

real        0.8
user        0.2
sys         0.2
BufferSize = 1024

real        0.8
user        0.1
sys         0.2
BufferSize = 10240

real        1.0
user        0.2
sys         0.1
```

10.12. Reading from Stdin

```
 1      import java.io.*;
 2
 3      public class stdin {
 4        public static void main( String args[] ) {
 5          LineNumberInputStream input;
 6
 7          if ( args.length > 1 )      {
 8              System.err.println(
 9                   "Usage: java stdin file-name");
10              System.exit(1);
11          }
12
13          try {
14              String line;
15      //
16              if ( args.length == 1 )
17                      input = new LineNumberInputStream(
18                                  new DataInputStream(
19                                    new FileInputStream(args[0]) ) );
20              else
21                      input = new LineNumberInputStream( System.in );
22
23
24              while ( ( input.read() )  != -1  ) {
25                      ;
26              }
27              System.out.println("# lines = " + input.getLineNumber() );
28              input.close();
29          }
30          catch ( FileNotFoundException e)    {
31              System.out.println(e.getMessage());
32          }
33          catch ( IOException e)      {
34              System.out.println(e.getMessage());
35          }
36          catch ( Exception e)        {
37              System.out.println("ExceptionType occurred: " +
38                      e.getMessage() );
39          }
40        }
41      }
```

Source Code: Src/14/stdin.java

10.13. Reading/Writing Compressed Files

```
 1
 2      /**
 3        * Opens the file with the data.
 4        * A rewind will happen, if this method gets called more than once.
 5        **/
 6          public void openFileForReading() {
 7              try {
 8                      if ( isBinaryInput )    {
 9                              bInputStream = new DataInputStream(
10                                                      new GZIPInputStream(
11                                                        new FileInputStream(inpu
12                                                      )
13                                                    );
14
15                      } else {
16                              inputStream = new BufferedReader(new FileReader(inputF
17                      }
18                      ...
19              } catch ( Exception e ) {
20                      e.printStackTrace();
21                      InOutErr.out.println("ParticleViewExtractor: -openFileForReadi
22              }
23          }
24
25
26         /*
27          * Convert Data to Binary format
28          */
29          public void doConvertToBinary(String fileName)      {
30              try {
31                      BufferedReader inputStream = new BufferedReader(new FileReader
32                      DataOutputStream outputStream = new DataOutputStream(
33                                                      new GZIPOutputStream(
34                                                        new FileOutputStream(fil
35                                                      )
36          ...                                          );
37          }
```

Source Code: Src/14/Compressed.java

10.14. A Grep Program

Extract from:
http://www.cs.rit.edu/usr/local/jdk/docs/api/java/io/BufferedReader.html
(http://www.cs.rit.edu/usr/local/jdk/docs/api/java/io/BufferedReader.html)

```
BufferedReader(Reader)
    Create a buffering character-input
    stream that uses a default-sized input buffer.
BufferedReader(Reader, int)
    Create a buffering character-input
    stream that uses an input buffer of the specified
    size.
```

```
 1      import java.io.*;
 2
 3      public class Grep {
 4        public static void main( String args[] ) {
 5
 6          try  (
 7              BufferedReader input = new BufferedReader( new FileReader(args[1]));
 8              BufferedWriter output   = new BufferedWriter( ( args.length == 3  ? ne
 9
10              ) {
11                      String line;
12                      while ( ( line = input.readLine() )  != null ) {
13                              if ( line.indexOf(args[0]) >= 0 )       {
14                                      output.write(line, 0, line.length() );
15                                      output.newLine();
16                              }
17                              output.flush();
18              }
19          }
20          catch ( FileNotFoundException e)    {
21              System.out.println(e.getMessage());
22          }
23          catch ( IOException e)      {
24              System.out.println(e.getMessage());
25          }
26          catch ( Exception e)        {
27              System.out.println("ExceptionType occurred: " +
28                      e.getMessage() );
29          }
30
31        }
32      }
```

Source Code: Src/9_was/Grep.java

Execution:

```
% java Grep Grep Grep.java
public class Grep {
             "Usage: java Grep search-string file-name [output-filename]");
% java Grep Grep
Usage: java Grep search-string file-name [output-filename]
```

If you are interested in line numbers, choose LineNumberReader

```
 1      import java.io.*;
 2
 3      public class Grep2 {
 4        public static void main( String args[] ) {
 5          LineNumberReader input;
 6          PrintWriter    output;
 7
 8          if ( args.length < 2 )      {
 9              System.err.println(
10                   "Usage: java Grep search-string file-name [outputfilename]");
11              System.exit(1);
12          }
13
14          try {
15              String line;
16              input = new LineNumberReader (
17                            new  BufferedReader(
18                                   new FileReader(args[1])
19                                                 )
20                                          );
21              if ( args.length == 3 )         {
22                  output = new PrintWriter( new FileWriter(args[2]) );
23              } else
24                  output = new PrintWriter(System.out);
25
26              while ( ( line = input.readLine() )  != null ) {
27                      if ( line.indexOf(args[0]) >= 0 )
28                              output.println(input.getLineNumber() +
29                                      ": " + line);
30              }
31              output.close();
32              input.close();
33          }
34          catch ( FileNotFoundException e)    {
35              System.out.println(e.getMessage());
36          }
37          catch ( IOException e)      {
38              System.out.println(e.getMessage());
39          }
40          catch ( Exception e)        {
41              System.out.println("ExceptionType occurred: " +
42                      e.getMessage() );
43          }
44
45        }
46      }
```

Source Code: Src/9_was/Grep2.java

Execution:

```
% java Grep2 Grep Grep.java
3: public class Grep {
10:          "Usage: java Grep search-string file-name [output-filename]");
%
```

10.15. Regular Expressions

• See also here: Regular Expressions (http://www.cs.rit.edu/~hpb/Lectures/20012/LP/all-5.3.html)

10.16. Regular Expressions in Java

See also here: Pattern (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/util/regex/Pattern.html)

• Classes for matching character sequences against patterns specified by regular expressions.

• An instance of the Pattern class represents a regular expression that is specified in string form in a syntax similar to that used by Perl.

• Instances of the Matcher class are used to match character sequences against a given pattern.

10.17. Example 1

```
1       /*
2        * Checks for invalid characters
3        * in email addresses
4        */
5
6       import java.util.regex.*;
7
8       public class EmailValidation {
9
10        /*
11         * Checks for email addresses starting with
12         * inappropriate symbols like dots or @ signs.
13         */
14         public static void checkForPorA(String aPossibleEmail )      {
15            Pattern p = Pattern.compile("^\\.|^\\@");
16            Matcher m = p.matcher(aPossibleEmail);
17            if (m.find())
18               System.err.println(aPossibleEmail + " - Email addresses don't start"
19                                  " with dots or @ signs.");
20            else
21               System.err.println(aPossibleEmail + " is valid.");
22         }
23
24        /*
25         * Checks for email addresses starting with
26         * www.
27         */
28         public static void checkForWWW(String aPossibleEmail )       {
29            Pattern p = Pattern.compile("^www\\.");
30            Matcher m = p.matcher(aPossibleEmail);
31            if (m.find())
32               System.err.println(aPossibleEmail + " - Email addresses don't start"
33                                  " with www.");
34            else
35               System.err.println(aPossibleEmail + " is valid.");
36         }
37
38
39        /*
```

```
40         * Checks for invalid characters in email addresses.
41         */
42         public static void checkForInvalidC(String aPossibleEmail )  {
43            Pattern p = Pattern.compile("[^A-Za-z0-9\\.\\@_\\-~#]+");
44            Matcher m = p.matcher(aPossibleEmail);
45            StringBuffer sb = new StringBuffer();
46            boolean result = m.find();
47            boolean deletedIllegalChars = false;
48
49            while(result) {
50               deletedIllegalChars = true;
51               m.appendReplacement(sb, "");
52               result = m.find();
53            }
54
55            if (deletedIllegalChars) {
56               System.out.println("It contained incorrect characters" +
57                                  " , such as spaces or commas.");
58            }
59         }
60
61
62         public static void main(String[] args) throws Exception {
63
64
65            checkForPorA("hpb@cs.rit.edu");
66            checkForPorA("@cs.rit.edu");
67
68            checkForWWW("www.cs.rit.edu");
69
70            checkForInvalidC("hpb@cs.rit.edu");
71            checkForInvalidC("p b@cs.rit.edu");
72
73         }
74      }
```

Source Code: Src/9_reg_ex/EmailValidation.java

10.18. Example 2

```
 1      /*
 2       * Checks for invalid characters
 3       * in email addresses
 4       */
 5
 6      import java.util.regex.*;
 7
 8      public class TheN {
 9
10        /*
11         * Palindroms
12         */
13         public static void checkForP(String aPossibleEmail ) {
```

```
14            Pattern p = Pattern.compile("^.$");
15            Matcher m = p.matcher(aPossibleEmail);
16            if (m.find())
17               System.err.println(aPossibleEmail + " one character");
18            else
19               System.err.println(aPossibleEmail + " more than one character");
20         }
21
22         public static void checkForP2(String aPossibleEmail )        {
23            Pattern p = Pattern.compile("^(.).\\1$");
24            Matcher m = p.matcher(aPossibleEmail);
25            if (m.find())
26               System.err.println(aPossibleEmail + " 2 char palindrom");
27            else
28               System.err.println(aPossibleEmail + " ! a 2 char palindrom");
29         }
30
31         public static void main(String[] args) throws Exception {
32
33
34            checkForP("a");
35            checkForP("aa");
36
37            checkForP2("a");
38            checkForP2("ata");
39
40            if (  Pattern.matches("^(.)\\1$", "aa" ))
41                      System.err.println("palindrom");
42
43         }
44      }
```

Source Code: Src/9_reg_ex/TheN.java

10.19. Serializing Objects

- Two of the byte streams, ObjectInputStream (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/io/ObjectInputStream.html)
  and ObjectOutputStream, (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/io/ObjectOutputStream.html)
  are specialized streams that let you read and write objects. Reading and writing objects is a process known as object serialization.

• The serialization interface has no methods or fields and serves only to identify the semantics of being serializable.

• Object serialization has many uses, including remote method invocation (RMI). In addition to the object streams, java/io has other classes and interfaces that define the API to help classes perform serialization for its instances.

- Only objects that support the java.io.Serializable (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/io/Serializable.html)
  interface can be written to streams.

• The class of each serializable object is encoded including the class name and signature of the class, the values of the object's fields and arrays, and the closure of any other objects referenced from the initial objects

- Reconstructing an object from a stream requires that the object first be written to a stream.
- The default serialization mechanism for an object writes the class of the object, the class signature, and the values of all non-transient and non-static fields. References to other objects (except in transient or static fields) cause those objects to be written also. Multiple references to a single object are encoded using a reference sharing mechanism so that graphs of objects can be restored to the same shape as when the original was written.
- Class variables are not part of the serialization mechanism.

Object writer:

```
 1      import java.io.*;
 2      import java.util.Date;
 3
 4      public class ObjectWriter_1 {
 5        public static void main( String args[] ) {
 6
 7          Date d = new Date();
 8
 9          try (
10              FileOutputStream ostream = new FileOutputStream("object_1.ser");
11              ObjectOutputStream p = new ObjectOutputStream(ostream);
12              )
13              {
14              p.writeInt(12345);
15                System.out.println("Integer = " + 1234);
16              p.writeObject("Today");
17                System.out.println("String = " + "Today");
18              p.writeObject(d);
19                System.out.println("Date = " + d);
20              p.flush();
21          }
22          catch ( IOException e)      {
23              System.out.println(e.getMessage());
24          }
25
26        }
27      }
```

Source Code: Src/9_was/ObjectWriter_1.java

Output:

```
% java ObjectWriter_1
Integer = 1234
String = Today
Date = Mon Nov  1 08:42:38 EDT 2010
```

Object Reader:

```
 1      import java.io.*;
 2      import java.util.Date;
 3
 4      public class ObjectReader_1 {
 5        public static void main( String args[] ) {
 6
 7          try (
 8              FileInputStream istream = new FileInputStream("object_1.ser");
 9              ObjectInputStream p = new ObjectInputStream(istream);
10              ) {
11              int i = p.readInt();
12                System.out.println("Integer = " + i);
13              String today = (String)p.readObject();
14                System.out.println("String = " + today);
15              Date date = (Date)p.readObject();
16                System.out.println("Date = " + date);
17          }
18          catch ( IOException e)      {
19              System.out.println(e.getMessage());
20          }
21          catch ( ClassNotFoundException e)   {
22              System.out.println(e.getMessage());
23          }
24
25        }
26      }
```

Source Code: Src/9_was/ObjectReader_1.java

Output:

```
% java ObjectReader_1
Integer = 1234
String = Today
Date = Mon Nov  1 08:42:38 EDT 2010
```

Octal dump:

```
ls -l o*a
-rw-------   1 hpb      fac           60 Oct  4 10:49 object_1.ser
yps 9 85 od -c object_1.ser
0000000 254 355  \0 005   w 004  \0  \0   0   9   t  \0 005   T   o   d
0000020   a   y   s   r  \0 016   j   a   v   a   .   u   t   i   l   .
0000040   D   a   t   e   h   j 201 001   K   Y   t 031 003  \0  \0   x
0000060   p   w  \b  \0  \0  \0 326 365   <   o 232   x
0000074
```

When an object is serialized, any object reference it contains are also serialized. A HashTable (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/gutil.Hashtable.html) example.

```
 1      import java.io.*;
 2      import java.util.*;
 3
 4      public class ObjectWriter_2 {
 5        public static void main( String args[] ) {
 6
 7          Hashtable aHashTable = new Hashtable();
 8          aHashTable.put("plus  Movie", "A little Voice");
 9          aHashTable.put("minus Movie", "Independence Day");
10
11          try (
12              FileOutputStream ostream = new FileOutputStream("object_2.ser");
13              ObjectOutputStream p = new ObjectOutputStream(ostream);
14              ) {
15              p.writeObject(aHashTable);
16                System.out.println("aHashTable = " + aHashTable.toString());
17              p.flush();
18          }
19          catch ( IOException e)      {
20              System.out.println(e.getMessage());
21          }
22
23        }
24      }
```

Source Code: Src/9_was/ObjectWriter_2.java

Output:

```
% java ObjectWriter_2
aHashTable = {minus Movie=Independence Day, plus  Movie=A little Voice}
```

```
 1      import java.io.*;
 2      import java.util.*;
 3
 4      public class ObjectReader_2 {
 5        public static void main( String args[] ) {
 6
 7          Hashtable aHashTable;
 8
 9          try (
10              FileInputStream istream = new FileInputStream("object_2.ser");
11              ObjectInputStream p = new ObjectInputStream(istream);
12              ) {
13              aHashTable= (Hashtable)p.readObject();
14                System.out.println("aHashTable = " + aHashTable.toString());
15          }
16          catch ( IOException e)      {
17              System.out.println(e.getMessage());
18          }
19          catch ( ClassNotFoundException e)   {
20              System.out.println(e.getMessage());
21          }
22        }
23      }
```

Source Code: Src/9_was/ObjectReader_2.java

Output:

```
% java ObjectReader_2
aHashTable = {minus Movie=Independence Day, plus  Movie=A little Voice}
```

10.20. Can an Object include itself?

```
 1      import java.io.*;
 2      import java.util.*;
 3
 4      public class Self {
 5        public static void main( String args[] ) {
 6
 7          Hashtable aHashTable = new Hashtable();
 8          aHashTable.put("plus  Movie", "A little Voice");
 9          aHashTable.put("The HashTable", aHashTable);
10
11          try {
12              FileOutputStream ostream =
13                           new FileOutputStream("self.ser");
14              ObjectOutputStream p = new ObjectOutputStream(ostream);
15              p.writeObject(aHashTable);
16              p.flush();
17              p.close();
18          }
19          catch ( IOException e)      {
20              System.out.println(e.getMessage());
21          }
22          catch ( Exception e)        {
23              System.out.println(e.getMessage());
24              e.printStackTrace();
25              System.exit(1);
26          }
27
28        }
29      }
```

Source Code: Src/9_was/Self.java

Output:

```
% java Self
% od -c self.ser
0000000 254 355  \0 005   s   r  \0 023   j   a   v   a   .   u   t   i
0000020   l   .   H   a   s   h   t   a   b   l   e 023 273 017   %   !
0000040   J 344 270 003  \0 002   F  \0  \n   l   o   a   d   F   a   c
 ...
0000160       M   o   v   i   e   t  \0 016   A       l   i   t   t   l
0000200   e       V   o   i   c   e   x
0000210
```

```
 1      import java.io.*;
 2      import java.util.*;
 3
 4      public class Self_Reader {
 5        public static void main( String args[] ) {
 6
 7          Hashtable aHashTable;
 8
 9          try {
10              FileInputStream istream =
11                           new FileInputStream("self.ser");
12              ObjectInputStream p = new ObjectInputStream(istream);
13
14              aHashTable= (Hashtable)p.readObject();
15              System.out.println("plus  Movie = " + aHashTable.get("plus  Movie"));
16              System.out.println("The HashTable" + aHashTable.get("The HashTable"));
17              System.out.println("aHashTable = " + aHashTable.toString());
18              p.close();
19          }
20          catch ( IOException e)      {
21              System.out.println(e.getMessage());
22          }
23          catch ( ClassNotFoundException e)   {
24              System.out.println(e.getMessage());
25          }
26        }
27      }
```

Source Code: Src/9_was/Self_Reader.java

Output:

```
% java Self_Reader 2>&1 | more // version jdk1.3.1
plus  Movie = A little Voice
java/lang/StackOverflowError
        at java/lang/StringBuffer.<init>(StringBuffer.java)
        at java/gutil.Hashtable.toString(Hashtable.java)
        at java/gutil.Hashtable.toString(Hashtable.java)
 ...
% java Self_Reader // version jdk1.4
plus  Movie = A little Voice
The HashTable{The HashTable=(this Map), plus  Movie=A little Voice}
aHashTable = {The HashTable=(this Map), plus  Movie=A little Voice}
```

10.21. Make it Serializable

• An object is serializable only if its class implements the Serializable interface.

•

• You don't have to write any methods. The serialization of instances of this class are handled by the defaultWriteObject method of ObjectOutputStream. This method automatically writes out everything required to reconstruct an instance of the class, including the following:

  — Class of the object

  — Class signature

  — Values of all non-transient and non-static members, including members that refer to other objects

```
1       import java.io.*;
2       import java.util.*;
3
4       public class ObjectWriter_4 implements Serializable {
5         int local = 42;
6         private void writeObject(ObjectOutputStream s) throws IOException {
7                  s.defaultWriteObject();
8                // customized serialization code
9         }
10
11        private void readObject(ObjectInputStream s) throws IOException  {
12              try {
13                      s.defaultReadObject();
14              }
15              catch ( ClassNotFoundException e)       {
16                  System.out.println(e.getMessage());
17                  e.printStackTrace();
18              }
19
20               // customized deserialization code
21               //  ...
22               // followed by code to update the object, if necessary
23        }
24
25        public static void main( String args[] ) {
26
27          ObjectWriter_4 aObjectWriter_4 = new ObjectWriter_4();
28
29          try (
30              FileOutputStream ostream = new FileOutputStream("object_4.ser");
31              ObjectOutputStream p = new ObjectOutputStream(ostream);
32              ) {
33              p.writeObject(aObjectWriter_4);
34                System.out.println("aObjectWriter_4 = " + aObjectWriter_4.toString()
35                System.out.println("aObjectWriter_4.local = " + aObjectWriter_4.loca
36              p.flush();
37          }
38          catch ( IOException e)      {
39              System.out.println(e.getMessage());
40              e.printStackTrace();
41          }
42
43        }
44      }
```

Source Code: Src/9_was/ObjectWriter_4.java

Output:

```
% java ObjectWriter_4
aObjectWriter_4 = ObjectWriter_4@1dc60810
aObjectWriter_4.local = 42
```

The reader program must not be modified.

10.22. Using readObject and writeObject

The class:

```
 1      import java.io.*;
 2      import java.util.*;
 3
 4      public class CheckLength implements Serializable {
 5        private int    MINIMUMlength = 10;
 6        private int    length;
 7        private String aString;
 8        // private int length = 4;
 9
10        // private static final long serialVersionUID = 1234567L;
11        private static final long serialVersionUID = 1234568L;
12
13        public CheckLength(String aString)    {
14                     this.aString = aString;
15                     this.length  = aString.length();
16        }
17        private void writeObject(ObjectOutputStream s) throws IOException {
18              System.out.println("CheckLength: writeObject");
19              s.defaultWriteObject();
20              if ( length < MINIMUMlength )
21                      s.writeObject(-1);
22              else
23                      s.writeObject(length);
24        }
25
26        private void readObject(ObjectInputStream s) throws IOException  {
27              System.out.println("CheckLength: readObject");
28              try {
29                      s.defaultReadObject();
30                      int length = (int)s.readObject();
31                      if ( this.aString.length() != length )
32                              this.aString = "";
33              }
34              catch ( ClassNotFoundException e)       {
35                  System.out.println(e.getMessage());
36                  e.printStackTrace();
37              }
38        }
39        public String toString()      {
40              return aString + "/" + length;
41        }
42
43      }
```

Source Code: Src/9_was/CheckLength.java

The reader:

```
 1      import java.io.*;
 2      import java.util.*;
 3
 4      public class CheckLengthRead {
 5
 6        public static void main( String args[] ) {
 7
 8          CheckLength aCheckLength;
 9          String fileName = "1234.ser";
10
11          if ( args.length == 1 )
12              fileName = args[0];
13          try (
14              ObjectInputStream p = new ObjectInputStream(new FileInputStream(fileNa
15              ) {
16              aCheckLength = (CheckLength)p.readObject();
17              System.out.println("aCheckLength = " + aCheckLength );
18              aCheckLength = (CheckLength)p.readObject();
19              System.out.println("aCheckLength = " + aCheckLength );
20          }
21          catch ( IOException e)      {
22              System.out.println(e.getMessage());
23              e.printStackTrace();
24          }
25          catch ( ClassNotFoundException e)   {
26              System.out.println(e.getMessage());
27              e.printStackTrace();
28          }
29        }
30      }
```

Source Code: Src/9_was/CheckLengthRead.java

The Writer:

```
 1      import java.io.*;
 2      import java.util.*;
 3
 4      public class CheckLengthWrite {
 5
 6        public static void main( String args[] ) {
 7
 8          CheckLength aCheckLength = new CheckLength("abcdef");
 9          CheckLength bCheckLength = new CheckLength("abcdefghijklmnopqrst");
10          String fileName    = "1234.ser";
11
12          if ( args.length == 1 )
13              fileName = args[0];
14
15          try (
16              ObjectOutputStream p = new ObjectOutputStream(new FileOutputStream(fil
17              ) {
18              p.writeObject(aCheckLength);
19              p.writeObject(bCheckLength);
```

```
20          }
21          catch ( IOException e)      {
22              System.out.println(e.getMessage());
23              e.printStackTrace();
24          }
25
26        }
27      }
```

Source Code: Src/9_was/CheckLengthWrite.java

Output:

```
% java CheckLengthWrite 1234.ser
CheckLength: writeObject
CheckLength: writeObject
% java CheckLengthRead 1234.ser
CheckLength: readObject
aCheckLength = /6  # why?
CheckLength: readObject
aCheckLength = abcdefghijklmnopqrst/20
```

10.23. Serialization Notes - General

Interitance Comments

• When a class implements the java.io.Serializable interface, all its sub-classes are serializable as well.

• When an object has a reference to another object, these objects must implement the Serializable interface separately

10.24. Serialization Notes - Security

• Serialization is not secure

• Serialized data can be signed and sealed

• JVM associates a long number with each serializable class, a Serial Version UID

• If a serializable class doesnot declare a serialVersionUID, the JVM will generate one automatically at run-time

• It is highly recommended that each class declares its serialVersionUID as the generated one is compiler dependent

• Modifiction of *serialVersionUID = 1234568L;* and recompiling of CheckLength.java results in:

```
javac CheckLength.java
java CheckLengthRead 1234.ser
CheckLength; local class incompatible: stream classdesc serialVersionUID = 1234567, lo
java.io.InvalidClassException: CheckLength;
 ...
```

• If serialVersionUID is not defined, Java will define one based on some properties of the class itself such as the class name, instance fields, and so on.

• This means you can not any more fields to the class.

• Adding *private int length = 4;* results in:

```
java CheckLengthRead 1234.ser
CheckLength; local class incompatible: stream classdesc serialVersionUID = 91318398384
java.io.InvalidClassException: CheckLength; local class incompatible: stream classdesc
 ...
```

• Further reading: Secure Coding Guidelines for Java SE (https://www.oracle.com/java/technologies/javase/seccodeguide.html)
published Sept/28/2020.

10.25. StreamTokenizer

The StreamTokenizer class takes an input stream and parses it into "tokens", allowing the tokens to be read one at a time. The parsing process is controlled by a table and a number of flags that can be set to various states. The stream tokenizer can recognize identifiers, numbers, quoted strings, and various comment styles.

Each byte read from the input stream is regarded as a character in the range '\u0000' through '\u00FF'. The character value is used to look up five possible attributes of the character: white space, alphabetic, numeric, string quote, and comment character. Each character can have zero or more of these attributes.

In addition, an instance has four flags. These flags indicate:

- Whether line terminators are to be returned as tokens or treated as white space that merely separates tokens.
- Whether C-style comments are to be recognized and skipped.
- Whether C++-style comments are to be recognized and skipped.
- Whether the characters of identifiers are converted to lowercase.

A typical application first constructs an instance of this class, sets up the syntax tables, and then repeatedly loops calling the nextToken method in each iteration of the loop until it returns the value TT_EOF.

The first program:

```
 1      import java.io.*;
 2      public class St_1 {
 3        public static void main( String args[] ) {
 4          StreamTokenizer input;
 5          if ( args.length > 1 )      {
 6              System.err.println("Usage: java St [file-name]");
 7              System.exit(1);
 8          }
 9          try {
10              String line;
11              if ( args.length == 1 )
12                  input = new StreamTokenizer( new FileReader(args[0]) );
13              else
14                  input = new StreamTokenizer(
15                              new InputStreamReader(System.in) );
16              while ( input.TT_EOF != input.nextToken() ) {
17                      System.out.println(input.lineno() + ": "
18                              + input.toString());
19              }
20          }
21          catch ( FileNotFoundException e)    {
22              System.out.println(e.getMessage());
23          }
24          catch ( IOException e)      {
25              System.out.println(e.getMessage());
26          }
27          catch ( Exception e)        {
28              System.out.println("Exception occurred: " + e.getMessage() );
29              e.printStackTrace();
30          }
31        }
32      }
```

Source Code: Src/9_was/St_1.java

Result:

```
% head -1 /etc/passwd
root:x:0:1:Super-User:/:/sbin/sh
% head -1 /etc/passwd  | java St_1
1: Token[root], line 1
1: Token[':'], line 1
1: Token[x], line 1
1: Token[':'], line 1
1: Token[n=0.0], line 1
1: Token[':'], line 1
1: Token[n=1.0], line 1
1: Token[':'], line 1
1: Token[Super-User], line 1
1: Token[':'], line 1

% java St_1 /etc/passwd | sed 7q
1: Token[root], line 1
1: Token[':'], line 1
1: Token[x], line 1
1: Token[':'], line 1
1: Token[n=0.0], line 1
1: Token[':'], line 1
1: Token[n=1.0], line 1

% java St_1
hello
1: Token[hello], line 1
a b:c d;e
2: Token[a], line 2
2: Token[b], line 2
2: Token[':'], line 2
2: Token[c], line 2
2: Token[d], line 2
2: Token[';'], line 2
2: Token[e], line 2

% java St_1
aa ///
1: Token[aa], line 1
sss // wwww
2: Token[sss], line 2
222 + #
3: Token[n=222.0], line 3
3: Token['+'], line 3
3: Token['#'], line 3
```

The second program is a beginning of a calculator:

```
 1      import java.io.*;
 2
 3      public class St_2 {
 4
 5        StreamTokenizer input;
 6
 7        public void adjustT() {
 8              input.resetSyntax();
 9              input.commentChar('#');          // comments from #
10                                               // to end-of-line
11              input.wordChars('0', '9');       // parse decimal
12                                               // numbers as words
13              input.wordChars('.', '.');
14              input.wordChars('+', '+');       // operators as words
15              input.wordChars('-', '-');       // operators as words
16              input.wordChars('*', '*');       // operators as words
17              input.wordChars('/', '/');       // operators as words
18              input.whitespaceChars(0, ' ');   // ignore white space
19              input.eolIsSignificant(true);    // need '\n'
20         }
21
22        public void processInput() throws IOException {
23              while ( input.TT_EOF != input.nextToken() ) {
24                      if ( input.ttype != input.TT_EOL )
25                              System.out.println(input.lineno() + ": " +
26                                                      input.sval);
27                              else
28                                      System.out.println("Saw EOL");
29              }
30       }
31
32
33        public static void main( String args[] ) {
34
35          St_2 aSt_2 = new St_2();
36
37          if ( args.length > 1 )      {
38              System.err.println(
39                   "Usage: java St [file-name]");
40              System.exit(1);
41          }
42
43          try {
44              String line;
45              if ( args.length == 1 )
46                  aSt_2.input = new StreamTokenizer(
47                                       new FileReader(args[0]) );
48              else
49                  aSt_2.input = new StreamTokenizer(
50                                        new InputStreamReader(System.in) );
51              aSt_2.adjustT();
```

```
52              aSt_2.processInput();
53
54          }
55          catch ( FileNotFoundException e)    {
56              System.out.println(e.getMessage());
57          }
58          catch ( IOException e)      {
59              System.out.println(e.getMessage());
60          }
61          catch ( Exception e)        {
62              System.out.println("ExceptionType occurred: " +
63                      e.getMessage() );
64              e.printStackTrace();
65          }
66        }
67      }
```

Source Code: Src/9_was/St_2.java

Result:

```
% java St_2
Saw EOL
2 + 3
1: 2
1: +
1: 3
Saw EOL
2 - 3
2: 2
2: -
2: 3
Saw EOL
```

A simple calculator:

The first idea for the process loop:

```
  public void processInput() throws IOException {
        while ( input.TT_EOF != new Expression(input) ) {
                ;
        }
 }
```

What do you think?

• Scanner:

```
 1
 2      import java.io.BufferedReader;
 3      import java.io.FilterReader;
 4      import java.io.IOException;
 5      import java.io.Reader;
 6      import java.io.StreamTokenizer;
 7
 8      /** lexical analyzer for arithmetic expressions.
 9              Comments extend from # to end of line.
10              Words are composed of digits and decimal point(s).
11              White space consists of control characters and space and is ignored;
12              however, end of line is returned.
13              Fixes the lookahead problem for TT_EOL.
14        */
15      public class Scanner extends StreamTokenizer {
16        /** kludge: pushes an anonymous Reader which inserts
17              a space after each newline.
18          */
19        public Scanner (Reader r) {
20          super (new FilterReader(new BufferedReader(r)) {
21            protected boolean addSpace;       // kludge to add space after \n
22            public int read () throws IOException {
23              int ch = addSpace ? ' ' : in.read();
24              addSpace = ch == '\n';
25              return ch;
26            }
27          });
28          resetSyntax();
29          commentChar('#');                   // comments from # to end-of-line
30          wordChars('0', '9');                // parse decimal numbers as words
31          wordChars('.', '.');
32          whitespaceChars(0, ' ');            // ignore control-* and space
33          eolIsSignificant(true);             // need '\n'
34        }
35      }
```

Source Code: Src/9_e/Scanner.java

• Expression.java

```
 1      /*
 2       * Thanks to ats
 3       */
 4      import java.io.InputStreamReader;
 5      import java.io.IOException;
 6      import java.io.ObjectOutputStream;
 7      import java.io.StreamTokenizer;
 8      import java.util.Vector;
 9
10      /** recognizes, stores, and evaluates arithmetic expressions.
11        */
12      public abstract class Expression {
13        final static int eol  = StreamTokenizer.TT_EOL; // switch use ...
14        final static int eof  = StreamTokenizer.TT_EOF; // must be const :(
15        final static int word  = StreamTokenizer.TT_WORD;
16
17        /** reads lines from standard input, parses, and evaluates them
18            or writes them as a Vector to standard output if -c is set.
19            @param args if -c is specified, a Vector is written.
20          */
21        public static void main (String args []) {
22          Scanner scanner = new Scanner(new InputStreamReader(System.in));
23          try {
24            do
25              try {
26                Number n = Expression.line(scanner);
27                System.out.println(n.floatValue());
28              } catch (java.lang.Exception e) {
29                System.err.println(scanner +": "+ e);
30                while (scanner.ttype != scanner.TT_EOL
31                       && scanner.nextToken() != scanner.TT_EOF)
32                  ;
33            } while (scanner.ttype == scanner.TT_EOL);
34          } catch (IOException ioe) { System.err.println(ioe); }
35        }
36        /** indicates parsing errors.
37          */
38        public static class Exception extends java.lang.Exception {
39          public Exception (String msg) {
40            super(msg);
41          }
42        }
43        /** recognizes line: sum '\n';
44            an empty line is silently ignored.
45            @param s source of first input symbol, may be at end of file.
46            @return tree for sum, null if only end of file is found.
47            @throws Exception for syntax error.
48            @throws IOException discovered on s.
49          */
50        public static Number line (Scanner s) throws Exception, IOException {
51          for (;;)
```

```
52            switch (s.nextToken()) {
53            default:
54              Number result = sum(s);
55              if (s.ttype != eol) throw new Exception("expecting nl");
56              return result;
57            case eol: continue;       // ignore empty line
58            case eof: return null;
59            }
60          }
61          /** recognizes product: term [{ ('*'|'%'|'/') term }];
62              @param s source of first input symbol, advanced beyond product.
63              @return tree with evaluators.
64              @see Expression#sum
65            */
66          public static Number product (Scanner s) throws Exception, IOException {
67            Number result = term(s);
68            for (;;)
69              switch (s.ttype) {
70              case '*':
71                s.nextToken();
72                result = new Node.Mul(result, term(s));
73                continue;
74              case '/':
75                s.nextToken();
76                result = new Node.Div(result, term(s));
77                continue;
78              case '%':
79                s.nextToken();
80                result = new Node.Mod(result, term(s));
81                continue;
82              default:
83                return result;
84              }
85          }
86          /** recognizes sum: product [{ ('+'|'-') product }];
87              @param s source of first input symbol, advanced beyond sum.
88              @return tree with evaluators.
89              @see Expression#line
90            */
91          public static Number sum (Scanner s) throws Exception, IOException {
92            Number result = product(s);
93            for (;;)
94              switch (s.ttype) {
95              case '+':
96                s.nextToken();
97                result = new Node.Add(result, product(s));
98                continue;
99              case '-':
00                s.nextToken();
01                result = new Node.Sub(result, product(s));
02                continue;
03              default:
04                return result;
05              }
```

```
06        }
07        /** recognizes term: '('sum')' | Number;
08            @param s source of first input symbol, advanced beyond term.
09            @return tree with evaluators.
10            @see Expression#sum
11          */
12        public static Number term (Scanner s) throws Exception, IOException {
13          switch (s.ttype) {
14          case '(':
15            s.nextToken();
16            Number result = sum(s);
17            if (s.ttype != ')') throw new Exception("expecting )");
18            s.nextToken();
19            return result;
20          case word:
21            result = s.sval.indexOf(".") < 0 ? (Number)new Long(s.sval)
22                                             : (Number)new Double(s.sval);
23            s.nextToken(); return result;
24          }
25          throw new Exception("missing term");
26        }
27      }       // end of class Expression
```

Source Code: Src/9_e/Expression.java

Node:

```
 1      /*
 2       * Thanks to ats
 3       */
 4
 5      import java.io.Serializable;
 6
 7      /** base class to store and evaluate arithmetic expressions.
 8          Defines most value-functions so that subclasses need only deal
 9          with long and double arithmetic.
10        */
11      public abstract class Node extends Number implements Serializable {
12
13        /** maps byte arithmetic to long.
14            @return truncated long value.
15          */
16        public byte byteValue () {
17          return (byte)longValue();
18        }
19
20        /** maps short arithmetic to long.
21            @return truncated long value.
22          */
23        public short shortValue () {
24          return (short)longValue();
25        }
26
27        /** maps int arithmetic to long.
```

```
28            @return truncated long value.
29          */
30        public int intValue () {
31          return (int)longValue();
32        }
33
34        /** maps float arithmetic to double.
35            @return truncated double value.
36          */
37        public float floatValue () {
38          return (float)doubleValue();
39        }
40
41        /** represents a binary operator.
42            Must be subclassed to provide evaluation.
43          */
44        protected abstract static class Binary extends Node {
45          /** left operand subtree.
46              @serial left operand subtree.
47            */
48          protected Number left;
49
50          /** right operand subtree.
51              @serial right operand subtree.
52            */
53          protected Number right;
54
55          /** builds a node with two subtrees.
56              @param left left subtree.
57              @param right right subtree.
58            */
59          protected Binary (Number left, Number right) {
60            this.left = left; this.right = right;
61          }
62        }
63
64        /** represents a unary operator.
65            Must be subclassed to provide evaluation.
66          */
67        protected abstract static class Unary extends Node {
68          /** operand subtree.
69              @serial operand subtree.
70            */
71          protected Number tree;
72
73          /** builds a node with a subtree.
74              @param tree subtree.
75            */
76          protected Unary (Number tree) {
77            this.tree = tree;
78          }
79        }
80
81        /** implements addition.
```

```
82          */
83        public static class Add extends Binary {
84          /** builds a node with two subtrees.
85              @param left left subtree.
86              @param right right subtree.
87            */
88          public Add (Number left, Number right) {
89            super(left, right);
90          }
91
92          /** implements long addition.
93              @return sum of subtree values.
94            */
95          public long longValue () {
96            return left.longValue() + right.longValue();
97          }
98
99          /** implements double addition.
00              @return sum of subtree values.
01            */
02          public double doubleValue () {
03            return left.doubleValue() + right.doubleValue();
04          }
05        }
06
07        /** implements subtraction.
08          */
09        public static class Sub extends Binary {
10          /** builds a node with two subtrees.
11              @param left left subtree.
12              @param right right subtree.
13            */
14          public Sub (Number left, Number right) {
15            super(left, right);
16          }
17
18          /** implements long subtraction.
19              @return difference of subtree values.
20            */
21          public long longValue () {
22            return left.longValue() - right.longValue();
23          }
24
25          /** implements double subtraction.
26              @return difference of subtree values.
27            */
28          public double doubleValue () {
29            return left.doubleValue() - right.doubleValue();
30          }
31        }
32
33        /** implements multiplication.
34          */
35        public static class Mul extends Binary {
```

```
36          /** builds a node with two subtrees.
37              @param left left subtree.
38              @param right right subtree.
39            */
40          public Mul (Number left, Number right) {
41            super(left, right);
42          }
43
44          /** implements long multiplication.
45              @return product of subtree values.
46            */
47          public long longValue () {
48            return left.longValue() * right.longValue();
49          }
50
51          /** implements double multiplication.
52              @return product of subtree values.
53            */
54          public double doubleValue () {
55            return left.doubleValue() * right.doubleValue();
56          }
57        }
58
59        /** implements division.
60          */
61        public static class Div extends Binary {
62          /** builds a node with two subtrees.
63              @param left left subtree.
64              @param right right subtree.
65            */
66          public Div (Number left, Number right) {
67            super(left, right);
68          }
69
70          /** implements long division.
71              @return quotient of subtree values.
72            */
73          public long longValue () {
74            return left.longValue() / right.longValue();
75          }
76
77          /** implements double division.
78              @return quotient of subtree values.
79            */
80          public double doubleValue () {
81            return left.doubleValue() / right.doubleValue();
82          }
83        }
84
85        /** implements modulus.
86          */
87        public static class Mod extends Binary {
88          /** builds a node with two subtrees.
89              @param left left subtree.
```

```
90              @param right right subtree.
91            */
92          public Mod (Number left, Number right) {
93            super(left, right);
94          }
95
96          /** implements long modulus.
97              @return remainder after division of subtree values.
98            */
99          public long longValue () {
00            return left.longValue() % right.longValue();
01          }
02
03          /** implements double modulus.
04              @return remainder after division of subtree values.
05            */
06          public double doubleValue () {
07            return left.doubleValue() % right.doubleValue();
08          }
09        }
10
11        /** implements sign change.
12          */
13        public static class Minus extends Unary {
14          /** builds a node with a subtree.
15              @param tree subtree.
16            */
17          public Minus (Number tree) {
18            super(tree);
19          }
20
21          /** implements long sign change.
22              @return negative of subtree value.
23            */
24          public long longValue () {
25            return - tree.longValue();
26          }
27
28          /** implements double sign change.
29              @return negative of subtree values.
30            */
31          public double doubleValue () {
32            return - tree.doubleValue();
33          }
34        }
35      }
```

Source Code: Src/9_e/Node.java

Result:

```
% java Expression
2 * ( 1 - 2 )
-2.0
```

10.26. Not Discussed.

• PipedInputStream

A piped input stream should be connected to a piped output stream; the piped input stream then provides whatever data bytes are written to the piped output stream. Typically, data is read from a PipedInputStream object by one thread and data is written to the corresponding PipedOutputStream by some other thread. Attempting to use both objects from a single thread is not recommended, as it may deadlock the thread. The piped input stream contains a buffer, decoupling read operations from write operations, within limits

• PushbackInputStream

A PushbackInputStream adds functionality to another input stream, namely the ability to "push back" or "unread" one byte. This is useful in situations where it is convenient for a fragment of code to read an indefinite number of data bytes that are delimited by a particular byte value; after reading the terminating byte, the code fragment can "unread" it, so that the next read operation on the input stream will reread the byte that was pushed back. For example, bytes representing the characters constituting an identifier might be terminated by a byte representing an operator character; a method whose job is to read just an identifier can read until it sees the operator and then push the operator back to be re-read.

• DigestInputStream

A transparent stream that updates the associated message digest using the bits going through the stream. To complete the message digest computation, call one of the digest methods on the associated message digest after your calls to one of this digest input stream's read methods.

• JCE (Java Crypto Package)

• ZipIn/Out-putStream

This class implements an input stream filter for reading files in the ZIP file format. Includes support for both compressed and uncompressed entries.

• and more .....

10.27. try-with-resources Statement

from here (https://docs.oracle.com/javase/tutorial/essential/exceptions/tryResourceClose.html)

• try-with-resources statement is a try statement that declares one or more resources

• resource is an object that must be closed after the program is finished with it

• try-with-resources statement ensures that each resource is closed at the end of the statemento

Example: Not using try-with-resources

```
 1      // first line
 2      import java.io.*;
 3      public class TryWithOutResourceAndFinally {
 4
 5
 6      static void readAndPrint(String inF ) throws IOException  {
 7              BufferedReader in = null;
 8
 9              try     {
10                      in = new BufferedReader( new FileReader(inF) );
11                      System.out.println(in.readLine() );
12              } catch (Exception e) {
13                      System.out.println("Could not open file");
```

```
14                      e.printStackTrace();
15              } finally {
16                      if ( in != null )
17                              in.close();
18              }
19        }
20
21        public static void main(String args[]) {
22              if ( args.length != 1 )  {
23                      System.out.println("Usage: java FileIO file");
24              } else  {
25                      System.out.println("Inputfile: " + args[0]);
26                      try {
27                              readAndPrint(args[0]);
28                      } catch (Exception e) {
29                              e.printStackTrace();
30                      }
31              }
32        }
33      }
```

Source Code: Src/7/TryWithOutResourceAndFinally.java

Example: Not using try-with-resources

```
1       // first line
2       import java.io.*;
3       public class TryWithResource {
4
5
6       static void readAndPrint(String inF ) throws IOException  {
7
8               try (
9                       BufferedReader in = new BufferedReader( new FileReader(inF) );
10                  ) {
11                              System.out.println(in.readLine() );
12              } catch (Exception e) {
13                      System.out.println("Could not open file");
14                      e.printStackTrace();
15              }
16        }
17
18        public static void main(String args[]) {
19              if ( args.length != 1 )  {
20                      System.out.println("Usage: java FileIO file");
21              } else  {
22                      System.out.println("Inputfile: " + args[0]);
23                      try {
24                              readAndPrint(args[0]);
25                      } catch (Exception e) {
26                              e.printStackTrace();
27                      }
28              }
```

```
29        }
30      }
```

Source Code: Src/7/TryWithResource.java

Output:

```
% java TryWithResource TryWithResource.java
Inputfile: TryWithResource.java
// first line
```

10.28. Examples from the JDK Distribution

The following are examples from the JDK 10 release I used them as are.

10.29. Examples from the JDK Distribution: CustomAutoCloseableSample.java

```
1       /*
2        * Copyright (c) 2014, Oracle and/or its affiliates. All rights reserved.
3        *
4        * Redistribution and use in source and binary forms, with or without
5        * modification, are permitted provided that the following conditions
6        * are met:
7        *
8        *   - Redistributions of source code must retain the above copyright
9        *     notice, this list of conditions and the following disclaimer.
10       *
11       *   - Redistributions in binary form must reproduce the above copyright
12       *     notice, this list of conditions and the following disclaimer in the
13       *     documentation and/or other materials provided with the distribution.
14       *
15       *   - Neither the name of Oracle nor the names of its
16       *     contributors may be used to endorse or promote products derived
17       *     from this software without specific prior written permission.
18       *
19       * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
20       * IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
21       * THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
22       * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
23       * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
24       * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
25       * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
26       * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
27       * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
28       * NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
29       * SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
30       */
31
32      /*
33       * This source code is provided to illustrate the usage of a given feature
34       * or technique and has been deliberately simplified. Additional steps
35       * required for a production-quality application, such as security checks,
36       * input validation, and proper error handling, might not be present in
```

```
37       * this sample code.
38       */
39
40      import java.io.BufferedOutputStream;
41      import java.io.IOException;
42      import java.io.OutputStream;
43      import java.io.PrintStream;
44      import java.nio.file.Files;
45      import java.nio.file.Path;
46      import java.nio.file.Paths;
47
48      /**
49       * This sample demonstrates the ability to create custom resource that
50       * implements the {@code AutoCloseable} interface. This resource can be used i
51       * the try-with-resources construct.
52       */
53      public class CustomAutoCloseableSample {
54
55          /**
56           * The main method for the CustomAutoCloseableSample program.
57           *
58           * @param args is not used.
59           */
60          public static void main(String[] args) {
61              /*
62               * TeeStream will be closed automatically after the try block.
63               */
64              try (TeeStream teeStream = new TeeStream(System.out, Paths.get("out.tx
65                   PrintStream out = new PrintStream(teeStream)) {
66                  out.print("Hello, world");
67              } catch (Exception e) {
68                  e.printStackTrace();
69                  System.exit(1);
70              }
71          }
72
73          /**
74           * Passes the output through to the specified output stream while copying
75           * The TeeStream functionality is similar to the Unix tee utility.
76           * TeeStream implements AutoCloseable interface. See OutputStream for deta
77           */
78          public static class TeeStream extends OutputStream {
79
80              private final OutputStream fileStream;
81              private final OutputStream outputStream;
82
83              /**
84               * Creates a TeeStream.
85               *
86               * @param outputStream an output stream.
87               * @param outputFile   an path to file.
88               * @throws IOException If an I/O error occurs.
89               */
90              public TeeStream(OutputStream outputStream, Path outputFile) throws IO
```

```
91                  this.fileStream = new BufferedOutputStream(Files.newOutputStream(o
92                  this.outputStream = outputStream;
93              }
94
95              /**
96               * Writes the specified byte to the specified output stream
97               * and copies it to the file.
98               *
99               * @param b the byte to be written.
00               * @throws IOException If an I/O error occurs.
01               */
02              @Override
03              public void write(int b) throws IOException {
04                  fileStream.write(b);
05                  outputStream.write(b);
06              }
07
08              /**
09               * Flushes this output stream and forces any buffered output bytes
10               * to be written out.
11               * The <code>flush</code> method of <code>TeeStream</code> flushes
12               * the specified output stream and the file output stream.
13               *
14               * @throws IOException if an I/O error occurs.
15               */
16              @Override
17              public void flush() throws IOException {
18                  outputStream.flush();
19                  fileStream.flush();
20              }
21
22              /**
23               * Closes underlying streams and resources.
24               * The external output stream won't be closed.
25               * This method is the member of AutoCloseable interface and
26               * it will be invoked automatically after the try-with-resources block
27               *
28               * @throws IOException If an I/O error occurs.
29               */
30              @Override
31              public void close() throws IOException {
32                  try (OutputStream file = fileStream) {
33                      flush();
34                  }
35              }
36          }
37      }
```

Source Code: Src/7_JDK/CustomAutoCloseableSample.java

10.30. Examples from the JDK Distribution: Unzip.java

```
/*
 * Copyright (c) 2014, Oracle and/or its affiliates. All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 *   - Redistributions of source code must retain the above copyright
 *     notice, this list of conditions and the following disclaimer.
 *
 *   - Redistributions in binary form must reproduce the above copyright
 *     notice, this list of conditions and the following disclaimer in the
 *     documentation and/or other materials provided with the distribution.
 *
 *   - Neither the name of Oracle nor the names of its
 *     contributors may be used to endorse or promote products derived
 *     from this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
 * IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
 * THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
 * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
 * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
 * NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
 * SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

/*
 * This source code is provided to illustrate the usage of a given feature
 * or technique and has been deliberately simplified. Additional steps
 * required for a production-quality application, such as security checks,
 * input validation, and proper error handling, might not be present in
 * this sample code.
 */

import java.io.IOException;
import java.io.UncheckedIOException;
import java.nio.file.*;

import static java.nio.file.StandardCopyOption.REPLACE_EXISTING;

/**
 * Extract (unzip) a file to the current directory.
 */
public class Unzip {

    /**
     * The main method for the Unzip program. Run the program with an empty
     * argument list to see possible arguments.
     *
```

```
55           * @param args the argument list for {@code Unzip}.
56           */
57          public static void main(String[] args) {
58              if (args.length != 1) {
59                  System.out.println("Usage: Unzip zipfile");
60              }
61              final Path destDir = Paths.get(".");
62              /*
63               * Create AutoCloseable FileSystem. It will be closed automatically
64               * after the try block.
65               */
66              try (FileSystem zipFileSystem = FileSystems.newFileSystem(Paths.get(ar
67
68                  Path top = zipFileSystem.getPath("/");
69                  Files.walk(top).skip(1).forEach(file -> {
70                      Path target = destDir.resolve(top.relativize(file).toString())
71                      System.out.println("Extracting " + target);
72                      try {
73                          Files.copy(file, target, REPLACE_EXISTING);
74                      } catch (IOException e) {
75                          throw new UncheckedIOException(e);
76                      }
77                  });
78              } catch (UncheckedIOException | IOException e) {
79                  e.printStackTrace();
80                  System.exit(1);
81              }
82          }
83      }
```

Source Code: Src/7_JDK/Unzip.java

10.31. Examples from the JDK Distribution: ZipCat.java

```
1       /*
2        * Copyright (c) 2014, Oracle and/or its affiliates. All rights reserved.
3        *
4        * Redistribution and use in source and binary forms, with or without
5        * modification, are permitted provided that the following conditions
6        * are met:
7        *
8        *   - Redistributions of source code must retain the above copyright
9        *     notice, this list of conditions and the following disclaimer.
10       *
11       *   - Redistributions in binary form must reproduce the above copyright
12       *     notice, this list of conditions and the following disclaimer in the
13       *     documentation and/or other materials provided with the distribution.
14       *
15       *   - Neither the name of Oracle nor the names of its
16       *     contributors may be used to endorse or promote products derived
17       *     from this software without specific prior written permission.
18       *
19       * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
```

```
20       * IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
21       * THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
22       * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
23       * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
24       * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
25       * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
26       * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
27       * LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
28       * NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
29       * SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
30       */
31
32      /*
33       * This source code is provided to illustrate the usage of a given feature
34       * or technique and has been deliberately simplified. Additional steps
35       * required for a production-quality application, such as security checks,
36       * input validation, and proper error handling, might not be present in
37       * this sample code.
38       */
39
40      import java.io.IOException;
41      import java.io.InputStream;
42      import java.nio.file.FileSystem;
43      import java.nio.file.FileSystems;
44      import java.nio.file.Files;
45      import java.nio.file.Paths;
46
47      /**
48       * Prints data of the specified file to standard output from a zip archive.
49       */
50      public class ZipCat {
51
52          /**
53           * The main method for the ZipCat program. Run the program with an empty
54           * argument list to see possible arguments.
55           *
56           * @param args the argument list for ZipCat
57           */
58          public static void main(String[] args) {
59              if (args.length != 2) {
60                  System.out.println("Usage: ZipCat zipfile fileToPrint");
61              }
62              /*
63               * Creates AutoCloseable FileSystem and BufferedReader.
64               * They will be closed automatically after the try block.
65               * If reader initialization fails, then zipFileSystem will be closed
66               * automatically.
67               */
68              try (FileSystem zipFileSystem
69                      = FileSystems.newFileSystem(Paths.get(args[0]),null);
70                      InputStream input
71                      = Files.newInputStream(zipFileSystem.getPath(args[1]))) {
72                          byte[] buffer = new byte[1024];
73                          int len;
```

```
74                          while ((len = input.read(buffer)) != -1) {
75                              System.out.write(buffer, 0, len);
76                          }
77
78              } catch (IOException e) {
79                  e.printStackTrace();
80                  System.exit(1);
81              }
82          }
83      }
```

Source Code: Src/7_JDK/ZipCat.java

11. Threads

See also: here. (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/lang/Thread.html)

11.1. Intro

• Java programs, applications, and applets can consists of threads which conceptually are executed in parallel.

• This section demonstrates with simple examples how threads are created and manipulated.

• How exclusive access to common variables can be achieved.

• Classical examples concern communication with semaphores and conditional access to resources will be discussed.

11.2. Principles and Features

- A thread is a thread of execution in a program. The Java Virtual Machine allows an application to have multiple threads of execution running concurrently.
- Every thread has a priority. Threads with higher priority are executed in preference to threads with lower priority.
- Each thread may or may not also be marked as a daemon. The Java Virtual Machine exits when the only threads running are daemon threads.
- When code running in some thread creates a new Thread object, the new thread has its priority initially set equal to the priority of the creating thread, and is a daemon thread if and only if the creating thread is a daemon, unless specified otherwise.
- As is everything else, during construction the here. (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/lang/Thread.html)
  object is controlled by method; afterwards it's execution can be controlled through *start()*, *setPriority()*,
  ... investigated with *getPriority()* and *isAlive()*. JDK 1.1 is supposed to implement interruptions as well

The Thread java doc page: here. (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/lang/Thread.html)

11.3. Creation and Using

- Threads can be created using an instance of a class which is a subclass of Thread.
- Threads can be created using an instance of a class that implements the Runnable (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/lang/Runnable.html) interface.

Why are there two different ways?

```
public interface Runnable  {
       public abstract void run();
}
```

Start:

```
YourClass extends Threads

                yourThread Object =  new YourClass
                yourThread.start();
------

Thread.start();
                put yourThread reference in scheduler

Scheduler:
               if ( yourThread.run() has not been execcuted )
                     yourThread.run
               else
                     continue whereEver thread_pointer is
```

Runnable:

```
YourClass implements Runnable

                yourThread Object =  new YourClass
                new Thread(Object =  new YourClass).start()

Thread.start();
                put yourThread reference in scheduler

Scheduler:
               if ( yourThread.run() has not been execcuted )
                     yourThread.run
               else
                     continue whereEver thread_pointer is
```

First example:

```
 1
 2      public class Thread_0 extends Thread    {
 3          private String info;
 4          static Object o = new Object();
 5
 6          public Thread_0 (String info) {
 7              this.info    = info;
 8          }
 9
10          public void run () {
11              synchronized ( o )      {
12                      System.err.println(info + " ---> ");
13                      try { sleep(1000); } catch (  InterruptedException e ) { }
14                      System.err.println(info + " <--- ");
15              }
16          }
17
18          public static void main (String args []) {
19              Thread_0 aT4_0 = new Thread_0("first");
20              Thread_0 aT4_1 = new Thread_0("second");
21              Thread   aT4_2 = new Thread_0("third");
22
23              aT4_0.start();
24              aT4_1.start();
25              aT4_2.start();
26          }
27      }
```

Source Code: Src/11/Thread_0.java

Example: run

```
 1
 2      public class Thread_1 extends Thread    {
 3
 4              private int info;
 5              static int x = 0;
 6
 7              public Thread_1 (int info) {
 8                      this.info = info;
 9              }
10
11              public void run () {
12                      if ( info == 1 )        {
13                              x = 3;
14                              try { sleep(100); } catch (Exception e ) {}
15                      } else
16                              x = 1;
17              }
18
19              public static void main (String args []) {
20                      Thread_1 aT1  = new Thread_1(1);
21                      Thread_1 aT2  = new Thread_1(2);
22                      aT1.start();
```

```
23                      aT2.start();
24                      System.err.println(x);
25              }
26      }
```

Source Code: Src/11/Thread_1.java

Result:

```
% java Thread_1
3
```

```
 1
 2      public class Thread_example extends Thread      {
 3
 4              static int value = 0;   // static?
 5              int id;
 6
 7              public Thread_example (int id) {
 8                      this.id = id;
 9              }
10
11              public int getValue() {
12                      return value;
13              }
14              public void compute()   {
15                      if ( id == 1 )
16                      try { sleep(1000); } catch (  InterruptedException e ) { }
17                      if ( id == 1 )  {
18                              value = 1;
19                      }
20                      if ( id == 2 )  {
21                              value = 2;
22                      }
23              }
24              public void run () {
25                      compute();
26              }
27
28              public static void main (String args []) {
29                      Thread_example aT1  = new Thread_example(1);
30                      Thread_example aT2  = new Thread_example(2);
31                      aT1.start();
32                      aT2.run();
33                      int aT2Value = aT2.getValue();
34                      try { sleep(2000); } catch (  InterruptedException e ) { }
35                      int aT1Value = aT1.getValue();
36                      System.out.println(aT1Value + aT2Value);
37              }
38      }
```

Source Code: Src/11/Thread_example.java

Result:

```
% java Thread_example
3
```

What kinds of output are possible?

Example: runnable

```
 1
 2      public class Thread_1b implements Runnable {
 3
 4              private String info;
 5              int x = 0;
 6
 7              public Thread_1b (String info) {
 8                      this.info = info;
 9              }
10
11              public void run () {
12                      x=1;
13                      System.out.print(info);
14              }
15
16              public static void main (String args []) {
17                      if (args != null)       {
18                              for (int n = 0; n < args.length; ++ n)  {
19                                      new Thread( new Thread_1b("" + n ) ).start();
20                              }
21                      }
22              }
23      }
```

Source Code: Src/11/Thread_1b.java

Output:

```
% java Thread_1 a b c d e f g h i j k l m n o p q r s
bdfhjlnpracegikmoqs%
```

Next

Example: Termination

```
 1
 2      public class Thread_2 extends Thread    {
 3
 4              private String info;
 5
 6              public Thread_2 (String info) {
 7                      this.info = info;
 8              }
 9
10              public void run () {
11                      long sleep = (int)(Math.random() * 10000);
12                      System.out.println(info + " sleeps for " + sleep );
13                      try {
14                              sleep(sleep);
15                      }
16                      catch (  InterruptedException e ) {
17                              e.getMessage();
18                      }
19              }
20
21              public static void main (String args []) {
22                      int count = 0;
23                      if (args != null)
24                      for (int n = 0; n < args.length; ++ n)  {
25                              Thread_2 aT1  = new Thread_2(args[n]);
26                              if ( n % 2 == 0 )
27                                      aT1.setPriority(Thread.MIN_PRIORITY);
28                              aT1.start();
29                      }
30                      while ( count != 1 )    {
31                              try {
32                                      count = activeCount();
33                                      System.out.println("activeCount() = " +
34                                              count );
35                                      sleep(500);
36                              }
37                              catch (  InterruptedException e ) {
38                                      e.getMessage();
39                              }
40                      }
41
42              }
43      }
```

Source Code: Src/11/Thread_2.java

```
java  Thread_2 a b c d
activeCount() = 5
b sleeps for 1063
d sleeps for 8295
a sleeps for 2197
c sleeps for 2619
activeCount() = 5
activeCount() = 5
activeCount() = 4
activeCount() = 4
activeCount() = 3
activeCount() = 2
 ..
activeCount() = 2
activeCount() = 1
```

Stolen from Java doc (sun)

- Threads belong to groups, represented as ThreadGroup objects and ordered hierarchically. Through a group, several threads can be controlled together; e.g., a group defines a maximum priority for it's members. As a thread can only influence threads in it's own group, the extent of a thread's influence on others can be limited.
- Threads and groups have names, which, however, are mostly useful for documentation and debugging. Theoretically, the classes can be used to identify all threads and all groups; this can be used to build a thread variant of *ps(1)*.
- Internally, the Java runtime system uses several threads that deal, for example, with unreachable objects (garbage collection). If a main program is started, this takes place in the thread main. Once a window is opened, more threads are added.

The execution of the Java Virtual Machine ends once most threads reach the end of their run() methods. ''Most'' means that there are user and daemon threads and thread groups; execution of the JVM ends when there are no more user threads. The distinction is made by calling setDaemon() ; the distinction is necessary, because threads that deliver events and manage painting in a window system are irrelevant for deciding if an application has completed it's job

11.4. Interruption

```
 1      public class InterruptExample extends Thread {
 2
 3
 4          public InterruptExample(String name) {
 5                  setName(name);
 6          }
 7
 8          public static void sleepForAbit(long sleepTime )    {
 9              try {
10                  sleep(sleepTime);
11              } catch (InterruptedException e) {
12                  System.err.println(Thread.currentThread().getName() + " was interr
13              }
14
15          }
16          public void run() {
17
18              System.err.println(getName() + " has started!");
19              double x = 1;
20              while ( x > 0 ) {               // forever loop
21                      x = x * 2 - x;          // x is constant
22                      sleepForAbit(200);
23                      if ( isInterrupted() )  {
24                              System.err.println(Thread.currentThread().getName() +
25                              return;
26                      }
27
28              }
29              System.err.println(getName() + " has exited!");
30          }
31
32          public static void main(String args[]){
33
34              InterruptExample aThread = new InterruptExample("aThread");
35              aThread.start();
36              sleepForAbit(100);      // should allow the thread to enter the while
37              aThread.interrupt();
38
39          }
40      }
```

Source Code: Src/11/InterruptExample.java

State: see java doc

```
% java InterruptExample
aThread has started!
aThreadis interrupted
aThread has exited
```

```
if (Thread.interrupted())  // Clears interrupted status!
      throw new InterruptedException();
```

Extract from Javadoc:

The interrupted status of the current thread is cleared when this exception is thrown.

11.5. Join

```
 1      public class Join extends Thread        {
 2              private String info;
 3              Join aT1;
 4
 5              public Join (String info) {
 6                      this.info = info;
 7              }
 8
 9              public void run () {
10                      System.out.println(info + " is running");
11                      try {
12                              sleep(1000);
13                      }
14                      catch (  InterruptedException e ) {
15                              System.err.println("Interrupted!");
16                      }
17                      System.out.println(info + ": exit run");
18
19              }
20              public static void main (String args []) {
21                      Join aT1  = new Join("first");
22
23                      aT1.start();
24
25                      try {
26                              aT1.join();
27                              System.err.println("Got it");
28                      }
29                      catch (  InterruptedException e ) {
30                              e.printStackTrace();
31                      }
32                      System.err.println("main end");
33              }
34      }
```

Source Code: Src/11/Join.java

Result:

```
first is running
third is running
second is running
second: exit run
third: exit run
first: exit run
Got it
```

Extract from Javadoc:

```
The interrupted status of the
current thread is cleared when
this exception is thrown.
```

11.6. Join II

Does this code produces the correct output?

```
 1      // fuer jedes resultat ein der es ausrechnet
 2      // auf das ende der ausrechner muss gewartet werden
 3      public class Evaluator extends Thread {
 4              int i, j;
 5              final static int MAX = 2;
 6              Evaluator()     {
 7              }
 8              Evaluator(int i, int j) {
 9                      this.i = i;
10                      this.j = j;
11              }
12              static int a[][] = new int[MAX][MAX];
13              static int b[][] = new int[MAX][MAX];
14              static int c[][] = new int[MAX][MAX];
15              public void run(){
16                      for ( int index = 0; index < MAX; index ++ )    {
17                              try { sleep (100); } catch (Exception e ){} ;
18                              c[i][j] += a[i][index] * b[index][j];
19                      }
20              }
21              public String print(int a[][], String whichOne){
22                      String rValue = whichOne + ": \n";
23                      for(int i = 0; i < MAX; i++){
24                              for(int j =0; j < MAX; j++){
25                                      rValue += a[i][j] + "  ";
26                              }
27                              rValue = rValue + "\n";
28                      }
29
30                      return rValue;
31              }
32              public void init()      {
33                      for(int i = 0; i < MAX; i++){                   // i-->
34                              for(int j =0; j < MAX; j++){            // j
35                                      a[i][j] = b[i][j] = 2 + i + j ;         // |
36                              }                                       // v
37                      }
38
39              }
40              public String toString(){
41                      String rValue = print(a, "A") + print(b, "B" ) + print(c, "C")
42                      return rValue;
43              }
44              public void multiply(){
45                      Evaluator et[] = new Evaluator[ MAX * MAX];
46                      for(int i = 0; i < MAX; i++){
47                              for(int j =0; j < MAX; j++){
48                                      new Evaluator(i, j).start();
49                              }
50                      }
```

```
51                      System.out.println(this);
52              }
53              public static void main(String[] args) {
54                      Evaluator eval = new Evaluator();
55                      eval.init();
56                      eval.multiply();
57              }
58      }
```

Source Code: Src/11/Evaluator.java

Does this code produces he correct output? Any problems?

```
 1      // es gibt immer nur einen ausrechner, aber das ergebniss is richtig
 2      public class Evaluator_2 extends Thread {
 3              int i, j;
 4              final static int MAX = 2;
 5              Evaluator_2()   {
 6              }
 7              Evaluator_2(int i, int j)       {
 8                      this.i = i;
 9                      this.j = j;
10              }
11              static int a[][] = new int[MAX][MAX];
12              static int b[][] = new int[MAX][MAX];
13              static int c[][] = new int[MAX][MAX];
14              public void run(){
15                      for ( int index = 0; index < MAX; index ++ )    {
16                              c[i][j] += a[i][index] * b[index][j];
17                      }
18              }
19              public String print(int a[][], String whichOne){
20                      String rValue = whichOne + ": \n";
21                      for(int i = 0; i < MAX; i++){
22                              for(int j =0; j < MAX; j++){
23                                      rValue += a[i][j] + "  ";
24                              }
25                              rValue = rValue + "\n";
26                      }
27
28                      return rValue;
29              }
30              public void init()      {
31                      for(int i = 0; i < MAX; i++){                   // i-->
32                              for(int j =0; j < MAX; j++){            // j
33                                      a[i][j] = b[i][j] = 2 + i + j ;         // |
34                              }                                               // v
35                      }
36
37              }
38              public String toString(){
39                      String rValue = print(a, "A") + print(b, "B" ) + print(c, "C")
40                      return rValue;
41              }
```

```
42              public void multiply(){
43                      System.out.println(this);
44                      for(int i = 0; i < MAX; i++){
45                              for(int j =0; j < MAX; j++){
46                                      Evaluator_2 et = new Evaluator_2(i, j);
47                                      et.start();
48                                      try {
49                                              et.join();
50                                      } catch ( Exception e ) {
51                                      }
52                              }
53                      }
54                      System.out.println(this);
55              }
56              public static void main(String[] args) {
57                      Evaluator_2 eval = new Evaluator_2();
58                      eval.init();
59                      eval.multiply();
60              }
61      }
```

Source Code: Src/11/Evaluator_2.java

Next

```
 1      public class Evaluator_3 extends Thread {
 2              int i, j;
 3              final static int MAX = 2;
 4              Evaluator_3()   {
 5              }
 6              Evaluator_3(int i, int j)       {
 7                      this.i = i;
 8                      this.j = j;
 9              }
10              static int a[][] = new int[MAX][MAX];
11              static int b[][] = new int[MAX][MAX];
12              static int c[][] = new int[MAX][MAX];
13              public void run(){
14                      for ( int index = 0; index < MAX; index ++ )    {
15                              c[i][j] += a[i][index] * b[index][j];
16                      }
17              }
18              public String print(int a[][], String whichOne){
19                      String rValue = whichOne + ": \n";
20                      for(int i = 0; i < MAX; i++){
21                              for(int j =0; j < MAX; j++){
22                                      rValue += a[i][j] + "  ";
23                              }
24                              rValue = rValue + "\n";
25                      }
26
27                      return rValue;
28              }
29              public void init()      {
```

```
30                      for(int i = 0; i < MAX; i++){                   // i-->
31                              for(int j =0; j < MAX; j++){            // j
32                                      a[i][j] = b[i][j] = 2 + i + j ;         // |
33                              }                                       // v
34                      }
35
36              }
37              public String toString(){
38                      String rValue = print(a, "A") + print(b, "B" ) + print(c, "C")
39                      return rValue;
40              }
41              public void multiply(){
42                      Evaluator_3 et[] = new Evaluator_3[ MAX * MAX];
43                      System.out.println(this);
44                      for(int counter = 0; counter < MAX; counter++){
45                              for(int j =0; j < MAX; j++){
46                                      et[counter] = new Evaluator_3(i, j);
47                                      et[counter].start();
48                              }
49                      }
50                      for(int counter = 0; i < MAX * MAX; i++){
51                              try{
52                                      et[counter].join();
53                              }
54                              catch(InterruptedException e){
55                                      System.out.println("Interrupted!");
56                              }
57                      }
58                      System.out.println(this);
59              }
60              public static void main(String[] args) {
61                      Evaluator_3 eval = new Evaluator_3();
62                      eval.init();
63                      eval.multiply();
64              }
65      }
```

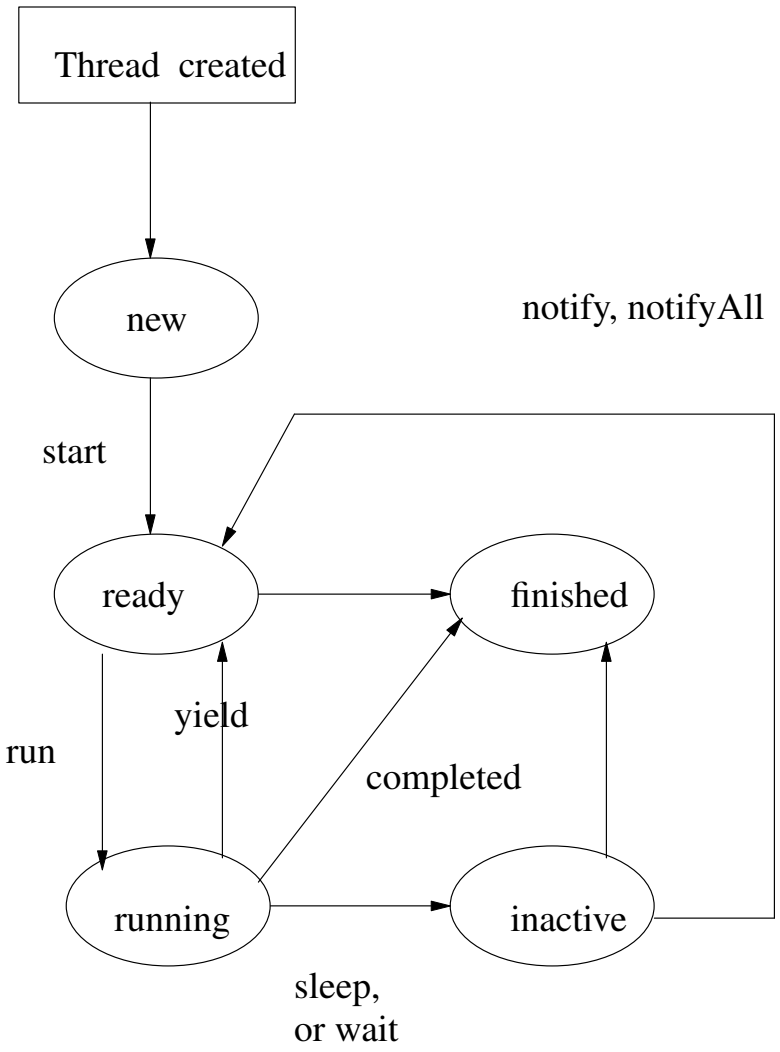
Source Code: Src/11/Evaluator_3.java

11.7. Using the Interface Runnable

```
 1      import java.util.*;
 2
 3      public class Thread_R extends Date implements Runnable  {
 4              private String name;
 5              private Vector aVector;
 6
 7              public Thread_R (String name) {
 8                      this.name = name;
 9              }
10
11              public void run () {
12                      System.out.println("Hi :) ... my name is: " + name );
13              }
14
15              public static void main (String args []) {
16                      String names[] = { "bono", "U2" };
17                      for ( int index = 0; index < names.length; index ++ )   {
18                              new Thread( new Thread_R( names[index] ) ).start();
19                      }
20              }
21      }
```

Source Code: Src/11/Thread_R.java

11.8. Thread States



Thread (http://download.oracle.com/javase/6/docs/api/java/lang/Thread.html)
Thread (http://download.oracle.com/javase/6/docs/api/java/lang/Thread.html)

11.9. Threads and Cores

```
 1      import java.util.*;
 2
 3      public class CoresTest extends Thread    {
 4          static final int soManyThreads  = Runtime.getRuntime().availableProcessors
 5          static final int soOftenPerThread = 10000000;
 6          static final int multiplier       = 100000000;
 7          static long milliSeconds = 0;
 8          double result = 0;
 9          int id;
10
11          public CoresTest(int index) {
12              id = index;
13          }
14          public static void init()  {
15              milliSeconds = System.currentTimeMillis();
16          }
17          public static void end(String s)   {
18              System.err.println(s + ":       " + ( System.currentTimeMillis() - mil
19              System.err.println(" # of cores" +   ":       " +
20                                                   Runtime.getRuntime().availableProcess
21          }
22          public void singleThreadTest (int soOften) {
23              for (int index = 0; index < soOften; index ++ )         {
24                      for (int k = 0; k < multiplier; k ++ )          {
25                              result = Math.sqrt(index) + Math.sqrt(index) + Math.sq
26                      }
27              }
28          }
29          public void run () {
30              singleThreadTest(soOftenPerThread);
31          }
32          public static void main (String args []) {
33              CoresTest single = new CoresTest(0);
34              CoresTest[] many   = new CoresTest[soManyThreads];
35              CoresTest o = null;
36              init();
37                      single.singleThreadTest(soOftenPerThread * soManyThreads);
38              end("Single Thread Test");
39
40              init();
41                      for ( int index = 0; index < soManyThreads; index ++ )  {
42                              many[index] = new CoresTest(soOftenPerThread);
43                              many[index].start();
44                      }
45                      try {
46                              for ( int index = 0; index < soManyThreads; index ++ )
47                                      many[index].join();
48                              }
49                      } catch (Exception e ) {
50                              e.printStackTrace();}
51              end("Multiple Core Test");
```

```
52          }
53      }
```

Source Code: Src/11/CoresTest.java

Output:

```
Single Thread Test:       8632
 # of cores:       16
Multiple Core Test:       745
 # of cores:       16
```

11.10. Limited Number of Threads

• You can only run a limited number of threads at the same time

```
 1      import java.util.*;
 2
 3      public class ThreadMax extends Thread {
 4          int id;
 5          public ThreadMax(int id)    {
 6              this.id = id;
 7          }
 8          public void run () {
 9              try {
10                      System.out.println("ThreadMax Start =   " + id);
11                      Thread.sleep(4000);
12                      System.out.println("ThreadMax End =     " + id);
13              } catch (Exception e)   {
14                      e.printStackTrace();
15                      System.exit(0);
16              }
17          }
18          private void processCommand() {
19              System.out.println("processCommand");
20          }
21
22          public String toString(){
23                  return "ThreadMax: " + id;
24          }
25          public static void main (String args []) {
26              try {
27                      for (int index = 0; index < 100000; index ++ )
28                              new ThreadMax(index).start();
29              } catch (Error e)       {
30                      System.out.println("Error");
31                      e.printStackTrace();
32                      System.exit(0);
33              } catch (Exception e)   {
34                      System.out.println("Exception");
35                      e.printStackTrace();
36                      System.exit(0);
```

```
37               }
38           }
39       }
```

Source Code: Src/11/ThreadMax.java

Output:

```
% java ThreadMax
 ...
ThreadMax Start =   3033
[1.048s][warning][os,thread] Failed to start thread - pthread_create failed (EAGAIN) f
Error
java.lang.OutOfMemoryError: unable to create native thread: possibly out of memory or
       at java.base/java.lang.Thread.start0(Native Method)
       at java.base/java.lang.Thread.start(Thread.java:813)
       at ThreadMax.main(ThreadMax.java:28)
```

11.11. Competing Threads

Threads can obtain exclusive access to an object if all competing threads use a synchronized statement or call a method with synchronized attribute. Class methods monitor the class description, other methods monitor their receiver, the statement monitors the indicated value. The attribute synchronized precedes the result type.

Adding or deleting a synchronized modifier of a method does not break compatibility with existing binaries.

- the execution of methods can be synchronized.

```
public synchronized method( ...) { ... }
public static synchronized method( ...) { ... }
```

- synchronized statements which allow access to an associated object.

```
synchronized(aObj) { ... }
```

```
public synchronized method( ...) { ... }
```

is aequivalent to

```
public method( ...) {
     synchronized ( this ) {
     }
}
```

Note: Interface methods can't be native, static, synchronized, final, private, or protected

11.12. Example 0

```
 1
 2      public class Thread_0 extends Thread    {
 3          private String info;
 4          static Object o = new Object();
 5
 6          public Thread_0 (String info) {
 7              this.info    = info;
 8          }
 9
10          public void run () {
11              synchronized ( o )      {
12                      System.err.println(info + " ---> ");
13                      try { sleep(1000); } catch (  InterruptedException e ) { }
14                      System.err.println(info + " <--- ");
15              }
16          }
17
18          public static void main (String args []) {
19              Thread_0 aT4_0 = new Thread_0("first");
20              Thread_0 aT4_1 = new Thread_0("second");
21              Thread   aT4_2 = new Thread_0("third");
22
23              aT4_0.start();
24              aT4_1.start();
25              aT4_2.start();
26          }
27      }
```

Source Code: Src/11/Thread_0.java

Next

```
 1      import java.util.*;
 2
 3      public class Thread_00 extends Thread    {
 4          private String info;
 5
 6          public Thread_00 (String info) {
 7              this.info    = info;
 8          }
 9
10          public synchronized void run () {
11              System.err.println(info + ": ----> ");
12              try { sleep(1000); } catch (  InterruptedException e ) { }
13              System.err.println(info + ": <---- ");
14          }
15
16          public static void main (String args []) {
17              Thread_00 aT4_00 = new Thread_00("first");
18              Thread_00 aT4_1 = new Thread_00("second");
19
```

```
20              aT4_00.start();
21              aT4_1.start();
22          }
23      }
```

Source Code: Src/11/Thread_00.java

11.13. Example I

```
 1      import java.util.*;
 2
 3      public class M extends Thread    {
 4          private String info;
 5          private Vector aVector;
 6
 7          public M (String info) {
 8              this.info    = info;
 9          }
10
11          public synchronized void run () {
12              System.err.println(info + ": --->");
13                      //try { sleep(1000); } catch (  InterruptedException e ) { }
14              System.err.println(info + "<---");
15          }
16
17          public static void main (String args []) {
18              Vector aVector = new Vector();
19              M aT4_0 = new M("first");
20
21              aT4_0.start();
22              aT4_0.run();
23          }
24      }
```

Source Code: Src/11/M.java

11.14. Example II

This example has a problem.

zwei underschiedlich empfaenger fuer die methode

```
 1      import java.util.*;
 2
 3      public class Thread_4 extends Thread    {
 4          private String info;
 5          private Vector aVector;
 6
 7          public Thread_4 (String info, Vector aVector) {
 8              this.info    = info;
 9              this.aVector = aVector;
10          }
11
```

```
12          private synchronized void inProtected () {
13              System.err.println(info + ": is in protected()");
14              aVector.addElement(info);
15              try {
16                  if ( info.equals("second") )
17                      sleep(1000);
18                  else
19                      sleep(3000);
20              }
21              catch (  InterruptedException e ) {
22                  System.err.println("Interrupted!");
23              }
24              System.err.println(info + ": exit run");
25          }
26
27          public void run () {
28              inProtected();
29          }
30
31          public static void main (String args []) {
32              Vector aVector = new Vector();
33              Thread_4 aT4_0 = new Thread_4("first",  aVector);
34              Thread_4 aT4_1 = new Thread_4("second", aVector);
35
36              aT4_0.inProtected();
37              aT4_0.start();
38          }
39      }
```

Source Code: Src/11/Thread_4.java

Result:

```
first: is in protected()
second: is in protected()
second: exit run
first: exit run
```

the only possible output?

11.15. Object Synchronization

```
 1      import java.util.*;
 2
 3      public class Thread_5 extends Thread    {
 4              private String info;
 5              static  Vector aVector;
 6
 7              public Thread_5 (String info, Vector aVector) {
 8                      this.info = info;
 9                      this.aVector = aVector;
10              }
11
12              public void inProtected () {
13                 synchronized ( aVector )     {
14                      System.err.println(info + ": is in protected()");
15                      try {
16                              if ( info.equals("second") )
17                                      sleep(1000);
18                              else
19                                      sleep(3000);
20                      }
21                      catch (  InterruptedException e ) {
22                              System.err.println("Interrupted!");
23                      }
24                      System.err.println(info + ": exit run");
25                 }
26              }
27
28              public void run () {
29                      inProtected();
30              }
31
32              public static void main (String args []) {
33                      Vector aVector = new Vector();
34                      Thread_5 aT5_0 = new Thread_5("first", aVector);
35                      Thread_5 aT5_1 = new Thread_5("second", new Vector());
36                      aT5_0.start();
37                      aT5_1.start();
38              }
39      }
```

Source Code: Src/11/Thread_5.java

Der Vector kann nich mehr veraendert werden, nachdem der Konstuctor geendet hat.

```
% java Thread_5
first: is in protected()
first: exit run
second: is in protected()
second: exit run
```

the only possible output?

11.16. Object Synchronization II

```
 1      import java.util.*;
 2
 3      public class Thread_5b extends Thread    {
 4              private String info;
 5              private Vector aVector = new Vector();
 6
 7              public Thread_5b (String info) {
 8                      this.info = info;
 9              }
10
11              public void inProtected () {
12                 synchronized ( aVector )     {
13                      System.err.println(info + ": is in protected()");
14                      try {
15                              sleep(3000);
16                      }
17                      catch (  InterruptedException e ) {
18                              System.err.println("Interrupted!");
19                      }
20                      System.err.println(info + ": exit run");
21                 }
22              }
23
24              public void run () {
25                      inProtected();
26              }
27
28              public static void main (String args []) {
29                      Thread_5b aT5_0 = new Thread_5b("first");
30                      Thread_5b aT5_1 = new Thread_5b("second");
31
32                      aT5_0.start();
33                      aT5_1.start();
34              }
35      }
```

Source Code: Src/11/Thread_5b.java

11.17. Object Synchronization III

What problem do you see?
How to fix it?

Das zu sychonizierende Object is zweimal vorhangen

```
 1
 2      import java.util.*;
 3
 4      public class Thread_5c extends Thread    {
 5              private String info;
 6              static Vector aVector;
 7
 8              public Thread_5c (Vector aVector, String info) {
 9                      this.aVector = aVector;
10                      this.info    = info;
11              }
12
13              public void inProtected () {
14                 synchronized ( aVector )     {
15                      System.err.println(info + ": is in protected()");
16                      try {
17                               sleep(100);
18                      }
19                      catch (  InterruptedException e ) {
20                              System.err.println("Interrupted!");
21                      }
22                      System.err.println(info + ": exit run");
23                 }
24              }
25
26              public void run () {
27                      inProtected();
28              }
29
30              public static void main (String args []) {
31                      Thread_5c aT5_0 = new Thread_5c(new Vector(), "first");
32                      aT5_0.start();
33
34                      Thread_5c aT5_1 = new Thread_5c(new Vector(), "second");
35                      aT5_1.start();
36              }
37      }
```

Source Code: Src/11/Thread_5c.java

Explain all possible outputs.

```
 1
 2      import java.util.*;
 3
 4      public class Thread_5d extends Thread    {
 5              private String info;
```

```
 6              static Vector aVector;
 7
 8              public Thread_5d (Vector aVector, String info) {
 9                      this.aVector = aVector;
10                      this.info    = info;
11              }
12
13              public void inProtected () {
14                 synchronized ( aVector )     {
15                      System.err.println(info + ": is in protected()");
16                      try {
17                               sleep(100);
18                      }
19                      catch (  InterruptedException e ) {
20                              System.err.println("Interrupted!");
21                      }
22                      System.err.println(info + ": exit run");
23                 }
24              }
25
26              public void run () {
27                      inProtected();
28              }
29
30              public static void main (String args []) {
31                      Vector aVector = new Vector();
32                      Thread_5d aT5_0 = new Thread_5d(aVector, "first");
33                      aT5_0.start();
34
35                      try { sleep(1000); } catch (  InterruptedException e ) { Syste
36
37                      aVector = new Vector();
38                      Thread_5d aT5_1 = new Thread_5d(aVector, "second");
39                      aT5_1.start();
40              }
41      }
```

Source Code: Src/11/Thread_5d.java

Explain all possible outputs.

Der Vektor kann ein oder zwei mal vorhanden sein

11.18. Object Synchronization IV

```
 1      public class Thread_5e extends Thread    {
 2          static Object o = new Object();
 3          static int counter = 0;
 4          int id;
 5          public Thread_5e(int id)  {
 6              this.id = id;
 7              o       = new Object();
 8          }
 9          public void run () {
10              if ( id == 0 )  {
```

```
11                      new Thread_5e(1).start();
12
13                      new Thread_5e(2).start();
14                      return;
15              }
16              synchronized ( o ) {
17                      System.err.println(id + " --->");
18                      try {
19                              if ( counter == 0 )     {
20                                      counter = 1;
21                                      o.wait();
22                              } else
23                                      o.notifyAll();
24                      }
25                      catch (  InterruptedException e ) { }
26                      System.err.println(id + " <---");
27                  }
28          }
29          public static void main (String args []) {
30              new Thread_5e(0).start();
31          }
32      }
33
```

Source Code: Src/11/Thread_5e.java

What problem do you see?
How to fix it?

Das zu sychonizierende Object is zweimal vorhangen

```
 1
 2      import java.util.*;
 3
 4      public class Thread_5c extends Thread    {
 5              private String info;
 6              static Vector aVector;
 7
 8              public Thread_5c (Vector aVector, String info) {
 9                      this.aVector = aVector;
10                      this.info    = info;
11              }
12
13              public void inProtected () {
14                 synchronized ( aVector )     {
15                      System.err.println(info + ": is in protected()");
16                      try {
17                               sleep(100);
18                      }
19                      catch (  InterruptedException e ) {
20                              System.err.println("Interrupted!");
21                      }
22                      System.err.println(info + ": exit run");
23                 }
24              }
```

```
25
26              public void run () {
27                      inProtected();
28              }
29
30              public static void main (String args []) {
31                      Thread_5c aT5_0 = new Thread_5c(new Vector(), "first");
32                      aT5_0.start();
33
34                      Thread_5c aT5_1 = new Thread_5c(new Vector(), "second");
35                      aT5_1.start();
36              }
37      }
```

Source Code: Src/11/Thread_5c.java

Explain all possible outputs.

```
 1
 2      import java.util.*;
 3
 4      public class Thread_5d extends Thread    {
 5              private String info;
 6              static Vector aVector;
 7
 8              public Thread_5d (Vector aVector, String info) {
 9                      this.aVector = aVector;
10                      this.info    = info;
11              }
12
13              public void inProtected () {
14                 synchronized ( aVector )     {
15                      System.err.println(info + ": is in protected()");
16                      try {
17                               sleep(100);
18                      }
19                      catch (  InterruptedException e ) {
20                              System.err.println("Interrupted!");
21                      }
22                      System.err.println(info + ": exit run");
23                 }
24              }
25
26              public void run () {
27                      inProtected();
28              }
29
30              public static void main (String args []) {
31                      Vector aVector = new Vector();
32                      Thread_5d aT5_0 = new Thread_5d(aVector, "first");
33                      aT5_0.start();
34
35                      try { sleep(1000); } catch (  InterruptedException e ) { Syste
36
```

```
37                      aVector = new Vector();
38                      Thread_5d aT5_1 = new Thread_5d(aVector, "second");
39                      aT5_1.start();
40              }
41      }
```

Source Code: Src/11/Thread_5d.java

Explain all possible outputs.

Der Vektor kann ein oder zwei mal vorhanden sein

11.19. Class Synchronization

```
 1      import java.util.*;
 2
 3      public class ClassT extends Thread    {
 4          private String info;
 5          private Vector aVector;
 6
 7          public ClassT (String info, Vector aVector) {
 8              this.info    = info;
 9              this.aVector = aVector;
10          }
11
12          static synchronized void staticInProtected1(String s) {
13              System.err.println(s + ": ---->");
14              try {
15                      sleep(1000);
16              }
17              catch (  InterruptedException e ) {
18                  System.err.println("Interrupted!");
19              }
20              staticInProtected2(s);
21              System.err.println(s + ": <----");
22          }
23
24          static synchronized void staticInProtected2(String s) {
25              System.err.println(s + ": ====>");
26              try {
27                      sleep(1000);
28              }
29              catch (  InterruptedException e ) {
30                  System.err.println("Interrupted!");
31              }
32              System.err.println(s + ": ====>");
33          }
34
35          public void run () {
36              staticInProtected1(info);
37          }
38
39          public static void main (String args []) {
40              Vector aVector = new Vector();
41              ClassT aClassT_0 = new ClassT("first",  aVector);
```

```
42              ClassT aClassT_1 = new ClassT("second", aVector);
43
44              ClassT.staticInProtected1("main");
45              aClassT_0.start();
46              aClassT_1.start();
47              aClassT_0.staticInProtected1("aClassT_0");
48              aClassT_1.staticInProtected1("aClassT_1");
49          }
50      }
```

Source Code: Src/11/ClassT.java

Result:

```
% java ClassT
main: ---->
main: ====>
main: ====>
main: <----
aClassT_0: ---->
```

11.20. Class Synchronization

```
 1      import java.util.*;
 2
 3      public class ClassT extends Thread    {
 4          private String info;
 5          private Vector aVector;
 6
 7          public ClassT (String info, Vector aVector) {
 8              this.info    = info;
 9              this.aVector = aVector;
10          }
11
12          static synchronized void staticInProtected1(String s) {
13              System.err.println(s + ": ---->");
14              try {
15                      sleep(1000);
16              }
17              catch (  InterruptedException e ) {
18                  System.err.println("Interrupted!");
19              }
20              staticInProtected2(s);
21              System.err.println(s + ": <----");
22          }
23
24          static synchronized void staticInProtected2(String s) {
25              System.err.println(s + ": ====>");
26              try {
27                      sleep(1000);
28              }
29              catch (  InterruptedException e ) {
30                  System.err.println("Interrupted!");
31              }
32              System.err.println(s + ": ====>");
33          }
34
35          public void run () {
36              staticInProtected1(info);
37          }
38
39          public static void main (String args []) {
40              Vector aVector = new Vector();
41              ClassT aClassT_0 = new ClassT("first",  aVector);
42              ClassT aClassT_1 = new ClassT("second", aVector);
```

```
43
44              ClassT.staticInProtected1("main");
45              aClassT_0.start();
46              aClassT_1.start();
47              aClassT_0.staticInProtected1("aClassT_0");
48              aClassT_1.staticInProtected1("aClassT_1");
49          }
50      }
```

Source Code: Src/11/ClassT.java

Result:

```
% java ClassT
main: ---->
main: ====>
main: ====>
main: <----
aClassT_0: ---->
aClassT_0: ====>
aClassT_0: ====>
aClassT_0: <----
aClassT_1: ---->
aClassT_1: ====>
aClassT_1: ====>
aClassT_1: <----
first: ---->
first: ====>
first: ====>
first: <----
second: ---->
second: ====>
second: ====>
second: <----
```

One more:

```
 1      import java.util.*;
 2
 3      public class Thread_6 extends Thread    {
 4          private String info;
 5          private Vector aVector;
 6
 7          public Thread_6 (String info, Vector aVector) {
 8              this.info    = info;
 9              this.aVector = aVector;
10          }
11
12          private void inProtected_1 () {
13              synchronized ( aVector )        {
14                      System.err.println("1: " + info + ": is in ");
15                      try {
16                              sleep(1000);
17                      }
18                      catch (  InterruptedException e ) {
19                          System.err.println("Interrupted!");
20                      }
21                      System.err.println("1: " + info + ": exit");
22              }
23          }
24
25          private void inProtected_2 () {
26              synchronized ( info )   {
27                      System.err.println("2: " + info + ": is IN ");
28                      try {
29                              sleep(5000);
30                      }
31                      catch (  InterruptedException e ) {
32                          System.err.println("Interrupted!");
33                      }
34                      System.err.println("2: " + info + ": EXIT");
35              }
36          }
37
38          private static void inProtected_3 () {
39              System.err.println("3: IN ");
40              try {
41                      sleep(9000);
42              }
43              catch (  InterruptedException e ) {
44                  System.err.println("Interrupted!");
45              }
46              System.err.println("3: EXIT");
47          }
48
49          public void run () {
50              inProtected_1();
51              inProtected_2();
```

```
52              Thread_6.inProtected_3();
53          }
54
55          public static void main (String args []) {
56              Vector aVector = new Vector();
57              Thread_6 aT6_0 = new Thread_6("first",  aVector);
58              Thread_6 aT6_1 = new Thread_6("second", aVector);
59
60              aT6_0.start();
61              aT6_1.start();
62          }
63      }
```

Source Code: Src/11/Thread_6.java

11.21. Wait and Notify

By using *wait* and *notify* a thread can give up its lock at an abritary point and the wait for another thread to give it back for continuation.

```
 1      import java.util.Vector;
 2
 3      public class WaitAndNotify_0 extends Thread     {
 4
 5              private static int counter = 0;
 6              private String  name = null;
 7              private static Vector aVector;
 8
 9              public WaitAndNotify_0 (String name, Vector aVector) {
10                      this.aVector = aVector;
11                      this.name = name;
12
13              }
14
15              public void run () {
16                 synchronized ( aVector )     {
17                      if (   name.equals("two")  )    {
18                              System.out.println(getName() + " will wait ...");
19                              aVector.notify();
20                              System.out.println(getName() + " done.");
21                      } else {
22                              try {
23                                      aVector.wait();
24                              } catch ( IllegalMonitorStateException  e )     {
25                                      System.out.println( ": IllegalMonitorStateExce
26                              } catch ( InterruptedException  e )     {
27                                      System.out.println(": InterruptedException");
28                              }
29                              System.out.println(getName() + " is awake!");
30                      }
31                }
32              }
33              public static void main (String args []) {
34                      Vector theVector = new Vector();
35                      new WaitAndNotify_0("one", theVector).start();
36                      new WaitAndNotify_0("two", theVector).start();
37              }
38      }
```

Source Code: Src/11/WaitAndNotify_0.java

Reihenfolge is nicht garatiert.

11.22. One after the Other

```
 1      import java.util.Vector;
 2
 3      public class WaitAndNotify_First extends Thread {
 4
 5              private static int counter = 0;
 6              private String  name = null;
 7              private Vector aVector;
 8
 9              public WaitAndNotify_First (String name, Vector aVector) {
10                      this.aVector = aVector;
11                      this.name = name;
12
13              }
14
15              public void run () {
16                 synchronized ( aVector )     {
17                      if (   name.equals("two")  )    {
18                              System.out.println(getName() + " will wait ...");
19                              aVector.notify();
20                              System.out.println(getName() + " done.");
21                      } else {
22                              System.out.println(getName() + " will wait ...");
23                              try {
24                                      new WaitAndNotify_First("two", aVector).start(
25                                      aVector.wait();
26                              } catch ( IllegalMonitorStateException  e )     {
27                                      System.out.println( ": IllegalMonitorStateExce
28                              } catch ( InterruptedException  e )     {
29                                      System.out.println(": InterruptedException");
30                              }
31                              System.out.println(getName() + " is awake!");
32                      }
33                }
34              }
35
36
37              public static void main (String args []) {
38                      Vector theVector = new Vector();
39                      new WaitAndNotify_First("one", theVector).start();
40              }
41      }
```

Source Code: Src/11/WaitAndNotify_First.java

Erzeugen verzeogert

11.23. Wait and Notify II

What is wrong here: Ordnung - der letzte muss nicht der letzte sein.

```
 1      import java.util.Vector;
 2
 3      public class WaitAndNotify extends Thread       {
 4
 5              private String info;
 6              static  Vector aVector = new Vector();
 7
 8              public WaitAndNotify (String info, Vector aVector) {
 9                      this.info = info;
10                      this.aVector = aVector;
11              }
12
13              public void doTheJob() {
14                 synchronized ( aVector )     {
15                      if (  info.equals("last") )     {
16                              System.out.println(info + " is waking up ...");
17                              aVector.notifyAll();
18                              System.out.println(info + " done.");
19                      } else {
20                              System.out.println(info + " is waiting");
21                              try {
22                                      aVector.wait();
23                              } catch ( IllegalMonitorStateException  e )     {
24                                      System.out.println(info +
25                                        ": IllegalMonitorStateException");
26                              } catch ( InterruptedException  e )     {
27                                      System.out.println(info +
28                                        ": InterruptedException");
29                              }
30                              System.out.println(info + " is awake!");
31                      }
32                }
33              }
34
35
36              public void run () {
37                      doTheJob();
38              }
39
40              public static void main (String args []) {
41                      new WaitAndNotify("first", aVector).start();
42                      new WaitAndNotify("second", aVector).start();
43                      new WaitAndNotify("last", aVector).start();
44              }
45      }
```

Source Code: Src/11/WaitAndNotify.java

Result:

```
% java WaitAndNotify
first is waiting
second is waiting
last is waking up ...
last done.
first is awake!
second is awake!
```

11.24. Wait and Notify III

```
1       import java.util.Vector;
2
3       public class WaitAndNotify_2 extends Thread     {
4
5               private String info;
6               static Integer monitor = new Integer(3);
7               static int count = 0;
8               static int max = 0;
9
10              public WaitAndNotify_2 (String info) {
11                      this.info = info;
12                      max ++;
13              }
14
15              public void doTheJob() {
16                 synchronized ( monitor )     {
17                              System.out.println(info + " is waiting");
18                              count ++;
19                              if ( count == max )
20                                      monitor.notifyAll();
21                              else
22                                      try {
23                                              monitor.wait();
24                                      } catch ( Exception  e )        {
25                                              System.out.println(info +
26                                                ": IllegalMonitorStateException");
27                                      }
28                              System.out.println(info + " is awake!");
29                      }
30              }
31
32
33              public void run () {
34                      doTheJob();
35              }
36
37              public static void main (String args []) {
38                      new WaitAndNotify_2("first").start();
39                      new WaitAndNotify_2("last").start();
40      /*
41                      WaitAndNotify_2 t1 = new WaitAndNotify_2("first");
42                      WaitAndNotify_2 t2 = new WaitAndNotify_2("last");
43                      t1.start();
44                      t2.start();
45      */
46              }
47      }
```

Source Code: Src/11/WaitAndNotify_2.java

Result:

```
% java WaitAndNotify_2
first is waiting
second is waiting
last is waiting
last is awake!
first is awake!
second is awake!
```

11.25. Be carefull with wait(long timeout)

```
 1      import java.util.Vector;
 2      import java.util.Date;
 3
 4
 5      public class WaitAndNotify_3 extends Thread     {
 6
 7              private String info;
 8              static  Vector aVector = new Vector();
 9
10              public WaitAndNotify_3 (String info, Vector aVector) {
11                      this.info = info;
12                      this.aVector = aVector;
13              }
14
15              public void doTheJob() {
16                 synchronized ( aVector )     {
17                      System.out.println(info + " is waiting. " + new Date() );
18                      try {
19                              aVector.wait(1000);
20                      } catch ( Exception  e )        {
21                              System.out.println(info + ": Exception");
22                              e.printStackTrace();
23                      }
24                      System.out.println(info + " is awake! " + new Date());
25                }
26              }
27
28
29              public void run () {
30                      doTheJob();
31              }
32
33              public static void main (String args []) {
34                      new WaitAndNotify_3("first", aVector).start();
35                      new WaitAndNotify_3("second", aVector).start();
36                      // new WaitAndNotify_3("last", aVector).start();
37              }
38      }
```

Source Code: Src/11/WaitAndNotify_3.java

```
% java WaitAndNotify_3
first is waiting. Mon Apr 16 15:02:10 EDT 2001
second is waiting. Mon Apr 16 15:02:11 EDT 2001
first is awake! Mon Apr 16 15:02:12 EDT 2001
second is awake! Mon Apr 16 15:02:12 EDT 2001
%
```

11.26. Lock Objects

See also: https://docs.oracle.com/javase/8/docs/api/java/util/concurrent/locks/package-summary.html (https://docs.oracle.com/javase/8/docs/api/java/util/concurrent/locks/package-summary.html)

TBD

Stole from java doc:

Interfaces and classes providing a framework for locking and waiting for conditions that is distinct from built-in synchronization and monitors. The framework permits much greater flexibility in the use of locks and conditions, at the expense of more awkward syntax. The Lock interface supports locking disciplines that differ in semantics (reentrant, fair, etc), and that can be used in non-block-structured contexts including hand-over-hand and lock reordering algorithms. The main implementation is ReentrantLock.

The ReadWriteLock interface similarly defines locks that may be shared among readers but are exclusive to writers. Only a single implementation, ReentrantReadWriteLock, is provided, since it covers most standard usage contexts. But programmers may create their own implementations to cover nonstandard requirements.

The Condition interface describes condition variables that may be associated with Locks. These are similar in usage to the implicit monitors accessed using Object.wait, but offer extended capabilities. In particular, multiple Condition objects may be associated with a single Lock. To avoid compatibility issues, the names of Condition methods are different from the corresponding Object versions. Executors

Copied from: http://docs.oracle.com/javase/7/docs/api/java/util/concurrent/Executor.html (http://docs.oracle.com/javase/7/docs/api/java/util/concurrent/Executor.html)

• An object that executes submitted Runnable tasks.

• This interface provides a way of decoupling task submission from the mechanics of how each task will be run, including details of thread use, scheduling, etc.

• An Executor is normally used instead of explicitly creating threads. For example, rather than invoking new Thread(new(RunnableTask())).start() for each of a set of tasks, you might use:

Stolen from java doc:

In all of the previous examples, there's a close connection between the task being done by a new thread, as defined by its Runnable object, and the thread itself, as defined by a Thread object. This works well for small applications, but in large-scale applications, it makes sense to separate thread management and creation from the rest of the application. Objects that encapsulate these functions are known as executors. The following subsections describe executors in detail.

Interface (https://docs.oracle.com/javase/10/docs/api/java/util/concurrent/Executors.html)

11.27. Executors: Thread Pools

• Using thread pools avoids the overhead of creating threads

• This reduces also memory management overheads

• Help to make usage of multiple processors

• Designed for problems which have recursive solutuions

11.28. Thread Pools

— Instead of staring a new thread for every task, the task is passed to the thread pool

— A thread from the thread pool will take over this taks if a thread is free, or completed the previous task

— The number of active threads at any given moment in time is bound

— Question: Recourses of thread pool management vs. thread creation

• A thread pool has typically:

— worker threads

— a thread factory (https://docs.oracle.com/javase/10/docs/api/java/util/concurrent/ThreadFactory.html)

— a executers (https://docs.oracle.com/javase/10/docs/api/java/util/concurrent/ThreadPoolExecutor.html)

An example:

```
 1      public class HpWorker_1 implements Runnable {
 2
 3          private int id;
 4          private int sleepTime;
 5
 6          public HpWorker_1(int id, int sleepTime){
 7              this.id         = id;
 8              this.sleepTime  = sleepTime;
 9          }
10          public void run() {
11              System.out.println("Start =     " + id);
12              try {
13                  System.err.println("active thread count: " + Thread.activeCount()
14                  Thread.sleep(sleepTime);
15                  } catch (InterruptedException e) {
16                          e.printStackTrace();
17              }
18              System.out.println("End =       " + id);
19          }
20          public String toString(){
21                  return "HpWorker_1: " + id;
22          }
23      }
```

Source Code: Src/11/HpWorker_1.java

Use:

```
 1      import java.util.concurrent.ExecutorService;
 2      import java.util.concurrent.Executors;
 3      import java.util.Random;
 4
 5      public class HpSimpleThreadPoolUse_1 {
 6          public static final int MAX = 2;
 7          public static final int SLEEP_TIME = 1000;
 8          public static void main(String[] args) {
 9
10              Random aRandom = new Random();
11              ExecutorService executor = Executors.newFixedThreadPool(MAX);
12              for (int id = 0; id < MAX * 3; id++) {  // max thread in pool
13                          executor.execute( new HpWorker_1(id,  aRandom.nextInt(20)
14              }
15              executor.shutdown();
```

```
16
17              while ( !executor.isTerminated() ) {
18                      try { Thread.sleep(100); } catch ( Exception e ) { }
19              }
20              System.out.println("all threads have terminated");
21          }
22
23      }
```

Source Code: Src/11/HpSimpleThreadPoolUse_1.java

Example:

```
% java HpSimpleThreadPoolUse_1
Start =      0
Start =      1
active thread count: 3
active thread count: 3
End =  0
Start =      2
active thread count: 3
End =  1
Start =      3
active thread count: 3
End =  3
Start =      4
active thread count: 3
End =  4
Start =      5
active thread count: 3
End =  2
End =  5
all threads have terminated
```

A Very Simplified Thread Pool Implementation

```
 1      import java.util.concurrent.LinkedBlockingQueue;
 2
 3      public class HpThreadPool {
 4          private final int nThreads;
 5          private final Worker[] threads;
 6          private final LinkedBlockingQueue queue;
 7
 8          private boolean shutDownThePool = false;
 9
10          public HpThreadPool(int nThreads) {
11              this.nThreads = nThreads;
12              queue = new LinkedBlockingQueue();
13              threads = new Worker[nThreads];
14
15              for (int i = 0; i < nThreads; i++)
16                  ( threads[i] = new Worker()).start();
```

```
17          }
18
19          public void execute(Runnable task) {
20              if ( ! shutDownThePool )    {
21                  synchronized (queue) {
22                      queue.add(task);
23                      queue.notify();
24                  }
25              }
26          }
27          public void shutdown() {
28              synchronized (queue) {
29                  shutDownThePool = true;
30              }
31          }
32          public boolean isTerminated() {
33              boolean rValue = false;
34                  synchronized (queue) {
35                          rValue = shutDownThePool && queue.isEmpty();
36                          if ( rValue )    {
37                              for (int i = 0; i < nThreads; i++) {
38                                      synchronized ( threads[i] )    {
39                                      threads[i].thisIsAnActiveThread = false;
40                                      queue.notify();
41                                  }
42                              }
43                  }
44              }
45              return rValue;
46          }
47
48          private class Worker extends Thread {
49              public boolean thisIsAnActiveThread = true;
50                  public void run() {
51                      Runnable task;
52
53                      while (thisIsAnActiveThread) {
54                          synchronized (queue) {
55                              while (queue.isEmpty()) {
56                                  try {
57                                      queue.wait();
58                                      if ( !  thisIsAnActiveThread )
59                                          return;
60                                  } catch (InterruptedException e) {
61                                      System.out.println("An error occurred while qu
62                                  }
63                              }
64                              task = (Runnable)queue.poll();
65                          }
66
67                          try {
68                              task.run();
69                          } catch (RuntimeException e) {
70                              System.out.println("HpThreadPool: Something went wrong
```

```
71                              e.printStackTrace();
72                          }
73                      }
74                  }
75              }
76      }
```

Source Code: Src/11/HpThreadPool.java

Use:

```
1       import java.util.concurrent.ExecutorService;
2       import java.util.concurrent.Executors;
3
4       public class HpSimpleThreadPoolUse_2 {
5           public static final int MAX = 1;
6           public static final int SLEEP_TIME = 1200; // 100
7           public static void main(String[] args) {
8               HpThreadPool executor = new HpThreadPool(MAX);
9               // is it possible to have less than max threads running?
10              for (int i = 0; i < 1 + MAX * 2; i++) {
11                  if ( i % 2 == 0 )
12                          executor.execute(new HpWorker_1(i, 4 * SLEEP_TIME));
13                  else
14                          executor.execute(new HpWorker_1(i, SLEEP_TIME));
15
16              }
17              executor.shutdown();
18
19              while ( !executor.isTerminated() ) {
20                      System.out.println("check if terminated ...");
21                      try { Thread.sleep(5000); } catch ( Exception e ) { }
22              }
23              System.out.println("all threads have terminated");
24
25          }
26      }
```

Source Code: Src/11/HpSimpleThreadPoolUse_2.java

Output:

```
% java HpSimpleThreadPoolUse_2
check if terminated ...
Start =       1
Start =       2
Start =       0
End =  1
Start =       3
End =  3
Start =       4
End =  0
End =  2
Start =       5
```

```
Start =      6
all threads have terminated # io latency
End =  5
End =  4
End =  6
```

11.29. Examples

```
 1      /*
 2       * is this output       1 0 1 0 1 ...
 3       *                      ...
 4       * the only possible output?
 5       *
 6       * Falsch: es ist nichtgarantiert, in welcher die
 7       * Threads eintreten.
 8       */
 9      public class X extends Thread   {
10              private String info;
11              static Object o = new Object();
12              public X (String info) {
13                      this.info    = info;
14              }
15              public void run () {
16                      while ( true )  {
17                              synchronized ( o ) {
18                                      System.out.println(info);
19                                      try {
20                                              o.notify();
21                                              sleep(100);
22                                              o.wait(1);
23                                      } catch ( Exception e ) { }
24                              }
25                      }
26              }
27              public static void main (String args []) {
28                      ( new X("0") ).start();
29                      ( new X("1") ).start();
30              }
31      }
```

Source Code: Src/Question_Week_5/X.java

Next

```
 1      /*
 2       * Should print out 0 1 0 1 0 1 ...
 3       * Is this correct?
 4       *
 5       * nicht richtig,
 6       * weil der Konstruktor fuer das Objekt mit der Id 0
 7       * nicht zuende gehen muss bevor der 2. Konstruktor
 8       * zuende geht.
```

```
 9       *
10       */
11      public class XX extends Thread  {
12              private String info;
13              static Object o = new Object();
14
15              public XX (String info) {
16                      this.info    = info;
17                      synchronized ( o ) {
18                              if ( info.equals("0") )
19                                      ( new XX("1") ).start();
20                      }
21              }
22              public void run () {
23                      while ( true )  {
24                              synchronized ( o ) {
25                                      System.out.println(info);
26                                      try {
27                                              o.notify();
28                                              sleep(100);
29                                              o.wait();
30                                      } catch ( Exception e ) { }
31                              }
32                      }
33              }
34              public static void main (String args []) {
35                      new XX("0").start();
36              }
37      }
```

Source Code: Src/Question_Week_5/XX.java

Next

```
 1      /*
 2       * Should print out 0 1 0 1 0 1 ...
 3       *
 4       *
 5       */
 6      public class XXX extends Thread {
 7              private String info;
 8              static Object o = new Object();
 9              static boolean oneIsRunning = false; // is static important?
10                                                   // es wird nur ein
11                                                   // Objekt erzeugt
12              public XXX (String info) {
13                      this.info    = info;
14              }
15              public void run () {
16                      while ( true )  {
17                              synchronized ( o ) {
18                                      o.notify();
19                                      System.out.println(info);
20                                      try {
```

```
21                                                  if ( ! oneIsRunning )   {
22                                                          ( new XXX("1") ).start();
23                                                          oneIsRunning = true;
24                                                  }
25                                                  sleep(300);
26                                                  o.wait();
27                                          } catch ( Exception e ) { }
28                                  }
29                          }
30                  }
31                  public static void main (String args []) {
32                          new XXX("0").start();
33                  }
34          }
```

Source Code: Src/Question_Week_5/XXX.java

More for you:

```
1       /*
2        * is this output       --->
3        *                      <--
4        *                      ...
5        * the only possible output?
6        */
7       public class T_1 extends Thread    {
8
9           private synchronized void inProtected () {
10             System.err.println("--> ");
11             try {
12                     sleep(1000);
13             }
14             catch (  InterruptedException e ) {
15                      System.err.println("Interrupted!");
16             }
17             System.err.println("<-- ");
18          }
19
20          public void run () {
21              inProtected();
22          }
23          public static void main (String args []) {
24              new T_1().start();
25              new T_1().start();
26          }
27      }
28
```

Source Code: Src/Question_Week_5/T_1.java

Next

```
1       /*
2        * is this output       --->
3        *                      <--
4        *                      ...
5        * the only possible output?
6        * nein unterschiedliche zwei objekte
7        */
8       public class T_2 extends Thread    {
9           static String info;
10
11          public T_2(String info )    {
12              this.info = info;
13          }
14          private void inProtected () {
15             synchronized ( info )       {
16                   System.err.println("--> " + info);
17                   try {
18                            sleep(1000);
19                   } catch (  Exception e ) {
20                       e.printStackTrace();
21                   }
22                   System.err.println("<-- " + info);
23             }
24          }
25
26          public void run () {
27              inProtected();
28          }
29          public static void main (String args []) {
30              String aString = "a";
31              T_2 one = new T_2(aString);
32              one.start();
33              T_2 two = new T_2(aString);
34              two.start();
35              aString = "b";
36              // new T_2("a").start();
37              // new T_2("b").start();
38
39          }
40      }
```

Source Code: Src/Question_Week_5/T_2.java

Next

```
1       /*
2        * is this output       --->
3        *                      <--
4        *                      ...
5        * the only possible output?
6        * ja ein objekt
7        */
8       public class T_2b extends Thread    {
9           private String info;
```

```
10
11          public T_2b(String info )   {
12              this.info = info;
13          }
14          private synchronized void inProtected () {
15                   System.err.println("--> " + info);
16                   try {
17                            sleep(1000);
18                   } catch (  Exception e ) {
19                       e.printStackTrace();
20                   }
21                   System.err.println("<-- " + info);
22          }
23
24          public void run () {
25              inProtected();
26          }
27          public static void main (String args []) {
28              new T_2b("hello").start();
29              new T_2b("hello").start();
30
31          }
32      }
```

Source Code: Src/Question_Week_5/T_2b.java

Next

```
1       /*
2        * is this output       --->
3        *                      <--
4        *                      ...
5        * the only possible output?
6        * ja ein objekt
7        */
8       public class T_2c extends Thread    {
9           private String info;
10
11          public T_2c(String info )   {
12              new T_2c("hello").start();
13              this.info = info;
14          }
15          private void inProtected () {
16             synchronized ( info )       {
17                   System.err.println("--> " + info);
18                   try {
19                            sleep(1000);
20                   } catch (  Exception e ) {
21                       e.printStackTrace();
22                   }
23                   System.err.println("<-- " + info);
24              }
25          }
26
```

```
27          public void run () {
28              inProtected();
29          }
30          public static void main (String args []) {
31              new T_2c("hello").start();
32
33          }
34      }
```

Source Code: Src/Question_Week_5/T_2c.java

Next

```
 1      /*
 2       * is this output       --->
 3       *                      <--
 4       *                      ...
 5       * the only possible output?
 6       * wievele objekte werden benutzt?
 7       */
 8      public class T_3 extends Thread    {
 9          private int info;
10
11          public T_3 (int info) {
12              this.info    = info;
13          }
14
15          public synchronized void run () {
16              System.err.println("--> " + info);
17               try {
18                        sleep(1000);
19               } catch (  Exception e ) {
20                      e.printStackTrace();
21               }
22               System.err.println("<-- " + info);
23          }
24
25          public static void main (String args []) {
26              for ( int i = 1; i < 100; i ++ )
27                      new T_3(i).start();
28          }
29      }
```

Source Code: Src/Question_Week_5/T_3.java

Next

```
 1      /*
 2       * is this output       --->
 3       *                      <--
 4       *                      ...
 5       * the only possible output?
 6       * nur ein objekt wird benutzt
 7       */
```

```
 8      public class T_4 extends Thread    {
 9
10          static Object o = new Object();
11          String info;
12
13          public T_4(String info )    {
14              this.info = info;
15          }
16
17          public void run () {
18              synchronized ( o ) {
19                  System.err.println("--->" + info);
20                  try {
21                          sleep(1000);
22                  }
23                  catch (  InterruptedException e ) {
24                      System.err.println("Interrupted!");
25                  }
26                  System.err.println("<---" + info);
27              }
28          }
29
30          public static void main (String args []) {
31              new T_4("1").start();
32              new T_4("2").start();
33              new T_4("3").start();
34          }
35      }
```

Source Code: Src/Question_Week_5/T_4.java

Next

```
 1      import java.util.Vector;
 2
 3      /* is 0 1 0 1 ... the only possible output? */
 4
 5      public class T_8 extends Thread {
 6
 7              private String info;
 8              private Vector aVector;
 9
10              public T_8(String info, Vector aVector) {
11                      this.info = info;
12                      this.aVector = aVector;
13              }
14
15              public void run() {
16                      inProtected();
17              }
18
19              public void inProtected() {
20                      int x = 0;
21                      // currently considering only 10 output pairs, but will work f
```

```
22                      // infinity while(true)
23                      while (x < 4) {
24                              synchronized (aVector) {
25
26                                      if (info.equals("zero")) {
27                                              System.out.println("0");
28                                              aVector.notify();
29                                              try {
30                                                      aVector.wait();
31                                              } catch (InterruptedException e) {
32                                                      System.out.println(": Interrup
33                                              }
34                                      } else {
35                                              System.out.println("1");
36                                              try {
37                                                      aVector.notify();
38                                                      aVector.wait();
39                                              } catch (InterruptedException e) {
40                                                      System.out.println(": Interrup
41                                              }
42
43                                      }
44                              }
45                              x++;
46                      }
47              }
48
49              public static void main(String args[]) {
50                      @SuppressWarnings("rawtypes")
51                      Vector aVector = new Vector();
52                      T_8 t0 = new T_8("zero", aVector);
53                      T_8 t1 = new T_8("one", aVector);
54
55                      t0.start();
56                      t1.start();
57              }
58      }
```

Source Code: Src/Question_Week_5/T_8.java

Next

```
1       /*
2        * is this output       1 0 1 0 1 ...
3        *                      ...
4        * the only possible output?
5        *
6        * Falsch: es ist nichtgarantiert, in welcher die
7        * Threads eintreten.
8        */
9       public class X extends Thread   {
10              private String info;
11              static Object o = new Object();
12              public X (String info) {
```

```
13                      this.info    = info;
14              }
15              public void run () {
16                      while ( true )  {
17                              synchronized ( o ) {
18                                      System.out.println(info);
19                                      try {
20                                              o.notify();
21                                              sleep(100);
22                                              o.wait(1);
23                                      } catch ( Exception e ) { }
24                              }
25                      }
26              }
27              public static void main (String args []) {
28                      ( new X("0") ).start();
29                      ( new X("1") ).start();
30              }
31      }
```

Source Code: Src/Question_Week_5/X.java

Next

```
 1      /*
 2       * Should print out 0 1 0 1 0 1 ...
 3       * Is this correct?
 4       *
 5       * nicht richtig,
 6       * weil der Konstruktor fuer das Objekt mit der Id 0
 7       * nicht zuende gehen muss bevor der 2. Konstruktor
 8       * zuende geht.
 9       *
10       */
11      public class XX extends Thread  {
12              private String info;
13              static Object o = new Object();
14
15              public XX (String info) {
16                      this.info    = info;
17                      synchronized ( o ) {
18                              if ( info.equals("0") )
19                                      ( new XX("1") ).start();
20                      }
21              }
22              public void run () {
23                      while ( true )  {
24                              synchronized ( o ) {
25                                      System.out.println(info);
26                                      try {
27                                              o.notify();
28                                              sleep(100);
29                                              o.wait();
30                                      } catch ( Exception e ) { }
```

```
31                              }
32                      }
33              }
34              public static void main (String args []) {
35                      new XX("0").start();
36              }
37      }
```

Source Code: Src/Question_Week_5/XX.java

Next

```
 1      /*
 2       * Should print out 0 1 0 1 0 1 ...
 3       *
 4       *
 5       */
 6      public class XXX extends Thread {
 7              private String info;
 8              static Object o = new Object();
 9              static boolean oneIsRunning = false; // is static important?
10                                                   // es wird nur ein
11                                                   // Objekt erzeugt
12              public XXX (String info) {
13                      this.info    = info;
14              }
15              public void run () {
16                      while ( true )  {
17                              synchronized ( o ) {
18                                      o.notify();
19                                      System.out.println(info);
20                                      try {
21                                              if ( ! oneIsRunning )   {
22                                                 ( new XXX("1") ).start();
23                                                 oneIsRunning = true;
24                                              }
25                                              sleep(300);
26                                              o.wait();
27                                      } catch ( Exception e ) { }
28                              }
29                      }
30              }
31              public static void main (String args []) {
32                      new XXX("0").start();
33              }
34      }
```

Source Code: Src/Question_Week_5/XXX.java

11.30. Deadlock — an Overview

• Problem:

Resource 1 and and resource 2 must be used exclusively

Process 1 holds resource 1 and is requestion resource 2

Process 2 holds resource 2 and is requestion resource 1

11.31. Deadlock

• A set of processes is in a deadlock state when every process in the set is waiting for an event that can be caused by only another process in the set

11.32. Necessary Conditions

• A deadlock can occur if the following four conditions hold

— mutual exclusion: least one resource must be held in a non-sharable mode

— hold and wait: there is a process that is holding a resource and is waiting to acquire another that is currently being held by other processes

— no preemption: resources can only be released voluntarily

— circular wait: See intro example

11.33. Resource Graphs

11.34. Addressing Deadlock

• Prevention: Design the system so that deadlock is impossible

• Avoidance: Construct a model of system states, then choose a strategy that will not allow the system to go to a deadlock state

• Detection & Recovery: Check for deadlock (periodically or sporadically), then recover

11.35. Prevention

• Necessary conditions for deadlock

— Mutual exclusion

— Hold and wait

— Circular waiting

— No preemption

• Ensure that at least one of the necessary conditions is false at all times

• Mutual exclusion must hold at all times (you can fudge things to get around this)

11.36. Hold and Wait

• Need to be sure a process does not hold one resource while requesting another

• Approach 1: Force a process to request all resources it needs at one time (usually at startup). The process dies, if not all ressources are available.

• Approach 2: If a process needs to acquire a new resource, it must first release all resources it holds, then reacquire all it needs

• Problems:

— resource utilization may be low

— starvation is possible

11.37. Circular Wait

Have a situation in which there are K processes holding units of K resources

• There is a cycle in the graph of processes and and resources

• Choose a resource request strategy by which no cycle will be introduced

• Total order on all resources, then can only ask for resources in numerical order (a minor variation is to merely insist that no process request a resource lower than what it is already holding).

• For example, if a set of resource types includes tape drives, disks, and printers, then the weights might be assigned as follows:

  — W(tape drive) = 1

  — W(disk drive) = 5

  — W(printer) = 12

• If A needs tape, disk and printer and B needs printer and disk

  — A → tape,

  — B → disk,

  — B → printer,

  — A → disk,

  — A → printer

• F should be defined in order of normal usage

• Proof by contradiction

  — Assume a circular wait exists

  — Let the set of processes involved in the circular wait be {P(0) , P(1) , P(2) , , P(n) }, where P(i) is waiting for a resource R(i), which is held by process P((i+1)%(n+1)) (modulo arithmetic is used on the indexes, so that P(n) is waiting on R(n) which is held by P(0))

  — Then, since P(i+1) is holding R(i) while requesting R(i+1), then W(R i) < W(R i+1) for all i

  — But this means that
    W(R 0) < W(R 1) < W(R 0)

  • By transitivity, W(R 0) < W(R 0), which is impossible

11.38. Avoid Starvation and Deadlock

- Fairness is a problem, if several concurrent threads are competing for resources.
- A system is fair when each thread gets enough access to a limited resource to make a reasonable progress.
- A fair system prevents starvation and deadlock. Starvation occurs when one or more threads in your program is blocked from gaining access to a resource and thus cannot make progress.
- Deadlock is the ultimate form of starvation; it occurs when two or more threads are waiting on a condition that cannot be satisfied. Deadlock most often occurs when two (or more) threads are each waiting for the other(s) to do something.

Note: kill -SIGQUIT pid

11.39. DeadLocks

• Is there a dead lock in this program?

  und ist im synchronozierten block, bevor dem Aufruf inprtected_1 und fuer den ersten gilt das

aequivalente, dann ist ein deadlock.

```
 1      import java.util.*;
 2
 3      public class DeadLock extends Thread    {
 4              private static String o1  = new String();
 5              private static String o2  = new String();
 6              private String info;
 7
 8              public DeadLock (String info) {
 9                      this.info    = info;
10              }
11
12              private void inProtected_1 () {
13                      System.out.println(info + ":inProtected_1 ");
14                      synchronized ( o2 )   {
15                              inProtected_2();
16                      }
17              }
18
19              private void inProtected_2 () {
20                      System.out.println(info + ":inProtected_2 ");
21                      synchronized ( o1 )   {
22                              inProtected_1();
23                      }
24              }
25
26              public void run () {
27                      if ( info.equals("first") )     {
28                              try { sleep(1000); } catch (  InterruptedException e )
29                              synchronized ( o1 )   {
30                                      inProtected_1();
31                              }
32                      } else
33                              synchronized ( o2 )   {
34                                      inProtected_2();
35                              }
36              }
37
38              public static void main (String args []) {
39                      new DeadLock("second").start();
40                      new DeadLock("first").start();
41              }
42      }
```

Source Code: Src/11/DeadLock.java

• Is there a dead lock in this program?

Ja, falls es jeder thread in den erste. s. block schaft. Sonst, Nein, aber ein StackOverflow wird eintreten.

• How about this one:

```
 1      /*
 2       * Will end up in a dead lock
 3       *
 4       *
 5       */
 6      public class X extends Thread   {
 7              private String info;
 8              Object o_1;
 9              Object o_2;
10              static boolean oneIsRunning = false;
11
12              public X (String info, Object o_1, Object o_2, Object stop) {
13                      this.info    = info;
14                      this.o_1    = o_1;
15                      this.o_2    = o_2;
16                      this.stop    = stop;
17              }
18              public void run () {
19                      synchronized ( o_1 ) {
20                              System.out.println(info);
21                              try {
22                                      if ( ! oneIsRunning )   {
23                                              new X("1", o_2, o_1, stop).start();
24                                              oneIsRunning = true;
25                                              }
26                                      synchronized ( o_2 ) {
27                                              o_2.wait();
28                                              System.out.println("I will not get the
29                                      }
30                              } catch ( Exception e ) { }
31                      }
32              }
33              public static void main (String args []) {
34                      Object o_1 = new Object();
35                      Object o_2 = new Object();
36                      new X("0", o_1, o_2, stop).start();
37              }
38      }
```

Source Code: Src/11/DeadLock_4.java

11.40. Dining Philosophers

The dining philosophers are often used to illustrate various problems that can occur when many synchronized threads are competing for limited resources.

> The story goes like this: Five philosophers are sitting at a round table. In front of each philosopher is a bowl of rice. Between each pair of philosophers is one chopstick. Before an individual philosopher can take a bite of rice he must have two chopsticks — one taken from the left, and one taken from the right. The philosophers must find some way to share chopsticks such that they all eat with an reasonable frequency.

```
Source Code: Src/11/Philosopher.java
```

What is wrong with this solution?

Angenommen es sind nur zwei Philosopher am Tisch un jeder greift nach seiner linken Gabel und is erfolgreich, dann hat er keinen Zugriff zu rechten Gabel.

11.41. Semaphore

1965, suggested Edsger. W. Dijkstra to using an integer variable to count the number of wake ups: Semaphores (semaphore is a greek word, it stands for signal). A synchronization variable that take a positive integer variable. A semaphore has two operations:

- P (dutch for "to test", proberen): an atomic operation that waits for the semaphore to become positive, then decrements it by 1.
- V (dutch for "to increment", verhogen): an atomic operation that increments the semaphore by 1.

The P(S) operation on Semaphore S is:

```
If S > 0 then
       S := S - 1
else
       (Wait on S)
```

The V(S) operation on Semaphore S is:

```
If (One or more processes are waiting on S) then
       (Let one of the processes proceed)
else
       S := S + 1
```

It is assumed that P() and V() are indivisible.

If a thread tries to make a semaphore value to become negative, the thread is blocked until another thread makes the semaphore value positive.

See here. (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/concurrent/Semaphore.html)

11.42. Producer-Consumer Problem

Two processes share a common fixed size buffer. One of them, the producer, puts information into the buffer and the consumer takes it out.

Trouble arises when the producer wants to put a new item in the buffer, but it is already full. Similarily, if the consumer wants to remove an item from the buffer when the buffer is empty.

We will use three semaphores to solve this problem.

- *full* for counting the number of slots that are full.
- *empty* for counting the number of slots that are empty.
- *mutex* to make sure the producer and consumer do not access the buffer at the same time.

- *full* is initially 0
- *empty* is initially N
- *mutex* is initially 1 $\rightarrow$ no process is in its critical region.

What do you think about:

```
 1      // from: https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java
 2      class Pool {
 3         private static final int MAX_AVAILABLE = 100;
 4         private final Semaphore available = new Semaphore(MAX_AVAILABLE, true);
 5
 6         public Object getItem() throws InterruptedException {
 7           available.acquire();
 8           return getNextAvailableItem();
 9         }
10
11         public void putItem(Object x) {
12           if (markAsUnused(x))
13             available.release();
14         }
15
16         // Not a particularly efficient data structure; just for demo
17
18         protected Object[] items = ... whatever kinds of items being managed
19         protected boolean[] used = new boolean[MAX_AVAILABLE];
20
21         protected synchronized Object getNextAvailableItem() {
22           for (int i = 0; i < MAX_AVAILABLE; ++i) {
23             if (!used[i]) {
24               used[i] = true;
25               return items[i];
26             }
27           }
28           return null; // not reached
29         }
30
31         protected synchronized boolean markAsUnused(Object item) {
32           for (int i = 0; i < MAX_AVAILABLE; ++i) {
33             if (item == items[i]) {
34               if (used[i]) {
35                 used[i] = false;
36                 return true;
37               } else
38                 return false;
39             }
40           }
41           return false;
42         }
43       }
```

Source Code: Src/11/Pool.java

11.43. Questions

What is going on here?

•

```
 1      public class T_1 extends Thread    {
 2
 3          private static synchronized void inProtected () {
 4             System.err.println("--> ");
 5             try {
 6                    sleep(1000);
 7             }
 8             catch (  InterruptedException e ) {
 9                       System.err.println("Interrupted!");
10             }
11             System.err.println("<-- ");
12          }
13
14          public void run () {
15              inProtected();
16          }
17          public static void main (String args []) {
18              new T_1().start();
19              new T_1().start();
20              new T_1().start();
21          }
22      }
```

Source Code: Src/11q/T_1.java

Execution:

```
-->
<--
-->
<--
-->
<--
```

•

```
 1      public class T_2 extends Thread    {
 2          private String info;
 3
 4          public T_2 (String info) {
 5              this.info    = new String(info);
 6          }
 7
 8          private void inProtected () {
 9             synchronized ( info )       {
10                   System.err.println("--> " + info);
11                   try {
12                            sleep(1000);
13                   } catch (  Exception e ) {
14                       e.printStackTrace();
15                   }
16                   System.err.println("<-- " + info);
17             }
18          }
19
20          public void run () {
21              inProtected();
22          }
23          public static void main (String args []) {
24              String a = "hello";
25              new T_2(a).start();
26              new T_2(a).start();
27
28          }
29      }
```

Source Code: Src/11q/T_2.java

Execution:

```
--> hello
--> hello
<-- hello
<-- hello
```

•

```
 1      import java.util.Vector;
 2      public class T_3 extends Thread    {
 3          static Vector aVector = new Vector();
 4          private int info;
 5
 6          public T_3 (int info) {
 7              this.info    = info;
 8          }
 9
10          public synchronized void run () {
11              System.err.println("--> " + info);
12               try {
13                        sleep(1000);
14               } catch (  Exception e ) {
15                      e.printStackTrace();
16               }
17               System.err.println("<-- " + info);
18          }
19
20          public static void main (String args []) {
21              for ( int i = 1; i < 100; i ++ )
22                      new T_3(i).start();
23          }
24      }
```

Source Code: Src/11q/T_3.java

Execution:

```
--> 1
--> 2
--> 3
--> 4
--> 5
 ...
<-- 94
<-- 95
<-- 96
<-- 97
<-- 98
<-- 99
```

•

```
 1      import java.util.*;
 2
 3      public class T_4_1 extends Thread    {
 4          static Object o = new Object();
 5          public T_4_1()      {
 6              o  = new Object();
 7          }
```

```
 8          public void run () {
 9              synchronized ( o ) {
10                  System.err.println("--->");
11                  try {
12                          sleep(1000);
13                  }
14                  catch (  InterruptedException e ) {
15                      System.err.println("Interrupted!");
16                  }
17                  System.err.println("<---" );
18              }
19          }
20
21          public static void main (String args []) {
22              new T_4_1().start();
23              new T_4_1().start();
24              new T_4_1().start();
25          }
26      }
```

Source Code: Src/11q/T_4_1.java

Execution:

```
--->
<---
--->
<---
--->
<---
```

•

```
 1      import java.util.*;
 2
 3      public class T_4 extends Thread    {
 4          static Object o = new Object();
 5          public void run () {
 6              synchronized ( o ) {
 7                  System.err.println("--->");
 8                  try {
 9                          sleep(1000);
10                  }
11                  catch (  InterruptedException e ) {
12                      System.err.println("Interrupted!");
13                  }
14                  System.err.println("<---" );
15              }
16          }
17
18          public static void main (String args []) {
19              new T_4().start();
20              new T_4().start();
21              new T_4().start();
22          }
23      }
```

Source Code: Src/11q/T_4.java

Execution:

```
--->
<---
--->
<---
--->
<---
```

•

```
 1      import java.util.*;
 2
 3      public class T_5 extends Thread    {
 4          static Object o = new Object();
 5          static int    counter = 0;
 6
 7          public void run () {
 8              if ( ++counter == 1 )
 9                      o = new Object();
10      //      read x
11
12              synchronized ( o ) {
13                  System.err.println("--->" );
14                  try {
15                          sleep(1000);
16                  }
17                  catch (  InterruptedException e ) {
18                      System.err.println("Interrupted!");
19                  }
20                  System.err.println("<---" );
21              }
22          }
23
24          public static void main (String args []) {
25              new T_5().start();
26              new T_5().start();
27              new T_5().start();
28          }
29      }
```

Source Code: Src/11q/T_5.java

Execution:

```
--->
<---
--->
<---
--->
<---
```

•

```
 1      import java.util.*;
 2
 3      public class T_6 extends Thread    {
 4          static Object o = new Object();
 5          static int    counter = 0;
 6
 7          public void run () {
 8              if ( counter++ == 1 )
 9                      o = new Object();
10
11              synchronized ( o ) {
12                  System.err.println("--->" );
13                  try {
14                          sleep(1000);
15                  }
16                  catch (  InterruptedException e ) {
17                      System.err.println("Interrupted!");
18                  }
19                  System.err.println("<---" );
20              }
21          }
22
23          public static void main (String args []) {
24              new T_6().start();
25              new T_6().start();
26              new T_6().start();
27          }
28      }
```

Source Code: Src/11q/T_6.java

Execution:

```
--->
--->
<---
<---
--->
<---
```

• Will the following program terminate?

```
import java.util.Vector;
public class T_1 extends Thread    {
    private String info;
    Vector aVector;
    Vector bVector;

    public T_1 (String info, Vector aVector) {
        this.info = info;
        this.aVector = aVector;
    }
```

```
     public void run() {
       synchronized ( aVector )    {
        if (  info.equals("last") )    {
            aVector.notifyAll();
        } else {
            System.out.println(info + " is waiting");
            try {
                aVector.wait();
            } catch ( Exception  e )    {
                System.out.println(info +
                  ": InterruptedException");
            }
            System.out.println(info + " is awake!");
        }
      }
    }
    public static void main (String args []) {
        Vector aVector = new Vector();
        Vector bVector = new Vector();

        new T_1("first",  aVector).start();
        new T_1("second", bVector).start();
        new T_1("last",   bVector).start();
    }
}
```

11.44. Questions

threadQuestion.pdf (Src/11/threadQuestion.pdf)

11.45. Questions from Students

```
 1      import java.util.*;
 2      public class M extends Thread    {
 3          private String info;
 4          private Vector aVector;
 5
 6          public M (String info) {
 7              this.info    = info;
 8          }
 9          private synchronized void inProtected () {
10              System.err.println(info + ": is in protected()");
11              try {
12                      sleep(1000);
13              }
14              catch (  InterruptedException e ) {
15                  System.err.println("Interrupted!");
16              }
17              System.err.println(info + ": exit run");
18          }
19          public void run () {
20              inProtected();
21          }
```

```
22          public static void main (String args []) {
23              Vector aVector = new Vector();
24              M aT4_0 = new M("first");
25              M at5_0 = new M("second");
26
27              aT4_0.start();
28              at5_0.start();
29              aT4_0.inProtected();
30              at5_0.inProtected();
31          }
32      }
33
```

Source Code: Src/StudentT_Q/M.java

Next

```
 1      /*
 2
 3      Q2. If the object is synchronized in the main method, what is the signific
 4
 5      Q3.In what scenario will a running thread go to a ready state?
 6
 7      Q4.In the slide numbered 12.8, it says Interface methods cannot be sychron
 8      */
 9      public class T_7 extends Thread    {
10          static String  theValue ;
11          T_7(String theValue)        {
12              this.theValue = theValue;
13          }
14          public void run () {
15              synchronized ( theValue )       {
16                      if ( this.theValue.equals("1") )
17                              theValue = "3";
18                      else
19                              theValue = "4";
20                      }
21          }
22
23          public static void main (String args []) {
24              T_7 aT_7_1 = new T_7("1");
25              T_7 aT_7_2 = new T_7("1");
26              aT_7_1.run();
27              aT_7_2.run();
28              synchronized ( theValue )       {
29                      System.out.println("aT_7_1.theValue _7.i = " + aT_7_1.theV
30                      System.out.println("aT_7_2.theValue   = " + aT_7_2.theValu
31              }
32          }
33      }
```

Source Code: Src/StudentT_Q/T_7.java

12. Networking

12.1. The Subject

Computer networks are the biggest *Big New Thing* for decades. They are now of central importance to all information technology. With the recent explosive growth of the internet, they are rapidly "she" becoming of crucial importance to all of modern society.

12.2. A Network Architecture Example: WWW

The World Wide Web is the *Big New Thing* in computer networking.

History:

> In 1989, Tim Berners Lee proposed a global hypertext project, to be known as the World Wide Web. Based on the earlier "Enquire" work, it was designed to allow people to work together by combining their knowledge in a web of hypertext documents. Tim Berners Lee wrote the first World Wide Web server and the first client, a wysiwyg hypertext browser/editor which ran in the NeXTStep environment. This work was started in October 1990, and the program "WorldWideWeb" was first made available within CERN in December, and on the Internet at large in the summer of 1991.
>
> Through 1991 and 1993, Tim Berners Lee continued working on the design of the Web, coordinating feedback from users across the Internet. His initial specifications of URIs, HTTP and HTML were refined and discussed in larger circles as the Web technology spread.

See also: Tim Berners-Lee. (http://www.w3.org/People/Berners-Lee/)

A browser, or viewer program is used to fetch and display "pages" of information from a server. A page is simply an ASCII text file, written using a simple markup language called Hypertext Meta Language (HTML). You may find an introduction here. (http://www.utoronto.ca/webdocs/HTMLdocs/NewHTML/index.html)

Uniform Resource Locators - URLs

The URL is the basis of the WWW. Think of a URL as an address that can lead you to any file on any machine anywhere in the world. Unlike the common postal address, however, these are written backwards. (Actually backwards makes more sense. My postal address was:

HP Bischof
3002 ST RT 48
Oswego, 13126 NY,
USA.

vs.

HP Bischof
Am Kaninchen Weg 12
8456 Laupheim

But if you want to deliver a letter to me, shouldn't you first go to the USA, then NY, then Oswego, then 3002 ST RT 48, then to HP Bischof? The URL is written in that more logical order.)

A URL defines the location of a WWW page in the following way:

service:host:port/file and resource details

For example:

http://www.cs.rit.edu:80/~hpb/CS3/all-2.2.html#section4
http://www.av.digital.com/cgi-bin/query?pg=q&what=web

URLs on the Web don't have to use the HTTP protocol. Some other URLs you might encounter are:

ftp file transfer protocol

news
for Usenet news groups

telnet
for telnet

mailto
to send email to a specific address

Connection Establishment

To fetch a WWW page, the browser application process running on your local computer first establishes a connection to the remote host.

What this means is that the browser process uses the facilities of the network connecting the two computers to send a "connection request" message to a server process running on the computer whose name was given in the URL.

If the remote server process is prepared to accept the connection, it responds with a "connection accepted" message.

Note that we are, for the moment, ignoring the process of "looking up" the remote host - discovering the network address associated with its domain name.

The HTTP Protocol

Once the two application processes have an established connection between them, they can communicate reliably.

The browser then sends a request, in ordinary plain text, to the server, thus:

```
GET /home.html
```

The string *GET something* is one of many commands defined in the Hypertext Transfer Protocol, HTTP. The server responds by returning the contents of a file.

Finally, the browser process interprets the HTML markup in the returned file, and displays it to the user.

12.3. What is the Internet

The Internet (short for inter networking, the practice of linking technologically different and independently operated networks), is a network of networks which allows users to communicate using electronic mail, to retrieve data stored in databases, and to access distant computers. The "core" of Internet includes the National Science Foundation's NSFNET, the Department of Energy's Energy Science Network (ESnet), the NASA Science Internet (NSI) as well as Defense's ARPANET and Terrestrial Wideband Network (TWBnet). Internet also includes a larger, and continually expanding, collection of interconnected regional, campus, and other networks throughout the U.S.
and overseas, as well as several networks that provide service on a for-profit basis.

These linked networks are independently operated; there is no central control of Internet. Internet began as an Advanced Research
Projects Agency research project to investigate computer networking technology. The networks that comprise the National Research and
Education Network (NREN), a component of the High Performance Computing and Communications Program, are a part of the current
Internet.

Copied from: www.onelook.com. (www.onelook.com)

12.4. Protocol

(From: An Internet Encyclopedia) (http://www.FreeSoft.org/CIE/index.htm)
Douglas Comer defines a protocol as "a formal description of message formats and the rules two or more machines must follow to exchange those messages."

Protocols usually exist in two forms. First, they exist in a textual form for humans to understand. Second, they exist as programming code for computers to understand. Both forms should ultimately specify the precise interpretation of every bit of every message exchanged across a network.

Protocols exist at every point where logical program flow crosses between hosts. In other words, we need protocols every time we want to do something on another computer. Every time we want to print something on a network printer we need protocols. Every time we want to download a file we need protocols. Every time we want to save our work on disk, we don't need protocols - unless the disk is on a network file server.

Usually multiple protocols will be in use simultaneously. For one thing, computers usually do several things at once, and often for several people at once. Therefore, most protocols support multitasking. Also, one operation can involve several protocols. For example, consider the NFS (Network File System) protocol. A write to a file is done with an NFS operation, that uses another protocol (RPC) to perform a function call on a remote host, that uses another protocol (UDP) to deliver a datagram to a port on a remote host, that uses another protocol to deliver a datagram on an Ethernet, and so on. Along the way we made need to lookup host names (using the DNS protocol), convert data to a network standard form (using the XDR protocol), find a routing path to the host (using one or many of numerous protocols) - I think you get the idea.

12.5. Protocol Layers

Protocol layering is a common technique to simplify networking designs by dividing them into functional layers, and assigning protocols to perform each layer's task.

For example, it is common to separate the functions of data delivery and connection management into separate layers, and therefore separate protocols. Thus, one protocol is designed to perform data delivery, and another protocol, layered above the first, performs connection management. The data delivery protocol is fairly simple and knows nothing of connection management. The connection management protocol is also fairly simple, since it doesn't need to concern itself with data delivery.

Protocol layering produces simple protocols, each with a few well-defined tasks. These protocols can then be assembled into a useful whole. Individual protocols can also be removed or replaced.

The most important layered protocol designs are the Internet's original DoD model, and the OSI Seven Layer Model. The modern Internet represents a fusion of both models.

- DoD Four-Layer Model (http://www.FreeSoft.org/CIE/Topics/16.htm)
- OSI Seven-Layer Model (http://www.FreeSoft.org/CIE/Topics/15.htm)

12.6. The OSI Seven-Layer Model

(From: An Internet Encyclopedia) In the 1980s, the European-dominated International Standards Organization (ISO), began to develop its Open Systems Interconnection (OSI) networking suite. OSI has two major components: an abstract model of networking (the Basic Reference Model, or — seven-layer model —), and a set of concrete protocols. The standard documents that describe OSI are for sale and not currently available online.

Parts of OSI have influenced Internet protocol development, but none more than the abstract model itself, documented in OSI 7498 and its various addenda. In this model, a networking system is divided into layers. Within each layer, one or more entities implement its functionality. Each entity interacts directly only with the layer immediately beneath it, and provides facilities for use by the layer above it. Protocols enable an entity in one host to interact with a corresponding entity at the same layer in a remote host.

```
[

boxht=0.5i boxwid=2i Application: box                    "Application"

Presentation: box with .n at last box .s          "Presentation"

Session: box with .n at last box .s          "Session"

Transport: box with .n at last box .s          "Transport"

Network: box with .n at last box .s          "Network"

DataLink: box with .n at last box .s          "Data Link"

Physiscal: box with .n at last box .s          "Physical"

]
```

The seven layers of the OSI Basic Reference Model are (from bottom to top):

- The Physical Layer describes the physical properties of the various communications media, as well as the electrical properties and interpretation of the exchanged signals. This layer defines the size of Ethernet coaxial cable, the type of BNC connector used, and the termination method.
- The Data Link Layer describes the logical organization of data bits transmitted on a particular medium. Ex: this layer defines the framing, addressing and checksumming of Ethernet packets.
- The Network Layer describes how a series of exchanges over various data links can deliver data between any two nodes in a network. Ex: this layer defines the addressing and routing structure of the Internet.
- The Transport Layer describes the quality and nature of the data delivery. Ex: this layer defines if and how retransmissions will be used to ensure data delivery.
- The Session Layer describes the organization of data sequences larger than the packets handled by lower layers. Ex: this layer describes how request and reply packets are paired in a remote procedure call.
- The Presentation Layer describes the syntax of data being transferred. Ex: this layer describes how floating point numbers can be exchanged between hosts with different math formats.

- The Application Layer describes how real work actually gets done. Ex: this layer would implement file system operations.

The original Internet protocol specifications defined a four-level model, and protocols designed around it (like TCP) have difficulty fitting neatly into the seven-layer model. Most newer designs use the seven-layer model.

12.7. TCP/IP

TCP/IP is the essential two-layer program that each Internet point-of-presence (POP) or SLIP/PPP user must use.

The *Transmission Control Protocol* (a protocol is a formal set of rules for communicating) manages the packaging of data into the packets that get routed on different paths over the Internet and reassembled at their destination.

The *Internet Protocol* handles the address part of each data packet so that it is routed to the right destination.

TCP/IP can be used on many data-link layers (can support many network hardware implementations).

These two protocols are the most important, TCP/IP is really a suite of protocols. (Some of these are viewed as alternative protocols and others as application protocols.) The ones you are most likely to use (directly or indirectly) are: HTTP, FTP, Telnet, Gopher, PPP, and SMTP.

Related protocols, some of them may be included in a TCP/IP package:

- User Datagram Protocol (UDP)
- Telnet
- File Transfer Protocol (FTP)
- Trivial File Transfer Protocol (TFTP)
- Simple Mail Transfer Protocol (SMTP)
- Gopher protocol
- Hypertext Transport Protocol (HTTP)

12.8. TCP/IP Layer

TCP/IP is normally considered to be a 4 layer system:

.so Pic/16/tcp_ip.pic

Link layer:

the network interface layer. Includes normally the device driver in an OS.

Network layer:

The network layer handles the movement of the packets around a network. Routing of packets takes place here. IP (Internet Protocol), ICMP (Internet Control Message Protocol), and IGMP (Internet Group Management Protocol) provide the network layer in the TCP/IP suite.

Transport Layer:

Provides a flow of data between two hosts for the application layer. There are two different protocols: TCP (Transmission Control Protocol) and UDP (User Datagram Protocol).

The function of the TCP protocol, is to provide a:

reliable	all data is delivered correctly
connection-oriented	the protocol provides procedures for establishing interprocess connections.
byte stream	ie, no visible packetisation so far as the application processes are concerned
end-to-end	

... interprocess communication service.

The User Datagram Protocol provides a connectionless alternative transport service to TCP for applications where reliable stream service is not needed. UDP datagrams can be dropped, duplicated or delivered out of order, same as for IP.

Application Layer:

The application layer handles the details of the particalur application.

- telnet for remote login
- ftp the file transfer protocol
- SMTP simple mail transfer protocol
- SNMP simple network management protocol

An example:

.so Pic/16/use_tcp_ip.pic

12.9. Internet Addresses

Every Internet-connected system has a unique Internet host address.

This is a 32 bit, or 4 byte, binary number.

Internet addresses are written as a dotted sequence of the form:

a.b.c.d

where *a, b, c* and *d* etc, are the decimal values (ranging from 0 to 255) of the 4 bytes which make up the internet address, for example:

129.21.36.56

129.21.36.56 is the IP address of ilon,or to use its full name

spiegel.cs.rit.edu

We will later see how the name of a computer is mapped to its IP-address.

12.10. IP Address Classes

```
[

boxht=0.5i boxwid=0.19i y_dist=0.8i line_ht=0.5 bit_border=0.1

define bits Y
  for i = 1 to 32 do X          box with .w at last box.e          if ( i % 8 == 0 ) then Z
                line from last box.ne + ( 0, bit_border) to                                  last box.se - ( 0,
bit_border)          Z
  X Y

define tags Y        # Class_box first second text
  line from $1.ne + ( $2 * boxwid, 0 ) to                $1.ne + ( $2 * boxwid, line_ht )
  line from $1.ne + ( $3 * boxwid, 0 ) to                $1.ne + ( $3 * boxwid, line_ht )
  box invis with .c at  $1.ne + ( ( ( $3-$2 )  / 2 + $2 ) * boxwid, boxht / 2 ) $4 Y

A: box invis wid 1i "Class A"          bits(A) A_1: box with .w at A.e + ( 0 * boxwid, 0 ) "0"
        tags(A, 1,   8,    "7 bits — netid")          tags(A, 8,   32,   "24 bits — hostid")

B: box invis wid 1i "Class B" with .n at A.s - ( 0, y_dist )          bits(B) B_1: box with .w at B.e
+ ( 0 * boxwid, 0 ) "1" B_1: box with .w at B.e + ( 1 * boxwid, 0 ) "0"          tags(B, 2,  16,
"14 bits — netid")          tags(B,16,  32,    "16 bits — hostid")

C: box invis wid 1i "Class C" with .n at B.s - ( 0, y_dist )          bits(C) C_1: box with .w at C.e
+ ( 0 * boxwid, 0 ) "1" C_1: box with .w at C.e + ( 1 * boxwid, 0 ) "1" C_1: box with .w at C.e
+ ( 2 * boxwid, 0 ) "0"          tags(C, 3,  24,    "21 bits — netid")          tags(C,24,  32,    "8 bits
— hostid")

D: box invis wid 1i "Class D" with .n at C.s - ( 0, y_dist )          bits(D) D_1: box with .w at
D.e + ( 0 * boxwid, 0 ) "1" D_1: box with .w at D.e + ( 1 * boxwid, 0 ) "1" D_1: box with .w at
D.e + ( 2 * boxwid, 0 ) "1" D_1: box with .w at D.e + ( 3 * boxwid, 0 ) "0"          tags(D, 4,
32,    "28 bits — multicast group id")

E: box invis wid 1i "Class E" with .n at D.s - ( 0, y_dist )          bits(E) E_1: box with .w at E.e
+ ( 0 * boxwid, 0 ) "1" E_1: box with .w at E.e + ( 1 * boxwid, 0 ) "1" E_1: box with .w at E.e
+ ( 2 * boxwid, 0 ) "1" E_1: box with .w at E.e + ( 3 * boxwid, 0 ) "1" E_1: box with .w at E.e
+ ( 4 * boxwid, 0 ) "0"          tags(E, 5,  32,    "27 bits — reserved for future use")

]
```

12.11. Ethernet Address

An IP-address only makes sense to the TCP/IP suite. A data link such as an Ethernet or a token ring has its own addressing scheme (often 48 bits).

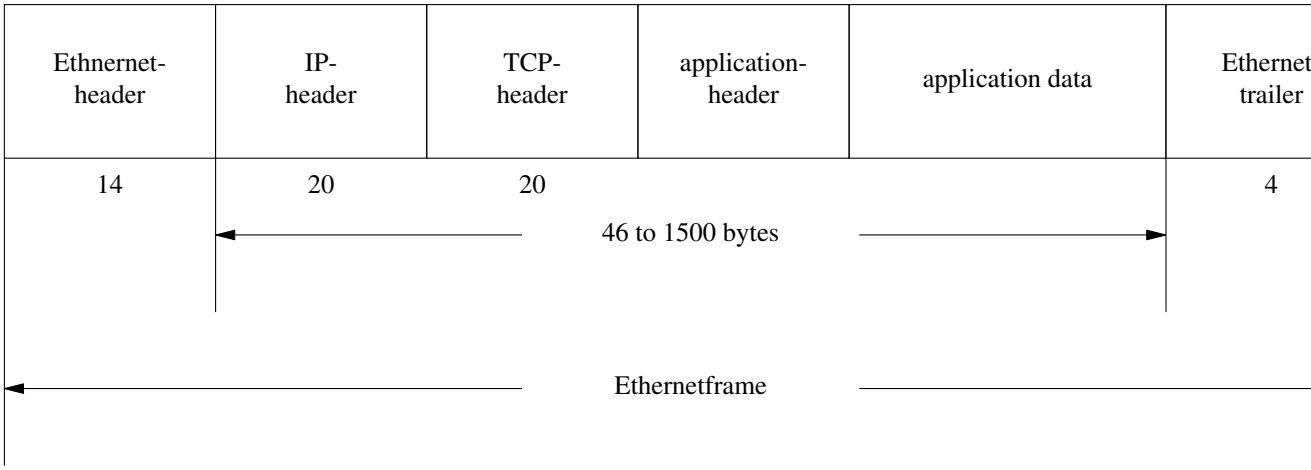
The-byte address is often used, which is divided into a 3-byte vendor ID and a 3-byte vendor-defined field. Ethernet manufacturers are assigned a unique vendor ID, and are then responsible for insuring that all of their devices have unique addresses in the last 3 bytes.

A network, such as an Ethernet can be used by different network layers at the same time. See also RFC 802. (http://www.FreeSoft.org/CIE/RFC/Orig/rfc802.txt)

12.12. Encapsulation

When an application sends data using TCP is sent down the protocol stack.

A physical property of an Ethernet frame is, that the size of its data must be between 46 and 1500 bytes.



12.13. TCP Ports

TCP is using protocol port numbers to identify the ultimate destination.

How does one determine the port to communicate with?

- Well known ports
- Randomly assigned ports.

See http://www.ietf.org/assignments/service-names-port-numbers/service-names-port-numbers.txt (http://www.ietf.org/assignments/service-names-port-numbers/service-names-port-numbers.txt)

12.14. Socket

Two popular Application Programming Interface using TCP/IP protocols are called sockets and TLI (transport layer interface).

A socket is one end-point of a two-way communication link between two programs running on the network. Socket classes are used to represent the connection between a client program and a server program. The java.net package provides two classes--Socket and ServerSocket--that implement the client side of the connection and the server side of the connection, respectively.

A socket is a network communications endpoint.

A socket is an object from which messages are sent and received.

Socket operations resemble file operations in many respects:

- Data transfer operations on sockets work just like read and write operations on files.
- A socket is closed, just like a file, when communications is finished.

See also: java.net (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/net/package-summary.html)

12.15. java.net

Through the classes in java.net, Java programs can use TCP or UDP to communicate over the Internet. The URL, URLConnection, Socket, and ServerSocket classes all use TCP to communicate over the network. The DatagramPacket, DatagramSocket, and MulticastSocket classes are for use with UDP.

12.16. Getting Information

Host info

```
 1      import java.net.*;
 2      import java.io.*;
 3      import java.util.*;
 4      public class HostInfo {
 5          public static void main(String argv[]) {
 6              InetAddress ipAddr;
 7              try {
 8                      ipAddr = InetAddress.getLocalHost();
 9                      System.out.println ("This is "+ipAddr);
10              } catch (UnknownHostException e) {
11                      System.out.println ("Unknown host");
12              }
13          }
14      }
```

Source Code: Src/16/HostInfo.java

Output:

```
% java HostInfo
This is spiegel.cs.rit.edu/129.21.36.56
```

12.17. Makefile for the Execution of the following Examples

```
 1      dayTimeRun:
 2              javac DayTimeServer.java
 3              java DayTimeServer      -port 3456 &
 4              sleep 1
 5              java DayTime            -port 3456
 6              kill `ps -t \`tty\` | grep DayTimeServer | sed 's/ .*//'`
 7              echo $(pid)
 8
 9      echoRun:
10              javac EchoServer.java
11              java EchoServer -port 4567 &
12              sleep 1
```

```
13              java HpEchoSocketTest -port 4567
14              kill `ps -t \`tty\` | grep EchoServer | sed 's/ .*//'`
15              echo $(pid)
16
17      mtsRun:
18              java MTS -port 63782 &
19              java MTSclient -port 63782; java MTSclient -port 63782
20              kill `ps -t \`tty\` | grep MTS | sed 's/ .*//'`
21              echo $(pid)
```

Source Code: Src/16/makefile

12.18. Daytime Client

see */etc/services*.

```
 1      import java.net.*;
 2      import java.io.*;
 3      import java.util.*;
 4      public class DayTime {
 5
 6          String hostName = "spiegel.cs.rit.edu";
 7          int    port = 13;
 8
 9          private void printMessage() {
10              System.out.println("-h          ---->   help");
11              System.out.println("[-host              hostName]");
12              System.out.println("[-port              port]");
13         }
14
15         /**
16           * Parse the commandlind arguments and sets variables.
17           */
18         public void parseArgs(String args[]) {
19
20              for (int i = 0; i < args.length; i ++) {
21                      if (args[i].equals("-h"))
22                              printMessage();
23                      else if (args[i].equals("-host"))
24                              hostName = args[++i];
25                      else if (args[i].equals("-port"))
26                              port = new Integer(args[++i]).intValue();
27              }
28         }
29
30         public void doTheJob()       {
31              try {
32                      System.out.println("host: " +  hostName );
33                      System.out.println("port: " +  port );
34                      Socket aReadSocket = new Socket(hostName, port);
35
36                      System.out.println("aReadSocket = " + aReadSocket );
37                      BufferedReader readFrom = new BufferedReader (
38                              new InputStreamReader (aReadSocket.getInputStream(
```

```
39                      String rTime = readFrom.readLine ();
40                      System.out.println (rTime);
41                      aReadSocket.close();
42              } catch (Exception e) {
43                      System.out.println (e);
44              }
45         }
46
47         public static void main(String argv[]) {
48              DayTime aDayTime = new DayTime();
49              aDayTime.parseArgs(argv);
50              aDayTime.doTheJob();
51
52         }
53      }
```

Source Code: Src/16/DayTime.java

Output:

```
% java DayTime
host: spiegel.cs.rit.edu
port: 13
java.net.ConnectException: Operation timed out
```

12.19. Daytime Server

```
 1      import java.net.*;
 2      import java.io.*;
 3      import java.util.*;
 4      // java DayTimeServer -port 12345
 5      public class DayTimeServer extends Thread {
 6
 7         ServerSocket         aServerSocket;
 8         int                  port     = 4242;
 9
10         public DayTimeServer()       {
11         }
12
13         public DayTimeServer(int port)       {
14              try {
15                  aServerSocket = new ServerSocket(port);
16                  System.out.println ("Listening on port: " + aServerSocket.getI
17              } catch(Exception e) {
18                  System.out.println(e);
19              }
20         }
21
22          private void printMessage() {
23              System.out.println("-h          ---->   help");
24              System.out.println(" -port              port");
25              System.out.println(" {-port             port}");
```

```
26              System.out.println("or ");
27              System.out.println(" no argument");
28          }
29
30          /**
31            * Parse the commandlind arguments and sets variables.
32            */
33          private  void parseArgs(String args[]) {
34
35              for (int i = 0; i < args.length; i ++) {
36                      if (args[i].equals("-h"))
37                              printMessage();
38                      else if (args[i].equals("-port")) {
39                              port = new Integer(args[++i]).intValue();
40                              new DayTimeServer(port).start();
41                      }
42              }
43          }
44
45          public void run()    {
46              try {
47                  for(;;) {
48                      Socket connectionToClientSocket = aServerSocket.accept();
49                      System.out.println(connectionToClientSocket.toString());
50                      PrintWriter out = new PrintWriter
51                          (connectionToClientSocket.getOutputStream (), true);
52                      out.println("It is now: " + new Date());
53                      connectionToClientSocket.close();
54                  }
55              } catch(Exception e) {
56                  System.out.println(e);
57                  e.printStackTrace();
58              }
59          }
60
61           public static void main(String argv[]) {
62              if ( argv.length == 0 )
63                      new DayTimeServer(0).start();
64              else
65                      new DayTimeServer().parseArgs(argv);
66           }
67      }
```

Source Code: Src/16/DayTimeServer.java

Output:

```
% java DayTimeServer &
java DayTimeServer      -port 3456 &
[1] 25419
spiegel.cs.rit.edu 16 166 Listening on port: 3456
spiegel.cs.rit.edu 16 166 java DayTime            -port 3456
host: spiegel.cs.rit.edu
port: 3456
```

```
Socket[addr=/129.21.36.56,port=59400,localport=3456]
aReadSocket = Socket[addr=spiegel.cs.rit.edu/129.21.36.56,port=3456,localport=5940
It is now: Wed Nov 13 10:57:18 EST 2019
```

12.20. Reading from and Writing to a Socket

```
 1      import java.io.*;
 2      import java.net.*;
 3
 4      class HpEchoSocketTest {
 5
 6             Socket aSocket;
 7             String hostName = "localhost";
 8             int port;
 9             PrintWriter outPutStream = null;
10             BufferedReader inPutStream = null;
11
12             public HpEchoSocketTest()        {
13             }
14             public HpEchoSocketTest(String name, int port) {
15                    hostName = name;
16                    this.port = port;
17             }
18
19             public void parseArgs(String args[]) {
20
21                  for (int i = 0; i < args.length; i ++) {
22                         if (args[i].equals("-host"))
23                                hostName = args[++i];
24                         else if (args[i].equals("-port"))
25                                port = new Integer(args[++i]).intValue();
26                  }
27             }
28
29             public void createIOconections() throws Exception {
30                    try {
31                           aSocket = new Socket(hostName, port);
32                           outPutStream = new PrintWriter( aSocket.getOutputStre
33                           inPutStream = new BufferedReader( new InputStreamRead
34                    } catch (Exception e )       {
35                           System.out.println(e.toString());
36                           System.exit(1);
37                    }
38             }
39             public void closeIOconections() throws Exception {
40                    try {
41                           inPutStream.close();
42                           outPutStream.close();
43                    } catch (Exception e )       {
44                           System.out.println(e.toString());
45                           System.exit(1);
46                    }
47             }
```

```
48              public void readAndPrint() throws Exception {
49                     InputStream ins;
50                     OutputStream os;
51
52                     BufferedReader stdIn = new BufferedReader(
53                                            new InputStreamReader(System.in));
54                     String userInput;
55
56                     while ((userInput = stdIn.readLine()) != null) {
57                              outPutStream.println(userInput);
58                              System.out.println("echo: " + inPutStream.readLine
59                     }
60                     stdIn.close();
61              }
62
63                public void doTheJob()       {
64                   try {
65                           System.out.println("host: " +  hostName );
66                           System.out.println("port: " +  port );
67                           HpEchoSocketTest aHpEchoSocketTest = new HpEchoSocketT
68                           createIOconections();
69                           readAndPrint();
70                           closeIOconections();
71                   } catch (Exception e) {
72                           System.out.println (e);
73                   }
74              }
75
76              public static void main(String[] args) {
77                     HpEchoSocketTest st;
78                     String host = "spiegel.cs.rit.edu";
79                     int    port = 12345;
80                     HpEchoSocketTest aHpEchoSocketTest = new HpEchoSocketTest();
81                     aHpEchoSocketTest.parseArgs(args);
82                     aHpEchoSocketTest.doTheJob();
83              }
84
85      }
```

Source Code: Src/16/HpEchoSocketTest.java

EchoServer:

```
 1      import java.net.*;
 2      import java.io.*;
 3      import java.util.*;
 4      // java EchoServer -port 12345
 5      public class EchoServer extends Thread {
 6
 7         ServerSocket         aServerSocket;
 8         int                  port     = 4242;
 9
10         public EchoServer()  {
11         }
```

```
12
13          public EchoServer(int port)  {
14               try {
15                   aServerSocket = new ServerSocket(port);
16                   System.out.println ("Listening on port: " + aServerSocket.getL
17               } catch(Exception e) {
18                   System.out.println(e);
19               }
20          }
21
22           private void printMessage() {
23               System.out.println("-h          ---->   help");
24               System.out.println(" -port              port");
25               System.out.println(" {-port             port}");
26               System.out.println("or ");
27               System.out.println(" no argument");
28          }
29
30          /**
31            * Parse the commandlind arguments and sets variables.
32            */
33          private  void parseArgs(String args[]) {
34
35               for (int i = 0; i < args.length; i ++) {
36                       if (args[i].equals("-h"))
37                               printMessage();
38                       else if (args[i].equals("-port")) {
39                               port = new Integer(args[++i]).intValue();
40                               new EchoServer(port).start();
41                       }
42               }
43          }
44
45          public void run()    {
46               try {
47                   for(;;) {
48                       Socket clientSocket = aServerSocket.accept();
49                       System.out.println(clientSocket.toString());
50                       PrintWriter out = new PrintWriter (clientSocket.getOutputS
51                       BufferedReader in = new BufferedReader( new InputStreamRea
52                       while ( true )  {
53                               String input = in.readLine();
54                               if ( input == null ) System.exit(0);
55                               System.out.println("sending back: " + input );
56                               out.println("back: " + input );
57                       }
58                       // clientSocket.close();
59                   }
60               } catch(Exception e) {
61                   System.out.println(e);
62                   e.printStackTrace();
63               }
64          }
65
```

```
66          public static void main(String argv[]) {
67              if ( argv.length == 0 )
68                      new EchoServer(12345).start();
69              else
70                      new EchoServer().parseArgs(argv);
71          }
72      }
```

Source Code: Src/16/EchoServer.java

Output:

```
% java  EchoServer
Listening on port: 12345
% java HpEchoSocketTest -port 12345
host: null
port: 12345
jkjkj
echo: back: jkjkj
lkklk
echo: back: lkklk
```

These are the typical steps:

1. Open a socket.
2. Open an input stream and output stream to the socket.
3. Read from and write to the stream according to the server's protocol.
4. Close the streams.
5. Close the socket.

12.21. Multi Client Server and Client

• telnet allows many clients to connect

12.22. MTS Server:

```
 1      // java MTS
 2
 3      import java.net.*;
 4      import java.io.*;
 5      import java.util.*;
 6
 7      public class MTS extends Thread {
 8
 9         ServerSocket         listen;
10         // static String     hostName = "spiegel.cs.rit.edu";
11         static String        hostName = "localhost";
12         int                  port     = 4242;
13         int                  id       = 0;
14
15         public MTS() {
```

```
16              listen = null;
17          }
18
19          public MTS(int port) {
20              try {
21                  listen = new ServerSocket(port);
22                  System.out.println ("Listening on port: " + getLocalPort());
23              } catch(Exception e) {
24                  System.out.println(e);
25              }
26          }
27          public MTS(int port, int id) {
28              this(port);
29              this.id = id;
30          }
31
32          public int getLocalPort ()   {
33              return listen.getLocalPort();
34          }
35
36           private void printMessage() {
37              System.out.println("-h          ---->   help");
38              System.out.println("[-host              hostName");
39              System.out.println(" -port              port");
40              System.out.println(" {-port             port}");
41              System.out.println("or ");
42              System.out.println(" no argument");
43          }
44
45          /**
46            * Parse the commandlind arguments and sets variables.
47            */
48          public void parseArgs(String args[]) {
49
50              for (int i = 0; i < args.length; i ++) {
51                      if (args[i].equals("-h"))
52                              printMessage();
53                      else if (args[i].equals("-host"))
54                              hostName = args[++i];
55                      else if (args[i].equals("-port")) {
56                              port = new Integer(args[++i]).intValue();
57                              new MTS(port).listenToPort();
58                      }
59              }
60          }
61
62          public void run()    {
63              try {
64                      System.out.println("Wating for client to connect " + liste
65                      Socket clientConnection = listen.accept();
66                      System.out.println(clientConnection.toString());
67                      PrintWriter out = new PrintWriter (clientConnection.getOut
68                      out.println("It is now: " + new Date());
69                      System.out.println(id + " .... falling asleep");
```

```
70                      sleep(1000);
71                      System.out.println("\t" + id + " .... wake up");
72                      listen.close();
73                  } catch(Exception e) {
74                      System.out.println(e);
75                      e.printStackTrace();
76                  }
77              }
78
79              public void listenToPort()   {
80                  try {
81                      int id = 0;
82                      for(;;) {
83                          System.out.println("Wating for client to connect " + liste
84                          Socket clientConnection = listen.accept();
85                          System.out.println("Somebody connected ... ");
86                          MTS aServer = new MTS(0, id++);
87                          aServer.start();
88                          System.out.println("offer ... " + aServer.getLocalPort());
89                          PrintWriter out = new PrintWriter (clientConnection.getOut
90                          out.println(aServer.getLocalPort());
91                          clientConnection.close();
92                      }
93                  } catch(Exception e) {
94                      System.out.println(e);
95                      e.printStackTrace();
96                  }
97              }
98
99               public static void main(String argv[]) {
00                  if ( argv.length == 0 )
01                          new MTS(0).listenToPort();
02                  else
03                          new MTS().parseArgs(argv);
04               }
05          }
```

Source Code: Src/16/MTS.java

12.23. MTS Client:

```
 1      // java MTSclient -host spiegel -port 50405
 2      import java.net.*;
 3      import java.io.*;
 4      import java.util.*;
 5      public class MTSclient {
 6
 7          // String hostName = "spiegel.cs.rit.edu";
 8          String hostName = "localhost";
 9          int    port = 63782;
10
11          private void printMessage() {
12              System.out.println("-h          ---->   help");
```

```
13              System.out.println("[-host              hostName]");
14              System.out.println("[-port              port]");
15          }
16
17          /**
18            * Parse the commandlind arguments and sets variables.
19            */
20          public void parseArgs(String args[]) {
21
22              for (int i = 0; i < args.length; i ++) {
23                      if (args[i].equals("-h"))
24                              printMessage();
25                      else if (args[i].equals("-host"))
26                              hostName = args[++i];
27                      else if (args[i].equals("-port"))
28                              port = new Integer(args[++i]).intValue();
29              }
30          }
31
32          public void doTheJob()       {
33              try {
34                      Socket socket = new Socket(hostName, port);
35
36                      BufferedReader dataStreamIn = new BufferedReader (
37                              new InputStreamReader (socket.getInputStream()));
38                      String newPort = dataStreamIn.readLine ();
39                      System.out.println ("Use from now in port: " + newPort);
40                      socket.close();
41                      dataStreamIn.close();
42                      socket = new Socket(hostName, new Integer(newPort).intValu
43                      dataStreamIn = new BufferedReader ( new InputStreamReader
44                      System.out.println("got: " + dataStreamIn.readLine () );
45              } catch (Exception e) {
46                      System.out.println (e);
47              }
48          }
49
50          public static void main(String argv[]) {
51              MTSclient aMTSclient = new MTSclient();
52              aMTSclient.parseArgs(argv);
53              aMTSclient.doTheJob();
54
55          }
56      }
```

Source Code: Src/16/MTSclient.java

• Result:

```
# Window 1:
Listening on port: 53660
Wating for client to connect ServerSocket[addr=0.0.0.0/0.0.0.0,localport=53660]
Somebody connected ...
Listening on port: 53665
```

```
offer ... 53665
Wating for client to connect ServerSocket[addr=0.0.0.0/0.0.0.0,localport=53665]
Wating for client to connect ServerSocket[addr=0.0.0.0/0.0.0.0,localport=53660]
Socket[addr=/129.21.36.56,port=53666,localport=53665]
0 .... falling asleep
      0 .... wake up
Somebody connected ...
Listening on port: 53674
offer ... 53674
Wating for client to connect ServerSocket[addr=0.0.0.0/0.0.0.0,localport=53674]
Wating for client to connect ServerSocket[addr=0.0.0.0/0.0.0.0,localport=53660]
Socket[addr=/129.21.36.56,port=53675,localport=53674]
1 .... falling asleep
      1 .... wake up

# Window 2:
% java MTSclient -host spiegel.cs.rit.edu -port 53660
Use from now in port: 53665
got: It is now: Mon Nov  9 08:44:03 EST 2020
```

12.24. Connection to an URL

Class URL represents a Uniform Resource Locator, a pointer to a "resource" on the World Wide Web. A resource can be something as simple as a file or a directory, or it can be a reference to a more complicated object, such as a query to a database or to a search engine. More information on the types of URLs and their formats can be found at:
http://www.ncsa.uiuc.edu/demoweb/url-primer.html (http://www.ncsa.uiuc.edu/demoweb/url-primer.html)

```
 1      import java.io.*;
 2      import java.net.URL;
 3      import java.net.MalformedURLException;
 4
 5      public class Url_Read {
 6
 7        public static void readFromUrl(String theUrl) {
 8
 9        URL aUrl = null;
10        BufferedReader in = null;
11        String line;
12
13        try {
14                aUrl = new URL(theUrl);
15                System.out.println("getPort() " + aUrl.getPort());
16                System.out.println("getHost() " + aUrl.getHost());
17                System.out.println("getProtocol() " + aUrl.getProtocol());
18                System.out.println("getFile() " + aUrl.getFile());
19                System.out.println("getRef() " + aUrl.getRef());
20
21                in = new BufferedReader(
22                         new InputStreamReader( aUrl.openStream() ) );
23
24                while ( ( line = in.readLine() ) != null ) {
25                      System.out.println(line);
```

```
26                }
27
28                in.close();
29
30        } catch (MalformedURLException e) {
31                System.err.println("Something is wrong with this " +
32                      theUrl +  ".");
33                System.exit(1);
34        } catch (IOException e) {
35                System.err.println("Couldn't get I/O for the connection to: "
36                      + theUrl );
37                System.exit(1);
38        }
39
40        }
41
42        public static void main( String args[] ) {
43
44          if ( args.length != 1 )     {
45              System.err.println(
46                   "Usage: java Url_Read url");
47              System.exit(1);
48          }
49
50          try {
51              readFromUrl(args[0]);
52
53          }
54          catch ( NumberFormatException e)    {
55              System.out.println(args[0] + " is not a number ;-(");
56          }
57
58        }
59      }
```

Source Code: Src/16/Url_Read.java

Output:

```
% java Url_Read http://www.cs.rit.edu/~hpb | sed 15q
getPort() -1
getHost() www.cs.rit.edu
getProtocol() http
getFile() /~hpb
getRef() null
<HTML>

<HEAD>
<title>Hans-Peter Bischof's Home Page</title>
</HEAD>


<FRAMESET cols="230,*">
  <frame name="toc"    TARGET="_main" src="toc.html"     scrolling="auto">
```

```
<frame name="intro"                src="intro.html"   scrolling="auto">
```

12.25. Datagram Socket

• A datagram socket is the sending or receiving point for a packet delivery service.

• Each packet sent or received on a datagram socket is individually addressed and routed.

• Multiple packets sent from one machine to another may be routed differently, and may arrive in any order.

• UDP broadcasts sends and receives are always enabled on a DatagramSocket.

An example:

12.26. Datagram Server:

```
 1      import java.net.*;
 2      import java.io.*;
 3      import java.util.*;
 4
 5      public class DayTimeUDPServer extends Thread {
 6
 7         DatagramSocket       socket;
 8         static String        hostName = "spiegel";
 9         int                  port     = 1313;
10
11
12         public DayTimeUDPServer()    {
13         }
14
15         public DayTimeUDPServer(int port)    {
16              try {
17                  socket = new DatagramSocket(port);
18                  System.out.println ("Listening on port: "
19                      + socket.getLocalPort());
20              } catch(Exception e) {
21                  System.out.println(e);
22              }
23         }
24
25          private void printMessage() {
26              System.out.println("-h          ---->   help");
27              System.out.println("[-host              hostName");
28              System.out.println(" -port              port");
29              System.out.println(" {-port             port}");
30              System.out.println("or ");
31              System.out.println(" no argument");
32         }
33
34         /**
35           * Parse the commandlind arguments and sets variables.
36           */
37         public void parseArgs(String args[]) {
38
39              for (int i = 0; i < args.length; i ++) {
40                      if (args[i].equals("-h"))
41                              printMessage();
```

```
42                      else if (args[i].equals("-host"))
43                              hostName = args[++i];
44                      else if (args[i].equals("-port")) {
45                              port = new Integer(args[++i]).intValue();
46                              new DayTimeUDPServer(port).start();
47                      }
48              }
49          }
50
51          public void run()    {
52              byte[] buf = new byte[256];
53              try {
54                  for(;;) {
55                      String sendThis = "it is now: " + new Date();
56                      DatagramPacket packet = new DatagramPacket(buf, buf.length
57                      socket.receive(packet);
58                      InetAddress address = packet.getAddress();
59                      int port = packet.getPort();
60                      buf = sendThis.getBytes();
61                      packet = new DatagramPacket(buf, buf.length, address, port
62                      System.out.println("Sending to port: " + port );
63                      System.out.println("Sending    data: " + new String(buf) )
64                      socket.send(packet);
65                  }
66              } catch(Exception e) {
67                  System.out.println(e);
68                  e.printStackTrace();
69              }
70          }
71
72           public static void main(String argv[]) {
73              if ( argv.length == 0 )
74                      new DayTimeUDPServer(0).start();
75              else
76                      new DayTimeUDPServer().parseArgs(argv);
77           }
78      }
```

Source Code: Src/16/DayTimeUDPServer.java

Datagram Client:

```
 1      import java.net.*;
 2      import java.io.*;
 3      import java.util.*;
 4      public class DayTimeUDP {
 5
 6           String hostName = "localhost";
 7           int    port = 1313;
 8
 9           private void printMessage() {
10               System.out.println("-h          ---->   help");
11               System.out.println("[-host              hostName]");
12               System.out.println("[-port              port]");
```

```
13          }
14
15          /**
16            * Parse the commandlind arguments and sets variables.
17            */
18          public void parseArgs(String args[]) {
19
20               for (int i = 0; i < args.length; i ++) {
21                       if (args[i].equals("-h"))
22                               printMessage();
23                       else if (args[i].equals("-host"))
24                               hostName = args[++i];
25                       else if (args[i].equals("-port"))
26                               port = new Integer(args[++i]).intValue();
27               }
28          }
29
30          public void doTheJob()       {
31               try {
32                       byte buf[] = new byte[64];
33                       InetAddress aInetAddress = InetAddress.getByName(hostName);
34                       DatagramPacket dp = new DatagramPacket(buf, buf.length);
35                       DatagramSocket socket = new DatagramSocket();
36                       DatagramPacket packet = new DatagramPacket(buf,
37                               buf.length, aInetAddress, port);
38                       socket.send(packet);
39
40                       System.out.println("host: " +  hostName );
41                       System.out.println("port: " +  port );
42                       System.out.println("after creation");
43                       socket.receive(dp);
44                       System.out.println("received: -" +
45                               new String(dp.getData() ) + "-"  );
46                       socket.close();
47               } catch (Exception e) {
48                       System.out.println (e);
49                       e.printStackTrace();
50               }
51          }
52
53          public static void main(String argv[]) {
54               DayTimeUDP aDayTimeUDP = new DayTimeUDP();
55               aDayTimeUDP.parseArgs(argv);
56               aDayTimeUDP.doTheJob();
57
58          }
59      }
```

Source Code: Src/16/DayTimeUDP.java

• Execution:

```
# Window 1:
% java DayTimeUDP –port  50666
host: localhost
port: 50666
after creation
received: –it is now: Wed Nov 11 08:31:28 EST 2020–

# Window 2:
% java DayTimeUDPServer
Listening on port: 50666
Sending to port: 56640
Sending    data: it is now: Wed Nov 11 08:31:28 EST 2020
```

12.27. Remote Method Invocation

See also http://java.sun.com/docs/books/tutorial/rmi. (http://java.sun.com/docs/books/tutorial/rmi)

Part of the text and programs are from there. Copyright belongs to Ann Wollrath (http://java.sun.com/docs/books/tutorial/information/bios.html#wollrath)
and Jim Waldo. (http://java.sun.com/docs/books/tutorial/information/bios.html#waldo)

See also http://java.sun.com/products/jdk/1.2/docs/guide/rmi/. (http://java.sun.com/products/jdk/1.2/docs/guide/rmi/)

The Java Remote Method Invocation (RMI) system allows an object running in one Java Virtual Machine (VM) to invoke methods on an object running in another Java VM. RMI provides for remote communication between programs written in the Java programming language.

Distributed object applications need to:

Locate remote objects

> Applications can use one of two mechanisms to obtain references to remote objects. An application can register its remote objects with RMI's simple naming facility, the rmiregistry, or the application can pass and return remote object references as part of its normal operation.

Communicate with remote objects

> Details of communication between remote objects are handled by RMI; to the programmer, remote communication looks like a standard Java method invocation.

Load class bytecodes for objects that are passed as parameters or return values

Because RMI allows a caller to pass pure Java objects to remote objects, RMI provides the necessary mechanisms for loading an object's code as well as transmitting its data.

The illustration below depicts an RMI distributed application that uses the registry to obtain references to a remote object. The server calls the registry to associate a name with a remote object. The client looks up the remote object by its name in the server's registry and then invokes a method on it.

12.28. Remote Method Invocation: Idea

The problem is: you need a reference of an object, before you can send a method to it.

12.29. Remote Method Invocation: Idea II

12.30. SenderProxy/Receiver Proxy.png

12.31. Stubs

```
1       // Stub class generated by rmic, do not edit.
2       // Contents subject to change without notice.
3
4       public final class HelloImplementation_Stub
5           extends java.rmi.server.RemoteStub
6           implements HelloInterface, java.rmi.Remote
7       {
8           private static final long serialVersionUID = 2;
9
10          private static java.lang.reflect.Method $method_sayHello_0;
11
12          static {
13              try {
14                  $method_sayHello_0 = HelloInterface.class.getMethod("sayHello"
15              } catch (java.lang.NoSuchMethodException e) {
16                  throw new java.lang.NoSuchMethodError(
17                      "stub class initialization failed");
18              }
19          }
20
21          // constructors
22          public HelloImplementation_Stub(java.rmi.server.RemoteRef ref) {
23              super(ref);
24          }
25
26          // methods from remote interfaces
27
28          // implementation of sayHello()
29          public java.lang.String sayHello()
30              throws java.rmi.RemoteException
31          {
32              try {
33                  Object $result = ref.invoke(this, $method_sayHello_0, null, 60
34                  return ((java.lang.String) $result);
35              } catch (java.lang.RuntimeException e) {
36                  throw e;
37              } catch (java.rmi.RemoteException e) {
38                  throw e;
39              } catch (java.lang.Exception e) {
40                  throw new java.rmi.UnexpectedException("undeclared checked exc
41              }
42          }
43      }
```

```
Source Code: Src/16_D/HelloImplementation_Stub.java
```

Note: *rmic -keep* can generate the stubs.

12.32. Remote Method Innovation Registry

Where does the sender proxy come from?

12.33. Passing Non-remote Objects

A non-remote object, that is passed as a parameter of a remote method invocation or returned as a result of a remote method invocation, is passed by copy; that is, the object is serialized using the Java Object Serialization mechanism.

So, when a non-remote object is passed as an argument or return value in a remote method invocation, the content of the non-remote object is copied before invoking the call on the remote object.

When a non-remote object is returned from a remote method invocation, a new object is created in the calling virtual machine

12.34. Advantages of Dynamic Code Loading

- download the bytecodes (or simply code) of an object's class if the class is not defined in the receiver's virtual machine.
- types and the behavior of an object, previously available only in a single virtual machine, can be transmitted to another, possibly remote, virtual machine.
- RMI passes objects by their true type, so the behavior of those objects is not changed when they are sent to another virtual machine.
- allows new types to be introduced into a remote virtual machine, thus extending the behavior of an application dynamically.

12.35. Remote Interfaces, Objects, and Methods

- distributed application built using Java RMI is made up of interfaces and classes.
- In a distributed application some of the implementations are assumed to reside in different virtual machines.
- Objects that have methods that can be called across virtual machines are remote objects.
- An object becomes remote by implementing a remote interface, which has the following characteristics.
  - — A remote interface extends the interface java/rmi.Remote.
  - — Each method of the interface declares java/rmi.RemoteException in its throws clause, in addition to any application-specific exceptions.
- RMI passes a remote stub for a remote object.
- The stub acts as the local representative, or proxy, for the remote object and basically is, to the caller, the remote reference.
- The caller invokes a method on the local stub, which is responsible for carrying out the method call on the remote object.
- A stub for a remote object implements the same set of remote interfaces that the remote object implements.
- This allows a stub to be cast to any of the interfaces that the remote object implements.
- This also means that only those methods defined in a remote interface are available to be called in the receiving virtual machine.

12.36. Creating Distributed Applications Using RMI

1. Design and implement the components of your distributed application.
2. Compile sources and generate stubs.
3. Make classes network accessible.
4. Start the application.

Compile Sources and Generate Stubs

Compile as usual ... see makefile

Make Classes Network Accessible

In this step you make everything--the class files associated with the remote interfaces, stubs, and other classes that need to be downloaded to clients.

Start the Application

Starting the application includes running the RMI remote object registry, the server, and the client.

12.37. Intro Example

We have to design a protocol that allows jobs to be submitted to the server and results of the job to be returned to the client. This protocol is expressed in interfaces supported by the server and by the objects that are submitted to the sever, as shown in the following figure.

box with .sw at (1.00,8.62) width 1.25 height 0.75 box with .sw at (3.75,8.62) width 1.25 height 0.75 line -> from 2.250,9.125 to 3.750,9.125 line from 3.750,8.750 to 3.750,8.750 line -> from 3.750,8.875 to 2.250,8.875 "Client" at 1.250,8.914 ljust "Server" at 4.000,8.914 ljust "submit task" at 2.375,9.289 ljust "return results" at 2.375,8.539 ljust

12.38. Hello World

The Client:

```
 1      import java.rmi.*;
 2
 3      public class HelloC {
 4              public static void main(String args[] ) {
 5              String message = "";
 6              try {
 7                      // HelloInterface obj = (HelloInterface)Naming.lookup("//s
 8                      HelloInterface obj = (HelloInterface)Naming.lookup("//loca
 9
10                      message = obj.sayHello();
11                      System.out.println(message);
12
13                      message = obj.sayHello(3);
14                      System.out.println(message);
15
16              } catch (Exception e) {
17                      System.out.println("HelloC exception: " +
18                      e.getMessage());
19                      e.printStackTrace();
20              }
21        }
22      }
```

Source Code: Src/16_D/HelloC.java

The interface:

```
1       public interface HelloInterface extends java.rmi.Remote {
2               String sayHello() throws java.rmi.RemoteException;
3               String sayHello(int i) throws java.rmi.RemoteException;
4       }
```

Source Code: Src/16_D/HelloInterface.java

The implementation of the interface:

```
 1      import java.rmi.*;
 2      import java.rmi.server.UnicastRemoteObject;
 3
 4      public class HelloImplementation
 5              extends UnicastRemoteObject
 6              implements HelloInterface {
 7
 8              int state;
 9              static int counter;
10
11              public HelloImplementation() throws RemoteException {
12              }
13
14              public String sayHello() throws RemoteException {
15                      counter ++;
16                      return  "Spiegel: Hello World my Friend!";
17              }
18              public String sayHello(int state) throws RemoteException {
19                      this.state = state;
20                      counter ++;
21                      return  "sayHello: " + counter + "/" + state;
22              }
23      }
```

Source Code: Src/16_D/HelloImplementation.java

The implementation of the server:

```
 1      import java.rmi.*;
 2
 3      public class HelloServer {
 4
 5              public static void main(String args[])
 6              {
 7                      // System.setSecurityManager(new RMISecurityManager());
 8
 9                      try {
10                          HelloInterface obj = new HelloImplementation();
11                          HelloInterface obj2 = new HelloImplementation();
12                          HelloInterface obj3 = new HelloImplementationSleep();
13                          Naming.rebind("//localhost/HelloServer", obj);
14                          Naming.rebind("//localhost/HelloServer2", obj2);
15                          Naming.rebind("//localhost/HelloServer3", obj3);
16                          // Naming.rebind("//129.21.36.56/HelloServer", obj);
17                          System.out.println("HelloServer bound in registry");
18                      } catch (Exception e) {
19                          System.out.println("HelloImpl err: " + e.getMessage())
20                          e.printStackTrace();
21                          System.exit(0);
22                      }
23              }
24      }
```

Source Code: Src/16_D/HelloServer.java

Compilation and Execution

```
all:
       javac HelloInterface.java
       javac HelloImplementation.java
       # rmic HelloImplementation
       javac HelloC.java HelloServer.java
       rmiregistry &
       sleep 1
       java HelloServer &
       sleep 4
       java HelloC

         Spiegel: Hello World my Friend!
         sayHello: 2/3
```

The stub:

Source Code: Src/16_D/Keep_HelloImplementation_Stub.java

Trying to use rmic:

```
rmic -keep CbVImplementation
Warning: generation and use of skeletons and static stubs for JRMP
is deprecated. Skeletons are unnecessary, and static stubs have
been superseded by dynamically generated stubs. Users are
encouraged to migrate away from using rmic to generate skeletons and static
stubs. See the documentation for java.rmi.server.UnicastRemoteObj
```

12.39. Modified Hello World

Client:

```
 1      import java.rmi.*;
 2
 3      public class HelloC2 {
 4              public static void main(String args[] ) {
 5              String message = "";
 6              try {
 7                      // HelloInterface obj = (HelloInterface)Naming.lookup("//s
 8                      HelloInterface obj1 = (HelloInterface)Naming.lookup("//loc
 9                      HelloInterface obj2 = (HelloInterface)Naming.lookup("//loc
10
11                      message = obj1.sayHello(1);
12                      System.out.println("obj1: " + message);
13
14                      message = obj2.sayHello(2);
15                      System.out.println("obj1: " + message);
16
17                      // obj = (HelloInterface)Naming.lookup("//localhost/HelloS
18                      message = obj.sayHello(2);
19                      System.out.println("obj1: " + message);
20
21              } catch (Exception e) {
22                      System.out.println("HelloC exception: " +
23                      e.getMessage());
24                      e.printStackTrace();
25              }
26        }
27      }
```

Source Code: Src/16_D/HelloC2.java

Output:

```
java HelloC2
obj1: sayHello: 1/1
obj1: sayHello: 2/2
obj1: sayHello: 3/2
```

12.40. RMI and Threads

The Client:

```
 1      import java.rmi.*;
 2
 3      public class HelloCSleep {
 4              public static void main(String args[] ) {
 5              String message = "";
 6              try {
 7                      // HelloInterface obj = (HelloInterface)Naming.lookup("//s
 8                      HelloInterface obj1 = (HelloInterface)Naming.lookup("//loc
 9
10                      message = obj1.sayHello(1);
11                      System.out.println("obj1: " + message);
12                      message = obj1.sayHello(2);
13                      System.out.println("obj1: " + message);
14
15              } catch (Exception e) {
16                      System.out.println("HelloC exception: " +
17                      e.getMessage());
18                      e.printStackTrace();
19              }
20        }
21      }
```

Source Code: Src/16_D/HelloCSleep.java

Implementation:

```
 1      import java.rmi.*;
 2      import java.rmi.server.UnicastRemoteObject;
 3
 4      public class HelloImplementationSleep
 5              extends UnicastRemoteObject
 6              implements HelloInterface {
 7
 8              int state;
 9              static int counter;
10
11              public HelloImplementationSleep() throws RemoteException {
12              }
13
14              public String sayHello() throws RemoteException {
15                      counter ++;
16                      System.out.println("sayHello before sleep");
17                      try { Thread.sleep(10000); } catch (  InterruptedException
18                      System.out.println("sayHello after sleep");
19                      return  "Spiegel: Hello World my Friend!";
20              }
21              public String sayHello(int state) throws RemoteException {
22                      System.out.println("sayHello(int) before sleep");
23                      try { Thread.sleep(10000); } catch (  InterruptedException
24                      System.out.println("sayHello(int) after sleep");
25                      this.state = state;
26                      counter ++;
27                      return  "sayHello: " + counter + "/" + state;
```

```
28              }
29      }
```

Source Code: Src/16_D/HelloImplementationSleep.java

Server:

```
 1      import java.rmi.*;
 2
 3      public class HelloServer {
 4
 5              public static void main(String args[])
 6              {
 7                      // System.setSecurityManager(new RMISecurityManager());
 8
 9                      try {
10                          HelloInterface obj = new HelloImplementation();
11                          HelloInterface obj2 = new HelloImplementation();
12                          HelloInterface obj3 = new HelloImplementationSleep();
13                          Naming.rebind("//localhost/HelloServer", obj);
14                          Naming.rebind("//localhost/HelloServer2", obj2);
15                          Naming.rebind("//localhost/HelloServer3", obj3);
16                          // Naming.rebind("//129.21.36.56/HelloServer", obj);
17                          System.out.println("HelloServer bound in registry");
18                      } catch (Exception e) {
19                          System.out.println("HelloImpl err: " + e.getMessage())
20                          e.printStackTrace();
21                          System.exit(0);
22                      }
23              }
24      }
```

Source Code: Src/16_D/HelloServer.java

Execution:

```
make allsleep
javac HelloInterface.java
javac HelloImplementationSleep.java
# rmic HelloImplementation
javac HelloCSleep.java HelloServer.java
rmiregistry &
sleep 4
java HelloServer&
sleep 4
HelloServer bound in registry
java HelloCSleep    &
sleep 1
sayHello(int) before sleep
java HelloCSleep    &
java HelloCSleep
sayHello(int) before sleep
sayHello(int) before sleep
sayHello(int) after sleep
```

```
obj1: sayHello: 1/1
sayHello(int) before sleep
sayHello(int) after sleep
sayHello(int) after sleep
obj1: sayHello: 2/1
obj1: sayHello: 3/1
sayHello(int) before sleep
sayHello(int) before sleep
sayHello(int) after sleep
obj1: sayHello: 4/2
sayHello(int) after sleep
obj1: sayHello: 5/2
sayHello(int) after sleep
obj1: sayHello: 6/2
```

12.41. Argument Passing Example

Interface:

```
1       public interface CbVInterface extends java.rmi.Remote {
2               public String callByValue(int anArray[]) throws java.rmi.RemoteExc
3       }
```

Source Code: Src/16_CbV/CbVInterface.java

Implementation:

```
1       import java.rmi.*;
2       import java.rmi.server.UnicastRemoteObject;
3
4       public class CbVImplementation
5               extends UnicastRemoteObject
6               implements CbVInterface {
7
8               int id = 0;
9               public CbVImplementation() throws RemoteException {
10              }
11
12              public String callByValue(int array[]) throws RemoteException {
13                      array[0] = -42;
14                      return  id + ": CbV World my Friend!";
15              }
16      }
```

Source Code: Src/16_CbV/CbVImplementation.java

Client:

```
1       import java.rmi.*;
2
3
4       public class CbVC {
5               public static void makeCall(String id)  {
6                       String message = "";
7                       try {
8                               CbVInterface obj = (CbVInterface)Naming.lookup("//
9                               int anArrayLocal[] = new int[2];
10
11                              anArrayLocal[0] = 42;
12                              System.out.println("    before Call: anArrayLocal[
13                              message = obj.callByValue(anArrayLocal);
14                              System.out.println("    after  Call: anArrayLocal[
15
16                              System.out.println(id + " - " + message);
17                      } catch (Exception e) {
18                              System.out.println("CbVC exception: " +
19                              e.getMessage());
20                              e.printStackTrace();
21                      }
22              }
23              public static void main(String args[] ) {
24                      makeCall("first");
25              }
26      }
```

Source Code: Src/16_CbV/CbVC.java

Server:

```
1       import java.rmi.*;
2
3       public class CbVServer {
4
5               public static void main(String args[])
6               {
7                       // System.setSecurityManager(new RMISecurityManager());
8
9                       try {
10                          CbVInterface obj = new CbVImplementation();
11                          Naming.rebind("//spiegel/IamACbVImplementationObject",
12                          // Naming.rebind("//129.21.36.56/CbVServer", obj);
13                          System.out.println("CbVServer bound in registry");
14                      } catch (Exception e) {
15                          System.out.println("CbVImpl err: " + e.getMessage());
16                          e.printStackTrace();
17                      }
18              }
19      }
```

Source Code: Src/16_CbV/CbVServer.java

Execution:

```
% java CbVC
       before Call: anArrayLocal[0] = 42
       after  Call: anArrayLocal[0] = 42
first - 0: CbV World my Friend!
```

12.42. RMISecurityManager

The RMISecurityManager (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/rmi.RMISecurityManager.html)
class defines a default security policy for RMI applications (not applets). For code loaded from a class loader, the security manager disables all functions except class definition and access. This class may be subclassed to implement a different policy. To set a RMISecurityManager, add the following to an application's main() method:

```
System.setSecurityManager(new RMISecurityManager());
```

If no security manager has been set, RMI will only load classes from local system files as defined by CLASSPATH.

Naming

(http://www.cs.rit.edu/usr/local/jdk/docs/api/java/rmi.Naming.html#_top_Naming)
is the bootstrap mechanism for obtaining references to remote objects based on Uniform Resource Locator (URL) syntax. The URL for a remote object is specified using the usual host, port and name:

```
rmi://host:port/name
```

host = host
: name of registry (defaults to the current host)

port = port
: number of registry (defaults to the registry port number)

name = name
: for remote object

The makefile:

```
 1
 2
 3      all:
 4              javac HelloInterface.java
 5              javac HelloImplementation.java
 6              # rmic HelloImplementation
 7              javac HelloC.java HelloServer.java
 8              rmiregistry &
 9              sleep 4
10              java HelloServer &
11              sleep 4
12              java HelloC
13
14      all2:
15              javac HelloInterface.java
16              javac HelloImplementation.java
17              # rmic HelloImplementation
18              javac HelloC2.java HelloServer.java
19              rmiregistry &
20              sleep 4
21              java HelloServer&
22              sleep 4
23              java HelloC2
24      allsleep:
25              javac HelloInterface.java
```

```
26              javac HelloImplementationSleep.java
27              # rmic HelloImplementation
28              javac HelloCSleep.java HelloServer.java
29              rmiregistry &
30              sleep 4
31              java HelloServer&
32              sleep 4
33              java HelloCSleep        &
34              sleep 1
35              java HelloCSleep        &
36              java HelloCSleep
37
38
39      clean:
40              rm -f *class
41
```

Source Code: Src/16_D/makefile

Execution:

```
% make -f Makefile
javac HelloInterface.java
javac HelloImplementation.java
# rmic HelloImplementation
javac HelloC.java HelloServer.java
rmiregistry &
sleep 1
java HelloServer &
sleep 4
HelloServer bound in registry
java HelloC
Hello World my friend.
```

Note: Make sure that *rmiregistry* is dead before you log out!

Note: Make sure that every *java server* is dead before you log out!

```
 1      #!/bin/sh
 2
 3      killThisProcess ()      {       # $1
 4              echo $1
 5              ps -axl                 |       \
 6              grep "$1"               |       \
 7              grep -v grep            |       \
 8              grep -v kill            |       \
 9              grep -v vi              |       \
10              awk ' { print $2 }'     |       \
11              while read x
12              do
13                      echo "kill -9 $x"
14                      kill -9 $x 2> /dev/null
15              done
16
17      }
18      killThisProcess "rmiregistry"
19      killThisProcess "java"
```

Source Code: Src/16_D/killIt

Beware of:

```
% ps -edf | grep java
```

The registry by default runs on port 1099. To start the registry on a different port, specify the port number in the command. For example, to start the registry on port 2001:

```
% rmiregistry 2001
```

For example, if the registry is running on port 2001 in the Hello World example, here is the call required to bind HelloServer to the remote object reference:

```
Naming.lookup("//yps:2001/HelloServer", obj);
```

12.43. Hello World II

The Client:

```
 1      import java.rmi.*;
 2
 3      public class HelloC {
 4              public static void main(String args[] ) {
 5              String message = "";
 6              try {
 7                      Hello obj =
 8                              (Hello)Naming.lookup("//spiegel.cs.rit.edu:2001/H
 9
10                      message = obj.sayHello();
11
12                      System.out.println(message);
13
14              } catch (Exception e) {
15                      System.out.println("Something went wrong: " +
16                              e.getMessage());
17                      e.printStackTrace();
18              }
19        }
20      }
```

Source Code: Src/16_P/HelloC.java

The Server:

```
 1      import java.rmi.*;
 2      import java.rmi.server.UnicastRemoteObject;
 3
 4      public class HelloImpl
 5              extends UnicastRemoteObject
 6              implements Hello
 7      {
 8              private String name;
 9
10              public HelloImpl(String s) throws RemoteException {
11                      name = s;
12              }
13
14              public String sayHello() throws RemoteException {
15                      return  "Stanley Kubrick was there!";
16              }
17
18              public static void main(String args[])
19              {
20                      // Create and install a security manager
21                      // System.setSecurityManager(new RMISecurityManager());
22
23                      try {
24                              HelloImpl obj = new HelloImpl("HelloServer");
25                              Naming.rebind("//spiegel.cs.rit.edu:2001/HelloServ
26                              System.out.println("HelloServer bound in registry"
27                      } catch (Exception e) {
28                              System.out.println("HelloImpl err: " + e.getMessag
29                              e.printStackTrace();
30                      }
31              }
32      }
```

Source Code: Src/16_P/HelloImpl.java

The interface:

```
1       public interface Hello extends java.rmi.Remote {
2               String sayHello() throws java.rmi.RemoteException;
3       }
```

Source Code: Src/16_P/Hello.java

The makefile:

```
 1
 2
 3      all: Hello.class HelloC.class HelloImpl.class   \
 4           HelloImpl_Skel.class HelloImpl_Stub.class
 5
 6              rmiregistry 2001 &
 7              sleep 1
 8              java HelloImpl &
 9              sleep 4
10              java HelloC
11              killIt java
12              killIt
13
14
15      HelloImpl_Skel.class HelloImpl_Stub.class: HelloImpl.java
16              rmic HelloImpl
17
18      Hello.class:    Hello.java
19              javac Hello.java
20
21      HelloC.class:   HelloC.java
22              javac HelloC.java
23
24      HelloImpl.class:        HelloImpl.java
25              javac HelloImpl.java
26
27      clean:
28              rm -f *class
```

Source Code: Src/16_P/makefile

12.44. Multiple Servers

Client:

```
 1      import java.rmi.*;
 2
 3      public class Client {
 4              public static void main(String args[] ) {
 5              String message = "";
 6              try {
 7                      MyServer obj =
 8                              (MyServer)Naming.lookup("//localhost:2001/Server1
 9                      message = obj.sayHello();
10                      System.out.println(message);
11
12
13                      obj = (MyServer)Naming.lookup("//localhost:2001/Server2");
14                      message = obj.sayHello();
15                      System.out.println(message);
16
17              } catch (Exception e) {
18                      System.out.println("Something went wrong: " +
19                      e.getMessage());
20                      e.printStackTrace();
21              }
22        }
23      }
```

Source Code: Src/16_M/Client.java

Server 1:

```
 1      import java.rmi.*;
 2      import java.rmi.server.UnicastRemoteObject;
 3
 4      public class Server1Impl
 5              extends UnicastRemoteObject
 6              implements MyServer
 7      {
 8              private String name;
 9
10              public Server1Impl(String s) throws RemoteException {
11                      name = s;
12              }
13
14              public String sayHello() throws RemoteException {
15                      return  "Server1()";
16              }
17
18              public static void main(String args[])
19              {
20                      // Create and install a security manager
21                      // System.setSecurityManager(new RMISecurityManager());
22
23                      try {
24                              Server1Impl obj = new Server1Impl("Server1");
25                              Naming.rebind("//localhost:2001/Server1", obj);
26                              System.out.println("Server1 bound in registry");
27                      } catch (Exception e) {
28                              System.out.println("Server1Impl err: "
29                                 + e.getMessage());
30                              e.printStackTrace();
31                      }
32              }
33      }
```

Source Code: Src/16_M/Server1Impl.java

Server:

Server 2:

```
 1      import java.rmi.*;
 2      import java.rmi.server.UnicastRemoteObject;
 3
 4      public class Server2Impl
 5              extends UnicastRemoteObject
 6              implements MyServer
 7      {
 8              private String name;
 9
10              public Server2Impl(String s) throws RemoteException {
11                      name = s;
12              }
13
14              public String sayHello() throws RemoteException {
15                      return  "Server2()";
16              }
17
18              public static void main(String args[])
19              {
20                      // Create and install a security manager
21                      // System.setSecurityManager(new RMISecurityManager());
22
23                      try {
24                              Server2Impl obj = new Server2Impl("Server2");
25                              Naming.rebind("//localhost:2001/Server2", obj);
26                              System.out.println("Server2Server bound in registr
27                      } catch (Exception e) {
28                              System.out.println("Server2Impl err: "
29                                 + e.getMessage());
30                              e.printStackTrace();
31                      }
32              }
33      }
```

Source Code: Src/16_M/Server2Impl.java

Make execution:

```
% make
rmic Server1Impl
rmic Server2Impl
javac Client.java
rmiregistry 2001 &
sleep 1
java Server1Impl &
java Server2Impl &
sleep 4
Server2Server bound in registry
Server1 bound in registry
java Client
Server1()
```

```
Server2()
```

12.45. Running Multiple Server on different Machines

- Before a client can connect to a server, a server process must run on this machine.
- Server can be started — on a UNIX box — during
  - — boot time
  - — via *inetd*
  - — "by hand"

- In order to start a a server by hand, you have to log on this machine.
- ssh
- rsh ????

12.46. Startup Multiple Server on different Machines

Client:

```
 1      import java.rmi.*;
 2      import java.math.*;
 3
 4      public class Client {
 5
 6          public static void doIt(String catServer, String mouseServer, int port
 7
 8              MyServer aCatServer;
 9              MyServer aMouseServer;
10              Point    aPoint = new Point(4, 2 );
11
12              System.out.println("In Client: cat   is on: " + catServer );
13              System.out.println("In Client: mouse is on: " + mouseServer );
14              System.out.println("In Client: port     is: " + port );
15              try {
16                      aCatServer = (MyServer)Naming.lookup("rmi://" +
17                              catServer + ":" + port + "/CatServer");
18
19                      aMouseServer = (MyServer)Naming.lookup("rmi://" +
20                              mouseServer + ":" + port + "/MouseServer");
21
22
23      // -------------- Cat --------------------
24              System.out.println("In Client: aCatServer.movePoint(aPoint): " +
25                   (aPoint = aCatServer.movePoint(aPoint)).toString() );
26              System.out.println("In Client: aCatServer.movePoint(aPoint): " +
27                              aCatServer.movePoint(aPoint).toString() );
28              System.out.println("In Client: aCatServer.movePoint(aPoint): " +
29                              aCatServer.movePoint(aPoint).toString() );
30
31      // -------------- Mouse --------------------
32
33              System.out.println("In Client: aMouseServer.movePoint(aPoint): " +
```

```
34                  (aPoint = aMouseServer.movePoint(aPoint)).toString() );
35              System.out.println("In Client: aMouseServer.movePoint(aPoint): " +
36                              aMouseServer.movePoint(aPoint).toString() );
37              System.out.println("In Client: aMouseServer.movePoint(aPoint): " +
38                              aMouseServer.movePoint(aPoint).toString() );
39
40
41
42              } catch (Exception e) {
43                      System.out.println("Something went wrong: " +
44                      e.getMessage());
45                      e.printStackTrace();
46              }
47
48          }
49
50           public static void main(String args[] ) {
51           int    port   = 1099;
52           String catServer   = "yps";
53           String mouseServer = "yps";
54
55           if ( args.length  >= 1 )
56               catServer = args[0];
57           if ( args.length  >= 2 )
58               mouseServer = args[1];
59           if ( args.length  == 3 )
60               try {
61                       port = Integer.parseInt(args[2]);
62               }
63               catch ( NumberFormatException e )       {
64                       System.out.println("Hm , port = " +
65                               args[2] + " is not valid.");
66                       System.exit(1);
67               }
68
69           if ( args.length  > 3 )     {
70               System.out.println("Usage: " +
71                               "java Client [CatServer [MouseServer [port]]]");
72               System.exit(1);
73           }
74
75           doIt(catServer, mouseServer, port);
76         }
77       }
```

Source Code: Src//16_MS/Client.java

Interface:

```
1       public interface MyServer extends java.rmi.Remote {
2               Point movePoint(Point aPoint) throws java.rmi.RemoteException;
3               int getX()                    throws java.rmi.RemoteException;
4               int getY()                    throws java.rmi.RemoteException;
5       }
```

Source Code: Src//16_MS/MyServer.java

Cat Server

```
 1      import java.rmi.*;
 2      import java.rmi.server.UnicastRemoteObject;
 3
 4      public class CatServer extends UnicastRemoteObject implements MyServer
 5      {
 6              final private int DELTA = 10;
 7              private int x;
 8              private int y;
 9              private Point aPoint;
10
11              public CatServer() throws RemoteException {
12                      ;
13              }
14
15
16              public Point movePoint(Point aPoint) throws RemoteException {
17                      System.out.println("\tIN CatServer: movePoint(): "
18                              + aPoint.toString() );
19                      return aPoint.move(DELTA, DELTA);
20              }
21              public int getX() throws RemoteException {
22                      System.out.println("\tCIN atServer: getX(): " + x );
23                      return  x;
24              }
25
26              public int getY() throws RemoteException {
27                      System.out.println("\tCIN atServer: getY(): " + y );
28                      return  x;
29              }
30
31      public static void main(String args[])
32        {
33                int port = 1099;
34
35                // System.setSecurityManager(new RMISecurityManager());
36
37                if ( args.length  == 1 )
38                        try {
39                                port = Integer.parseInt(args[0]);
40                         }
41                         catch ( NumberFormatException e )    {
42                                System.out.println("Hm , port = " +
43                                args[0] + " is not valid.");
44                                System.exit(1);
45                          }
46
47                try {
48                        CatServer obj = new CatServer();
49                        System.out.println("\tIN CatServer: " +
50                                        "rmi://:" + port + "/CatServer");
51                        Naming.rebind("rmi://:" + port + "/CatServer", obj);
```

```
52                      System.out.println("\tIN CatServer bound in registry");
53              } catch (RemoteException  e) {
54                      System.out.println("CatServer RemoteException ");
55                      e.printStackTrace();
56              } catch (Exception e) {
57                      System.out.println("CatServer err: "
58                      + e.getMessage());
59                      e.printStackTrace();
60              }
61        }
62      }
```

Source Code: Src//16_MS/CatServer.java

Point Class:

```
 1      /**
 2       * This class implements a point in a two dimensional
 3       * area.
 4       * All methods print the method name, when they are called.
 5       * state information includes:
 6       *
 7       * @version   $Id$
 8       *
 9       * RIT's home page: <a href="http://www.cs.rit.edu/~hpb">RIT</a>
10       *
11       * Revisions:
12       *      $Log$
13       */
14      import java.io.*;
15
16
17      public class Point implements Serializable {
18
19        private int x;                // x coordinate of the point
20        private int y;                // y cooridnate of the point
21
22      /**
23       * Constructor.
24       * initialize x and y values of a point
25       *
26       * @param       x       x coordinate
27       * @param       y       y coordinate
28       *
29       * @return      a Point object
30       */
31        public Point(int _x, int _y){
32              this.x = _x;
33              this.y = _y;
34        }
35
36        private void writeObject(ObjectOutputStream s) throws IOException {
37                 s.defaultWriteObject();
38        }
39
40        private void readObject(ObjectInputStream s) throws IOException  {
41              try {
42                      s.defaultReadObject();
43              }
44              catch ( ClassNotFoundException e)       {
45                  System.out.println(e.getMessage());
46                  e.printStackTrace();
47              }
48        }
49      /**
50       * initialzes x and y of a point.
51       *
```

```
52       * @param       x       int x coordinate
53       * @param       y       int y coordinate
54       *
55       * @return      a Point object
56       */
57        public Point initPoint(int _x, int _y){
58
59              this.x = _x;
60              this.y = _y;
61
62              return this;
63        }
64
65      /**
66       * moves a point
67       *
68       * @param       _x      int delta x value
69       * @param       _y      int delta y value
70       *
71       * @return      a Point object
72       */
73        public Point move(int _x, int _y){
74
75              this.x += _x;
76              this.y += _y;
77              return this;
78        }
79
80      /**
81       * Returns the x coordinate of a point
82       *
83       * @return x value
84       */
85        public int getX(){
86              return this.x;
87        }
88
89      /**
90       * Returns the y coordinate of a point
91       *
92       * @return y value
93       */
94        public int getY(){
95              return this.y;
96        }
97
98      /**
99       * Returns a String reperesentation of the point
00       *
01       * @return String reprasentation of the point
02       */
03        public String toString(){
04              return "Point at (" + x + "/" + y + ")";
05        }
```

```
06      }
```

Source Code: Src/16_MS/Point.java

Makefile:

```
 1
 2
 3      all: Point.class                                          \
 4           CatServer_Skel.class CatServer_Stub.class  \
 5           MouseServer_Skel.class MouseServer_Stub.class      \
 6           MyServer.class Client.class
 7
 8              fireItUp
 9
10
11      CatServer_Skel.class CatServer_Stub.class: CatServer.java
12              rmic CatServer
13      MouseServer_Skel.class MouseServer_Stub.class: MouseServer.java
14              rmic MouseServer
15
16      MyServer.class: MyServer.java
17              javac MyServer.java
18
19      Client.class:   Client.java
20              javac Client.java
21
22      CatServer.class:        CatServer.java
23              javac CatServer.java
24
25      MouseServer.class:      MouseServer.java
26              javac MouseServer.java
27
28      Point.class:    Point.java
29              javac Point.java
30
31      clean:
32              rm -f *class
```

Source Code: Src/16_MS/makefile

Result:

```
        IN CatServer: //yps:2001/CatServer
        IN MouseServer: /yps:2001/MouseServer
        IN CatServer bound in registry
In Client: cat   is on: yps
In Client: mouse is on: yps
In Client: port     is: 2001
        IN MouseServer bound in registry
        IN CatServer: movePoint(): Point at (4/2)
In Client: aCatServer.movePoint(aPoint): Point at (14/12)
        IN CatServer: movePoint(): Point at (14/12)
In Client: aCatServer.movePoint(aPoint): Point at (24/22)
        IN CatServer: movePoint(): Point at (14/12)
In Client: aCatServer.movePoint(aPoint): Point at (24/22)
```

Start of a fireItUp Script:

```
#!/bin/sh

KILL_IT="killIt; killIt java"
ME="`who am i | sed 's/ .*//'`"
HOSTNAME="`hostname`"
USEDHOSTS="yps yps yps" # <-- modify here ...
WD=`pwd`


remote_cmd()            # bg host cmd
{
        echo "$HOSTNAME $ME" > $HOME/.rhosts
        if [ $1 = "bg" ]
        then
                rsh $2 "rm -f $HOME/.rhosts; cd $WD && $3" &
        else
                rsh $2 "rm -f $HOME/.rhosts; cd $WD && $3"
        fi
}

kill_all()
{
        for i in $USEDHOSTS
        do
                remote_cmd fg $i "$KILL_IT" 2>&1 > /dev/null
        done
}









kill_all
sleep 2

echo 1
rmiregistry &
echo "Waiting for rmiregistry .... chrr ... "; sleep 1
java CatServer &

echo 2
remote_cmd bg yps "rmiregistry &"
```

```
52
53      echo 3
54      echo "Waiting for rmiregistry .... chrr ... "; sleep 2
55      remote_cmd bg yps "java MouseServer &"
56
57      echo 4
58      echo "Waiting for the servers  .... chrr ... "; sleep 2
59      remote_cmd fg yps "java Client $HOSTNAME stones"
60
61      kill_all
62
63      exit 0
64
```

Source Code: Src/16_MS/fireItUp

12.47. Calculating PI

```
% make
javac Client.java
rmiregistry &
sleep 1
java PiServer &
sleep 4
        PiServer: PiServer()
PiServer bound in registry
java Client
3.1415926536
% java Client 100
3.14159265358979323846264338327950288419716939937510582097494459230781640628620899
86280348253421170680
%
```

Server Interface:

```
1       public interface MyServer extends java.rmi.Remote {
2               String sayHello() throws java.rmi.RemoteException;
3       }
```

Source Code: Src/16_M/MyServer.java

Class java/gmath.BigDecimal

Immutable, arbitrary-precision signed decimal numbers. A BigDecimal consists of an arbitrary precision integer value and a non-negative integer scale, which represents the number of decimal digits to the right of the decimal point. (The number represented by the BigDecimal is intVal/7**scale.) BigDecimals provide operations for basic arithmetic, scale manipulation, comparison, format conversion and hashing.

The compute engine, a remote object in the server, takes tasks from clients, runs them, and returns any results. The tasks are run on the machine where the server is running. This sort of distributed application could allow a number of client machines to make use of a particularly powerful machine or one that has specialized hardware.

Client:

```
 1      import java.rmi.*;
 2      import java.math.*;
 3
 4      public class Client {
 5
 6          public static void doIt(String host, String port, int digits) {
 7              String message = "";
 8              try {
 9                      MyServer obj = (MyServer)Naming.lookup("//" +
10                              host + ":" + port + "/PiServer");
11                      System.out.println(obj.computePi(digits));
12
13              } catch (Exception e) {
14                      System.out.println("Something went wrong: " +
15                      e.getMessage());
16                      e.printStackTrace();
17              }
18
19         }
20
21          public static void main(String args[] ) {
22          int    digits = 10;
23          String host   = "yps";
24          String port   = "";
25
26
27          if ( args.length  >= 1 )    {
28              try {
29                      digits = Integer.parseInt(args[0]);
30              }
31              catch ( NumberFormatException e )       {
32                      System.out.println("Hm , digits = " + args[0]);
33                      System.exit(1);
34              }
35
36          }
37          if ( args.length >= 2 )     {
38              host = args[1];
39          }
40
41          if ( args.length  == 3 )    {
42              try {
43                      port = args[2];
44                      Integer.parseInt(port);
45              }
46              catch ( NumberFormatException e )       {
47                      System.out.println("Port = " + port + " is not valid.");
```

```
48                      System.exit(1);
49              }
50
51          }
52          if ( args.length  > 3 )     {
53              System.out.println("Usage: java Client [digits [host [port]]]");
54              System.exit(1);
55          }
56          doIt(host, port, digits);
57        }
58      }
```

Source Code: Src/16_C/Client.java

Interface:

```
1       import java.math.*;
2
3       public interface MyServer extends java.rmi.Remote {
4               BigDecimal computePi(int digits)
5               throws java.rmi.RemoteException;
6       }
```

Source Code: Src/16_C/MyServer.java

Server:

```
 1      import java.rmi.*;
 2      import java.math.*;
 3      import java.rmi.server.UnicastRemoteObject;
 4
 5      public class PiServer
 6              extends UnicastRemoteObject
 7              implements MyServer
 8      {
 9          /** constants used in pi computation */
10          private static final BigDecimal ZERO =
11              BigDecimal.valueOf(0);
12          private static final BigDecimal  ONE =
13              BigDecimal.valueOf(1);
14          private static final BigDecimal FOUR =
15              BigDecimal.valueOf(4);
16
17          /** rounding mode to use during pi computation */
18          private static final int roundingMode =
19              BigDecimal.ROUND_HALF_EVEN;
20
21          /** digits of precision after the decimal point */
22          private int digits;
23
24
25          /**
26           * Construct a task to calculate pi to the specified
27           * precision.
28           */
29          public PiServer() throws RemoteException {
30              System.out.println("\tPiServer: PiServer()");
31          }
32          /**
33           * Compute the value of pi to the specified number of
34           * digits after the decimal point.  The value is
35           * computed using Machin's formula:
36           *
37           *          pi/4 = 4*arctan(1/5) - arctan(1/239)
38           *
39           * and a power series expansion of arctan(x) to
40           * sufficient precision.
41           */
42          public BigDecimal computePi(int digits) throws RemoteException {
43              int scale = digits + 5;
44              BigDecimal arctan1_5 = arctan(5, scale);
45              BigDecimal arctan1_239 = arctan(239, scale);
46              BigDecimal pi = arctan1_5.multiply(FOUR).subtract(
47                                          arctan1_239).multiply(FOUR);
48              return pi.setScale(digits,
49                                 BigDecimal.ROUND_HALF_UP);
50          }
51          /**
```

```
52          * Compute the value, in radians, of the arctangent of
53          * the inverse of the supplied integer to the speficied
54          * number of digits after the decimal point.  The value
55          * is computed using the power series expansion for the
56          * arc tangent:
57          *
58          * arctan(x) = x - (x^3)/3 + (x^5)/5 - (x^7)/7 +
59          *     (x^9)/9 ...
60          */
61         public static BigDecimal arctan(int inverseX,
62                                         int scale)
63         {
64             BigDecimal result, numer, term;
65             BigDecimal invX = BigDecimal.valueOf(inverseX);
66             BigDecimal invX2 =
67                 BigDecimal.valueOf(inverseX * inverseX);
68
69             numer = ONE.divide(invX, scale, roundingMode);
70
71             result = numer;
72             int i = 1;
73             do {
74                 numer =
75                     numer.divide(invX2, scale, roundingMode);
76                 int denom = 2 * i + 1;
77                 term =
78                     numer.divide(BigDecimal.valueOf(denom),
79                                  scale, roundingMode);
80                 if ((i % 2) != 0) {
81                     result = result.subtract(term);
82                 } else {
83                     result = result.add(term);
84                 }
85                 i++;
86             } while (term.compareTo(ZERO) != 0);
87             return result;
88         }
89
90
91             public static void main(String args[])
92             {
93                     // Create and install a security manager
94                     // System.setSecurityManager(new RMISecurityManager());
95
96                     try {
97                             PiServer obj = new PiServer();
98                             Naming.rebind("//yps:2042/PiServer", obj);
99                             System.out.println("PiServer bound in registry");
00                     } catch (Exception e) {
01                             System.out.println("PiServer err: " + e.getMessage
02                             e.printStackTrace();
03                     }
04             }
05     }
```

Source Code: Src/16_C/PiServer.java

12.48. Receiving and Sending Objects

Interface:

```
1       import java.util.*;
2
3       public interface TheInterface extends java.rmi.Remote {
4           Hashtable modifyAHashTableAndReturn(Hashtable aHashTable) throws java.
5           void modifyAHashTable(Hashtable aHashTable) throws java.rmi.RemoteExce
6       }
```

Source Code: Src/16_Hash/TheInterface.java

Implementation:

```
1       import java.util.*;
2       import java.rmi.*;
3       import java.rmi.server.UnicastRemoteObject;
4
5       public class Implementation
6               extends UnicastRemoteObject
7               implements TheInterface {
8
9               public Implementation() throws RemoteException {
10              }
11
12              public Hashtable modifyAHashTableAndReturn(Hashtable aHashTable) t
13                      aHashTable.put("two", "2");
14                      return aHashTable;
15              }
16              public void modifyAHashTable(Hashtable aHashTable) throws java.rmi
17                      aHashTable.put("three", "3");
18              }
19      }
```

Source Code: Src/16_Hash/Implementation.java

Client:

```
1       import java.rmi.*;
2       import java.util.*;
3
4       public class Client {
5         public static void main(String args[] ) {
6
7               Hashtable aHashTable = new Hashtable();
8               aHashTable.put("one", "1");
9
10              try {
11                      TheInterface aImplementation = new ProvideObject().provide
12
13                      aHashTable = aImplementation.modifyAHashTableAndReturn(aHa
14                              System.out.println("1. " + aHashTable);
```

```
15                      aImplementation.modifyAHashTable(aHashTable);
16                              System.out.println("2. " + aHashTable);
17
18                      aImplementation = new ProvideObject().provideTheObject(2);
19
20                      aHashTable = aImplementation.modifyAHashTableAndReturn(aHa
21                              System.out.println("3. " + aHashTable);
22                      aImplementation.modifyAHashTable(aHashTable);
23                              System.out.println("4. " + aHashTable);
24
25              } catch (Exception e) {
26                      System.out.println("HelloApplet exception: " +
27                      e.getMessage());
28                      e.printStackTrace();
29              }
30              System.out.println("bye. ");
31        }
32      }
```

Source Code: Src/16_Hash/Client.java

Server:

```
 1      import java.util.*;
 2      import java.rmi.*;
 3      import java.rmi.server.UnicastRemoteObject;
 4
 5      public class TheServer
 6              extends UnicastRemoteObject {
 7              private String name;
 8
 9              public TheServer() throws RemoteException {
10              }
11
12              public static void main(String args[])
13              {
14                      // System.setSecurityManager(new RMISecurityManager());
15
16                      try {
17                          Implementation obj = new Implementation();
18                          Naming.rebind("//localhost/HelloServer", obj);
19                          System.out.println(" object ... bound in registry");
20                      } catch (Exception e) {
21                          System.out.println("TheServer err: " + e.getMessage())
22                          e.printStackTrace();
23                      }
24              }
25      }
```

Source Code: Src/16_Hash/TheServer.java

Explain this result:

```
 ... # setup
% java Client
1. {three=3, one=1}
2. {three=3, one=1}
3. {three=3, one=1}
4. {three=3, one=1, two=2}
```

Object generation:

```
 1      import java.rmi.*;
 2      import java.util.*;
 3
 4      public class ProvideObject {
 5
 6        static TheInterface provideTheObject(int whichOne) throws Exception {
 7              TheInterface aHashtable;
 8              if ( whichOne == 1 )
 9                      aHashtable = (TheInterface)Naming.lookup("//localhost/Hell
10              else
11                      aHashtable = (TheInterface)new Implementation();
12              return aHashtable;
13        }
14      }
```

Source Code: Src/16_Hash/ProvideObject.java

12.49. RMI and Multi Threaded Systems

How does a remote method gets executed?

The Interface:

```
 1      public interface MultiTInterface extends java.rmi.Remote {
 2              String comeBackASAP() throws java.rmi.RemoteException;
 3              String sleepForAwhile() throws java.rmi.RemoteException;
 4      }
```

Source Code: Src/16_T/MultiTInterface.java

The Client:

```
 1      import java.rmi.*;
 2
 3      public class MultiTC extends Thread {
 4              static MultiTInterface obj;
 5              int id;
 6              public MultiTC()        {
 7              }
 8              public MultiTC(int id)  {
 9                      this.id = id;
10              }
11              public void run()       {
12                      String message = "";
```

```
13                      try {
14                              MultiTInterface obj = (MultiTInterface)Naming.look
15                              System.out.println(id +" Client Call sleepForAwhil
16                              message = obj.sleepForAwhile();
17                              System.out.println(message);
18                              System.out.println(id + " Client Call comeBackASAP
19                              message = obj.comeBackASAP();
20                              System.out.println(id + " " + message);
21                      } catch (Exception e) {
22                              e.printStackTrace();
23                      }
24              }
25
26              public static void main(String args[] ) {
27              try {
28                      new MultiTC(1).start();
29                      new MultiTC(111).start();
30                      new MultiTC(111111).start();
31              } catch (Exception e) {
32                      e.printStackTrace();
33              }
34        }
35      }
```

Source Code: Src/16_T/MultiTC.java

The Server:

```
 1      import java.rmi.*;
 2      import java.rmi.server.UnicastRemoteObject;
 3
 4      public class MultiTImpl
 5              extends UnicastRemoteObject
 6              implements MultiTInterface
 7      {
 8              private String name;
 9
10              public MultiTImpl(String s) throws RemoteException {
11                      name = s;
12              }
13
14              public String sleepForAwhile() throws RemoteException {
15                      System.out.println("MultiTImpl: going to sleep ...");
16                      try {
17                              Thread.sleep(2000);
18                      } catch ( Exception e )         {
19                              e.printStackTrace();
20                      }
21                      System.out.println("MultiTImpl: woke up ...");
22                      return  "sleepForAwhile";
23              }
24
25              public String comeBackASAP() throws RemoteException {
26                      return  "comeBackASAP";
```

```
27              }
28
29              public static void main(String args[])
30              {
31                      try {
32                              MultiTImpl obj = new MultiTImpl("MultiTServer");
33                              Naming.rebind("//localhost:2001/MultiTServer", obj
34                              System.out.println("MultiTServer bound in registry
35                      } catch (Exception e) {
36                              System.out.println("MultiTImpl err: " + e.getMessa
37                              e.printStackTrace();
38                      }
39              }
40      }
```

Source Code: Src/16_T/MultiTImpl.java

The makefile:

```
 1
 2
 3      all: MultiTInterface.class MultiTC.class MultiTImpl.class
 4              rmiregistry 2001 &
 5              sleep 1
 6              java MultiTImpl &
 7              sleep 4
 8              java MultiTC 1 &
 9              java MultiTC  &
10              killIt java
11              killIt
12
13      MultiTInterface.class:  MultiTInterface.java
14              javac MultiTInterface.java
15
16      MultiTC.class:  MultiTC.java
17              javac MultiTC.java
18
19      MultiTImpl.class:       MultiTImpl.java
20              javac MultiTImpl.java
21
22      clean:
23              rm -f *class
```

Source Code: Src/16_T/makefile

12.50. Dynamic Class Loading

RMI allows parameters, return values and exceptions passed in RMI calls to be any object that is serializable. RMI uses the object serialization mechanism to transmit data from one virtual machine to another and also annotates the call stream with the appropriate location information so that the class definition files can be loaded at the receiver.

When parameters and return values for a remote method invocation are unmarshaled to become live objects in the receiving VM, class definitions are required for all of the types of objects in the stream. The unmarshmaling process first attempts to resolve classes by name in its local

class loading context (the context class loader of the current thread). RMI also provides a facility for dynamically loading the class definitions for the actual types of objects passed as parameters and return values for remote method invocations from network locations specified by the transmitting endpoint. This includes the dynamic downloading of remote stub classes corresponding to particular remote object implementation classes (and used to contain remote references) as well as any other type that is passed by value in RMI calls, such as the subclass of a declared parameter type, that is not already available in the class loading context of the unmarshmaling side.

To support dynamic class loading, the RMI runtime uses special subclasses of java/io.ObjectOutputStream and java/io.ObjectInputStream for the marshal streams that it uses for marshaling and unmarshmaling RMI parameters and return values.

12.51. Java Object Serialization Security Issues

See here:

http://www.slideshare.net/frohoff1/appseccali-2015-marshalling-pickles
(http://www.slideshare.net/frohoff1/appseccali-2015-marshalling-pickles)

Problem:

- Classes/Objects are to remote process (JVM)
- Unmarshalled
- Developer:
  - — Trust communication channel
  - — Assume binary objects can not be changed
  - — Assume Serialization is safe
- Idea: how to allow "xxx" to login, instead of "hpb"

```
 1      import java.io.*;
 2      import java.util.Date;
 3
 4      public class ObjectWriter_5 {
 5        public static void main( String args[] ) {
 6          try (
 7              FileOutputStream ostream = new FileOutputStream("object_5.ser");
 8              ObjectOutputStream p = new ObjectOutputStream(ostream);
 9              ) {
10              p.writeObject("User: " + "hpb");
11          }
12          catch ( IOException e)      {
13              System.out.println(e.getMessage());
14          }
15        }
16      }
```

Source Code: Src/9_was/ObjectWriter_5.java

Od output:

```
% od -c object_5.data
0000000  254 355    005   t        U   s   e   r   :       h   p   b
0000020
% od -c x.data
0000000  254 355    005   t        U   s   e   r   :
0000015
% echo -n  xxx  >> x.data
% od -c x.data
0000000  254 355    005   t        U   s   e   r   :       x   x   x
0000020
```

• the same can be done with classes, and objects

Abritary Code Execution

• Code Reuse attack (return-oriented programming)

— control of the call stack

— Executed carefully chosen machine instructions (gadgets)

—

— http://heartbleed.com/ (http://heartbleed.com/)

— http://www.cs.rit.edu/~hpb/Lectures/20135/652/652-110.html (http://www.cs.rit.edu/~hpb/Lectures/20135/652/652-110.html)

```
if (1 + 2 + payload + 16 > s->s3->rrec.length) return 0; /* silently discard p
```

• https://dzone.com/articles/point-of-viewwhy-the-java-serialization-vulnerabil (https://dzone.com/articles/point-of-viewwhy-the-java-serialization-vulnerabil)

• defaultReadObject (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/io/ObjectInputStream.html)

Restrict Deserialization

• Default ObjectInputStream will deserialize any serializable class

• Class Black/White Listening

13. Collections

40 questions: http://www.journaldev.com/1330/java-collections-interview-questions-and-answers#map-vs-collection (http://www.journaldev.com/1330/java-collections-interview-questions-and-answers#map-vs-collection)

13.1. What is a Collection

Stolen from here. (http://java.sun.com/docs/books/tutorial/collections/intro/index.html)

• A collection (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/util/Collection.html) is simply an object that groups multiple elements into a single unit.

• Collections are used to store, retrieve and manipulate data, and to transmit data from one method to another.

• The primary use of the Collection interface is to pass around collections of objects where maximum generality is desired.

• Collections typically represent data items that form a natural group.

Sorting:

```
 1      import java.util.*;
 2
 3      public class Sort {
 4          public static void main(String args[]) {
 5              List l = Arrays.asList(args);
 6              Collections.sort(l);
 7              System.out.println(l);
 8          }
 9      }
10
```

Source Code: Src/9/Sort.java

Execution:

```
% java Sort X Mac OS
[Mac, OS, X]
```

• How does this work? In detail.... See here: http://www.cs.rit.edu/usr/local/jdk/docs/api/java/util/Collections.html (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/util/Collections.html)

13.2. How could we Implement the Previous Example?

• Assume the list is an array

• We handle only String objects

• How about this:

```
 1      public class BubbleSort {
 2
 3        public static void printIt(String aCollection[] )      {
 4                for (int index=0; index<aCollection.length; index++)
 5                      System.out.println(index + "\t" + aCollection[index] );
 6        }
 7
 8        public static void sort(String aCollection[] )      {
 9          for (int index=0; index < aCollection.length - 1; index++)      {
10            for (int walker=0; walker < aCollection.length - 1; walker++)  {
11              if ( aCollection[walker].compareTo(aCollection[walker+1]) > 0 )
12                              String tmp = aCollection[walker];
13                              aCollection[walker] = aCollection[walker + 1];
14                              aCollection[walker+1] = tmp;
15              }
16            }
17          }
18        }
```

```
19
20        public static void main( String args[] ) {
21          String[] aCollection = new String[3];
22          aCollection[0] =  "c";
23          aCollection[1] =  "b";
24          aCollection[2] =  "a";
25
26          sort(aCollection);
27          printIt(aCollection);
28        }
29      }
```

Source Code: Src/9/BubbleSort.java

• What do we need:

— a way to access every object in the array

— we use the String objects compareTo method

— Will this work for other kind of objects?

13.3. Implementation of Sort

```
1       import java.util.*;
2
3       public class HpCollections {
4
5         static Object anArray[] = null;
6
7
8         public static void sort(List aList) {
9               anArray = aList.toArray();
10
11              for (int index=0; index<anArray.length - 1; index++)     {
12                  for (int walker=0; walker<anArray.length - index - 1; walker++
13                      String left = (String) anArray[walker];
14                      String right = (String) anArray[walker+1];
15                      if ( left.compareTo( right ) > 0 )        {
16                              Object tmp = anArray[walker];
17                              anArray[walker] = anArray[walker + 1];
18                              anArray[walker+1] = tmp;
19                      }
20                  }
21              }
22              aList = Arrays.asList(anArray);
23
24        }
25        public String toString()      {
26              String s = new String ();
27              for (Object o: anArray )
28                      s = s + "/" + o ;
29              return s;
30         }
31
```

```
32        public static void main(String args[]) {
33              args = new String[4];
34              args[0] =  "z"; args[1] =  "x";
35              args[2] =  "a"; args[3] =  "t";
36              List l = Arrays.asList(args);
37              // HpCollections_remove.sort(l);
38              // HpOKCollections.sort(l);
39              // Collections.sort(l);
40              HpCollections.sort(l);
41              System.out.println(l);
42          }
43      }
44
```

Source Code: Src/Collection_5/HpCollections.java

Output:

```
% java HpOKCollections
[z, x, a, t]
```

Next:

```
 1      import java.util.*;
 2
 3      public class HpCollections_remove {
 4
 5        public static void sort(List aList) {
 6              Object anArray[] = aList.toArray();
 7
 8              for (int index=0; index<anArray.length - 1; index++)     {
 9                  for (int walker=0; walker<anArray.length - index - 1; walker++
10                      Comparable left = (Comparable) anArray[walker];
11                      Comparable right = (Comparable) anArray[walker+1];
12                      if ( left.compareTo( right ) > 0 )        {
13                              Object tmp = anArray[walker];
14                              anArray[walker] = anArray[walker + 1];
15                              anArray[walker+1] = tmp;
16                      }
17                  }
18              }
19
20              for (Object o: anArray )
21                      System.out.println("anArray: " + o );
22              for (int index=0; index<anArray.length ; index++)       {
23                      aList.remove(index );
24      /*
25      Exception in thread "main" java.lang.UnsupportedOperationException
26              at java.base/java.util.AbstractList.remove(AbstractList.java:167)
27              at HpCollections_remove.sort(HpCollections_remove.java:23)
28              at HpCollections.main(HpCollections.java:37)
29      */
30                      aList.add(anArray[index] );
31              }
32        }
```

```
33
34      }
35
```

Source Code: Src/Collection_5/HpCollections_remove.java

Next:

```
1       import java.util.*;
2
3       public class HpOKCollections {
4
5         public static void sort(List list) {
6               Object a[] = list.toArray();
7               Arrays.sort(a);                 // ...
8               ListIterator i = list.listIterator();
9               for (int j=0; j<a.length; j++) {
10                      i.next();               // this is it
11                      i.set(a[j]);            // modification of the list
12              }
13        }
14
15      }
16
```

Source Code: Src/Collection_5/HpOKCollections.java

Next:

```
1       import java.util.*;
2
3       public class Sort {
4           public static void main(String args[]) {
5               args = new String[4];
6               args[0] =  "z"; args[1] =  "x";
7               args[2] =  "a"; args[3] =  "a";
8               List l = Arrays.asList(args);
9               //  Collections.sort(l);
10              HpOKCollections.sort(l);
11              System.out.println(l);
12          }
13      }
14
```

Source Code: Src/Collection_5/Sort.java

Why does it not work?

Strings (http://docs.oracle.com/javase/7/docs/api/java/lang/String.html)
- take a look at this class

13.4. What Is a Collections Framework?

A collections framework is a unified architecture for representing and manipulating collections.
All collections frameworks contain three things:

• Interfaces: abstract data types representing collections. Interfaces allow collections to be manipulated independently of the details of their representation.

• Implementations: concrete implementations of the collection interfaces. In essence, these are reusable data structures.

• Algorithms: methods that perform useful computations, like searching and sorting, on objects that implement collection interfaces. These algorithms are said to be polymorphic because the same method can be used on many different implementations of the appropriate collections interface. In essence, algorithms are reusable functionality.

13.5. Iterators

• The object returned by the iterator (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/util/Iterator.html)
method is very similar to an Enumeration, but differs in two respects:

— Iterator allows the caller to remove elements from the underlying collection during the iteration with well-defined semantics.

— Method names have been improved.

Use of an iterator:

```
 1
 2      import java.util.*;
 3
 4      public class UseIterator {
 5          public static void main(String args[]) {
 6              int index = 0;
 7              List l = Arrays.asList(args);
 8              Iterator aIterator = l.iterator();
 9              while ( aIterator.hasNext() )   {
10                      System.out.println(++index + ": " +
11                              (String)aIterator.next() );
12              }
13          }
14      }
```

Source Code: Src/9/UseIterator.java

```
% java UseIterator A Day at the Races
1: A
2: Day
3: at
4: the
5: Races
```

13.6. Benefits of a Collections Framework

• It reduces programming effort

• It increases program speed and quality

• It allows interoperability among unrelated APIs

• It reduces effort to design new APIs

• It fosters software reuse

13.7. Collection Interface

Overview (https://docs.oracle.com/javase/8/docs/technotes/guides/collections/overview.html)

• The core collection (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/util/Collection.html) interfaces is:

copied from http://java.sun.com/docs/books/tutorial/collections/interfaces/index.html (http://java.sun.com/docs/books/tutorial/collections/interfaces/index.html)

• The Collection (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/util/Collections.html) interface is the root of the collection hierarchy. A Collection represents a group of objects, known as its elements. Some Collection implementations

— allow duplicate elements and others do not.

— some are ordered and others unordered.

Collection is used to pass collections around and manipulate them when maximum generality is desired.

• Interfaces:

Collection

- Enumeration
- List
- Map
- Queue
- RandomAccess
- Set
- SortedMap
- SortedSet

13.8. Set Interface

A Set (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/util/Set.html) is a collection that cannot contain duplicate elements.

• Two Set objects are equal if they contain the same elements.

• See also HashSet (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/util/HashSet.html) and TreeSet (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/util/TreeSet.html)

• A List (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/util/List.html) is an ordered collection

13.9. Lists

• Lists can contain duplicate elements.

13.10. Maps

• A Map (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/util/Map.html) is an object that maps keys to values.

• Maps cannot contain duplicate keys: Each key maps to one value.

13.11. Maps vs Collections

• Collections: add, remove, lookup

• Maps: key value pair, access values stored by key

13.12. A Picture

13.13. See Here

Tutorial. (http://docs.oracle.com/javase/tutorial/collections/index.html)

13.14. General Purpose Implementations

	Implementation			
	Hash Table	Resizable Array	Balanced Tree	Linked List
Interface Set	HashSet		TreeSet	
Interface List		ArrayList		LinkedList
Interface Map	HashMap		TreeMap	

• The fact that the new implementations are unsynchronized represents a break with the past

• If you need a synchronized collection, the synchronization wrappers, allow any collection to be transformed into a synchronized collection. Thus, synchronization is optional for the new collection implementations where it was mandatory for the old.

• As a rule of thumb, you should be thinking about the interfaces rather than the implementations.

13.15. Implementations: Set

• HashSet and

• TreeSet.

• HashSet is much faster (constant time vs. log time for most operations), but offers no ordering guarantees.

• If it is needed to use the operations in the SortedSet, or in-order iteration is important to use TreeSet.

13.16. Implementations: List

• ArrayList and

• LinkedList.

• ArrayList offers constant time positional access

13.17. Implementations: Map

• HashMap and

• TreeMap

• The situation for Map is exactly analogous to Set.

13.18. Algorithms

• prints out its arguments in lexicographic order :

```
 1      import java.util.*;
 2
 3      public class Sort {
 4          public static void main(String args[]) {
 5              List l = Arrays.asList(args);
 6              Collections.sort(l);
 7              System.out.println(l);
 8          }
 9      }
10
```

Source Code: Src/9/Sort.java

13.19. Examples: HashSet

```
 1
 2      import java.util.HashSet;
 3      import java.util.Set;
 4
 5      public class HashSetEx_1 {
 6
 7          private Set<Integer> universe;
 8
 9          private Set<Integer> fill(int soMany) {
10              Set<Integer> universe = new HashSet<Integer>();
11              for ( int index = 0; index < soMany; index ++ )
12                      universe.add(new Integer(9999999 * index));
13              return universe;
14          }
15          public static void main(String args[])      {
16              Set<Integer> universe = null;
17              HashSetEx_1 aHashSetEx_1 = new HashSetEx_1();
18              universe = aHashSetEx_1.fill(253);
19              System.out.println("1: " + universe );
20
21              // universe.remove( new Integer(1) );
22              // System.out.println("2: " + universe );
23
24              universe.remove( new Integer(10) );
25              // System.out.println("3: " + universe );
26          }
27      }
28             /*
29              for(Integer id : stars.keySet()) {
30                  ids.add(id);
31
32                  if(ids.size() >= keepStars)
33                      break;
34              }
35
36              return ids;
37              */
```

Source Code: Src/9/HashSetEx_1.java

13.20. Examples: HashMap I

```
 1
 2      import java.util.HashMap;
 3      import java.util.Map;
 4      import java.util.Set;
 5      import java.util.Iterator;
```

```
 6
 7      public class HashMapEx {
 8
 9          private Map<Integer, String> universe;
10
11          private Map<Integer, String> fill(int soMany) {
12              universe = new HashMap<Integer, String>();
13              for ( int index = 0; index < soMany; index ++ )
14                      universe.put(new Integer(index), "_" + index);
15              return universe;
16          }
17
18          private Map<Integer, String> delete(int what) {
19              try {
20                      for (Integer id : universe.keySet() ) {
21                              System.out.println("try to delete: " + id);
22                              if ( id.equals(new Integer(what) ) )
23                                      universe.remove(id);
24                              System.out.println("deleted: " + id);
25                      }
26              } catch ( Exception e ) {
27                      System.out.println("Exception ..... ");
28                      e.printStackTrace();
29              }
30              return universe;
31
32          }
33      /*
34      The iterators returned by all of this class's "collection view methods" ar
ConcurrentModificationException. Thus, in the face of concurrent modification, the
35
36      Note that the fail-fast behavior of an iterator cannot be guaranteed as it
on on a best-effort basis. Therefore, it would be wrong to write a program that de
37
38      */
39          private Map<Integer, String> deleteUsingKeySetCorrect(int what) {
40              try {
41                      Iterator aIterator = universe.keySet().iterator();
42                      while ( aIterator.hasNext() )           {
43                              aIterator.next();
44                              aIterator.remove();
45                      }
46              } catch ( Exception e ) {
47                      System.out.println("Exception ");
48                      e.printStackTrace();
49              }
50              return universe;
51
52          }
53
54          public static void main(String args[])      {
55              Map<Integer, String> universe;
56              HashMapEx aHashMapEx = new HashMapEx();
57              universe = aHashMapEx.fill(3);
```

```
58
59              System.out.println("1: " + universe );
60              aHashMapEx.deleteUsingKeySetCorrect(1);
61
62
63              universe = aHashMapEx.fill(3);
64              aHashMapEx.delete(1);
65              System.out.println("2: " + universe );
66
67          }
68      }
```

Source Code: Src/9/HashMapEx.java

Execution

```
1: {0=_0, 1=_1, 2=_2}
0
1
Exception .....
java.util.ConcurrentModificationException
       at java.util.HashMap$HashIterator.nextEntry(HashMap.java:793)
       at java.util.HashMap$KeyIterator.next(HashMap.java:828)
       at HashMapEx.delete(HashMapEx.java:18)
       at HashMapEx.main(HashMapEx.java:37)
2: {0=_0, 2=_2}
```

From: http://www.cs.rit.edu/usr/local/jdk/docs/api/java/util/HashMap.html (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/util/HashMap.html)
Note that this implementation is not synchronized. If multiple threads access this map concurrently, and at least one of the threads modifies the map structurally, it must be synchronized externally. (A structural modification is any operation that adds or deletes one or more mappings; merely changing the value associated with a key that an instance already contains is not a structural modification.) This is typically accomplished by synchronizing on some object that naturally encapsulates the map. If no such object exists, the map should be "wrapped" using the Collections.synchronizedMap method. This is best done at creation time, to prevent accidental unsynchronized access to the map:

```
Map m = Collections.synchronizedMap(new HashMap(...));
```

The iterators returned by all of this class's "collection view methods" are fail-fast: if the map is structurally modified at any time after the iterator is created, in any way except through the iterator's own remove or add methods, the iterator will throw a ConcurrentModificationException. Thus, in the face of concurrent modification, the iterator fails quickly and cleanly, rather than risking arbitrary, non-deterministic behavior at an undetermined time in the future.

Note that the fail-fast behavior of an iterator cannot be guaranteed as it is, generally speaking, impossible to make any hard guarantees in the presence of unsynchronized concurrent modification. Fail-fast iterators throw ConcurrentModificationException on a best-effort basis. Therefore, it would be wrong to write a program that depended on this exception for its correctness: the fail-fast behavior of iterators should be used only to detect bugs.

13.21. List Iterator

• An iterator for lists allows the programmer to traverse the list

— in either direction

— modify the list during iteration

— obtain the iterator's current position in the list.

• A ListIterator has no current element

• its cursor position always lies between the element that would be returned by a call to previous() and the element that would be returned by a call to next()

• In a list of length n, there are n+1 valid index values, from 0 to n, inclusive.

```
 1
 2      import java.util.Stack;
 3      import java.util.ListIterator;
 4      import java.util.Collection;
 5
 6      public class ListItereatorEx {
 7          // private Collection<String> palindrom;
 8          private Stack<String> palindrom;
 9
10          private Collection<String> fill(String words[]) {
11              palindrom  = new Stack<String>();
12              for (String id : words ) {
13                      palindrom.push(id);
14              }
15              return palindrom;
16          }
17
18          private Collection<String> leftRight()      {
19              ListIterator<String> aListIterator = palindrom.listIterator(2);
20              String s  = aListIterator.next();
21              System.out.println("s = " + s );
22              aListIterator.set("ZZ top");
23              return palindrom;
24          }
25
26          public static void main(String args[])      {
27              Collection<String> aStack;
28              String theOnes[] = { "a", "b", "c", "d" };
29              ListItereatorEx o = new ListItereatorEx();
30
31              aStack = o.fill(theOnes );
32              System.out.println("1: " + aStack );
33
34              aStack = o.leftRight();
35              System.out.println("2: " + aStack );
36
37          }
38      }
```

Source Code: Src/9/ListItereatorEx.java

Why is this important?

13.22. Collections.sort()

How does Collections.sort() work?

```
 1      import java.util.*;
 2
 3      public class Sort {
 4          public static void main(String args[]) {
 5              List l = Arrays.asList(args);
 6              Collections.sort(l);
 7              System.out.println(l);
 8          }
 9      }
10
```

Source Code: Src/9/Sort.java

From java doc:

```
sort

public static void sort(List list)
Sorts the specified list into ascending order, according to the natural ordering o
This sort is guaranteed to be stable: equal elements will not be reordered as a re

The specified list must be modifiable, but need not be resizable.

The sorting algorithm is a modified mergesort (in which the merge is omitted if th

Parameters:
list - the list to be sorted.
Throws:
ClassCastException - if the list contains elements that are not mutually comparabl
UnsupportedOperationException - if the specified list's list-iterator does not sup
See Also:
Comparable
```

13.23. Object Ordering

• A List l may be sorted as follows:

```
Collections.sort(l);
```

Class	Natural Ordering
Byte	signed numerical
Character	unsigned numerical
Long	signed numerical
Integer	signed numerical
Short	signed numerical
Double	signed numerical
Float	signed numerical
BigInteger	signed numerical
BigDecimal	signed numerical
File	system-dependent lexicographic on pathname.

String	lexicographic
Date	chronological
CollationKey	locale-specific lexicographic

• The Comparable interface consists of a single method:

```
public interface Comparable {
      public int compareTo(Object o);
}
```

Example:

```
 1      import java.util.*;
 2
 3      public class Name implements Comparable {
 4          protected String  firstName, lastName;
 5
 6          public Name(String firstName, String lastName) {
 7              if (firstName==null || lastName==null)
 8                  throw new NullPointerException();
 9              this.firstName = firstName;
10              this.lastName = lastName;
11          }
12
13          public String firstName()    {
14              return firstName;
15          }
16          public String lastName()     {
17              return lastName;
18          }
19
20          public boolean equals(Object o) {
21              if (!(o instanceof Name))
22                  return false;
23              Name n = (Name)o;
24              return n.firstName.equals(firstName) &&
25                     n.lastName.equals(lastName);
26          }
27
28          public String toString() {
29              return firstName + " " + lastName;
30          }
31
32          public int compareTo(Object o) {
33              Name n = (Name)o;
34              int lastCmp = lastName.compareTo(n.lastName);
35              return (lastCmp!=0 ? lastCmp :
36                      firstName.compareTo(n.firstName));
37          }
38
39
40          public static void main(String args[]) {
41              Name n[] = {
```

```
42                  new Name("Bond",    "James"),
43                  new Name("Jack",    "Blues"),
44                  new Name("Elwood",  "Blues"),
45                  new Name("You",     "Me")
46              };
47              List l = Arrays.asList(n);
48              Collections.sort(l);
49              System.out.println(l);
50          }
51      }
52
```

Source Code: Src/9/Name.java

See also here. (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/lang/Object.html)

13.24. Filling a HashTable and using a reasonable hashfunction

• Object.hashCode contract: s1.equals(s2) implies that s1.hashCode()==s2.hashCode() for any two sets s1 and s2,

```
public int hashCode() {
    return 31*super.firstName.hashCode() + super.lastName.hashCode();
}
```

Example:

```
1       import java.util.*;
2
3       public class Hash_1 extends Name_1 {
4           static final int MAX = 20000;
5           static HashMap aHashMap = new HashMap();
6
7           public Hash_1(String firstName, String lastName) {
8               super(firstName, lastName);
9           }
10
11
12          public static void init()   {
13              long milliSeconds = System.currentTimeMillis();
14              for ( int index = 0; index <= MAX; index ++ ) {
15                      if ( index % 1000  == 0 )
16                              System.out.println(index + "/" + MAX );
17                      aHashMap.put( new Hash_1( "A" + index, "A" + index),
18                            new Hash_1( "A" + index, "A" + index)
19                           );
20              }
21              System.out.println("Time for filling: " +
22                      ( System.currentTimeMillis() - milliSeconds) );
23          }
24
25          public static void findIt(Hash_1 aHash_1)   {
26              long milliSeconds = System.currentTimeMillis();
```

```
27              if ( aHashMap.containsKey( aHash_1 ) )
28                      System.out.print("\taHashMap: containsKey takes: ");
29              System.out.println(System.currentTimeMillis() - milliSeconds);
30
31           }
32
33           public static void findMax()        {
34              Hash_1 aHash_1 = new Hash_1( "A" + MAX, "A" + MAX);
35              System.out.println("Find Max = " + aHash_1);
36              findIt(aHash_1);
37          }
38
39           public static void findMiddle()     {
40              Hash_1 aHash_1 = new Hash_1( "A" + ( MAX/2), "A" + ( MAX/2));
41              System.out.println("Find Middle = " + aHash_1);
42              findIt(aHash_1);
43          }
44
45           public static void findMin()        {
46              Hash_1 aHash_1 = new Hash_1( "A" + 0, "A" + 0);
47              System.out.println("Find Min = " + aHash_1);
48              findIt(aHash_1);
49          }
50
51
52          public static void main(String args[] )      {
53              long milliSeconds = System.currentTimeMillis();
54
55              init();
56              findMax();
57              findMiddle();
58              findMin();
59              System.exit(0);
60          }
61      }
62
```

Source Code: Src/9/Hash_1.java

Output:

```
Time for filling: 1638
Find Max = A20000 A20000
        aHashMap: containsKey takes: 0
Find Middle = A10000 A10000
        aHashMap: containsKey takes: 0
Find Min = A0 A0
        aHashMap: containsKey takes: 0
```

13.25. Filling a HashTable and not using a reasonable hashfunction

```
public int hashCode() {
    return 1;
}
```

Example

```
 1      import java.util.*;
 2
 3      public class Hash_2 extends Name_2 {
 4          static final int MAX = 20000;
 5          static HashMap aHashMap = new HashMap();
 6
 7          public Hash_2(String firstName, String lastName) {
 8              super(firstName, lastName);
 9          }
10
11
12          public static void init()   {
13              long milliSeconds = System.currentTimeMillis();
14              for ( int index = 0; index <= MAX; index ++ ) {
15                      if ( index % 3000  == 0 )
16                              System.out.println(new Date() +
17                              ": "  + index + "/" + MAX );
18                      aHashMap.put( new Hash_2( "A" + index, "A" + index),
19                            new Hash_2( "A" + index, "A" + index)
20                           );
21              }
22              System.out.println("Time for filling: " +
23                      ( System.currentTimeMillis() - milliSeconds) );
24          }
25
26          public static void findIt(Hash_2 aHash_2)   {
27              long milliSeconds = System.currentTimeMillis();
28              if ( aHashMap.containsKey( aHash_2 ) )
29                      System.out.print("\taHashMap: containsKey takes: ");
30              System.out.println(System.currentTimeMillis() - milliSeconds);
31
32          }
33
34          public static void findMax()        {
35              Hash_2 aHash_2 = new Hash_2( "A" + MAX, "A" + MAX);
36              System.out.println("Find Max = " + aHash_2);
37              findIt(aHash_2);
38         }
39
40          public static void findMiddle()     {
41              Hash_2 aHash_2 = new Hash_2( "A" + ( MAX/2), "A" + ( MAX/2));
42              System.out.println("Find Middle = " + aHash_2);
43              findIt(aHash_2);
44         }
45
46          public static void findMin()        {
47              Hash_2 aHash_2 = new Hash_2( "A" + 0, "A" + 0);
48              System.out.println("Find Min = " + aHash_2);
```

```
49              findIt(aHash_2);
50          }
51
52
53          public static void main(String args[] )      {
54              long milliSeconds = System.currentTimeMillis();
55
56              init();
57              findMax();
58              findMiddle();
59              findMin();
60              System.exit(0);
61          }
62      }
63
```

Source Code: Src/9/Hash_2.java

Output:

```
% java Hash_2
Wed Sep 18 12:25:54 EDT 2002: 0/20000
Wed Sep 18 12:25:58 EDT 2002: 3000/20000
Wed Sep 18 12:26:08 EDT 2002: 6000/20000
Wed Sep 18 12:26:31 EDT 2002: 9000/20000
Wed Sep 18 12:27:11 EDT 2002: 12000/20000
Wed Sep 18 12:27:58 EDT 2002: 15000/20000
Wed Sep 18 12:29:01 EDT 2002: 18000/20000
Time for filling: 252173
Find Max = A20000 A20000
        aHashMap: containsKey takes: 1
Find Middle = A10000 A10000
        aHashMap: containsKey takes: 85
Find Min = A0 A0
        aHashMap: containsKey takes: 16
```

13.26. Routine Data Manipulation

Routine Data Manipulation (http://docs.oracle.com/javase/7/docs/api/java/util/Collections.html)

The Collections class provides three algorithms for doing routine data manipulation on List objects.

• reverse: Reverses the order of the elements in a List.

• fill: Overwrites every element in a List with the specified value.

• copy: Takes two arguments, a destination List and a source List, and copies the elements of the source into the destination.

• See java doc ...

13.27. Sorting Maps

• A Map is not sorted

• Collections.sort():

```
public static void sort(List list)
```

• Convert Maps to Lists.

```
 1      import java.util.*;
 2
 3      public class UseCollectionS     {
 4          static ArrayList aArrayList = new ArrayList();
 5          static HashMap aHashMap = new HashMap();
 6
 7         public static void main(String args[] )      {
 8
 9              for ( int index = 0; index < args.length; ++index)
10                      aHashMap.put(args[index], args[index] + " " + new Date() )
11
12              System.out.println("The HashMap: " + aHashMap);
13
14              List l = new ArrayList(aHashMap.values());
15              Collections.sort(l);
16              System.out.println("The List: " + l);
17
18         }
19      }
20
```

Source Code: Src/9/UseCollectionS.java

Output:

```
javac UseCollectionS.java && java UseCollectionS a b c d
The HashMap: {a=a Wed Nov 06 10:23:47 EST 2019, b=b Wed Nov 06 10:23:47 EST 2019,
The List: [a Wed Nov 06 10:23:47 EST 2019, b Wed Nov 06 10:23:47 EST 2019, c Wed N
```

13.28. Shuffling

The shuffle algorithm does the opposite of what sort does:

• it destroys any trace of order that may have been present in a List.

• It's useful in implementing games of chance.

• It's useful for generating test cases.

13.29. Searching

The binarySearch algorithm searches for a specified element in a sorted List using the binary search algorithm.

```
 1      import java.util.*;
 2
 3      public class Name implements Comparable {
 4          protected String  firstName, lastName;
 5
 6          public Name(String firstName, String lastName) {
```

```
 7              if (firstName==null || lastName==null)
 8                  throw new NullPointerException();
 9              this.firstName = firstName;
10              this.lastName = lastName;
11          }
12
13          public String firstName()    {
14              return firstName;
15          }
16          public String lastName()     {
17              return lastName;
18          }
19
20          public boolean equals(Object o) {
21              if (!(o instanceof Name))
22                  return false;
23              Name n = (Name)o;
24              return n.firstName.equals(firstName) &&
25                     n.lastName.equals(lastName);
26          }
27
28          public String toString() {
29              return firstName + " " + lastName;
30          }
31
32          public int compareTo(Object o) {
33              Name n = (Name)o;
34              int lastCmp = lastName.compareTo(n.lastName);
35              return (lastCmp!=0 ? lastCmp :
36                      firstName.compareTo(n.firstName));
37          }
38
39
40          public static void main(String args[]) {
41              Name n[] = {
42                  new Name("Bond",    "James"),
43                  new Name("Jack",    "Blues"),
44                  new Name("Elwood",  "Blues"),
45                  new Name("You",     "Me")
46              };
47              List l = Arrays.asList(n);
48              Collections.sort(l);
49              System.out.println(l);
50          }
51      }
52
```

Source Code: Src/9/Name.java

Next:

```
 1      import java.util.*;
 2
 3      public class Name_1 extends Name {
 4          static final int MAX = 5;
 5
 6          public Name_1(String firstName, String lastName) {
 7              super(firstName, lastName);
 8          }
 9
10          public int hashCode() {
11              return 31*super.firstName.hashCode() + super.lastName.hashCode();
12          }
13
14
15         public static void main(String args[] )      {
16              HashMap aHashMap = new HashMap();
17              long milliSeconds = System.currentTimeMillis();
18
19
20              milliSeconds = System.currentTimeMillis();
21              for ( int i = 1; i < MAX; i ++ )                {
22                      System.out.println("1: " + i );
23                      for ( int index = 0; index < 10000; index ++ )
24                              aHashMap.put( new Name_1( "A" + index, "A" + index
25                                    new Name_1( "B" + index, "A" + index)
26                                   );
27              }
28              System.out.println("Time Name_1: " + ( System.currentTimeMillis()
29                                  milliSeconds )  );
30
31         }
32
33      }
34
```

Source Code: Src/9/Name_1.java

Next:

```
 1      import java.util.*;
 2
 3      public class SortTest extends Name_1 {
 4          static final int MAX = 20000;
 5          static HashMap aHashMap = new HashMap();
 6          static TreeMap aTreeMap = new TreeMap();
 7          static ArrayList aArrayList = new ArrayList();
 8
 9          public SortTest(String firstName, String lastName) {
10              super(firstName, lastName);
11          }
12
13
14          public static void init()   {
15              long milliSeconds = System.currentTimeMillis();
```

```
16              for ( int index = 0; index <= MAX; index ++ ) {
17                      if ( index % 1000  == 0 )
18                              System.out.println(index + "/" + MAX );
19                      aTreeMap.put( new SortTest( "A" + index, "A" + index),
20                            new SortTest( "A" + index, "A" + index)
21                           );
22                      aHashMap.put( new SortTest( "A" + index, "A" + index),
23                            new SortTest( "A" + index, "A" + index)
24                           );
25                      aArrayList.add( new SortTest( "A" + index, "A" + index));
26              }
27              System.out.println("Time for filling: " +
28                      ( System.currentTimeMillis() - milliSeconds) );
29          }
30
31          public static void findIt(SortTest aSortTest)       {
32              long milliSeconds = System.currentTimeMillis();
33              if ( aArrayList.contains( aSortTest ) )
34                      System.out.print("\taArrayList: contains takes: ");
35              System.out.println( System.currentTimeMillis() - milliSeconds);
36
37              milliSeconds = System.currentTimeMillis();
38              if ( aArrayList.indexOf( aSortTest ) >= 0 )
39                      System.out.print("\taArrayList: indexOf takes: ");
40              System.out.println(System.currentTimeMillis() - milliSeconds);
41
42              milliSeconds = System.currentTimeMillis();
43              if ( aHashMap.containsKey( aSortTest ) )
44                      System.out.print("\taHashMap: containsKey takes: ");
45              System.out.println(System.currentTimeMillis() - milliSeconds);
46
47              milliSeconds = System.currentTimeMillis();
48              if ( aTreeMap.containsKey( aSortTest ) )
49                      System.out.print("\taTreeMap: containsKey takes: ");
50              System.out.println(System.currentTimeMillis() - milliSeconds);
51
52          }
53
54          public static void findMax()        {
55              SortTest aSortTest = new SortTest( "A" + MAX, "A" + MAX);
56              System.out.println("Find Max = " + aSortTest);
57              findIt(aSortTest);
58         }
59
60          public static void findMiddle()     {
61              SortTest aSortTest = new SortTest( "A" + ( MAX/2), "A" + ( MAX/2))
62              System.out.println("Find Middle = " + aSortTest);
63              findIt(aSortTest);
64         }
65
66          public static void findMin()        {
67              SortTest aSortTest = new SortTest( "A" + 0, "A" + 0);
68              System.out.println("Find Min = " + aSortTest);
69              findIt(aSortTest);
```

```
70          }
71
72
73          public static void main(String args[] )      {
74               long milliSeconds = System.currentTimeMillis();
75
76               init();
77               findMax();
78               findMiddle();
79               findMin();
80               System.exit(0);
81          }
82
83       }
84
```

Source Code: Src/9/SortTest.java

Output:

```
0/20000
1000/20000
2000/20000
3000/20000
4000/20000
5000/20000
6000/20000
7000/20000
8000/20000
9000/20000
10000/20000
11000/20000
12000/20000
13000/20000
14000/20000
15000/20000
16000/20000
17000/20000
18000/20000
19000/20000
20000/20000
Time for filling: 128
Find Max = A20000 A20000
       aArrayList: contains takes: 5
       aArrayList: indexOf takes: 1
       aHashMap: containsKey takes: 0
       aTreeMap: containsKey takes: 0
Find Middle = A10000 A10000
       aArrayList: contains takes: 1
       aArrayList: indexOf takes: 0
       aHashMap: containsKey takes: 0
       aTreeMap: containsKey takes: 0
Find Min = A0 A0
       aArrayList: contains takes: 0
```

```
aArrayList: indexOf takes: 0
aHashMap: containsKey takes: 0
aTreeMap: containsKey takes: 0
```

13.30. Finding Extreme Values

The min and max algorithms return, respectively, the minimum and maximum element contained in a specified Collection.

13.31. Comparable

How about comparing based on different criterias?

• See also
http://www.cs.rit.edu/usr/local/jdk/docs/api/java/lang/Comparable.html
(http://www.cs.rit.edu/usr/local/jdk/docs/api/java/lang/Comparable.html)

• The Comparable interface consists of a single method:

```
public interface Comparable {
      public int compareTo(Object o);
}
```

•

a < b → a.compareTo(b) < 0
a == b → a.compareTo(b) == 0
a > b → a.compareTo(b) > 0

• Lists and arrays of objects that implement this interface can be sorted automatically by

Collections.sort()
Arrays.sort() (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/util/Arrays.html)

• A class's natural ordering is said to be consistent with equals if and only if

(e1.compareTo((Object)e2)==0)

has the same boolean value as

e1.equals((Object)e2)

for every e1 and e2 of class C.

• It is strongly recommended, but not strictly required that

(x.compareTo(y)==0) == (x.equals(y))

• o = (name, age)

— equals: name + age

— compareTo: name

•

compareTo (http://www.cs.rit.edu/usr/local/jdk/docs/api/java/lang/Comparable.html#compareTo(java.lang.Object))
must ensure that

— sgn(x.compareTo(y)) == -sgn(y.compareTo(x)) for all x and y.

— (x.compareTo(y)>0 && y.compareTo(z)>0) implies x.compareTo(z)>0.

— x.compareTo(y)==0 implies that sgn(x.compareTo(z)) == sgn(y.compareTo(z)), for all z.

13.32. Example I:

• Let's take a look at this example:

```
 1      /*
 2       * "Note: this class has a natural ordering
 3       *        that is inconsistent with equals.
 4       */
 5
 6      import java.util.*;
 7
 8      public class ComparableEx implements Comparable {
 9          protected String  firstName;
10          protected String  lastName;
11
12          public ComparableEx(String firstName, String lastName) {
13              this.firstName = firstName;
14              this.lastName  = lastName;
15          }
16
17          public boolean equals(Object o) {
18              if (!(o instanceof ComparableEx))
19                  return false;
20              ComparableEx n = (ComparableEx)o;
21              return firstName.equals(n.firstName)    &&
22                     lastName.equals(n.lastName);
23          }
24
25          public int compareTo(Object o) {
26              ComparableEx n = (ComparableEx)o;       // cast execption
27              return lastName.compareTo(lastName);
28          }
29          public String toString()    {
30                      return firstName + "/" + lastName;
31          }
32
33          public static void main(String args[]) {
34              ComparableEx n[] = {
35                  new ComparableEx("James",   "Bond"),
36                  new ComparableEx("James",   "Bond"),
37                  new ComparableEx("Jack",    "Blues"),
38                  new ComparableEx("Elwood",  "Blues")
39              };
40              List l = Arrays.asList(n);
41              Collections.sort(l);
42              System.out.println(l);
43          }
44      }
45
```

Source Code: Src/9/ComparableEx.java

Es wird ein Teil der Information des kompletten Objects verwendet.

13.33. Warning

What is the problem here?

Nur ein Nachname wird eingefuegt... doppelte Nachnamen werde ignoriert...

```
 1
 2      import java.util.*;
 3
 4      public class ComparatorExTree {
 5
 6          protected String  firstName;
 7          protected String  lastName;
 8
 9          static final Comparator nameC = new Comparator() {
10                  public int compare(Object o1, Object o2) {
11                      ComparatorExTree n1 = (ComparatorExTree)o1;     // cas
12                      ComparatorExTree n2 = (ComparatorExTree)o2;     // cas
13                      return n1.lastName.compareTo(n2.lastName);
14                  }
15          };
16
17          public ComparatorExTree(String firstName, String lastName) {
18              this.firstName = firstName;
19              this.lastName  = lastName;
20          }
21
22          public boolean equals(Object o) {
23              if (!(o instanceof ComparatorExTree))
24                  return false;
25              ComparatorExTree n = (ComparatorExTree)o;
26              return firstName.equals(n.firstName)    &&
27                     lastName.equals(n.lastName);
28          }
29
30         public String toString() {
31              return firstName + "; " + lastName;
32         }
33
34         public int compareTo(Object o) {
35              ComparableEx n = (ComparableEx)o;       // cast execption
36              if ( firstName.compareTo(firstName) == 0 )
37                      return lastName.compareTo(lastName);
38              else
39                      return  0;
40          }
41
42          public static void main(String args[]) {
43              ComparatorExTree n[] = {
44                  new ComparatorExTree("You",         "Name"),
45                  new ComparatorExTree("Roger",       "Bond"),
46                  new ComparatorExTree("James",       "Bond"),
47                  new ComparatorExTree("Jack",        "Blues"),
48                  new ComparatorExTree("Elwood",      "Blues")
49              };
```

```
50              TreeSet l = new TreeSet(nameC);
51
52              for ( int i = 0; i < n.length; i ++ )   {
53                      System.out.println(i + " " + n[i]);
54                      l.add(n[i]);
55              }
56              System.out.println("the TreeSet: " + l);
57          }
58      }
59
```

Source Code: Src/9/ComparatorExTree.java

Output:

```
java ComparatorExTree
0 You; Name
1 Roger; Bond
2 James; Bond
3 Jack; Blues
4 Elwood; Blues
the TreeSet: [Jack; Blues, Roger; Bond, You; Name]
```

13.34. Comparator

• Interface (http://docs.oracle.com/javase/7/docs/api/java/util/Comparator.html)

(From java doc)

The relation that defines the imposed ordering that a given comparator c imposes on a given set of objects S is:

For all x, y ∈ S:
{(x, y) such that c.compare(x, y) <= 0}.

The quotient for this total order is:
{(x, y) such that c.compare(x, y) == 0}.

13.35. Example

```
1
2       import java.util.*;
3
4       class HpComparator implements Comparator {
5               public int compare(Object o1, Object o2)        {
6                       String s1 = (String)o1;
7                       String s2 = (String)o2;
8                       return s1.compareTo(s2) ;
9               }
10              public boolean equals(Object o) {
11                      return true;
12              }
13      }
14      public class HpC_C_ex   {
15        public static void sort(List aList) {
```

```
16              Object anArray[] = aList.toArray();
17
18              for (int index=0; index<anArray.length - 1; index++)     {
19                  for (int walker=0; walker<anArray.length - index - 1; walk
20                      Comparable left = (Comparable) anArray[walker];
21                      Comparable right = (Comparable) anArray[walker+1];
22                      if ( left.compareTo( right ) > 0 )        {
23                              Object tmp = anArray[walker];
24                              anArray[walker] = anArray[walker + 1];
25                              anArray[walker+1] = tmp;
26                      }
27                  }
28              }
29              ListIterator anIterator  = aList.listIterator();
30              for (int j=0; j<anArray.length; j++) {
31                      anIterator.next();
32                      anIterator.set(anArray[j]);
33              }
34
35      }
36      public static void sort(List aList, Comparator aComparator) {
37              Object anArray[] = aList.toArray();
38
39              for (int index=0; index<anArray.length - 1; index++)     {
40                  for (int walker=0; walker<anArray.length - index - 1; walk
41                      Object left = anArray[walker];
42                      Object right = anArray[walker+1];
43                      if ( aComparator.compare(left, right ) > 0 )        {
44                              Object tmp = anArray[walker];
45                              anArray[walker] = anArray[walker + 1];
46                              anArray[walker+1] = tmp;
47                      }
48                  }
49              }
50              ListIterator anIterator  = aList.listIterator();
51              for (int j=0; j<anArray.length; j++) {
52                      anIterator.next();
53                      anIterator.set(anArray[j]);
54              }
55
56      }
57
58      public static void main(String args[]) {
59              args = new String[4];
60
61              args[0] = "x"; args[1] = "b"; args[2] = "a"; args[3] = "d";
62              List l = Arrays.asList(args);
63              HpC_C_ex.sort(l);
64              System.out.println("1. " + l);
65
66              args[0] = "x"; args[1] = "b"; args[2] = "a"; args[3] = "d";
67              l = Arrays.asList(args);
68              HpC_C_ex.sort(l, new HpComparator()  );
69              System.out.println("2. " + l);
```

```
70
71        }
72      }
73
```

Source Code: Src/9/HpC_C_ex.java

13.36. Collections.sort()

```
1       import java.util.*;
2
3       public class HpComparator {
4
5         public static void sort(List list ) {
6               Object anArray[] = list.toArray();
7                       // Arrays.sort(a, c);           /////// this is the tr
8                       // http://www.cs.rit.edu/~hpb/Jdk5/api/java/util/ListI
9
10              for (int index=0; index<anArray.length - 1; index++)     {
11                  for (int walker=0; walker<anArray.length - index - 1; walk
12                      Object left =  anArray[walker];
13                      Object right =  anArray[walker+1];
14                      if ( left.compareTo(right ) > 0 )        {
15                              Object tmp = anArray[walker];
16                              anArray[walker] = anArray[walker + 1];
17                              anArray[walker+1] = tmp;
18                      }
19                  }
20              }
21
22
23              ListIterator i = list.listIterator();
24              for (int j=0; j<anArray.length; j++) {
25                      i.next();
26                      i.set(anArray[j]);
27              }
28        }
29
30      }
31
```

Source Code: Src/9/HpComparator_1.java

Next:

The List: [a Sun Oct 04 13:08:09 EDT 2010, b Sun Oct 04 13:08:10 EDT 2010, c Sun Oct 04 13:08:10 EDT 2010, d Sun Oct 04 13:08:10 EDT 2010]

```
1       import java.util.*;
2
3       public class HpComparator {
4
5         public static void sort(List list, Comparator c) {
```

```
 6              Object anArray[] = list.toArray();
 7                      // Arrays.sort(a, c);           /////// this is the tr
 8                      // http://www.cs.rit.edu/~hpb/Jdk5/api/java/util/ListI
 9
10              for (int index=0; index<anArray.length - 1; index++)     {
11                  for (int walker=0; walker<anArray.length - index - 1; walk
12                      Object left =  anArray[walker];
13                      Object right =  anArray[walker+1];
14                      if ( c.compare(left,  right ) > 0 )        {
15                              Object tmp = anArray[walker];
16                              anArray[walker] = anArray[walker + 1];
17                              anArray[walker+1] = tmp;
18                      }
19                  }
20              }
21
22
23              ListIterator i = list.listIterator();
24              for (int j=0; j<anArray.length; j++) {
25                      i.next();
26                      i.set(anArray[j]);
27              }
28        }
29
30      }
31
```

Source Code: Src/9/HpComparator_3.java

Function pointer in C/C++

13.37. Comparator in separate Classes

The use:

```
 1      import java.util.*;
 2
 3      public class ComparatorExTreeClass {
 4          static HpbComparator theNth = new HpbComparator();
 5
 6          protected static int  soManyS;
 7          protected String  name;
 8          protected int     waitingListN;
 9
10          public ComparatorExTreeClass(String name) {
11              if (name==null)
12                  throw new NullPointerException();
13              this.name = name;
14              this.waitingListN = soManyS ++;
15          }
16
17          public ComparatorExTreeClass(String name, int waitingListN) {
18              this(name);
19              this.waitingListN = waitingListN;
20          }
```

```
21
22
23          public String getName()    {
24              return name;
25          }
26
27          public String toString() {
28              return name + " - " + waitingListN;
29          }
30
31          public static void main(String args[]) {
32              WaitingList n[] = {
33                  new WaitingList("Bond"),
34                  new WaitingList("Jack"),
35                  new WaitingList("Elwood"),
36                  new WaitingList("You", -1),
37                  new WaitingList("Me", -1)
38              };
39              TreeSet l = new TreeSet(theNth);
40
41              for ( int i = 0; i < n.length; i ++ )   {
42                      System.out.println(i + " " + n[i]);
43                      l.add(n[i]);
44              }
45              System.out.println("the TreeSet: " + l);
46          }
47      }
48
```

Source Code: Src/9/ComparatorExTreeClass.java

The comparator:

```
 1
 2      import java.util.*;
 3
 4      public class HpbComparator implements Comparator {
 5
 6          public int compare(Object o1, Object o2) {
 7
 8              if ( ( o1 instanceof WaitingList )  &&
 9                   ( o2 instanceof WaitingList )  )   {
10                     WaitingList n1 = (WaitingList)o1;
11                     WaitingList n2 = (WaitingList)o2;
12                     int nameCompareV = n1.name.compareTo(n2.name);
13                     return ( nameCompareV == 0 ?
14                             n1.waitingListN - n2.waitingListN :
15                             nameCompareV);;
16              } else
17                  return -1;
18          }
19      }
20
```

Source Code: Src/9/HpbComparator.java

Questions:

— Which classes do you need?

— Comparator, do you need them?

— which kind of collections do you need?

13.38. Exercise II

• What do you have to be aware of, if you write a compareTo method?

• What do you have to be aware of, if you write a compare method?

Under Construction

14. Lambda Expressions

See Java Language Specifications (https://docs.oracle.com/javase/specs/jls/se8/html/jls-15.html#jls-15.27)

Material copied from here (https://docs.oracle.com/javase/tutorial/java/javaOO/lambdaexpressions.html)
and here. (http://tutorials.jenkov.com/java/lambda-expressions.html)

• Enable to treat functionality as a method argument, or code as data.

• A function that can be created without belonging to any class.

• A lambda expression can be passed around as if it was an object and executed on demand.

• Lambda expressions provide a clear and concise way to represent one method interface using an expression.

• Lambda expressions helps to iterate, filter and extract data from collection.

• Java lambda expressions are Java's first step into functional programming (https://en.wikipedia.org/wiki/Functional_programming)

• Java lambda expressions are mainky used to implement simple event listeners/callbacks, or in functional programming with the Java Streams API

Good Use Cases:

• Selection

• Pre/Post-conditions

14.1. Java Lambda Expression Example

```
 1      @FunctionalInterface
 2      interface Addable{
 3          int add(int a,int b);
 4      }
 5
 6      public class WithLambda_0{
 7          public static void main(String[] args) {
 8              //int a; not a new scope, like anonymous classes
 9
10              Addable ad1=(a,b)->(a+b);
11              System.out.println(ad1.add(10,20));
12
13              Addable ad2=(int a,int b)->(a+b);
14              System.out.println(ad2.add(100,200));
15          }
16      }
```

Source Code: Src/18_1/WithLambda_0.java

Other Example:

```
 1      @FunctionalInterface
 2      interface PrintInterface {
 3          void printIt(String s);
 4      }
 5      public class Print {
 6          public static void main(String args[]) {
 7                  PrintInterface myPrinterLamda           =  s ->  System.ou
 8                  PrintInterface myPrinterLamdaAlternativ = ( String s) -> {
 9                  PrintInterface myPrinterMethodRef       = System.out::prin
10
11                  myPrinterLamda.printIt("He is early");
12                  myPrinterLamdaAlternativ.printIt("He is punctual");
13                  myPrinterLamdaAlternativ.printIt("He is late");
14          }
15      }
```

Source Code: Src/18_1/Print.java

14.2. Java Lambda Expression Example

```
 1      @FunctionalInterface
 2      interface LambdaExpression1Interface {
 3              void anExample(int aIntegerBasicType);
 4      }
 5
 6      public class LambdaExpression1 {
 7
 8         public static void main(String args[]) {
 9              LambdaExpression1Interface lambdaObj = (int aInt)->System.out.
10
11              lambdaObj.anExample(42);
12          }
13
14      }
```

Source Code: Src/18_1/LambdaExpression1.java

Output:

```
java LambdaExpression1
42
```

Next (backwards compbility):

```
 1      @FunctionalInterface
 2      interface PrintInterface {
 3          void printIt(String s);
 4      }
 5      public class Print {
 6          public static void main(String args[]) {
 7                  PrintInterface myPrinterLamda           =  s ->  System.ou
 8                  PrintInterface myPrinterLamdaAlternativ = ( String s) -> {
```

```
 9                  PrintInterface myPrinterMethodRef       = System.out::prin
10
11                  myPrinterLamda.printIt("He is early");
12                  myPrinterLamdaAlternativ.printIt("He is punctual");
13                  myPrinterLamdaAlternativ.printIt("He is late");
14          }
15      }
```

Source Code: Src/18_1/Print.java

Next:

```
 1      import java.util.ArrayList;
 2      class LambdaExpression2
 3      {
 4          public static void main(String args[])
 5          {
 6              ArrayList<Integer> arrL = new ArrayList<Integer>();
 7              arrL.add(1); arrL.add(2); arrL.add(3); arrL.add(4);
 8
 9              arrL.forEach( n -> { if (n % 2 == 0) System.out.println(n); }
10          }
11      }
```

Source Code: Src/18_1/LambdaExpression2.java

Output:

```
java LambdaExpression2
2
4
```

Next:

```
 1      import java.util.ArrayList;
 2      class LambdaExpression3
 3      {
 4          public static void main(String args[])
 5          {
 6              ArrayList<Integer> arrL = new ArrayList<Integer>();
 7              ArrayList<Integer> arrN = new ArrayList<Integer>();
 8              arrL.add(1); arrL.add(2); arrL.add(3); arrL.add(4);
 9
10              arrL.forEach(n -> { if (n % 2 == 0) arrN.add(n); });
11
12              System.out.println("arrL:");
13              arrL.forEach(n -> { System.out.println(n); });
14              System.out.println("arrN:");
15              arrN.forEach(n -> { System.out.println(n); });
16
17          }
18      }
```

Source Code: Src/18_1/LambdaExpression3.java

Output:

```
java LambdaExpression3
arrL:
1
2
3
4
arrN:
2
4
```

14.3. Java Lambda Expression Syntax

```
(argument-list) -> {body}
```

• Argument-list: It can be empty or non-empty as well.

• Arrow-token: It is used to link arguments-list and body of expression.

• Body: It contains expressions and statements for lambda expression.

• No return type required: java 8 compiler and higher is able to infer the return type by checking the code.

14.4. Matching Lambdas to Interfaces

• A single method interface is referred to as a functional interface.

• Note: For JDK ≤ 8 Java interface can contain both default methods and static methods

• Matching a Java lambda expression against a functional interface is divided into these steps:

Requirement:

• Does the interface have only one method?

• Does the parameters of the lambda expression match the parameters of the single method?

• Does the return type of the lambda expression match the return type of the single method?

• If the answer is yes to these three questions, then the given lambda expression is matched successfully against the interface.

14.5. Lambda Expressions vs. Anonymous Interface Implementations

• An anonymous interface implementation can have state (member variables) whereas a lambda expression cannot.

• A lambda expression is thus said to be stateless

14.6. Lambda Type Inference

• With lambda expressions the type can often be inferred from the surrounding code

• The compiler infers the type of a parameter by looking elsewhere for the type - in this case the method definition.

14.7. Lambda Parameters

• Java lambda expressions are effectively just methods, lambda expressions can take parameters just like methods

• Zero Parameters:

```
() -> System.out.println("Zero parameter lambda");
```

• One Parameters:

```
(id) -> System.out.println("One parameter lambda " + id);
```

• Multiple Parameters:

```
(id1, id2) -> System.out.println("Mutltiple parameter lambda " + id1 + "/" + i
```

• var Parameter Types from Java 11:

```
Function<String, String> toLowerCase = (var input) -> input.toLowerCase();
```

14.8. Returning a Value From a Lambda Expression

• Return statement as ususal

14.9. Lambdas as Objects

• Java lambda expression is essentially an object

```
public interface MyComparator {

    public boolean compare(int a1, int a2);

}
MyComparator myComparator = (a1, a2) -> return a1 > a2;

boolean result = myComparator.compare(2, 5);
```

14.10. Examples

The following examples show different ways om solution for the same problem.

14.11. Example - Without Lambda

```
1       public class Id {
2
3           private String name;
4           private int number;
5
6           public Id(String name, int number) {
7               this.name = name;
8               this.number = number;
9           }
```

```
10          public String toString() {
11              return "my name is: " + name + "/" + number;
12          }
13          public String getName() {
14              return name;
15          }
16          public int getNumber() {
17              return number;
18          }
19          public void printNumber() {
20              System.out.println("    " + number);
21          }
22          public void printName() {
23              System.out.println("    " + name);
24          }
25      }
```

Source Code: Src/18_1/Id.java

Next

```
 1      import java.util.Arrays;
 2      import java.util.Comparator;
 3      import java.util.Collections;
 4      import java.util.ArrayList;
 5      import java.util.List;
 6      import java.util.stream.Collectors;
 7
 8      public class WithoutLambda {
 9
10          public static void printIdsHigherThan(List<Id> roster, int low, ir
11              for (Id p : roster) {
12      //------------------------------------------------
13                  if (  ( low <= p.getNumber() )
14                     && ( p.getNumber() <= high ) )
15                      p.printNumber();
16      //------------------------------------------------
17              }
18          }
19          public static void main(String[] args) {
20              List<Id> persons = Arrays.asList(
21                      new Id("Joe",   1234),
22                      new Id("Jane",  123),
23                      new Id("You",   12) );
24
25              printIdsHigherThan(persons, 12, 123);
26          }
27
28      }
```

Source Code: Src/18_1/WithoutLambda.java

14.12. Example - Without Lambda - With Interface

```
1       @FunctionalInterface
2       interface Check {
3           boolean doTheCheck(Id thisId, int low, int high);
4       }
```

Source Code: Src/18_1/Check.java

Next:

```
1       import java.util.Arrays;
2       import java.util.Comparator;
3       import java.util.Collections;
4       import java.util.ArrayList;
5       import java.util.List;
6       import java.util.stream.Collectors;
7
8       public class WithoutLambdaInterfaceA implements Check {
9
10      //-----------------------------------------------
11          public boolean doTheCheck(Id thisId, int low, int high) {
12              return (  ( low <= thisId.getNumber() )
13                     && ( thisId.getNumber() <= high ) );
14          }
15      //-----------------------------------------------
16          public void printIdsHigherThan(List<Id> roster, int low, int high)
17              for (Id p : roster) {
18                  if ( doTheCheck(p, low, high) )
19                      p.printNumber();
20              }
21          }
22          public void work()  {
23              List<Id> persons = Arrays.asList(
24                      new Id("Joe",   1234),
25                      new Id("Jane",  123),
26                      new Id("You",   12) );
27
28              System.out.println("printIdsHigherThan(persons, 12, 123);");
29              printIdsHigherThan(persons, 12, 123);
30
31          }
32          public static void main(String[] args) {
33              new WithoutLambdaInterfaceA().work();
34          }
35      }
```

Source Code: Src/18_1/WithoutLambdaInterfaceA.java

and

```
 1      import java.util.Arrays;
 2      import java.util.Comparator;
 3      import java.util.Collections;
 4      import java.util.ArrayList;
 5      import java.util.List;
 6      import java.util.stream.Collectors;
 7
 8      public class WithoutLambdaInterfaceB implements Check {
 9
10      //-----------------------------------------------
11          public boolean doTheCheck(Id thisId, int low, int high) {
12              return (  ( low <= thisId.getNumber() )
13                     && ( thisId.getNumber() <= high ) );
14          }
15      //-----------------------------------------------
16          public void printIdsHigherThanWithTester(List<Id> roster, Check te
17              for (Id p : roster) {
18      //-----------------------------------------------
19                  if ( tester.doTheCheck(p, low, high) )
20      //-----------------------------------------------
21                      p.printNumber();
22                   }
23          }
24          public void work()  {
25              List<Id> persons = Arrays.asList(
26                      new Id("Joe",   1234),
27                      new Id("Jane",  123),
28                      new Id("You",   12) );
29
30              System.out.println("printIdsHigherThanWithTester(persons, new
31              printIdsHigherThanWithTester(persons, new WithoutLambdaInterfa
32          }
33          public static void main(String[] args) {
34              new WithoutLambdaInterfaceB().work();
35          }
36      }
```

Source Code: Src/18_1/WithoutLambdaInterfaceB.java

14.13. Example - Without Lambda - With Anonymous Class

```
 1      import java.util.Arrays;
 2      import java.util.Comparator;
 3      import java.util.Collections;
 4      import java.util.ArrayList;
 5      import java.util.List;
 6      import java.util.stream.Collectors;
 7
 8      public class WithoutLambdaAnonmymousClass {
 9
10          public void printIdsHigherThanWithTester(List<Id> roster,
11                      Check tester,
12                      int low, int high) {
```

```
13              for (Id p : roster) {
14                  if ( tester.doTheCheck(p, low, high) )
15                      p.printNumber();
16                   }
17          }
18          public void work()  {
19              List<Id> persons = Arrays.asList(
20                      new Id("Joe",   1234),
21                      new Id("Jane",  123),
22                      new Id("You",   12) );
23
24      //-----------------------------------------------
25              printIdsHigherThanWithTester(persons, new Check() {
26                      public boolean doTheCheck(Id thisId, int low, int high
27                              return (  ( low <= thisId.getNumber() )
28                                      && ( thisId.getNumber() <= high ) );
29
30                              } }, 12, 123);
31      //-----------------------------------------------
32          }
33          public static void main(String[] args) {
34              new WithoutLambdaAnonmymousClass().work();
35          }
36      }
```

Source Code: Src/18_1/WithoutLambdaAnonmymousClass.java

14.14. Example - With Lambda 1

```
 1      import java.util.Arrays;
 2      import java.util.Comparator;
 3      import java.util.Collections;
 4      import java.util.ArrayList;
 5      import java.util.List;
 6      import java.util.stream.Collectors;
 7
 8      public class WithLambda_1 {
 9
10          public void work()  {
11              List<Id> persons = Arrays.asList(
12                      new Id("Joe",   1234),
13                      new Id("Jane",  123),
14                      new Id("You",   12) );
15
16              persons.forEach(  (p)->System.out.println(p.getName())  );
17          }
18          public static void main(String[] args) {
19              new WithLambda_1().work();
20          }
21      }
```

Source Code: Src/18_1/WithLambda_1.java

14.15. Example - With Lambda 2

The interface:

```
1       @FunctionalInterface
2       interface CheckNoLimit {
3           boolean doTheCheck(Id thisId);
4       }
```

Source Code: Src/18_1/CheckNoLimit.java

The implementation:

```
1       import java.util.Arrays;
2       import java.util.Comparator;
3       import java.util.Collections;
4       import java.util.ArrayList;
5       import java.util.List;
6       import java.util.stream.Collectors;
7
8       public class WithLambda_3 {
9
10          public static void printPersons(List<Id> roster, CheckNoLimit test
11                  for (Id p : roster) {
12                      if (tester.doTheCheck(p))
13                          p.printName();
14                  }
15          }
16
17          public void work()  {
18              List<Id> persons = Arrays.asList(
19                      new Id("Joe",   1234),
20                      new Id("Jane",  123),
21                      new Id("You",   12) );
22
23              printPersons(persons, (Id p) -> p.getNumber() >= 12 && p.getNu
24          }
25          public static void main(String[] args) {
26              new WithLambda_3().work();
27          }
28      }
```

Source Code: Src/18_1/WithLambda_3.java

14.16. Example - With Lambda 3

The interface:

```
1       @FunctionalInterface
2       interface Check {
3           boolean doTheCheck(Id thisId, int low, int high);
4       }
```

Source Code: Src/18_1/Check.java

The implementation:

```
1       import java.util.Arrays;
2       import java.util.Comparator;
3       import java.util.Collections;
4       import java.util.ArrayList;
5       import java.util.List;
6       import java.util.stream.Collectors;
7
8       public class WithLambda_5 implements Check {
9
10          public boolean doTheCheck(Id p, int low, int high)  {
11                      return p.getNumber() >= low  && p.getNumber() <= high;
12          }
13          public static void printPersons(List<Id> roster, Check tester, int
14                  for (Id p : roster) {
15                      if (tester.doTheCheck(p, low, high))
16                          p.printName();
17                  }
18          }
19
20          public void work()  {
21              List<Id> persons = Arrays.asList(
22                      new Id("Joe",   1234),
23                      new Id("Jane",  123),
24                      new Id("You",   12) );
25
26              printPersons(persons, (Id p, int low, int high) -> p.getNumber
27          }
28          public static void main(String[] args) {
29              new WithLambda_5().work();
30          }
31      }
```

Source Code: Src/18_1/WithLambda_5.java

14.17. Example

Homework: Convert

```
1       import java.util.*;
2
3       public class Sort {
4           public static void main(String args[]) {
5               List l = new ArrayList<String>();
6               l.add("a"); l.add("b");
7               l.add("0"); l.add("a");
8               Collections.sort(l);
9               System.out.println(l);
10          }
11      }
```

```
12
```

```
Source Code: Src/18_1/Sort.java
```

• Using anonymous classes

• Lampda Expression using a Lambda variable: Collections.sort(l, compareLambda);

• Direct Lampda Expression as argumment: Collections.sort(l, (S ...));

14.18. Lambdas and Method Reference

Java Language Specification (https://docs.oracle.com/javase/specs/jls/se13/html/jls-15.html#jls-15.13)

Copied from here. (http://what-when-how.com/Tutorial/topic-624i83o/Java-8-in-Action-Lambdas-Streams-and-Functional-Style-Programming-82.html)

Method references let you reuse existing method definitions and pass them just like lambdas.

```
Lambda Method: (Apple a) -> a.getWeight()
Method References: Apple::getWeight

Lambda Method: () -> Thread.currentThread().dumpStack()
Method References: Thread.currentThread()::dumpStack

Lambda Method: (str, i) -> str.substring(i)
Method References: String::substring

Lambda Method: (Apple a) -> a.getWeight
Method References: Apple::getWeight

Lambda References: (String s) -> System.out.println(s)
Method References: System.out::println
```

How to access

Type	Syntax
Reference to a static method	ClassName::staticMethodName
Reference to an instance method of an existing object	object::instanceMethodNames
Reference to an instance method of an arbitrary object of a particular type	ClassName::instanceMethodName
Reference to a constructor	ClassName::new

Example:

```
 1
 2      import java.util.Arrays;
 3      import java.util.ArrayList;
 4
 5      public class MethodReferences {
 6          public static void main(String args[])
 7          {
 8              ArrayList<String> arrL = new ArrayList<String>();
 9              arrL.add("a"); arrL.add("X"); arrL.add("b");
10              arrL.sort( String::compareToIgnoreCase);
```

```
11                      System.out.println("compareToIgnoreCase:\n      ");
12                      arrL.forEach(n -> {  System.out.print(n); });
13              System.out.println();
14              arrL.sort( String::compareTo);
15                      System.out.println("compareTo:\n        ");
16                      arrL.forEach(n -> {  System.out.println(n); });
17          }
18      }
```

Source Code: Src/18_1/MethodReferences.java

Next

```
 1
 2      import java.util.ArrayList;
 3      import java.util.Set;
 4      import java.util.HashSet;
 5
 6      @FunctionalInterface
 7      interface IdI {
 8          int getId();
 9      }
10
11      class Id implements IdI {
12              int idNumber = 0;
13
14              Id()                                    { this.idNumber = -1;
15              Id(int idNumber)                        { this.idNumber = idNu
16
17              public int getId()                      { return idNumber; }
18              int compareIdObject2(Id a1, Id a2)      { return a1.idNumber -
19              int compareIdObject(Id a2)              { return a2.idNumber -
20              static int compareId(Id a1, Id a2)      { return a1.idNumber -
21              public String toString()                { return "" + idNumber
22      }
23      public class MethodReferences_2 {
24          public static void main(String args[]) {
25              ArrayList<Id> arrL = new ArrayList<Id>();
26              Id id1 = new Id(1);
27              Id id2 = new Id(2);
28              Id id3 = new Id(3);
29              arrL.add(id1); arrL.add(id2); arrL.add(id3);
30
31              arrL.sort( Id::compareId);
32                      System.out.println("Id::compareId:      ");
33                      arrL.forEach(n -> {  System.out.print(n); });
34
35              arrL.sort((a, b) -> Id.compareId(a, b) );
36                      System.out.println("\n(a, b) -> Id.compareId(a, b):");
37                      arrL.forEach(n -> {  System.out.print(n); });
38
39              arrL.sort( Id::compareIdObject);
40                      System.out.println("\nId::compareIdObject:");
41                      arrL.forEach(n -> {  System.out.print(n); });
```

```
42
43              arrL.sort( id1::compareIdObject2);
44                      System.out.println("\nid1::compareIdObject2:");
45                      arrL.forEach(n -> {  System.out.print(n); });
46
47          }
48      }
```

Source Code: Src/18_1/MethodReferences_2.java

Next

```
 1      @FunctionalInterface
 2      interface NoteI {
 3          Note getNote(String msg);
 4      }
 5      class Note{
 6          Note(String theNote){
 7              System.out.print("--" + theNote);
 8          }
 9      }
10      public class MethodReferenceNew {
11          public static void main(String[] args) {
12              NoteI aNote = Note::new;
13              aNote.getNote("hello my friend");       // when does this do?
14          }
15      }
```

Source Code: Src/18_1/MethodReferenceNew.java

Next

14.19. Constructing Method References

• A method reference to a static method (Integer::parseInt)

• A method reference to an instance method of an arbitrary type (String::length)

• A method reference to an instance method of an existing object (expensiveTransaction::getValue)

15. Streams

Images and text copied and modified from here. (http://www.oracle.com/technetwork/articles/java/ma14-java-se-8-streams-2177646.html)

See also here. (https://docs.oracle.com/javase/8/docs/api/java/util/stream/package-summary.html)

Text is copied from: java doc (https://docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html)

15.1. Streams: Idea 1

Code before JDK 8:

```
List<Transaction> groceryTransactions = new Arraylist<>();
for ( Transaction t: transactions ) {
         if ( t.getType() == Transaction.GROCERY ) {
           groceryTransactions.add(t);
         }
       }
Collections.sort(groceryTransactions, new Comparator() {
                public int compare(Transaction t1, Transaction t2) {
                  return t2.getValue().compareTo(t1.getValue());
                }
          });

List<Integer> transactionIds = new ArrayList<>();
       for(Transaction t: groceryTransactions){
                transactionsIds.add(t.getId());
       }
```

Code after or with JDK 8 as a stream:

```
List<Integer> transactionsIds =
    transactions.stream()
                .filter(t -> t.getType() == Transaction.GROCERY)
                .sorted(comparing(Transaction::getValue).reversed())
                .map(Transaction::getId)
                .collect(toList());
```

15.2. Streams: Idea 2 - Parallelizing the Code

Example 1:

```
List<Integer> transactionsIds =
    transactions.parallelStream()
                .filter(t -> t.getType() == Transaction.GROCERY)
                .sorted(comparing(Transaction::getValue).reversed())
                .map(Transaction::getId)
                .collect(toList());
```

15.3. Streams - A Introcuction

Defintion of a stream: A short definition is 'a sequence of elements from a source that supports aggregate operations'.

• Sequence of elements: A stream provides an interface to a sequenced set of values of a specific element type.

• Streams don't actually store elements; they are computed on demand.

• Source: Streams consume from a data-providing source such as collections, arrays, or I/O resources.

• Aggregate operations: Streams support SQL-like operations and common operations from functional programing languages, such as filter, map, reduce, find, match, sorted, and so on.

The classes Stream, IntStream, LongStream, and DoubleStream are streams over objects and the primitive int, long and double types.

15.4. Streams vs Collections

• Pipelining: Many stream operations return a stream themselves.

• Allows operations to be chained to form a larger pipeline which enables certain optimizations, such as laziness and short-circuiting, which we explore later.

• Internal iteration: In contrast to collections, which are iterated explicitly (external iteration), stream operations do the iteration behind the scenes for you.

15.5. A More Detailed Look

• No storage. A stream is not a data structure that stores elements; instead, it conveys elements from a source such as a data structure, an array, a generator function, or an I/O channel, through a pipeline of computational operations.

• Functional in nature. An operation on a stream produces a result, but does not modify its source. For example, filtering a Stream obtained from a collection produces a new Stream without the filtered elements, rather than removing elements from the source collection.

• Laziness-seeking. Many stream operations, such as filtering, mapping, or duplicate removal, can be implemented lazily, exposing opportunities for optimization. For example, "find the first String with three consecutive vowels" need not examine all the input strings. Stream operations are divided into intermediate (Stream-producing) operations and terminal (value- or side-effect-producing) operations. Intermediate operations are always lazy.

• Intermediate operations are further divided into stateless and stateful operations.

  — Stateless operations, such as filter and map, retain no state from previously seen element when processing a new element.

  — Stateful operations, such as distinct and sorted, may incorporate state from previously seen elements when processing new elements.

• Possibly unbounded. While collections have a finite size, streams need not. Short-circuiting operations such as limit(n) or findFirst() can allow computations on infinite streams to complete in finite time.

• Consumable. The elements of a stream are only visited once during the life of a stream. Like an Iterator, a new stream must be generated to revisit the same elements of the source.

15.6. Creation of Streams

Streams can be obtained in a number of ways:

• From a Collection via the stream() and parallelStream() methods;

• From an array via Arrays.stream(Object[]);

• From static factory methods on the stream classes, such as Stream.of(Object[]), IntStream.range(int, int) or Stream.iterate(Object, UnaryOperator);

• The lines of a file can be obtained from BufferedReader.lines();

• Streams of file paths can be obtained from methods in Files;

• Streams of random numbers can be obtained from Random.ints();

• Numerous other stream-bearing methods in the JDK, including BitSet.stream(), Pattern.splitAsStream(java.lang.CharSequence), and JarFile.stream().

```
Stream<Integer> stream = Stream.of(1,2,3,4);
SWithStreamIdMultipleConditionstream<Integer> sequentialStream = myList.stream
Arrays.stream(Object[]);

Stream<Integer> sequentialStream = myList.stream();

Stream<Integer> parallelStream = myList.parallelStream();
```

15.7. Linear - Examples 1 - filter and collect

Without streams:

```
 1      import java.util.ArrayList;
 2      import java.util.Arrays;
 3      import java.util.List;
 4
 5      public class WithoutStreams {
 6
 7          public static void main(String[] args) {
 8
 9              List<String> lines = Arrays.asList("one", "two", "three");
10              List<String> result = getFilterOutput(lines, "two");
11              for (String tmp : result) {
12                  System.out.println("main: " + tmp);
13              }
14
15          }
16
17          private static List<String> getFilterOutput(List<String> lines, St
18              List<String> result = new ArrayList<>();
19              for (String line : lines) {
20                  if (!"three".equals(line)) {
21                      result.add(line);
22                  }
23              }
24              return result;
25          }
26
```

```
27      }
```

Source Code: Src/18/WithoutStreams.java

With streams:

```
 1      import java.util.Arrays;
 2      import java.util.List;
 3      import java.util.stream.Collectors;
 4
 5      public class WithStreams {
 6
 7          public static void main(String[] args) {
 8
 9              List<String> lines = Arrays.asList("one", "two", "three");
10
11              List<String> result = lines.stream()
12                      .filter(line -> !"three".equals(line))
13                      .collect(Collectors.toList());
14
15              result.forEach(System.out::println);
16
17          }
18
19      }
```

Source Code: Src/18/WithStreams.java

15.8. Linear - Examples 2 - filter(), findAny() and orElse()

```
 1      import java.util.Arrays;
 2      import java.util.List;
 3
 4      public class WithoutStreamId {
 5
 6          public static void main(String[] args) {
 7
 8              List<Id> persons = Arrays.asList(
 9                      new Id("Joe",   1234),
10                      new Id("Jane",  123),
11                      new Id("You",   12) );
12              System.out.println("getByName(persons, You): " + getByName(per
13              System.out.println("getByName(persons, x): " + getByName(perso
14          }
15
16          private static Id getByName(List<Id> persons, String name) {
17              Id result = null;
18              for (Id tmp : persons) {
19                  if (name.equals(tmp.getName())) {
20                      result = tmp;
21                  }
22              }
```

```
23              return result;
24          }
25      }
26
```

Source Code: Src/18/WithoutStreamId.java

Next

```
 1      import java.util.Arrays;
 2      import java.util.List;
 3
 4      public class WithStreamId {
 5
 6          public static void main(String[] args) {
 7
 8              List<Id> persons = Arrays.asList(
 9                      new Id("Joe",   1234),
10                      new Id("Jane",  123),
11                      new Id("You",   12) );
12
13              System.out.println("You: " +
14                       persons.stream()
15                      .filter(x -> "You".equals(x.getName()))
16                      .findAny()
17                      .orElse(null) );
18
19              System.out.println("x: " +
20                       persons.stream()
21                      .filter(x -> "x".equals(x.getName()))
22                      .findAny()
23                      .orElse(null) );
24
25          }
26
27      }
```

Source Code: Src/18/WithStreamId.java

Next

```
 1      import java.util.Arrays;
 2      import java.util.List;
 3
 4      public class WithStreamIdMultipleConditions {
 5
 6          public static void main(String[] args) {
 7
 8              List<Id> persons = Arrays.asList(
 9                      new Id("Joe",   1234),
10                      new Id("Jane",  123),
11                      new Id("You",   12) );
12
```

```
13              System.out.println("You && 12: " +
14                      persons.stream()
15                      .filter((p) -> "You".equals(p.getName()) && 12 == p.ge
16                      .findAny()
17                      .orElse(null) );
18
19              System.out.println("You && 123: " +
20                      persons.stream()
21                      .filter(p -> {
22                          if ("You".equals(p.getName()) && 123 == p.getNumbe
23                              return true;
24                          }
25                          return false;
26                      }).findAny()
27                      .orElse(null) );
28
29          }
30      }
```

Source Code: Src/18/WithStreamIdMultipleConditions.java

15.9. Mapping and Collecting

```
 1      import java.math.BigDecimal;
 2      import java.util.ArrayList;
 3      import java.util.Arrays;
 4      import java.util.List;
 5      import java.util.stream.Collectors;
 6
 7      public class Getting {
 8
 9          public static void main(String[] args) {
10
11              List<Id> all = Arrays.asList(
12                      new Id("Joe",   1234),
13                      new Id("Jane",  123),
14                      new Id("You",   12) );
15
16              //Before Java 8
17              List<String> result = new ArrayList<>();
18              for (Id person : all) {
19                  result.add(person.getName());
20              }
21              System.out.println(result);
22
23              // With Java 8
24              List<String> collect = all.stream().map(x -> x.getName()).coll
25              System.out.println(collect);
26
27          }
28
29      }
30
```

Source Code: Src/18/Getting.java

15.10. Parallelism

```
double average = roster
    .parallelStream()
    .filter(p -> p.getGender() == Person.Sex.MALE)
    .mapToInt(Person::getAge)
    .average()
    .getAsDouble();
```

```
 1      import java.util.Arrays;
 2      import java.util.Comparator;
 3      import java.util.Collections;
 4      import java.util.ArrayList;
 5      import java.util.List;
 6      import java.util.stream.Collectors;
 7
 8      public class P {
 9
10          public static void main(String[] args) {
11
12                  Arrays.asList("a", "b", "c", "d", "e")
13                  .parallelStream()
14                  .filter(s -> {
15                      System.out.format("filter: %s [%s]\n",
16                          s, Thread.currentThread().getName());
17                      return true;
18                  })
19                  .map(s -> {
20                      System.out.format("map: %s [%s]\n",
21                          s, Thread.currentThread().getName());
22                      return s.toUpperCase();
23                  })
24                  .sorted((s1, s2) -> {
25                      System.out.format("sort: %s <> %s [%s]\n",
26                          s1, s2, Thread.currentThread().getName());
27                      return s1.compareTo(s2);
28                  })
29                  .forEach(s -> System.out.format("forEach: %s [%s]\n",
30                      s, Thread.currentThread().getName()));
31                  }
32
33      }
```

Source Code: Src/18/P.java

15.11. Example:

```
1       // /Volumes/home_8tb_hpb/hpb/Desktop/Download/jdk1.8.0_231/sample/lamb
2       import java.io.IOException;
3       import java.io.UncheckedIOException;
4       import java.nio.file.Files;
5       import java.nio.file.Path;
6       import java.nio.file.Paths;
7       import java.util.Arrays;
8       import java.util.List;
9       import java.util.regex.Pattern;
10      import java.util.stream.Stream;
11
12      import static java.util.stream.Collectors.toList;
13
14      public class Ex {
15
16          private static void printUsageAndExit(String... str) {
17              System.out.println("Usage: " + Ex.class.getSimpleName()
18                      + " [OPTION]... PATTERN FILE...");
19              System.out.println("Search for PATTERN in each FILE. "
20               + "If FILE is a directory then whole file tree of the directo
21               + " will be processed.");
22              System.out.println("Example: grep -m 100 'hello world' main.c'
23              System.out.println("Options:");
24              System.out.println("    -m NUM: stop analysis after NUM matche
25              Arrays.asList(str).forEach(System.err::println);
26              System.exit(1);
27          }
28
29          public static void main(String[] args) throws IOException {
30              long maxCount = Long.MAX_VALUE;
31              if (args.length < 2) {
32                  printUsageAndExit();
33              }
34              int i = 0;
35              //parse OPTIONS
36              while (args[i].startsWith("-")) {
37                  switch (args[i]) {
38                      case "-m":
39                          try {
40                              maxCount = Long.parseLong(args[++i]);
41                          } catch (NumberFormatException ex) {
42                              printUsageAndExit(ex.toString());
43                          }
44                          break;
45                      default:
46                          printUsageAndExit("Unexpected option " + args[i]);
47                  }
48                  i++;
49              }
50              //parse PATTERN
51              Pattern pattern = Pattern.compile(args[i++]);
```

```
52              if (i == args.length) {
53                  printUsageAndExit("There are no files for input");
54              }
55
56              try {
57                  List<Path> files = Arrays.stream(args, i, args.length)
58                          .map(Paths::get)
59                          // flatMap ensures each I/O-based stream will be c
60                          .flatMap(Ex::getPathStream)
61                          .filter(Files::isRegularFile)
62                          .collect(toList());
63                  files.parallelStream()
64                          // flatMap ensures each I/O-based stream will be c
65                          .flatMap(Ex::path2Lines)
66                          .filter(pattern.asPredicate())
67                          .limit(maxCount)
68                          .forEachOrdered(System.out::println);
69              } catch (UncheckedIOException ioe) {
70                  printUsageAndExit(ioe.toString());
71              }
72          }
73          private static Stream<Path> getPathStream(Path path) {
74              try {
75                  return Files.walk(path);
76              } catch (IOException e) {
77                  throw new UncheckedIOException(e);
78              }
79          }
80          private static Stream<String> path2Lines(Path path) {
81              try {
82                  return Files.lines(path);
83              } catch (IOException e) {
84                  throw new UncheckedIOException(e);
85              }
86          }
87      }
```

Source Code: Src/18_1/Ex.java

16. Reflection API

Most of the stuff 'stolen' from

The represents, or reflects, the classes, interfaces, and objects in the current Java Virtual Machine.

Typical use:

- debuggers,
- class browsers,
- GUI builders.

With the reflection API you can:

- Determine the class of an object.
- Get information about a class's modifiers, fields, methods, constructors, and superclasses.

- Find out what constants and method declarations belong to an interface.
- Create an instance of a class whose name is not known until runtime.
- Get and set the value of an object's field, even if the field name is unknown to your program until runtime.
- Invoke a method on an object, even if the method is not known until runtime.
- Create a new array, whose size and component type are not known until runtime, and then modify the array's component

16.1. Intro example

```
 1      import java.lang.reflect.*;
 2
 3      class Intro {
 4
 5              static void printName(Object o) {
 6                      Class c = o.getClass();
 7                      String s = c.getName();
 8                      System.out.println(s);
 9              }
10
11              public static void main(String[] args) {
12                      String aS = new String();
13                      printName(aS);
14              }
15      }
```

Source Code: Src/17/Intro.java

Result:

```
% javac Intro.java
% java Intro
java.lang.String
```

16.2. Discovering Class Modifiers

See

```
 1      import java.lang.reflect.*;
 2
 3      public final class MyModifier {
 4
 5              void printName(Object o) {
 6                      Class c = o.getClass();
 7                      String s = c.getName();
 8                      System.out.println(s);
 9              }
10
11              public void printModifiers(Object o) {
12              Class c = o.getClass();
13              int m = c.getModifiers();
14
15              if (Modifier.isPrivate(m))
16                      System.out.println("private");
17              if (Modifier.isPublic(m))
18                      System.out.println("public");
19              if (Modifier.isAbstract(m))
20                      System.out.println("abstract");
21              if (Modifier.isFinal(m))
22                      System.out.println("final");
23              }
24
25              public static void main(String[] args) {
26                      MyModifier aM = new MyModifier();
27                      aM.printName(aM);
28                      aM.printModifiers(aM);
29              }
30
31      }
```

Source Code: Src/17/MyModifier.java

Result:

```
% java MyModifier
MyModifier
public
final
```

16.3. Identifying Class Fields

Application such as a class browser, might want to find out what fields belong to a particular class. This can be identified by invoking the getFields method on a Class object. The getFields method returns an array of Field objects containing one object per accessible public field.

A public field is accessible if it is a member of either:

- this class
- a superclass of this class
- an interface implemented by this class
- an interface extended from an interface implemented by this class

```
 1
 2      import java.lang.reflect.*;
 3
 4      class What {
 5              public int publicVar;;
 6              private int privateVar;;
 7              static int staticVar;;
 8
 9              static void printFieldNames(Object o) {
10                      Class c = o.getClass();
11                      Field[] publicFields = c.getFields();
12                      for (int i = 0; i < publicFields.length; i++) {
13                              String fieldName = publicFields[i].getName();
14                              Class typeClass = publicFields[i].getType();
15                              String fieldType = typeClass.getName();
16                              System.out.println("\tName: " + fieldName +
17                                  ", Type: " + fieldType);
18                      }
19              }
20
21              public static void main(String[] args) {
22                      String aS = new String();
23                      Thread aT = new Thread();
24                      What aW = new What();
25                      System.out.println("String: ");
26                      printFieldNames(aS);
27
28                      System.out.println("Thread: ");
29                      printFieldNames(aT);
30
31                      System.out.println("What: ");
32                      printFieldNames(aW);
33              }
34      }
```

Source Code: Src/17/What.java

Result:

```
% java What
tring:
        Name: serialVersionUID, Type: long
        Name: CASE_INSENSITIVE_ORDER, Type: java.util.Comparator
Thread:
        Name: MIN_PRIORITY, Type: int
        Name: NORM_PRIORITY, Type: int
        Name: MAX_PRIORITY, Type: int
What:
        Name: publicVar, Type: int
```

16.4. Getting Values

A development tool such as a debugger, must be able to obtain field values. This is a three-step process:

1. Create a Class object.
2. Create a Field object by invoking getField on the Class object.
3. Invoke one of the get methods on the Field object

```
 1
 2      import java.lang.reflect.*;
 3
 4      class Get {
 5              public int publicVar = 42;
 6              private int privateVar;
 7              static int staticVar;
 8
 9              static void getValue(Object o) {
10                      Class c = o.getClass();
11                      Integer value;
12                      try {
13                              Field publicVarField = c.getField("publicVar")
14                              value = (Integer) publicVarField.get(o);
15                              System.out.println("value: " + value);
16                      } catch (NoSuchFieldException e) {
17                              System.out.println(e);
18                              } catch (SecurityException e) {
19                              System.out.println(e);
20                              } catch (IllegalAccessException e) {
21                              System.out.println(e);
22                              }
23              }
24
25              public static void main(String[] args) {
26                      Get aG = new Get();
27
28                      System.out.println("Get: ");
29                      getValue(aG);
30              }
31      }
```

Source Code: Src/17/Get.java

```
% java Get
Get:
value: 42
```

16.5. Setting Values

Some debuggers allow users to change field values during a debugging session. A tool that has this capability, must call one of the Field class's set methods.

1. To modify the value of a field, perform the following steps: Create a Class object. For more information, see the section Retrieving Class Objects.
2. Create a Field object by invoking getField on the Class object.
3. Class Fields shows you how. Invoke the appropriate set method on the Field object.

The Field class provides several set methods. Specialized methods, such as setBoolean and setInt, are for modifying primitive types. If the field you want to change is an object invoke the set method. It is possible to set to modify a primitive type, but the appropriate wrapper object for the value parameter must be used.

```
 1
 2      import java.lang.reflect.*;
 3
 4      class Set {
 5              public int publicVar = 42;
 6              private int privateVar;
 7              static int staticVar;
 8
 9              static void setValue(Object o) {
10                      Class c = o.getClass();
11                      Integer value;
12                      try {
13                              Field publicVarField = c.getField("publicVar")
14                              publicVarField.set(o, new Integer(24) );
15                      } catch (NoSuchFieldException e) {
16                              System.out.println(e);
17                              } catch (SecurityException e) {
18                              System.out.println(e);
19                              } catch (IllegalAccessException e) {
20                              System.out.println(e);
21                              }
22              }
23
24              public static void main(String[] args) {
25                      Set aS = new Set();
26
27                      System.out.println("before: aS.publicVar = "
28                              + aS.publicVar);
29                      setValue(aS);
30                      System.out.println("after:  aS.publicVar = "
31                              + aS.publicVar);
32              }
33      }
```

Source Code: Src/17/Set.java

```
% java Set
before: aS.publicVar = 42
after:  aS.publicVar = 24
```

16.6. Obtaining Method Information

To find out what public methods belong to a class, invoke the method named getMethods. The array returned by getMethods contains Method objects. This can be used to uncover a method's name, return type, parameter types, set of modifiers, and set of throwable exceptions. All of this information would be useful to write a class browser or a debugger. A metod can be called with Method.invoke.

1. It retrieves an array of Method objects from the Class object by calling getMethods.
2. For every element in the Method array, the program:
   a. retrieves the method name by calling getName
   b. gets the return type by invoking getReturnType
   c. creates an array of Class objects by invoking getParameterTypes
3. The array of Class objects created in the preceding step represents the parameters of the method. To retrieve the class name for every one of these parameters, the program invokes getName against each Class object in the array.

```
 1
 2      import java.lang.reflect.*;
 3
 4      class Show {
 5              public int publicVar = 42;
 6              private int privateVar;
 7              static int staticVar;
 8
 9              public static int HPclassM(String aString)    {
10                                      return 1;
11              }
12              public int        HPobjectM(int i, Boolean aBoolean) {
13                                      return 1;
14              }
15
16              void showMethods(Object o) {
17                      Class c = o.getClass();
18                      Method[] theMethods = c.getMethods();
19                      for (int i = 0; i < theMethods.length; i++) {
20                              String methodString = theMethods[i].getName();
21                              System.out.println("Name: " + methodString);
22                              String returnString =
23                              theMethods[i].getReturnType().getName();
24                              System.out.println("   Return Type: " + return
25                              Class[] parameterTypes = theMethods[i].getPara
26                              System.out.print("   Parameter Types:");
27                              for (int k = 0; k < parameterTypes.length; k +
28                                      String parameterString = parameterType
29                                      System.out.print(" " + parameterString
30                                      }
31                              System.out.println();
32                      }
33              }
34
35              public static void main(String[] args) {
```

```
36                      Show aS = new Show();
37                      System.out.println("Show: ");
38                      aS.showMethods(aS);
39
40              }
41      }
```

Source Code: Src/17/Show.java

```
% java Show
Show:
Name: HPclassM
   Return Type: int
   Parameter Types: java.lang.String
Name: main
   Return Type: void
   Parameter Types: [Ljava.lang.String;
 ...
Name: wait
   Return Type: void
   Parameter Types: long int
Name: HPobjectM
   Return Type: int
   Parameter Types: int java.lang.Boolean
```

16.7. Invoking Methods

A debugger should hallows a user to select and then invoke methods during a debugging session. Since it is not know at compile time which methods the user will invoke, the method name can not be hardcoded in the source code.

1. Create a Class object that corresponds to the object whose method you want to invoke. See the section Retrieving Class Objects for more information.
2. Create a Method object by invoking getMethod on the Class object. The getMethod method has two arguments: a String containing the method name, and an array of Class objects. Each element in the array corresponds to a parameter of the method you want to invoke. For more information on retrieving Method objects, see the section Obtaining Method Information
3. Invoke the method by calling invoke. The invoke method has two arguments: an array of argument values to be passed to the invoked method, and an object whose class declares or inherits the method.

```
 1
 2      import java.lang.reflect.*;
 3
 4      class Invoke {
 5              public int publicVar = 42;
 6              private int privateVar;
 7              static int staticVar;
 8
 9              public String objectM(Boolean aBoolean, Integer i) {
10                      System.out.println("objectM: i        = "
11                              + i);
12                      System.out.println("objectM: aBoolean = "
13                              + aBoolean);
14                      return "Alles Verloren, alles Geboren ...";
15              }
16
17              void invokeMethod(Object o) {
18                      Class c = Invoke.class;
19                      Class[] parameterTypes = new Class[] {Boolean.class,
20                                                            Integer.class};
21                      Method objectM;
22                      Object[] arguments = new Object[] {new Boolean(true),
23                                                         new Integer(2)    }
24                      try {
25                              objectM = c.getMethod("objectM", parameterType
26                              System.out.println(
27                                      (String) objectM.invoke((Invoke)o, arg
28                      } catch (NoSuchMethodException e) {
29                              System.out.println(e);
30                      } catch (IllegalAccessException e) {
31                              System.out.println(e);
32                      } catch (InvocationTargetException e) {
33                              System.out.println(e);
34                      }
35
36              }
37
```

```
38              public static void main(String[] args) {
39                      Invoke aS = new Invoke();
40                      System.out.println("Invoke: ");
41                      aS.invokeMethod(aS);
42
43              }
44      }
```

Source Code: Src/17/Invoke.java

```
% java Invoke
Invoke:
objectM: i        = 2
objectM: aBoolean = true
Alles Verloren, alles Geboren ...
```

```
 1
 2      import java.lang.reflect.*;
 3
 4      class Stubs {
 5              public int publicVar = 42;
 6              private int privateVar;
 7              static int staticVar;
 8
 9              public String objectM(Boolean aBoolean, Integer i) {
10                      System.out.println("objectM: i        = "
11                              + i);
12                      System.out.println("objectM: aBoolean = "
13                              + aBoolean);
14                      return "Alles Verloren, alles Geboren ...";
15              }
16
17              static void invokeMethod(Object o) {
18                      System.out.println("invokeMethod: ");
19                      Class c = o.getClass();
20                      System.out.println("class: " + c );
21                      Class[] parameterTypes = new Class[] {String.class};
22
23                      Method objectM;
24                      Object[] arguments = new Object[] {new String("Merlin'
25                      try {
26                              objectM = c.getMethod("sayHello", parameterTyp
27                              System.out.println(
28                                      (String) objectM.invoke((HelloInterfac
29                      } catch (NoSuchMethodException e) {
30                              System.out.println(e);
31                      } catch (IllegalAccessException e) {
32                              System.out.println(e);
33                      } catch (InvocationTargetException e) {
34                              System.out.println(e);
35                      }
36
```

```
37              }
38
39              public static void main(String[] args) {
40                      try {
41                              HelloInterface aHello = new HelloImplementatio
42                              invokeMethod(aHello);
43                              System.exit(0);
44                      } catch ( Exception e )         {
45                              e.printStackTrace();
46                      }
47
48              }
49      }
```

Source Code: Src/17/Stubs.java

17. Under Construction

18. Question Regarding the Final Exam

18.1. TreeSets their utility vs HashMap

18.2. Question 1/Was asked twice

```
 1      public class X_3 extends Thread    {
 2          private String info;
 3          Object o_1;
 4          Object o_2;
 5          Object stop;
 6          public X_3 (String info, Object o_1, Object o_2, Object stop) {
 7              this.info        = info;
 8              this.o_1        = o_1;
 9              this.o_2        = o_2;
10              this.stop        = stop;
11          }
12          public void run () {
13              synchronized ( o_1 ) {
14                  try {
15                      synchronized ( stop ) {
16                          if ( info == "0")    {
17                              new X_3("1", o_2, o_1, stop).start();
18                              stop.wait();
19                          } else
20                              synchronized ( stop ) {
21                                  stop.notify();
22                              }
23                      }
24                      synchronized ( o_2 ) {
25                          System.out.println(info + ": I am here.");
26                      }
27                  } catch ( Exception e ) { }
28              }
```

```
29          }
30          public static void main (String args []) {
31              new X_3("0", new Object(), new Object(), new Object()).start()
32          }
33      }
```

Source Code: Src/17/X_3.java

Is it possible, by adding sleep statements, that this program will not end up in a dead lock? You can, if needed, mark the code, with numbers where you would like to add sleep statements.

Answer: yes/no

Explanation:

18.3. Question 6 - 2161 Final

```
 1      public class X_6 extends Thread    {
 2          private String info;
 3          Object o_1;
 4          Object o_2;
 5          Object stop;
 6          static boolean oneIsRunning = false; // is static important?
 7                                  // es wird nur ein
 8                                  // Objekt erzeugt
 9          public X_6 (String info, Object o_1, Object o_2, Object stop) {
10              this.info    = info;
11              this.o_1     = o_1;
12              this.o_2     = o_2;
13              this.stop    = stop;
14          }
15          public void tryIt () {
16              synchronized ( o_1 ) {
17                  try {
18                      synchronized ( stop ) {
19                          if ( ! oneIsRunning )    {
20                              new X_6("1", o_2, o_1, stop).start();
21                              oneIsRunning = true;
22                              stop.wait();
23                          } else
24                              stop.notify();
25                      }
26                      System.out.println("1. info: " + info );
27                      synchronized ( o_2 ) {
28                              System.out.println("2. info: " + info );
29                      }
30                  } catch ( Exception e ) { }
31              }
32          }
33          public void run () {
34              tryIt();
35          }
36          public static void main (String args []) {
37              Object o_1 = new Object();
38              Object o_2 = new Object();
39              Object stop = new Object();
40              new X_6("0", o_1, o_2, stop).start();
41          }
42      }
```

Source Code: Src/17/X_6.java

What is/are the output/s of this program?

18.4. Comparator

I am not sure about *Comparator* and its specific usage. You gave us the exam questionsâ12/13/2018 Version5.3, the third question is about *Comparator class*, I donât know the exactly answer.

```
 1
 2      import java.util.*;
 3
 4      class LengthComparator implements Comparator {
 5              public int compare(Object o1, Object o2)        {
 6                      String s1 = (String)o1;
 7                      String s2 = (String)o2;
 8                      return s1.length() - s2.length() ;
 9              }
10      }
11      public class Sort {
12
13              public static void main(String args[]) {
14                      String[] words = { "aaaa", "bbb", "CC", "D"};
15                      List l = Arrays.asList(words);
16                      Collections.sort(l);
17                              System.out.println("default sort: " + l);
18                      Collections.sort(l, new LengthComparator());
19                              System.out.println("length sort: " + l);
20
21              }
22      }
```

Source Code: Src/17/Sort.java

18.5. Semaphores and Monitors

I have question about *Semaphores* and *Monitors* usage.

See chapter 11

18.6. Deadlock

I have question about: if the final exam ask us to make a deadlock, could we use *sleep()*?

18.7. RMI

Given is the following code:

```
 1      import java.rmi.*;
 2      import java.rmi.server.UnicastRemoteObject;
 3
 4      public class WaitImplementation extends UnicastRemoteObject
 5              implements WaitInterface {
 6
 7              int id;
 8              public WaitImplementation() throws RemoteException {
 9              }
10              public WaitImplementation(int id) throws RemoteException {
11                      this.id = id;
12              }
13              public int waitForAWhile() throws RemoteException {
14                      System.out.println("---> Client/" + id );
```

```
15                      try {
16                              if ( id == 1 )  Thread.sleep(100);
17                      } catch (Exception e ) {}
18                      System.out.println("<--- Client/" + id );
19                      return id;
20              }
21      }
```

Source Code: Src/17/WaitImplementation.java

Next

```
1       public interface WaitInterface extends java.rmi.Remote {
2               int waitForAWhile() throws java.rmi.RemoteException;
3       }
```

Source Code: Src/17/WaitInterface.java

Next

```
1       import java.rmi.*;
2
3       public class WaitServer {
4
5               public static void main(String args[])
6               {
7                       // System.setSecurityManager(new RMISecurityManager())
8
9                       try {
10                          System.out.println("WaitServer trying to bind in r
11                          WaitInterface obj_1 = new WaitImplementation(1);
12                          WaitInterface obj_2 = new WaitImplementation(2);
13                          Naming.rebind("//spiegel/Wait_1", obj_1);
14                          Naming.rebind("//spiegel/Wait_2", obj_2);
15                          Naming.rebind("//spiegel/Wait_3", obj_2);
16                          System.out.println("WaitServer bound in registry")
17                      } catch (Exception e) {
18                          System.exit(0);
19                      }
20              }
21      }
```

Source Code: Src/17/WaitServer.java

Next

```
1       import java.rmi.*;
2       import java.util.*;
3
4       public class WaitClient extends Thread {
5               int id;
6               public WaitClient(int id)       {
7                       this.id = id;
```

```
 8              }
 9              public void run()       {
10                      try {
11                              WaitInterface obj = (WaitInterface)Naming.look
12                              obj.waitForAWhile();
13                      } catch (Exception e) { System.exit(0); }
14              }
15              public static void main(String args[] ) {
16                      new WaitClient(1).start();
17                      new WaitClient(2).start();
18                      new WaitClient(3).start();
19              }
20      }
```

Source Code: Src/17/WaitClient.java

Assume all is working and no exception will be thrown.

Is the following output possible?

```
---> Client/2
---> Client/1
<--- Client/1
---> Client/2
<--- Client/2
<--- Client/2
```

Answer: yes/no

18.8. 11.27

few examples related to synchronization on string objects is confusing for me. T_2.java,T_2b.java

```
 1      /*
 2       * is this output       --->
 3       *                      <--
 4       *                      ...
 5       * the only possible output?
 6       * nein unterschiedliche zwei objekte
 7       */
 8      public class T_2 extends Thread    {
 9          static String info;
10
11          public T_2(String info )    {
12              this.info = info;
13          }
14          private void inProtected () {
15             synchronized ( info )       {
16                   System.err.println("--> " + info);
17                   try {
18                            sleep(1000);
19                   } catch (  Exception e ) {
20                       e.printStackTrace();
```

```
21                  }
22                  System.err.println("<-- " + info);
23              }
24          }
25
26          public void run () {
27              inProtected();
28          }
29          public static void main (String args []) {
30              String aString = "a";
31              T_2 one = new T_2(aString);
32              one.start();
33              T_2 two = new T_2(aString);
34              two.start();
35              aString = "b";
36              // new T_2("a").start();
37              // new T_2("b").start();
38
39          }
40      }
```

Source Code: Src/Question_Week_5/T_2.java

Next

```
1       /*
2        * is this output       --->
3        *                      <--
4        *                      ...
5        * the only possible output?
6        * ja ein objekt
7        */
8       public class T_2b extends Thread    {
9           private String info;
10
11          public T_2b(String info )   {
12              this.info = info;
13          }
14          private synchronized void inProtected () {
15                  System.err.println("--> " + info);
16                  try {
17                          sleep(1000);
18                  } catch (  Exception e ) {
19                      e.printStackTrace();
20                  }
21                  System.err.println("<-- " + info);
22          }
23
24          public void run () {
25              inProtected();
26          }
27          public static void main (String args []) {
28              new T_2b("hello").start();
29              new T_2b("hello").start();
```

```
30
31          }
32      }
```

Source Code: Src/Question_Week_5/T_2b.java

Next

18.9. 12.19

Method references for constructors : eg of MethodReferences_2.java

```
 1
 2      import java.util.ArrayList;
 3      import java.util.Set;
 4      import java.util.HashSet;
 5
 6      @FunctionalInterface
 7      interface IdI {
 8          int getId();
 9      }
10
11      class Id implements IdI {
12              int idNumber = 0;
13
14              Id()                                    { this.idNumber = -1;
15              Id(int idNumber)                        { this.idNumber = idNu
16
17              public int getId()                      { return idNumber; }
18              int compareIdObject2(Id a1, Id a2)      { return a1.idNumber -
19              int compareIdObject(Id a2)              { return a2.idNumber -
20              static int compareId(Id a1, Id a2)      { return a1.idNumber -
21              public String toString()                { return "" + idNumber
22      }
23      public class MethodReferences_2 {
24          public static void main(String args[]) {
25              ArrayList<Id> arrL = new ArrayList<Id>();
26              Id id1 = new Id(1);
27              Id id2 = new Id(2);
28              Id id3 = new Id(3);
29              arrL.add(id1); arrL.add(id2); arrL.add(id3);
30
31              arrL.sort( Id::compareId);
32                      System.out.println("Id::compareId:      ");
33                      arrL.forEach(n -> {  System.out.print(n); });
34
35              arrL.sort((a, b) -> Id.compareId(a, b) );
36                      System.out.println("\n(a, b) -> Id.compareId(a, b):");
37                      arrL.forEach(n -> {  System.out.print(n); });
38
39              arrL.sort( Id::compareIdObject);
40                      System.out.println("\nId::compareIdObject:");
41                      arrL.forEach(n -> {  System.out.print(n); });
42
43              arrL.sort( id1::compareIdObject2);
```

```
44                      System.out.println("\nid1::compareIdObject2:");
45                      arrL.forEach(n -> {  System.out.print(n); });
46
47          }
48      }
```

Source Code: Src/18_1/MethodReferences_2.java

18.10. Laziness Streams

What do we mean by laziness seeking in Streams?

19. Under Construction

20. Under Construction

21. Homework 1

Posted: June/3/2020
Due: August/25/2020 24.00

The solutions for this homework are due August/25/2020 24.00. I recommend to submit at least one version of all homework solutions long before due date.

21.1. Submission of your Homework

This course uses the myCourses submission functionality to allow students to electronically submit their work.

I suggest to submit often, meaning submit long before the deadline. We will only accept submitted code. You will receive 0 points for the hw, if your code is not submitted.

21.2. Programming Standard

Please make sure that your code is compatible with the Coding Standard (https://www.cs.rit.edu/~f2y-grd/java-coding-standard.html)

We will take points of, if your code does not follow this standard.

21.3. Teamwork

All work has to be submitted as a team of 2. You have to appear to the grading sessions on time. You have to select a grading slot during the first week. A schedule will be posted at the grad lab door at the beginning of the first week.

You will receive 0 points if you are late for your grading session.

The graders determine who answers the questions.

The signup sheet for your team can be found here: https://docs.google.com/spreadsheets/d/1So-jVaBhfyG_IiGCWTDM4wgazpbG_nTPVQGk61rdlf0/edit?usp=sharing (https://docs.google.com/spreadsheets/d/1So-jVaBhfyG_IiGCWTDM4wgazpbG_nTPVQGk61rdlf0/edit?usp=sharing)

21.4. Homework 1.1 (10 Points)

Objective: Compilation of a Java program, designing, implementing, and testing of an algorithm.

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

All homework are submitted as a team of 2.

Definition: Prime Number

A number p is a prime number if p has the following properties:

• p must be a integer

• $p > 1$ and

• the factors of p are 1 and itself.

The following program prints out all prime numbers in the range of [2 ... 10]:

```
 1      class Prime {
 2
 3        public static boolean isPrime(int n) {
 4
 5              for ( int index = 2; index < n; index ++ ) {
 6                      if ( n % index  == 0 )
 7                              return false;
 8              }
 9
10              return true;
11        }
12        public static void main( String args[] ) {
13          for ( int index = 2; index <= 10; index ++ )
14              if ( isPrime(index) )
15                      System.out.println(index + " " );
16        }
17      }
```

Source Code: Src/21/Prime.java

Your Work:

Modify *Prime.java* in such a way that is tests all numbers between n 3 and 73939233 for the following properties:

• n is a prime number

• Remove the right digit and and this number is also a prime number. Continue this process until there is only one digit left or the number is not a prime number.

• Your program has to print all numbers which have these properties.

Explanation:

An example for a number which has these properties is 233.

• 233 is a prime number

• 23 is a prime number

• 2 is a prime number

An example for a number which does not these properties is 397.

• 397 is a prime number

• 39 is not a prime number (3 * 13)

Example:

An example of a solution execution:

```
% java Prime
3 has the properties.
5 has the properties.
7 has the properties.
23 has the properties.
```

```
29 has the properties.
31 has the properties.
37 has the properties.
53 has the properties.
59 has the properties.
71 has the properties.
73 has the properties.
79 has the properties.
233 has the properties.
239 has the properties.
293 has the properties.
311 has the properties.
313 has the properties.
317 has the properties.
373 has the properties.
379 has the properties.
 ...
```

Idea for a Solution:

A statement like *thisNumber = (int)(thisNumber / 10)* cuts of the last digit.

Requirements:

• You have to name your program Prime.java

• You can only use basic arithmetic operations, casting, boolean expressions, print statements, Math.sqrt, basic types and native arrays.

• You can not use any publicly available class or library which can determine if a number is a prime number.

• Your program has to compute the properties of the number. In other words your program can not be something like:

```
 1      class PrimeWrong {
 2
 3        public static void main( String args[] ) {
 4              System.out.println("3 has the properties.");
 5              System.out.println("5 has the properties.");
 6              System.out.println("7 has the properties.");
 7              System.out.println("23 has the properties.");
 8              System.out.println("29 has the properties.");
 9              System.out.println("...");
10        }
11      }
```

Source Code: Src/21/PrimeWrong.java

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
1       public class Prime {
2
3           // final static int MAXIMUM = 73939133 + 1;
4           final static int MAXIMUM = 400;
5           final static int MINIMUM = 1;
6           final static boolean  thisNumberIsAprimeNumber[] = new boolean[ MA
7
8           private static boolean isItAPrimeNumber(int n)      {       // is
9               boolean rValue = true;
10              int index = 1;
11              if ( n <= 1 )
12                      return false;
13              if ( n == 2 )
14                      rValue = true;
15              while ( ( ++index <= ( (int)Math.sqrt(n) ) ) &&  rValue ) {
16                      if ( n % index == 0 )   {
17                          rValue = false;
18                      }
19              }
20              return rValue;
21          }
22          private static void initPrimeOrNotSieve()   {
23              for ( int index = 0; index < thisNumberIsAprimeNumber.length;
24                      thisNumberIsAprimeNumber[index] = false;
25
26              for ( int factor1 = 2; factor1 <=  ( (int)Math.sqrt(thisNumber
27                      for ( int factor2 = 2; factor2 * factor1 <   thisNumbe
28                              thisNumberIsAprimeNumber[factor1 * factor2 ] =
29                      }
30              }
31          }
32          private static boolean hasProperty(int thisNumber) {
33
34              while ( ( thisNumber > 1 ) && ! thisNumberIsAprimeNumber[thisN
35                      thisNumber = thisNumber / 10;
36              }
37              return ( thisNumber == 0 );
38          }
39          private static void findPrimesWithTheseProperties() {
40              initPrimeOrNotSieve();
41              // print();
42              for ( int index = MINIMUM; index < MAXIMUM; index +=2 ) {
43                      if ( ! thisNumberIsAprimeNumber[index] && hasProperty
44                              System.out.println(index + " has the propertie
45              }
46          }
47          private static void print() {
48              for ( int index = 0; index < thisNumberIsAprimeNumber.length;
49                      System.out.println(index + ":   " + thisNumberIsAprime
50              }
51          }
52
53          public static void main( String[] args ) {
```

```
54                new Prime().findPrimesWithTheseProperties();
55            }
56        }
57
58
```

Source Code: Src/21/PrimeSol.java

21.5. Homework 1.2 (10 Points)

Objective: Creating and implementing an algorithm.

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

This homework was inspired by having too many coins in my wallet. You have a given, fixed amount of coins and you can only pay the cashier with these coins, and the cashier accepts only the exact amount. If can have the exact amount you have to use the maximum number of coins possible to pay the bill.

The coins distribution will be initialized to the original state after each paying process.

Explanation:

Let's assume you have the following coins: { 1, 1, 2, 5, 25, 25, 25 } cents, and you have to pay 7 cents.

1 + 1 + 5 == 7, and

2 + 5 == 7.

The sequence { 1, 1, 5 } contains 3 coins and the sequence { 2, 5 } contains 2 coins. You have to use the sequence { 1, 1, 5 } to pay.

Let's assume you have the following coins: { 1, 1, 2, 5, 25, 25, 25 } cents, and you have to pay 11 cents. This is not possible.

Your Work:

Your work is to write a program which determines if a sequence of coins exist adding up to a given value. Your program has to print sequence, or the longest sequence if more than one sequence exist.

Assume you have the following coins: *coins = { 1, 1, 2, 5, 25, 25, 25 };* and you have to pay the following sums *toPay = { 0, 1, 4, 5, 7, 8 };*.

```
0 cents: can not be paid
1 cents: = 1 cents
4 cents: = 2 cents 1 cents 1 cents
5 cents: = 5 cents
7 cents: = 5 cents 1 cents 1 cents
8 cents: = 5 cents 2 cents 1 cents
```

Example:

An example of a solution execution:

Look at the following program snippet:

```
public class Coins {

    static int[] coins   = { 1, 1, 2, 5, 25, 25, 25 };
    static int[] toPay   = { 0, 1, 4, 5, 7, 8 };
    ...
}
```

My program produces the following output:

```
% java Coins
0 cents:            can not be paid
1 cents:            yes; used coins = 1 cents
4 cents:            yes; used coins = 2 cents 1 cents 1 cents
5 cents:            yes; used coins = 5 cents
7 cents:            yes; used coins = 5 cents 1 cents 1 cents
8 cents:            yes; used coins = 5 cents 2 cents 1 cents
```

Your program has to function correctly you change only the amount and values of the available coins.

If I change the values to, and this is the only change:

```
    static int[] coins = { 1, 5, 25, 25, 25 };
    static int[] toPay = { 0, 2, 30, 75 };
```

My program produces the following output::

```
% java Coins
0 cents:            can not be paid
2 cents:            no; can not be paid with the following sequence of coins:
30 cents:           yes; used coins = 25 cents 5 cents
75 cents:           yes; used coins = 25 cents 25 cents 25 cents
```

Idea for a Solution:

It might help to understand how to find all possible combinations of *n* items. For a moment assume the set is { 1, 2, 3 }.

{ } ∪ { 1 } = { 1 }
{ 1 } ∪ { 2 } = { 1, 2 }
{ 1 } ∪ { 3 } = { 1, 3 }
and so on. This is the staring point of a recursive solution.

Requirements:

• You have to name your program *Coins.java*

• Your program must produce the correct output, if the only the amount and values of the available coins are changed

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
1       /*
2        * Coinis = { c_1, c_2, ... c_n }
3        * k = | Coins | = (2 ^ n)  - 1
4        */
5
6       import java.util.Arrays;
7
8
9       public class Coins {
10
11          static int[] myCoins = { 0, 1, 1, 2, 5, 25, 25, 25 };
12          static int soManyCoins = myCoins.length;
13          static int[] toPay = { 1, 4, 5, 7, 8};
14          static int setSize          = (int)Math.pow(2,  myCoins.length );
15
16          private static void sortCoins() {
17              for (int index = 0; index < myCoins.length - 1; index++)      {
18                      for (int walker = 0; walker < myCoins.length - 1; walk
19                              if ( myCoins[walker] > myCoins[walker+1] )
20                                      int tmp = myCoins[walker];
21                                      myCoins[walker] = myCoins[walker + 1];
22                                      myCoins[walker+1] = tmp;
23                              }
24                      }
25              }
26          }
27          private static String printUsedCoins(int value) {
28              String returnValue = "";
29              for ( int index = soManyCoins; index >= 0 ; index --)   {
30                      if ( ( ( 1 << index ) & value ) == ( 1 << index ) )
31                              returnValue += myCoins[index] + " cents ";
32
33              }
34              if ( returnValue == "" )
35                      returnValue = "empty set";
36              return returnValue;
37          }
38
39          private static int soManyBitsAre1(int value) {
40              int returnValue = 0;
41              for ( int index = soManyCoins; index >= 0 ; index --)   {
```

```
42                      if ( ( ( 1 << index ) & value ) == ( 1 << index ) )
43                              returnValue ++;
44
45              }
46              return returnValue;
47          }
48
49          private static int calculteSumForThisSet(int value) {
50              int sum = 0;
51              for ( int index = soManyCoins - 1; index >= 0 ; index --)     {
52                      if ( ( ( 1 << index ) & value ) == ( 1 << index ) )
53                              sum += myCoins[index];
54                      }
55
56              }
57              return sum;
58          }
59          private static void  testIfaCombinationForThisSumExist(int thisSum
60              boolean foundAset       = false;
61              int largestSetSoFar     = 0;
62
63              int index = 0;                  // see comment in loop
64
65              if ( thisSum == 0 )     {
66                      System.out.println("0 cents:\t\tcan not be paid");
67              } else {
68
69                      while  ( index < setSize ) {
70                              int sum = calculteSumForThisSet(index);
71                              if ( ( thisSum == sum ) )       {
72                                      if ( soManyBitsAre1(largestSetSoFar) <
73                                              largestSetSoFar = index;
74                              }
75                              index ++;
76                      }
77                      if ( soManyBitsAre1(largestSetSoFar) > 0 )
78                              System.out.println(thisSum + " cents:   "
79                              "\tyes; used coins = "  + printUsedCoins(large
80                      else
81                              System.out.println(thisSum + " cents:   "
82                              "\tno; can not be paid with the following sequ
83              }
84          }
85          public static void main( String[] arguments ) {
86              // sortCoins();
87              for ( int index = 0; index < toPay.length; index ++ )
88                      testIfaCombinationForThisSumExist(toPay[index]);
89          }
90      }
```

Source Code: Src/21/CoinsSol.java

21.6. Homework 1.3 (10 Points)

Objective: Designing and implementing a simple algorithm and design a testing strategy.

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

You have to design a program which finds all numbers between 0, and 10000 who have the following property: A number n exists, so such the sum of each digit ^n is equal to the number. The number is a sum of 'nth' power of each digit of a n digit number is equal to n digit of a number

An example of a number which has this property is 153.

```
1^3 + 5^3 + 3^3  = 1 + 125 + 27 = 153.
```

1 also has this property.

Explanation:

This homework should be done using the following classes:
String (https://docs.oracle.com/en/java/javase/12/docs/api/java.base/java/lang/String.html)
Integer (https://docs.oracle.com/en/java/javase/12/docs/api/java.base/java/lang/Integer.html)
Math (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/lang/Math.html)

You should solve this homework using as much of the functionality provided by the classes mentioned above.

Your Work:

You should develop this program in stages. Start with the methods you can implement and test first. Develop a test scenario before you develop the method.

My solution produces the following output for the above variables.

```
% java Numbers
  1     ==      1     has the desired property
       1 ^ 1
  2     ==      2     has the desired property
       2 ^ 1
  3     ==      3     has the desired property
       3 ^ 1
  4     ==      4     has the desired property
       4 ^ 1
  5     ==      5     has the desired property
       5 ^ 1
  6     ==      6     has the desired property
       6 ^ 1
  7     ==      7     has the desired property
       7 ^ 1
  8     ==      8     has the desired property
```

```
     8 ^ 1
9     ==      9     has the desired property
     9 ^ 1
153   ==      153   has the desired property
     1 ^ 3 + 5 ^ 3 + 3 ^ 3
370   ==      370   has the desired property
     3 ^ 3 + 7 ^ 3 + 0 ^ 3
371   ==      371   has the desired property
     3 ^ 3 + 7 ^ 3 + 1 ^ 3
407   ==      407   has the desired property
     4 ^ 3 + 0 ^ 3 + 7 ^ 3
```

Your solution should produce a similar output. You output should be adjusted similar to my output.

Requirements:

You have to name the file *Numbers.java*.

Idea for a Solution:

No hint

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
 1      // andersons numbers
 2      import java.lang.Math;
 3
 4      public class Numbers {
 5
 6          final static int MINIMUM = 1;       // 0 is excluded
 7          final static int MAXIMUM = 100000000;
 8          static int[]  theDigits;
 9
10
11          private static void printDigits()   {
12                  for ( int index = 0; index < theDigits.length; index ++ )
13                          System.out.println(index + ":= " + theDigits[index
14          }
15          private static void extractDigits(int thisNumber) {
16              theDigits = new int[ ( "" + thisNumber ).length() ];
17              int index = 0;
18              if ( theDigits.length > 1 )     {
19                      do {
20                              theDigits[index++ ] = thisNumber % 10;
21                              thisNumber = thisNumber / 10;
22                      } while  ( thisNumber % 10 != thisNumber        );
23              }
24              theDigits[index++] = thisNumber;
25          }
```

```
26          private static boolean includesOnlyZerosAndOnes()   {
27              boolean rValue = true;
28              for ( int index = 0; rValue && ( index < theDigits.length); in
29                      rValue &= ( theDigits[index] <= 1 );
30              }
31              return rValue;
32          }
33          private static void printResult(int thisNumber, int sumOfPower, in
34              String format = "%" + ( theDigits.length + 2 ) + "d\t%s\t%" +
35              System.out.printf(format, thisNumber, " == ", sumOfPower, " ha
36              System.out.print("      ");
37              for ( int index = 0; index < theDigits.length; index ++ )
38                      System.out.print(theDigits[theDigits.length - index -
39                      if ( index < theDigits.length - 1 )
40                              System.out.print(" + " );
41              }
42              System.out.println();
43          }
44          // the 'do while' loop will never terminate, if the number include
45          private static void testNotZeroAndOnes(int thisNumber) {
46              int power = 0;
47              int sumOfPower = 0;
48              do {
49                      sumOfPower = 0;
50                      for ( int index = 0; index < theDigits.length; index +
51
52                              sumOfPower += Math.pow( (double)theDigits[inde
53                      }
54                      power ++;
55              } while  ( sumOfPower <  thisNumber );
56
57              if ( sumOfPower == thisNumber )
58                      printResult(thisNumber, sumOfPower, power - 1);
59          }
60
61          private static void hasProperty(int thisNumber) {
62              extractDigits(thisNumber);
63
64              if (!  includesOnlyZerosAndOnes() )
65                      testNotZeroAndOnes(thisNumber);
66          }
67          private static void findNumbersWithTheseProperties() {
68              for ( int index = MINIMUM; index < MAXIMUM; index ++ )
69                      hasProperty(index);
70          }
71
72          public static void main( String[] args ) {
73              new Numbers().findNumbersWithTheseProperties();
74          }
75      }
76
77
```

Source Code: Src/21/NumbersSol.java

22. Homework 2

NOTE: From now on you have to submit all files for your solutions for all parts of a give home work in one zip file. The name of the zip file has to be: *your_login_name.gz, or your_login_name.gzip*

Posted: June/10/2020
Due: September/1/2020 24.00

The solutions for this homework are due September/1/2020 24.00. I recommend to submit at least one version of all homework solutions long before due date.

22.1. Homework 2.1 (10 Points)

Objective: Modifying of an existing algorithm.

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

The solution for hw 1.2 needs to be modified for this homework. The objective is to have the least amount of coins in your wallet after the transaction is completed. The modification of the problem is that the cashier can also give you coins back, in other words you do not need the exact amount.

Explanation:

Assuming you have to pay 8 cents, and the coins in your wallet are:

{ 1, 1, 1, 1, 1, 5 }

and the cashier has the following coin:

{ 2 }

If you pay the cashier { 1, 1, 1, 5 } you will have { 1, 1 }; 2 coins in your wallet.

If you pay the cashier { 1, 1, 1, 1, 1, 5 } you will have { 2 }; 1 coin in your wallet, because the cashier gives you one coin with the value of 2 cents back.

Your Work:

Your work is to write a program to achieve the objective. The objective is to have the least amount of coins in your wallet after the transaction is completed. The modification of the problem is that the cashier can also give you coins back, in other words you do not need the exact amount.

Look at the snippet of this program:

```
public class Coins {

    static int[] coinsInMyWallet = { 1, 1, 1, 1, 5, 5, 10, 10, 10 };
    static int[] cashiersCoins = { 2  };
    static int[] toPay = { 7 };
```

My program produces the following output:

```
7 cents:  I gave the cashier the following coins 5 cents 1 cents 1 cents 1 cen
          and the cashier gave me 2 cents
```

Your program must work correctly if you change only the values of the coins you have, the coins the cashier has, and the amount you have to pay.

Idea for a Solution:

One hint: How about the wallets the cashier has, are negative values in your wallet?

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
 1
 2      import java.util.Arrays;
 3
 4      public class Coins {
 5
 6          static int[] coinsInMyWallet =  { 1, 1, 1, 1, 5, 5, 10, 10, 10};
 7          static int[] cashiersCoins = { 1, 1, 5};
 8          static int[] toPay = { 23 };
 9          // static int[] coinsInMyWallet = {1, 5, 1, 1, 1 };
10          // static int[] cashiersCoins = { 1, 2, 3 };
11          // static int[] toPay = { 9, 6, 7, 8 };
12
13              // cashier and my coins, but the chasier coins will be added a
14          static int[] myCoins = new int[coinsInMyWallet.length + cashiersCo
15          static int soManyCoins = myCoins.length;
16
17          static String myCoinsInAstring;
18
19          private static void printMyCoins() {
20              for ( int index = 0; index < myCoins.length; index ++ )
21                      System.out.println(index + " " + myCoins[index]);
22          }
23          private static void initMyCoins() {
24              for ( int index = 0; index < coinsInMyWallet.length; index ++
25                      myCoins[index] = coinsInMyWallet[index];
26              for ( int index = 0; index < cashiersCoins.length; index ++ )
27                      myCoins[index + coinsInMyWallet.length] = -cashiersCoi
28              myCoinsInAstring = Arrays.toString(myCoins);
29              myCoinsInAstring = myCoinsInAstring.replace("[", " ");
30              myCoinsInAstring = myCoinsInAstring.replace("]", " ");
31
32          }
33          private static void sortCoins() {
34              for (int index = 0; index < myCoins.length - 1; index++)     {
35                      for (int walker = 0; walker < myCoins.length - 1; walk
36                              if ( myCoins[walker] > myCoins[walker+1] )
37                                      int tmp = myCoins[walker];
```

```
38                                      myCoins[walker] = myCoins[walker + 1];
39                                      myCoins[walker+1] = tmp;
40                              }
41                      }
42              }
43          }
44          private static String printUsedCoins(int value) {
45              String cashiersCoinsReturnValue = "";
46              String myCoinsReturnValue = "";
47
48              for ( int index = soManyCoins; index >= 0 ; index --)   {
49                      if ( ( ( 1 << index ) & value ) == ( 1 << index ) )
50                              if ( myCoins[index] > 0 )
51                                      myCoinsReturnValue += myCoins[index] +
52                              else
53                                      cashiersCoinsReturnValue += ( -1 * myC
54
55              }
56              if ( myCoinsReturnValue == "" )
57                      myCoinsReturnValue = "empty set";
58              if ( cashiersCoinsReturnValue == "" )
59                      cashiersCoinsReturnValue = "nothing";
60              return myCoinsReturnValue + "\n\t\tand the cashier gave me " +
61          }
62
63          private static int soManyBitsAre1(int value) {
64              int returnValue = 0;
65              for ( int index = soManyCoins; index >= 0 ; index --)   {
66                      // coins in my wallet from cashier do not count in orc
67                      // number of coins in my wallet
68                      if ( ( ( ( 1 << index ) & value ) == ( 1 << index ) )
69                              returnValue ++;
70                      }
71              }
72              return returnValue;
73          }
74
75          private static int calculteSumForThisSet(int value) {
76              int sum = 0;
77              for ( int index = soManyCoins - 1; index >= 0 ; index --)    {
78                      if ( ( ( 1 << index ) & value ) == ( 1 << index ) )
79                              sum += myCoins[index];
80                      }
81
82              }
83              return sum;
84          }
85          private static void  testIfaCombinationForThisSumExist(int thisSum
86              int setSize             = (int)Math.pow(2,  myCoins.length );
87              boolean foundAset       = false;
88              int largestSetSoFar     = 0;
89
90              int index = 0;                  // see comment in loop
91
```

```
92              if ( thisSum == 0 )     {
93                      System.out.println("0 cents:\t\tcan not be paid");
94              } else {
95
96                      while  ( index < setSize ) {    // set must be smaller
97                              int sum = calculteSumForThisSet(index);
98                              if ( ( thisSum == sum ) )       {
99                                      if ( soManyBitsAre1(largestSetSoFar) <
00                                              largestSetSoFar = index;
01                              }
02                              index ++;
03                      }
04                      if ( soManyBitsAre1(largestSetSoFar) > 0 )
05                              System.out.println(thisSum + " cents:   "
06                              "I gave the cashier the following coins "
07                      else
08                              System.out.println(thisSum + " cents:   "
09                              "no; can not be paid with the following sequer
10              }
11          }
12          public static void main( String[] arguments ) {
13              initMyCoins();
14              sortCoins();    // not needed but produces sorted output
15              for ( int index = 0; index < toPay.length; index ++ )
16                      testIfaCombinationForThisSumExist(toPay[index]);
17          }
18      }
```

Source Code: Src/22/Coins.java

22.2. Homework 2.2 (10 Points)

Objective: Designing and implementing a algorithm.

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

A palindrome (https://en.wikipedia.org/wiki/Palindrome)
is a series of characters which reads backwards and forwards the same.

A Lychrel number, (https://en.wikipedia.org/wiki/Lychrel_number)
is a natural number that cannot form a palindrome through the iterative process of repeatedly reversing its digits and adding the resulting numbers.

56 is not a Lychrel number because it becomes palindromic after 1 iteration:
56 + 65 = 121. 121 is a palindrome

57 is not a Lychrel number because it becomes palindromic after 2 iterations:
57 + 75 = 132; 132 + 231 = 363. 363 is a palindrome.

59 is not a Lychrel number because it becomes palindromic after 3 iterations:
59 + 95 = 154; 154 + 451 = 605; 605 + 506 = 1111. 1111 is a palindrome.

If a number is not a Lychrel number, it is called a "delayed palindrome"
56 is delayed by 1,

You have to design and implement an algorithm to determine if a number becomes palindromic within up to 3 interations.

78 does not bebecomes palindromic within 3 iterations.
78 + 87 = 165; 165 + 561 = 726; 726 + 627 = 1353; 1353 != 3531

Explanation:

This homework should be done using the following classes:
StringBuilder (https://docs.oracle.com/en/java/javase/12/docs/api/java.base/java/lang/StringBuilder.html)
String (https://docs.oracle.com/en/java/javase/12/docs/api/java.base/java/lang/String.html)
Integer (https://docs.oracle.com/en/java/javase/12/docs/api/java.base/java/lang/Integer.html)

You should solve this homework using as much of the functionality provided by the classes mentioned above.

Your Work:

Please take a look at the following program snippet:

```
public class Palindrome {
        static int START               = 69;
        static int MAXIMUM             = 91;
        static int MAXIMUM_DELAYED     = 3;
 ...
```

My solution produces the following output for the above variables.

```
% java Palindrome
69:          delayed 3:   does not become palindromic within 3 iterations (72
70:          delayed 1:   70     +      07     =      77
71:          delayed 1:   71     +      17     =      88
72:          delayed 1:   72     +      27     =      99
73:          delayed 2:   110    +      011    =      121
74:          delayed 1:   74     +      47     =      121
75:          delayed 2:   132    +      231    =      363
76:          delayed 2:   143    +      341    =      484
77:          delayed 3:   605    +      506    =      1111
78:          delayed 3:   does not become palindromic within 3 iterations (72
79:          delayed 3:   does not become palindromic within 3 iterations (84
80:          delayed 1:   80     +      08     =      88
81:          delayed 1:   81     +      18     =      99
82:          delayed 2:   110    +      011    =      121
83:          delayed 1:   83     +      38     =      121
84:          delayed 2:   132    +      231    =      363
85:          delayed 2:   143    +      341    =      484
86:          delayed 3:   605    +      506    =      1111
87:          delayed 3:   does not become palindromic within 3 iterations (72
88:          delayed 3:   does not become palindromic within 3 iterations (84
89:          delayed 3:   does not become palindromic within 3 iterations (96
90:          delayed 1:   90     +      09     =      99
```

Your solution should produce a similar output.

Requirements:

You have to name the file *Palindrome.java*.

Idea for a Solution:

No hint

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
 1      import java.lang.StringBuilder;
 2
 3      public class Palindrome {
 4              static int END                  = 170;
 5              static int MAXIMUM_DELAYED      = 60; // 20
 6
 7
 8              public static int  delayedPalindrome(int firstNumberofSequence
 9                      boolean isDelayed       = false;
10                      int rValue = 0;
11
12                      String  original = Integer.toString(theNumber);
13                      StringBuilder aStringBuilder = new StringBuilder(origi
14                      String  lanigiro = aStringBuilder.reverse().substring
15                      int     rebmuNeht   = Integer.valueOf(lanigiro);
16                      int result = theNumber + rebmuNeht;
17                      String resultString  = Integer.toString(result);
18                      String resultStringReverse  = new StringBuilder(result
19                      rValue =  theNumber + rebmuNeht;                // mig
20                      String leftPadding = "";
21                      if  ( ( (theNumber / 10 ) * 10 ) == theNumber )
22                              leftPadding = "0";
23                      boolean notAlychrelNumber = ( ( delayed <= MAXIMUM_DEI
24                      boolean lastTry           = ( ( delayed == MAXIMUM_DEI
25                      if  ( notAlychrelNumber || lastTry ) {
26                              System.out.print(firstNumberofSequence + ":
27                              System.out.print("\tdelayed " + delayed);
28                              if  ( notAlychrelNumber )       {
29
30                                      System.out.println(":   " + theNumber
31                                      rValue = 0;   //  BigInt
32                              } else {
33                                      System.out.println(":   does not becom
34                                              + rebmuNeht + " = "  + result
35                                              + resultString + " != " + res
36                              }
37                      }
38                      return rValue;
39              }
```

```
40              public static void main( String args[] ) {
41                              int theNextnumber = 1;
42                              for ( int delayed = 1; (theNextnumber > 0 ) &&
43                                                                      // num
44                                      theNextnumber = delayedPalindrome(numb
45                              }
46                      }
47              }
48      }
```

Source Code: Src/22/Palindrome.java

BigInt solution:

```
 1      /*
 2       * From: https://en.wikipedia.org/wiki/Lychrel_number
 3       * In the 1980s, the 196 palindrome problem attracted the attention of
 4       * with search programs by Jim Butterfield and others appearing in sev
 5       * In 1985 a program by James Killman ran unsuccessfully for over 28 d
 6       * time java %
 7       * real 0m7.714s
 8       * user 0m15.951s
 9       * sys  0m0.516s
10       */
11      import java.lang.StringBuilder;
12      import java.math.BigInteger;
13
14      public class PalindromeBigInt {
15      /*
16              static int END                  = 196;
17              static int MAXIMUM_DELAYED      = 12955;
18      */
19      /*
20              static BigInteger END                   = new BigInteger("1186
21              static int MAXIMUM_DELAYED      = 12955;
22      */
23              static BigInteger END           = new BigInteger("170");
24              static int MAXIMUM_DELAYED      = 60;
25      /*
26              From Palindrom.java
27              static int END                  = 170;
28              static int MAXIMUM_DELAYED      = 60; // 20
29              // Exception in thread "main" java.lang.NumberFormatException:
30      */
31
32              public static BigInteger  delayedPalindrome(BigInteger firstNu
33                      BigInteger rValue = new BigInteger("0");
34
35                      String      original    = theNumber.toString();
36                      BigInteger  rebmuNeht   = new BigInteger(new StringBui
37
38                      String resultString  = theNumber.add(rebmuNeht).toStri
39                      String resultStringReverse  = new StringBuilder(result
40                      rValue =  theNumber.add(rebmuNeht);             // mig
```

```
41                      // String leftPadding =  ( ( (theNumber / 10 ) * 10 )
42                      String leftPadding =  "";
43
44                      boolean notAlychrelNumber = ( ( delayed <= MAXIMUM_DEI
45                      boolean lastTry           = ( ( delayed == MAXIMUM_DEI
46                      if  ( notAlychrelNumber || lastTry ) {
47                              System.out.print(firstNumberofSequence + ":
48                              System.out.print("\tdelayed " + delayed);
49                              if  ( notAlychrelNumber )       {
50                                      System.out.println(":   " + theNumber
51                                      rValue = new BigInteger("0");   // -1
52                              } else {
53                                      System.out.println(":   does not becom
54                                              + rebmuNeht + " = "  + result
55                                              + resultString + " != " + res
56                              }
57                      }
58                      return rValue;
59              }
60              public static void main( String args[] ) {
61                      BigInteger end = new BigInteger("0");
62                      while (  ! numberToTestBigInteger.equals(END) ) {
63                              BigInteger theNextnumber = new BigInteger("1")
64                              int  delayed = 1;
65                              do {
66                                      theNextnumber = delayedPalindrome(numb
67                              } while  ( (! theNextnumber.equals(end) ) &&
68                              numberToTestBigInteger = numberToTestBigIntege
69                      }
70              }
71      }
```

Source Code: Src/22/PalindromeBigInt.java

22.3. Homework 2.3 (10 Points)

Objective: Understanding strings and string literals.

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Look at the following program:

```
 1      class StringThing {
 2        public static String method1()        {
 3              return "123";
 4        }
 5        public static String method2()        {
 6              return method1();
 7        }
 8        public static void main( String args[] ) {
 9              int counter = 0;
10              int theNumberThree = 3;
11      // first block begins           ////////////////////////////////////
12              String aString = "123";
13              String bString = "123";
14              String cString = "12" + "3";
15              String dString = "12" + theNumberThree;
16              String eString = "123" + aString;
17              String fString = "123123";
18
19              System.out.println("" + ++counter + ".  " +   ( aString  == bS
20              System.out.println("" + ++counter + ".  " +   ( bString  == cS
21              System.out.println("" + ++counter + ".  " +   ( cString  == dS
22              System.out.println("" + ++counter + ".  " +   ( eString  == fS
23      // first block ends             ////////////////////////////////////
24
25      // second block begins          ////////////////////////////////////
26              String aaString = "123";
27              String bbString = new String("123");
28              String ccString = new String(aString);
29              String ddString = method1();
30              String eeString = method2();
31              String ffString = method1() + method2();
32
33              System.out.println("" + ++counter + ".  " +   ( aaString  == b
34              System.out.println("" + ++counter + ".  " +   ( aaString  == c
35              System.out.println("" + ++counter + ".  " +   ( aaString  == d
36              System.out.println("" + ++counter + ".  " +   ( aaString  == e
37              System.out.println("" + ++counter + ".  " +   ( ffString  == "
38      // second block ends            ////////////////////////////////////
39
40        }
41      }
```

`Source Code: Src/23/StringThing.java`

Explanation:

You should not run the code in order to conclude the answer. You should use the knowledge you have, without executing the code. Java evaluates expressions strictly from left to right. Keep this in mind when you think about line a). *"1." + "x" == "x"* becomes *"1.x" == "x"*, which is false. The compiler will create literals if possible.

Your Work:

You have to draw the memory model we use in class to explain the output of the program. Keep in mind that time is in issue. The lines in the writeup are numbered. You should use the lines to describe the order how the memory model changes. I suggest you draw a memory model for the first and second block. the 4 lines

Idea for a Solution:

It might help to understand: https://docs.oracle.com/javase/specs/jls/se11/html/jls-3.html#jls-3.10.5 (https://docs.oracle.com/javase/specs/jls/se11/html/jls-3.html#jls-3.10.5)

Requirements:

- You have to name your file *hw2.3.pdf*.

Submission:

Submit your files via myCourses.

23. Homework 3

Posted: June/4/2018
Due: September/8/2020 24.00

The solutions for this homework are due September/8/2020 24.00. I recommend to submit at least one version of all homework solutions long before due date.

23.1. Homework 3.1 (10 Points)

Objective: Understanding of Java Class documentation.

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

You should read and understand the following java classes before you start this homework:

- String, (https://docs.oracle.com/en/java/javase/12/docs/api/java.base/java/lang/String.html)
- Integer, and (https://docs.oracle.com/en/java/javase/12/docs/api/java.base/java/lang/Integer.html)
- Pattern, and (https://docs.oracle.com/en/java/javase/12/docs/api/java.base/java/util/regex/Pattern.html)
- Arrays. (https://docs.oracle.com/en/java/javase/12/docs/api/java.base/java/util/Arrays.html)

Your Work:

Given are the following declarations:

```
String aString = "2513";
String bString = "2513";
String cString = "ABCDEFG";
String dString = "abcDEFG";
String eString = aString + ( bString + cString ) + dString;
```

You have to write a program which performs the following operation I to X. You can use the classes: String, Integer, Arrays, and Pattern and a Arrays. You are not allowed to use control structures like loops, or if-then-else. You are allowed to use additional variables.

I. Determine if *aString* and *bString* are identical.
Answer: True

II. Determine if *cString* and *dString* are identical ignoring case differences.
Answer: True

III. Cutting out the the first character of *aString* and print it.
Answer: '2'

IV. Cut out the the last two characters of *aString* and print it.
Answer: '13'

V. Cut out the string of *cString* from the beginning of the string to the first occurrence of 'C' including 'C' and print it.
Answer: ABC

VI. Cut out of *dString* from the first digit in *aString* and the last digit in *aString* and print it.
Answer: c

VII. Sort the characters in *aString* and cut out from *dString* beginning with the lowest number to the second lowest number.

Answer: b

VIII. Print *eString* in such a ways that the output is sorted.
Answer: 11223355ABCCCDDEEFFGGabc

IX. Determine if *aString* is part of *eString*.
Answer: true

X. Determine if *dString* is part of *cString* ignoring case differences.
Answer: true

A snippet of my program:

```
import java.util.Arrays;

class StringIntegerArrays_1 {

  public static void main( String args[] ) {
        String aString = "2513";                                    // 1
        String bString = "2513";                                    // 2
        String cString = "ABDECFG";                                 // 3
        String dString = "abDECFG";                                 // 4
        String eString = aString + ( bString + cString ) + dString; // 5
        ...

  }
}
```

My program produces the following output:

```
% java StringIntegerArrays
I.     true
II.    true
III.   2
IV.    13
V.     ABC
VI.    c
VII.   b
VIII.  11223355ABCCCDDEEFFGGabc
IX.    true
X.     true
```

Let's assume the only modification of the programs are the lines marked 1 - 5 are changed to:

```
aString = "215";
bString = "13";
cString = "123ABCDECFGT";
dString = "123abcDECFG";
eString = bString + "10293847465" + ( bString + cString ) + dString;
```

The output of the otherwise unmodified program is now:

```
% java StringIntegerArrays useSecondSet
I.     false
II.    false
III.   2
IV.    15
V.     123ABC
VI.    3ab
VII.   b
VIII.  011111222333334456789ABCCCDDEEFFGGTabc
IX.    false
X.     true
```

A snippet of my code:

```
import java.util.Arrays;

class StringIntegerArrays {

  public static void main( String args[] ) {
        String aString;
        String bString;
        String cString;
        String dString;
        String eString;

        if ( args.length == 0 ) {
                aString = "2513";
                bString = "2513";
                cString = "ABCDECFG";
                dString = "abcDECFG";
                eString = aString + ( bString + cString ) + dString;
        } else {
                aString = "213";
                bString = "2513";
                cString = "ABCDECFGT";
                dString = "abcDECFG";
                eString = bString + ( bString + cString ) + dString;
        }
        ...
```

Your program has to behave in the same way.

Example:

An example of a solution execution:

```
% javac StringIntegerArrays.java
% java StringIntegerArrays
I.    true
II.   true
III.  2
IV.   13
V.    ABDEC
```

```
VI.   D
VII.  5
VIII. 11223355ABCCCDDEEFFGGabc
IX.   true
X.    true

% java StringIntegerArrays useSecondSet
I.    false
II.   false
III.  2
IV.   13
V.    ABC
VI.   c
VII.  1
VIII. 11223355ABCCCDDEEFFGGTabc
IX.   false
X.   true
page 3
```

Requirements:

- You have to name the file: StringIntegerArrays.java.
- You can not hardcode the *from, to* values for VI, and VII, in your program.
- You can use only the classes: String, Integer, Arrays, and Pattern and a character array.
- You are not allowed to use control structures like loops, or if-then-else.
- You are allowed to use additional variables.

Idea for a Solution:

Most methods you need to use are obvious. It is important to remember you can convert a String to an Array.

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
 1      //
 2      // there is no checking if the values for
 3      //      Cut out of dString from the first digit in aString and the las
 4      // and other requirements are valid. This hw is bout String operations
 5      //
 6
 7      import java.util.Arrays;
 8
 9      class StringIntegerArrays {
10      //  char[] anArray = aString.toCharArray();
11      //  Arrays.sort(anArray);
12
13        public static void main( String args[] ) {
14              String aString;
```

```
15              String bString;
16              String cString;
17              String dString;
18              String eString;
19
20              if ( args.length == 0 ) {
21                      aString = "2513";
22                      bString = "2515";
23                      cString = "ABCDECFG";
24                      dString = "abDECFG";
25                      eString = aString + ( bString + cString ) + dString;
26              } else {
27                      aString = "213";
28                      bString = "2513";
29                      cString = "ABCDECFGT";
30                      dString = "abcDECFG";
31                      eString = bString + "10293847465" + ( bString + cStrin
32              }
33
34              System.out.println("I.  " +     ( bString.equals(aString) )
35              System.out.println("II. " +     ( 0 == cString.compareToIgnore
36              System.out.println("III.       " +     aString.substring(0, 1
37              System.out.println("IV. " +     aString.substring(aString.leng
38              System.out.println("V.  " +     cString.substring( 0, cString.
39              int left  = Integer.parseInt( aString.substring(0, 1 ) );
40              int right = Integer.parseInt(  aString.substring(aString.lengt
41              System.out.println("VI. " +     dString.substring(left, right
42              char[] anArray = aString.toCharArray();
43              Arrays.sort(anArray);
44              System.out.println("VII.       " +     dString.substring(anAr
45              anArray = eString.toCharArray();
46              Arrays.sort(anArray);
47              System.out.println("VIII.      " +     new String(anArray) );
48              System.out.println("IX. " +     eString.matches(".*" + aString
49              System.out.println("X.  " +     cString.toUpperCase().matches
50        }
51      }
```

Source Code: Src/23/StringIntegerArrays.java

23.2. Homework 3.2 (10 Points)

Objective: Working with the Scanner Class.

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Your assignment is to create a word guessing program using the Scanner class.

Implement a version of the hangman game. (https://en.wikipedia.org/wiki/Hangman_(game))

You have to create your own dictionary file. Your program has to read a dictionary, which consists of one word per line and a picture file. Then the program has to pick a random word and the player has 9 tries to guess all characters of the word. A word can only be selected once. One round of the game is over, when the every character of the word has been guessed. The game ends when all words have been selected. At the beginning every character of the word is replaced with a ".". Your program aks from the user a guess and if the user guesses correctly one "." will be replaced with the character. Every correct guess will show more of the picture.

With every correct guess more of a pciture is shown.

Explanation:

You have to write a program which allows to specify on the command line a file which contains the words and a picture. One way to call my program is *java Picture -words words.txt -picture batman.txt*.

You can use the Vector class and the classes we have covered in class for your solution.

https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/Vector.html (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/Vector.html)

My words file looks like:

```
1       aaa
2       abcd
```

Source Code: Src/23/words.txt

and my picture file looks like:

```
 1      11111111111111111111111111
 2      11111100111111111100111111
 3      11110001111100111110001111
 4      11000001111000011110000011
 5      10000000111000011100000001
 6      10000000000000000000000001
 7      00000000000000000000000000
 8      00000000000000000000000000
 9      10000000000000000000000001
10      10000110001000010001100001
11      11001111111100111111110011
12      11100111111100111111100111
13      11111111111111111111111111
```

Source Code: Src/23/batman.txt

I used this tool (https://manytools.org/hacker-tools/convert-images-to-ascii-art/go/) to create images like

```
1
2                                          .,,,,,,,,,.
3                      .........,,,,,,,,,,,,*%@@@@@@@@@@&*,,,,,,
4                   ,,/@@@@&&&@@@@@@@@@@@@@@%//@(///(%((/(@@@@@@,,,,,,,
5                   ,,@@@@*.,****,,. //@/////////////*, */////@@(@@@@@,,,,
6                    ,,(@@@@@@@@@@@@@&#///////*..,#@@@@@@( */////////%@@*,,
```

Source Code: Src/23/ritTiger.txt

The structure of both files are a suggestion.

My program prints every *int printEverXthWord = (lengthOfWord - soManyCorrectGuesses) + 1;* character.

An execution of my program looks like this. Comments have been added to explain the execution. A comment starts with '#'.

```
% java Picture -words words.txt -picture batman.txt
       ...1...1...1...1...1...1..  # lengthOfWord = 3
       .1...1...1...1...1...1...1  # soManyCorrectGuesses = 0
       ...1...1...1...1...0...1..  # every 4 character is shown
       .1...0...1...0...1...0...1
       ...0...0...0...1...0...0..
       .0...0...0...0...0...0...1
       ...0...0...0...0...0...0..
       .0...0...0...0...0...0...0
       ...0...0...0...0...0...0..
       .0...1...0...0...0...0...1
       ...0...1...1...1...1...0..
       .1...1...1...0...1...0...1
       ...1...1...1...1...1...1..
       # words are "aaa" and "abcd"
       # the word selected was "aaa"
       # and this word is now removed
0: ... # the word to guess is "aaa"
w
       ...1...1...1...1...1...1..
       .1...1...1...1...1...1...1
       ...1...1...1...1...0...1..
```

```
        .1...0...1...0...1...0...1
        ...0...0...0...1...0...0..
        .0...0...0...0...0...0...1
        ...0...0...0...0...0...0..
        .0...0...0...0...0...0...0
        ...0...0...0...0...0...0..
        .0...1...0...0...0...0...1
        ...0...1...1...1...1...0..
        .1...1...1...0...1...0...1
        ...1...1...1...1...1...1..
1: ... # "w" was a wrong guess, 0 has changed to 1
a
        ..1..1..1..1..1..1..1..1..  # lengthOfWord = 3
        1..1..0..1..1..1..0..1..1.  # soManyCorrectGuesses = 1
        .1..0..1..1..0..1..0..1..1  # every 3 character is shown
        ..0..0..1..0..0..1..0..0..
        1..0..0..1..0..1..0..0..0.
        .0..0..0..0..0..0..0..0..1
        ..0..0..0..0..0..0..0..0..
        0..0..0..0..0..0..0..0..0.
        .0..0..0..0..0..0..0..0..1
        ..0..1..0..0..0..0..1..0..
        1..0..1..1..0..1..1..1..1.
        .1..0..1..1..0..1..1..0..1
        ..1..1..1..1..1..1..1..1..
1: a.. # an "a" was correct
a
        .1.1.1.1.1.1.1.1.1.1.1.1.1
        .1.1.1.0.1.1.1.1.1.0.1.1.1
        .1.1.0.1.1.1.0.1.1.0.0.1.1
        .1.0.0.1.1.0.0.1.1.0.0.0.1
        .0.0.0.0.1.0.0.1.1.0.0.0.1
        .0.0.0.0.0.0.0.0.0.0.0.0.1
        .0.0.0.0.0.0.0.0.0.0.0.0.0
        .0.0.0.0.0.0.0.0.0.0.0.0.0
        .0.0.0.0.0.0.0.0.0.0.0.0.1
        .0.0.1.0.0.0.0.1.0.1.0.0.1
        .1.0.1.1.1.1.1.0.1.1.1.1.0.1
        .1.0.1.1.1.1.1.0.1.1.1.0.1.1
        .1.1.1.1.1.1.1.1.1.1.1.1.1.1
1: aa. # an "a" was correct
a
        11111111111111111111111111
        11111100111111111100111111
        11110001111100111110001111
        11000001111000011110000011
        10000000111000011100000001
        10000000000000000000000001
        00000000000000000000000000
        00000000000000000000000000
        10000000000000000000000001
        10000110001000010001100001
        11001111111100111111110011
        11100111111100111111100111
```

```
       1111111111111111111111111
       # the word has been guessed and all characters
       # of the picture are shown
The word was: aaa

       # new round
       # a new word has been selected
       # the word is "abcd"
       # no word left to select from
       # this is the last round
       ....1....1....1....1....1.
       ...1....1....1....0....1..
       ..1....1....0....1....1...
       .1....0....0....1....0....
       1....0....1....1....0....1
       ....0....0....0....0....0.
       ...0....0....0....0....0..
       ..0....0....0....0....0...
       .0....0....0....0....0....
       1....1....1....1....1....1
       ....1....1....1....1....1.
       ...0....1....0....1....1..
       ..1....1....1....1....1...
0: .... # the word is "abcd"
       # "aaa" was removed, there is no
       # more left
a      # keyboard input
       ...1...1...1...1...1...1..
       .1...1...1...1...1...1...1
       ...1...1...1...1...0...1..
       .1...0...1...0...1...0...1
       ...0...0...0...1...0...0..
       .0...0...0...0...0...0...1
       ...0...0...0...0...0...0..
       .0...0...0...0...0...0...0
       ...0...0...0...0...0...0..
       .0...1...0...0...0...0...1
       ...0...1...1...1...1...0..
       .1...1...1...0...1...0...1
       ...1...1...1...1...1...1..
0: a...
b
       ..1..1..1..1..1..1..1..1..
       1..1..0..1..1..1..0..1..1.
       .1..0..1..1..0..1..0..1..1
       ..0..0..1..0..0..1..0..0..
       1..0..0..1..0..1..0..0..0.
       .0..0..0..0..0..0..0..0..1
       ..0..0..0..0..0..0..0..0..
       0..0..0..0..0..0..0..0..0.
       .0..0..0..0..0..0..0..0..1
       ..0..1..0..0..0..0..1..0..
       1..0..1..1..0..1..1..1..1.
       .1..0..1..1..0..1..1..0..1
```

```
       ..1..1..1..1..1..1..1..1..
0: ab.. incorrect guess
w
       ..1..1..1..1..1..1..1..1..
       1..1..0..1..1..1..0..1..1.
       .1..0..1..1..0..1..0..1..1
       ..0..0..1..0..0..1..0..0..
       1..0..0..1..0..1..0..0..0.
       .0..0..0..0..0..0..0..0..1
       ..0..0..0..0..0..0..0..0..
       0..0..0..0..0..0..0..0..0.
       .0..0..0..0..0..0..0..0..1
       ..0..1..0..0..0..0..1..0..
       1..0..1..1..0..1..1..1..1.
       .1..0..1..1..0..1..1..0..1
       ..1..1..1..1..1..1..1..1..
1: ab..
c
       .1.1.1.1.1.1.1.1.1.1.1.1.1
       .1.1.1.0.1.1.1.1.1.0.1.1.1
       .1.1.0.1.1.1.0.1.1.0.0.1.1
       .1.0.0.1.1.1.0.0.1.1.0.0.0.1
       .0.0.0.0.1.0.0.1.1.0.0.0.1
       .0.0.0.0.0.0.0.0.0.0.0.0.1
       .0.0.0.0.0.0.0.0.0.0.0.0.0
       .0.0.0.0.0.0.0.0.0.0.0.0.0
       .0.0.0.0.0.0.0.0.0.0.0.0.1
       .0.0.1.0.0.0.0.1.0.1.0.0.1
       .1.0.1.1.1.1.1.0.1.1.1.1.0.1
       .1.0.1.1.1.1.1.0.1.1.1.0.1.1
       .1.1.1.1.1.1.1.1.1.1.1.1.1.1
1: abc.
d
       11111111111111111111111111
       11111100111111111100111111
       11110001111100111110001111
       11000001111000011110000011
       10000000111000011100000001
       10000000000000000000000001
       00000000000000000000000000
       00000000000000000000000000
       10000000000000000000000001
       10000110001000010001100001
       11001111111100111111110011
       11100111111100111111100111
       11111111111111111111111111
The word was: abcd
       # there are no words left to select a word from

No more words left to guess
I hope you enjoyed the game, bye!
```

I expect that your output looks similar, but it does not have to be identical.

Requirements:

• You have to name your file Picture.java

• You may use the following classes *java.util.Scanner; java.util.Vector; java.io.File; java.util.Vector; java.util.Random*.

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
 1      /**
 2       * This class implements a Picture Game.
 3       */
 4
 5      import java.util.Scanner;
 6      import java.util.Vector;
 7      import java.io.File;
 8      import java.util.Vector;
 9      import java.util.Random;
10      import java.io.FileNotFoundException;
11
12
13      public class Picture {
14              private static Scanner userGuessInput           = new Scanner
15              private static String theOriginalWordToGuess;
16              private static String theWordToGuess;
17              private static String theWordShownToTheUser     = null;
18              private static int soManyGuesses                = 0;
19              private static int soManyCorrectGuesses         = 0;
20              final static private int MAX_GUESSES            = 3;
21              final static private Vector<String> pickAWordFromHere = new Ve
22              static Vector<String> thePicture                = new Vector<S
23              final static String DOT = ".";
24              final static Random randomNumberGenerator       = new Random()
25
26              private static void printTheWords()     {
27                      for ( int index = 0; index < pickAWordFromHere.size();
28                              System.out.println(index  + "/" + pickAWordFro
29                      }
30              }
31              private static void printThePicture()   {
32                      int lengthOfWord = theWordToGuess.length();
33                      int printEverXthWord = ( lengthOfWord - soManyCorrectG
34
35                      int globalCharCounter = 1;
36                      for ( int index = 0; index <  thePicture.size(); index
37                              System.out.print("       ");
38                              for ( int xOuter = 0; xOuter < thePicture.elem
39                                      if ( globalCharCounter++ % printEverXt
40                                              System.out.print(thePicture.el
41                                      else
```

```
42                                          System.out.print(DOT);
43                          }
44                          System.out.println();
45                  }
46          }
47          private static void fillTheWordsVector(Scanner theWords)
48                  while ( theWords.hasNext() )
49                          pickAWordFromHere.add(theWords.next());
50          }
51          private static void fillThePicture(Scanner thePictureInput)
52                  while ( thePictureInput.hasNextLine() ) {
53                          thePicture.add( thePictureInput.nextLine() );
54
55                  }
56          }
57          private static void parseArgs(String[] args)    {
58                  try {
59                          for ( int index = 0; index < args.length; inde
60                                  if ( args[index].equals("-words") )
61                                          Scanner theWords = new Scanner
62                                          fillTheWordsVector(theWords);
63                                          theWords.close();
64                                  } else if ( args[index].equals("-pictu
65                                          Scanner thePicture = new Scanr
66                                          fillThePicture(thePicture);
67                                          thePicture.close();
68                                  }
69                          }
70                  } catch (Exception e )  {
71                          System.err.println("Arguments could not be par
72                          e.printStackTrace();
73                          System.exit(1);
74                  }
75          }
76          private static void parse(String[] args) throws FileNotFoundEx
77                  for ( int index = 0; index < args.length; index++ )
78                          if ( args[index].equals("-words") )     {
79                                  Scanner theWords = new Scanner(new Fil
80                                  fillTheWordsVector(theWords);
81                                  theWords.close();
82                          } else if ( args[index].equals("-picture") )
83                                  Scanner thePicture = new Scanner(new F
84                                  fillThePicture(thePicture);
85                                  thePicture.close();
86                          }
87                  }
88          }
89          private static int createRandomNumber(int bound )       {
90                  return randomNumberGenerator.nextInt(bound);
91          }
92          private static boolean areThereAnyWordsLeft()   {
93                  boolean rValue;
94                  if ( ( rValue = ( pickAWordFromHere.size() == 0 ) ) )
95                          System.out.println("No more words left to gues
```

```
96                              System.out.println("I hope you enjoyed the gam
97                      }
98                      return !rValue;
99              }
00              private static String extractWord()     {
01                      String randomWord = null;
02                      int thisWord = 0;
03                      if ( pickAWordFromHere.size() > 1 )     {
04                              thisWord = createRandomNumber(pickAWordFromHer
05                      }
06                      randomWord =  pickAWordFromHere.elementAt(thisWord);
07                      pickAWordFromHere.remove(thisWord);
08                      theWordShownToTheUser = randomWord.replaceAll(DOT, DOT
09                      soManyCorrectGuesses = 0;
10                      return randomWord;
11              }
12              private static boolean guessedCorrectly(String theGuess)
13                      int positionOfChar;
14                      boolean correctGuess  = ( ( positionOfChar = theWordTo
15
16                      if ( correctGuess )     {
17                              theWordShownToTheUser = theWordShownToTheUser.
18                                                      theGuess + theWordShow
19                              }
20                      return  correctGuess;
21      /*
22                      if ( ( positionOfChar = theWordToGuess.indexOf(theGues
23                              theWordShownToTheUser = theWordShownToTheUser.
24                                                      theGuess + theWordShow
25                              return true;
26                      } else
27                              return  false;
28      */
29              }
30              private static boolean notDone()        {
31                      return   ( (soManyGuesses >= MAX_GUESSES ) || ( theWor
32              }
33              private static void printPicture()      {
34                      for ( int line = 0; line < thePicture.size(); line++ )
35                              System.out.println(thePicture.elementAt(line))
36                      }
37              }
38              private static void printWord() {
39                      System.out.println(soManyGuesses + ": " + theWordShown
40              }
41              private static void startTheGame(String[] args) {
42                      parseArgs(args);
43                      String guess = null;
44                      while ( areThereAnyWordsLeft() )        {
45                              theWordToGuess = extractWord();
46                              theOriginalWordToGuess = theWordToGuess;
47                              soManyGuesses = 0;
48                              do {
49                                      printThePicture();
```

```
50                                      printWord();
51                                      if ( userGuessInput.hasNext() )
52                                              guess = userGuessInput.next();
53                                      else {
54                                              System.err.println("Can not re
55                                              System.exit(2);
56                                      }
57                                      if ( ! guessedCorrectly(guess) )
58                                              soManyGuesses++;
59                                      else
60                                              soManyCorrectGuesses++;
61                                      if ( theWordShownToTheUser.indexOf(DOT
62                                              printThePicture();
63                                              System.out.println("The word w
64                                      }
65                                      if (soManyGuesses >= MAX_GUESSES )
66                                              System.out.println("You failed
67                              } while ( ! notDone() );
68                      }
69                      userGuessInput.close();
70
71              }
72
73              public static void main( String[] args ) {
74                      startTheGame(args);
75              }
76      }
```

Source Code: Src/23/Picture.java

23.3. Homework 3.3 (10 Points)

Objective: Understanding ?: operator.

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Look at the following program:

```
 1
 2      public class QuestionMark {
 3        public static boolean aGreaterB(int a, int b) {
 4              return ( a > b );
 5              boolean rValue = false;
 6      /*
 7              if ( a > b )
 8                      rValue = true;
 9              else
10                      rValue = false;
11              return rValue;
12      */
13        }
14        public static int findMaximum(int a, int b)   {
15              return ( a > b ? a : b );
16      /*
17              int rValue;
18              if ( a > b )
19                      rValue = a;
20              else
21                      rValue = b;
22              return rValue;
23      */
24        }
25        public static int findMaximum(int a, int b, int c, int d)     {
26              int rValue;
27              int maxAndB = 0;
28              int maxCndD = 0;
29              if ( a > b )
30                      maxAndB = a;
31              else
32                      maxAndB = b;
33              if ( c > d )
34                      maxCndD = a;
35              else
36                      maxCndD = b;
37              if ( maxAndB > maxCndD )
38                      rValue = maxAndB;
39              else
40                      rValue = maxCndD;
41              return rValue;
```

```
42        }
43        public static int leftToRight(int a, int b)   {
44              int rValue;
45              if ( a != 0 ) {
46                      if ( b != 0 )
47                              rValue = a/b;
48                      else
49                              rValue = -1;
50              } else {
51                      rValue = 0;
52              }
53              return rValue;
54        }
55
56        public static void main( String[] args ) {
57              int a = 5;
58              int b = 1;
59              int c = 2;
60              int d = 1;
61              System.out.println("aGreaterB(" + a + "," + b + ") = " + aGrea
62              System.out.println("findMaximum(" + a + "," + b + ") = " + fir
63              System.out.println("findMaximum(" + a + ", " + b + ", " + c +
64                              findMaximum(a, b, c, d ) );
65              a = 0;
66              b = 0;
67              System.out.println("leftToRight(" + a++ + "," + b++ + ") = " +
68              System.out.println("leftToRight(" + --a + "," + b + ") = " + f
69        }
70      }
```

Source Code: Src/23/QuestionMark.java

As you can seem this program is not elegant. You have to re-write this code in such a way that it is elegant.

Explanation:

Elegant is a way to describe a program, or algorithm. Similarly to the evaluation of a book or a movie, it is subjective, but it must be possible to reason about the rating.

Your Work:

I would use to make the program shorter.

Idea for a Solution:

Requirements:

• You have to name your file *QuestionMark.java*.

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
 1
 2      public class QuestionMark {
 3        public static boolean aGreaterB(int a, int b) {
 4              return ( a > b );
 5              boolean rValue = false;
 6      /*
 7              if ( a > b )
 8                      rValue = true;
 9              else
10                      rValue = false;
11              return rValue;
12      */
13        }
14        public static int findMaximum(int a, int b)   {
15              return ( a > b ? a : b );
16      /*
17              int rValue;
18              if ( a > b )
19                      rValue = a;
20              else
21                      rValue = b;
22              return rValue;
23      */
24        }
25        public static int findMaximum(int a, int b, int c, int d)     {
26              int rValue;
27              int maxAndB = 0;
28              int maxCndD = 0;
29              if ( a > b )
30                      maxAndB = a;
31              else
32                      maxAndB = b;
33              if ( c > d )
34                      maxCndD = a;
35              else
36                      maxCndD = b;
37              if ( maxAndB > maxCndD )
38                      rValue = maxAndB;
39              else
40                      rValue = maxCndD;
41              return rValue;
42        }
43        public static int leftToRight(int a, int b)   {
44              int rValue;
45              if ( a != 0 ) {
46                      if ( b != 0 )
47                              rValue = a/b;
48                      else
49                              rValue = -1;
50              } else {
51                      rValue = 0;
52              }
53              return rValue;
```

```
54        }
55
56        public static void main( String[] args ) {
57              int a = 5;
58              int b = 1;
59              int c = 2;
60              int d = 1;
61              System.out.println("aGreaterB(" + a + "," + b + ") = " + aGrea
62              System.out.println("findMaximum(" + a + "," + b + ") = " + fin
63              System.out.println("findMaximum(" + a + ", " + b + ", " + c +
64                                      findMaximum(a, b, c, d ) );
65              a = 0;
66              b = 0;
67              System.out.println("leftToRight(" + a++ + "," + b++ + ") = " +
68              System.out.println("leftToRight(" + --a + "," + b + ") = " + f
69        }
70      }
```

Source Code: Src/23/QuestionMark.java

24. Homework 4

Posted: June/4/2018
Due: September/8/2020 24.00

The solutions for this homework are due September/8/2020 24.00. I recommend to submit at least one version of all homework solutions long before due date.

24.1. Homework 4.1 (10 Points)

Objective: Working with the Scanner Class.

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Your assignment is to implement a one person 3-dimensional battleship (https://en.wikipedia.org/wiki/Battleship_(game))
game.

I recommend to read in detail the following Scanner documentation. (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/Scanner.html)

You program should present that a 3-dimensional ocean.

A visual representation could look like:

Play

Your program should print the battleship scenario and then prompt the user for column/row position.

Hits

If a boat is hit then the complete boat turns to water, in other word it sinks.

End State

The program ends when all parts of all boats are hit.

Board

Each boat has its uniq id. The range of an id is defined as: $1 \leq id \leq 128$. In the image above you can see 3 boats: one is 1 squares long, one is 2 squares long, one is 3 squares long.

The ocean is defined with a simple language, where each is statement is one line, and each keyword is at the beginning of the line. The keywords are *width, height, row*. It is guaranteed that the keywords *width, height* are on line 1 and 2, but the order is not guaranteed. *row* can only be followed by:

```
w       which stands for water
id      indicate a square of boat id, You can assume that
        all id's are ≥ 0, and all id's are different
```

It is guaranteed that the keyword *row* starts with line 3, and that there are *height row* lines and the are *width* water or boat id's following each row line.

```
Keyword     Explanation
width       so many squares in x direction
height      so many squares in y direction
row         next row starts
```

An example for the image above:

```
width  3
height 3
row  b1 b2 b1
row  w  b2 w
row  w  b2 b3
```

It is guaranteed that the boats are only in horizontal or vertical directions. This would be an example of a illegal input:

```
width 3
height 3
row  b2 b2 b1     # boat 2 to is not horizontal or vertical
row  w b2         # not enough entries
row  w b2 b3
```

In other words you can assume the input is correct.

Explanation:

This is an example of my solution played with the input from above. Comments have been added to explain the execution. A comment starts with '#'.

```
cat b_1.txt
width  3
height 3
row  b1 b2 b1
row  w  b2 w
row  w  b2 b3
% java BattleShip
battleField file name: b_1.txt # user typed in  b_1.txt
x indicates a hit.
w indicates a miss, but you know now there is water.

    0 1 2 ---> columns
0:  . . .
1:  . . .
2:  . . .

column coordinate (0 <= column < 3): 0   # user typed in 0
row    coordinate (0 <= row      < 3): 0  # user typed in 0
HIT

x indicates a hit.
w indicates a miss, but you know now there is water.

    0 1 2 ---> columns
0:  x . x
1:  . . .
2:  . . .

column coordinate (0 <= column < 3): 0   # user typed 0
row    coordinate (0 <= row      < 3): 1  # user typed 1

x indicates a hit.
w indicates a miss, but you know now there is water.
```

```
    0 1 2 ---> columns
0:  x . x
1:  w . .
2:  . . .

column coordinate (0 <= column < 3): 1   # user typed 1
row    coordinate (0 <= row      < 3): 1  # user typed 1
HIT

x indicates a hit.
w indicates a miss, but you know now there is water.

    0 1 2 ---> columns
0:  x x x
1:  w x .
2:  . x .

column coordinate (0 <= column < 3): 2   # user typed 2
row    coordinate (0 <= row      < 3): 2  # user typed 2
HIT

x indicates a hit.
w indicates a miss, but you know now there is water.

    0 1 2 ---> columns
0:  x x x
1:  w x .
2:  . x x
$   # game over, all boars have been hit
```

I expect that your output looks similar, but it does not have to be identical.

Requirements:

- You have to name your file BattleShip.java
- The name of the ocean must be specified on the command line
- You may only use basic Java types and arrays
- You have to test your solution

Submission:

Submit your files via myCourses.

24.2. Homework 4.2 (10 Points)

Objective: Working with the Pattern Class.

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

your assignment is to create patterns which will match a particular set of words.

I recommend to read in detail the following https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/regex/Pattern.html (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/regex/Pattern.html)

Explanation:

The following program might serve as a starting point for your solution.

```
 1      import java.io.File;
 2      import java.io.FileNotFoundException;
 3      import java.util.Scanner;
 4      import java.util.regex.Matcher;
 5      import java.util.regex.Pattern;
 6      import java.lang.Integer;
 7      /*
 8      */
 9      public class RegularExample
10      {
11              static String[] allPatternsToTest = {
12                                      ".",
13                                              "a word which is one character long",
14                                                      "^.$",
15                                      "a000",
16                                              "starts with 'a' followed by one digi
17                                                      "your pattern here",
18                                      "avenious",
19                                              "a word with the vowels 'aeiou' in or
20                                                      "your pattern here",
21                                      "a222",
22                                              "starts with 'a' followed by 3 digits
23                                                      "your pattern here",
24                                      "a123",
25                                              "starts with 'a' followed by 3 or mor
26                                                      "your pattern here",
27                                      "a88",
28                                              "starts with 'a' followed by 2 digits
29                                                      "your pattern here",
30                                      "world",
31                                              "includes only lower case characters,
32                                                      "your pattern here",
33                                      };
34
35              public static void processStatic()      {
36                      int index = 0;
37                      while  ( index < allPatternsToTest.length )     {
38                              System.out.println("Word to test: -" + allPatt
39                              if  ( Pattern.matches(allPatternsToTest[index
40                                      System.out.println("This regular expre
41                                              + " matches the following inpu
42                                      System.out.println("    verbal explana
43                              }
44                              index += 3;
45                      }
```

```
46              }
47              public static void main(String[] args) {
48                      processStatic();
49              }
50      }
```

Source Code: Src/24/RegularExample.java

This program produces the following output:

```
java RegularExample
Word to test: -.=
This regular expression ^.$ matches the following input: -.=
       verbal explanation: a word which is one character long
Word to test: -a000=
Word to test: -avenious=
Word to test: -a222=
Word to test: -a123=
Word to test: -a88=
Word to test: -world=
```

As you can see the string, a textual explanation, and the the regular expression are stored in *allPatternsToTest*.

For the solution you have to define the regular expression described in the textual explanation using as much as possible from the Pattern class offers.

Your program has to read the input strings from a file, and it must be possible to set the delimiter via command line argument. The program has to print all words which matches the description in RegularExample.java using regular expression described in the textual explanation.

In other words, your program has to read word for word for the file, and every word read is then matched with all patters. The word, the regular expression, and the verbal explanation will be printed for every pattern that matches. Assume the pattern is '^.$' and the input is

```
:a:
:b:
:cdef:
```

the inputs *a*, and *b* match the pattern.

An execution of a my solution produces the following output. The regular expression I used have been marked with 'X's.:

```
spiegel 24 104 cat test_3.txt
:1:
:a:
:h:
:aeiou:
:a111:
:a222:
:a99:
:a88:
:a8:
spiegel 24 105 make
```

```
javac RegularExample.java
java RegularExample -d ':' -input test_3.txt
-------------------------------------------------------
Input: -1=
This regular expression "XX" matches the following input: -1=
       verbal explanation a word which is one character long
-------------------------------------------------------
Input: -a=
This regular expression "XX" matches the following input: -a=
       verbal explanation a word which is one character long
This regular expression "XX" matches the following input: -a=
       verbal explanation includes only lower case characters, but not the cha
-------------------------------------------------------
Input: -h=
This regular expression "XX" matches the following input: -h=
       verbal explanation a word which is one character long
-------------------------------------------------------
Input: -aeiou=
This regular expression "XX" matches the following input: -aeiou=
       verbal explanation a word with the vowels 'aeiou' in order and each vow
This regular expression "XX" matches the following input: -aeiou=
       verbal explanation includes only lower case characters, but not the cha
-------------------------------------------------------
Input: -a111=
This regular expression "XX" matches the following input: -a111=
       verbal explanation starts with 'a' followed by one digit or more digits
This regular expression "XX" matches the following input: -a111=
       verbal explanation starts with 'a' followed by 3 digits
This regular expression "XX" matches the following input: -a111=
       verbal explanation starts with 'a' followed by 3 or more digits
-------------------------------------------------------
Input: -a222=
This regular expression "XX" matches the following input: -a222=
       verbal explanation starts with 'a' followed by one digit or more digits
This regular expression "XX" matches the following input: -a222=
       verbal explanation starts with 'a' followed by 3 digits
This regular expression "XX" matches the following input: -a222=
       verbal explanation starts with 'a' followed by 3 or more digits
-------------------------------------------------------
Input: -a99=
This regular expression "XX" matches the following input: -a99=
       verbal explanation starts with 'a' followed by one digit or more digits
This regular expression "XX" matches the following input: -a99=
       verbal explanation starts with 'a' followed by 2 digits in the range be
-------------------------------------------------------
Input: -a88=
This regular expression "XX" matches the following input: -a88=
       verbal explanation starts with 'a' followed by one digit or more digits
This regular expression "XX" matches the following input: -a88=
       verbal explanation starts with 'a' followed by 2 digits in the range be
-------------------------------------------------------
Input: -a8=
This regular expression "XX" matches the following input: -a8=
       verbal explanation starts with 'a' followed by one digit or more digits
```

Your output does not have to be identical to mine, but similar.

Your Work:

You have to write a program which allows to define the delimiter on the command line and the program has to find all the words which match the following descriptions:

You can assume the intout file is correct.

- starts with 'a' followed by one digit or more digits,
- a word with the vowels 'aeiou' in order and each vowel can appear only once,
- starts with 'a' followed by 3 digits in the range between 1 and 3 only,
- starts with 'a' followed by least 3 digits in the range between 1 and 3 only,
- starts with 'a' followed by between 1 and 2 digits in the range between 8 and 9 only,
- includes only lower case characters, but not the character 'h', 'p', and 'b'

You have to be able to test your program which words stored in an array and read from file.

You have to test your program.

Requirements:

Example:

An example of a solution execution:

- You have to name your program RegularExample.java.
- You have to provide a description of your test cases. Submit this file with your code.

Submission:

Submit your files via myCourses.

25. Homework 5

Posted: June/29/2018
Due: September/27/2020 24.00

The solutions for this homework are due September/27/2020 24.00. I recommend to submit at least one version of all homework solutions long before due date.

25.1. Homework 5.1 (10 Points)

Objective: Creating your on classes.

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Your assignment is to implement a system which can represent a solar system. A solar system has typically one sun and a known number of planets.

Explanation:

You have to have for each instance variable set and get methods. You have to use the best possible access controls, and *final* when ever possible. You have to explain your choice of access control to the grader.

First, lets take a look at the *Planet main* method:

```
public static void main(String args[])  {
        Planet aPlanet = new Planet("Mercury", 5.427, 87.97, 0);
        System.out.println(aPlanet);
        aPlanet.setName("Saturn");
        aPlanet.setDensity(0.687);
        aPlanet.setOrbitalPeriod(10759.22);
        aPlanet.setNumberOfMoons(82);
        System.out.println(aPlanet);

        System.out.println("1: " + aPlanet.getName() );
        System.out.println("2: " + aPlanet.getDensity() );
        System.out.println("3: " + aPlanet.getOrbitalPeriod() );
        System.out.println("4: " + aPlanet.getNumberOfMoons() );

}
```

The output of is:

```
Mercury/5.427/87.97/0
Saturn/0.687/10759.22/82
1: Saturn
2: 0.687
3: 10759.22
4: 82.0
```

You can see all methods a potential solution is using. You need to determine what kind of instance/class variables you need.

Second, lets take a look at the *SolarSystem*. The solar system class should be able to store the planets of a solarsystem. The constructor looks like: *public SolarSystem(int soManyPlanets)*.

The main method of my *SolarSystem* looks like:

```
        public static void main(String args[])  {
         SolarSystem aSolarSystem = new SolarSystem(8);  // sadly Pluto was dem

         Planet aPlanet = new Planet("Mercury", 5.427, 87.97, 0);
         aSolarSystem.setPlanet(1, new Planet("Mercury", 5.427, 87.97, 0));

         aPlanet.setName("Saturn");
         aPlanet.setDensity(0.687);
         aPlanet.setOrbitalPeriod(10759.22);
         aPlanet.setNumberOfMoons(82);
         aSolarSystem.setPlanet(6, aPlanet);
/*
         aPlanet.setName("Earth");
         aPlanet.setDensity(5.514);
         aPlanet.setOrbitalPeriod(365.256363004);
         aPlanet.setNumberOfMoons(1);
         aSolarSystem.setPlanet(3, aPlanet);
*/

         System.out.println(aSolarSystem);
}
```

The output looks like:

```
1: Mercury/5.427/87.97/0
6: Saturn/0.687/10759.22/82

       average density: 3.057
```

There is no available method to get the average density. My program calculates the average density every time a planet is added.

Your output does not have to be identical to mine, but similar.

Requirements:

• You have to name your file Planet.java and SolarSystem.java

• You may only use basic Java types, Date class, and arrays

• You have to test your solution

• You have to explain what will happen if the code in *SolarSystem* main method would not be commented out. It might be best to submit the answer to this queation in a pdf using a memory model.

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

Planet:

```
 1      import java.util.Date;
 2
 3      public class  Planet {
 4              String name = "Planet";
 5              double  density;
 6              double  orbitalPeriod;
 7              int    numberOfMoons;
 8
 9              public Planet() {
10              }
11              public Planet(String name, double density, double orbitalPeric
12                      this.name =  name;
13                      this.density =  density;
14                      this.orbitalPeriod =  orbitalPeriod;
15                      this.numberOfMoons =  numberOfMoons;
16              }
17              public String getName() {
18                      return name;
19              }
20              public double getDensity()      {
21                      return density;
22              }
23              public double getOrbitalPeriod()        {
24                      return orbitalPeriod;
25              }
26              public double getNumberOfMoons()        {
27                      return numberOfMoons;
28              }
29              public void setName(String name)        {
30                      this.name =  name;
31              }
32              public void setDensity(double density)  {
33                      this.density =  density;
34              }
35              public void setOrbitalPeriod(double orbitalPeriod)      {
36                      this.orbitalPeriod =  orbitalPeriod;
37              }
38              public void setNumberOfMoons(int numberOfMoons) {
39                      this.numberOfMoons =  numberOfMoons;
40              }
41              public String toString()        {
42                      return name + "/" + density + "/" + orbitalPeriod + "/
43              }
44
45              public static void main(String args[])  {
46                      Planet aPlanet = new Planet("Mercury", 5.427, 87.97, 0
47                      System.out.println(aPlanet);
48                      aPlanet = new Planet();
49                      aPlanet.setName("Saturn");
50                      aPlanet.setDensity(0.687);
51                      aPlanet.setOrbitalPeriod(10759.22);
52                      aPlanet.setNumberOfMoons(82);
53                      System.out.println(aPlanet);
54
```

```
55                      System.out.println("1: " + aPlanet.getName() );
56                      System.out.println("2: " + aPlanet.getDensity() );
57                      System.out.println("3: " + aPlanet.getOrbitalPeriod()
58                      System.out.println("4: " + aPlanet.getNumberOfMoons()
59
60              }
61      }
62
63
```

Source Code: Src/25_1/Planet.java

Solarsystem:

```
 1      import java.util.Date;
 2
 3      public class  SolarSystem {
 4              String name = "SolarSystem";
 5              int       soManyPlanets = 0;
 6              int       soManyPlanetsSet = 0;
 7              private double densityOnAverage = 0;
 8              Planet[]  thePlanets;
 9              public SolarSystem()    {
10              }
11              public SolarSystem(int soManyPlanets)   {
12                      this.soManyPlanets = soManyPlanets;
13                      thePlanets = new Planet[soManyPlanets];
14                      for ( int index = 0; index < soManyPlanets; index ++ )
15                              thePlanets[index] = null;
16              }
17              private void calculateDensityOnAverage() {
18                      double allDensity = 0;
19                      for ( int index = 0; index < soManyPlanets; index ++ )
20                              if ( thePlanets[index] != null )        {
21                                      allDensity += thePlanets[index].getDen
22                              }
23                      }
24                      densityOnAverage = allDensity / soManyPlanetsSet;
25              }
26              public void setPlanet(int nTh, Planet thePlanet)        {
27                      thePlanets[nTh] = thePlanet;
28                      soManyPlanetsSet++;
29                      calculateDensityOnAverage();
30              }
31              public String toString()        {
32                      String theSolarSystem = "";
33                      for ( int index = 0; index < soManyPlanets; index ++ )
34                              if ( thePlanets[index] != null )        {
35                                      theSolarSystem += index + ": " + thePl
36                              }
37                      }
38                      return theSolarSystem + "\n     average density: " + d
39              }
40
```

```
41              public static void main(String args[])  {
42                      SolarSystem aSolarSystem = new SolarSystem(8);  // we
43
44                      Planet aPlanet = new Planet("Mercury", 5.427, 87.97, (
45                      aSolarSystem.setPlanet(1, new Planet("Mercury", 5.427,
46
47                      aPlanet.setName("Saturn");
48                      aPlanet.setDensity(0.687);
49                      aPlanet.setOrbitalPeriod(10759.22);
50                      aPlanet.setNumberOfMoons(82);
51                      aSolarSystem.setPlanet(6, aPlanet);
52      /*
53                      aPlanet.setName("Earth");
54                      aPlanet.setDensity(5.514);
55                      aPlanet.setOrbitalPeriod(365.256363004);
56                      aPlanet.setNumberOfMoons(1);
57                      aSolarSystem.setPlanet(3, aPlanet);
58      */
59
60                      System.out.println(aSolarSystem);
61              }
62      }
63
64
```

Source Code: Src/25_1/SolarSystem.java

25.2. Homework 5.2 (10 Points)

Objective: Working with a Class Hierarchy.

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Your assignment is to create a class which can store a fixed amount of objects, and one which can store a as many objects as memory allows. The first class is called Array.java, the second is called Flexible.java I highy recommend to use inheritance to solve this problem.

I highly recommend to read the provided code and output very carefully. This information provides a lot of information how to solve the problem.

Objects of these classes must be timestamped with the time the object was generated and when the last modification of the object took place. Adding or removing an element modify the object.

Explanation:

It must be possible to store a null object to objects of these classes.

The methods used to manipulate an **Array or Flexible object** are:

• public boolean add(element) - Returns true if element could be added

• public boolean delete(element) - Returns true if element could be deleted

• public int getMax - Returns the maximum number of elements can be stored

• public int size - Returns the current number of elements stored

• public boolean contains(element) - Returns true if a particular element is stored

• public boolean isFull - Returns true if no more elements can be stored unless an element is deleted

• public boolean isEmpty - Returns true if 0 ore elements are stored

• public String toString - Returns a textual representation

A possible main program might look like:

```
public static void main(String args[])  {
        Array aArray = new Array();
        aArray.add(null);
        aArray.add("abc");
        System.out.println(aArray);
}
```

My program produces the following output:

```
name: Array
creationTime: Mon Jun 29 16:47:49 EDT 2020
unlimited: false
soMany = 2
nullObjectAdded = true
modificationTime: Mon Jun 29 16:47:49 EDT 2020
0/abc,
```

As you can see

• a null object has been added and the String *abc*.

• soMany == 2, because *abc* and null has been added

• it is not an unlimited storage

• the creationTime and modificationTime are known

• and the content of the anArray object.

My Array class is tested with String objects and Integer objects.

My test class with Strings looks like:

```
 1      public class TestString {
 2              Array aArray = new Array();
 3
 4              private static void error(String errorMessage)  {
 5                      System.out.println("Error: " + errorMessage);
 6              }
 7              private void testNull() {
 8                      // System.out.println(this);
 9                      if ( ! aArray.add(null) )
10                              error("1: Adding null element");
11                      if ( ! aArray.contains(null )   )
12                              error("2: null element not found");
13                      if ( ! aArray.delete(null) )
14                              error("3: Deleting null element");
```

```
15                      if ( aArray.delete(null) )
16                              error("4: Deleting null element");
17                      // System.out.println(this);
18              }
19              private void testAdd()  {
20                      if ( ! aArray.add("a") )
21                              error("Adding element");
22                      aArray.add("a");
23                      aArray.add("a");
24                      aArray.add("a");
25                      if ( aArray.size() != aArray.getMax() )
26                              error("Adding size failed");
27                      if ( aArray.add("a") )
28                              error("Adding one too many");
29
30              }
31              private void testDelete()       {
32                      aArray.add(null);
33                      if ( ! aArray.delete(null) )
34                              error("Deleting null element");
35                      if (  aArray.delete(null) )
36                              error("Deleting non exciting null element");
37                      if ( ! aArray.delete("a") )
38                              error("Deleting first element");
39                      aArray.delete("a");  aArray.delete("a");  aArray.delet
40                      if ( aArray.delete("a") )
41                              error("Deleting one too many");
42                      System.out.println(this);
43
44              }
45              private void test()     {
46                      testNull();
47                      testAdd();
48                      testDelete();
49              }
50              public static void main(String args[])  {
51              }
52      }
```

Source Code: Src/25/TestString.java

It produces no output.

My test class looks like:

```
 1      public class Test {
 2              Array aArray            = new Array();
 3              Flexible aFlexible      = new Flexible();
 4              Integer one             = Integer.valueOf(1);
 5              Integer two             = Integer.valueOf(2);
 6              Integer three           = Integer.valueOf(3);
 7
 8              private void testAddArray()     {
 9                      aArray.add(one);
10                      aArray.add(two);
```

```
11                      aArray.add(three);
12                      aArray.add(null);
13                      System.out.println(aArray);
14              }
15              private void testAddFlexible()  {
16                      aFlexible.add(one);
17                      aFlexible.add(one);
18                      aFlexible.add(two);
19                      aFlexible.add(two);
20                      aFlexible.add(three);
21                      aFlexible.add(null);
22                      aFlexible.delete(two);
23                      System.out.println(aFlexible);
24
25              }
26              private void test()     {
27                      testAddArray();
28                      testAddFlexible();
29              }
30              public static void main(String args[])  {
31                      new Test().test();
32              }
33      }
```

Source Code: Src/25/Test.java

It produces the following output:

```
name: Array
creationTime: Wed Jul 15 13:16:28 EDT 2020
unlimited: false
soMany = 4
nullObjectAdded = true
modificationTime: Wed Jul 15 13:16:28 EDT 2020
1, 2, 3,

name: Flexible
creationTime: Wed Jul 15 13:16:28 EDT 2020
unlimited: true
soMany = 5
nullObjectAdded = true
modificationTime: Wed Jul 15 13:16:28 EDT 2020
1, 1, 2, 3,
```

Design and test both classes accordingly.

Your output does not have to be identical to mine, but similar.

Your Work:

You have to create the as described above, and a test environment.

Requirements:

• Your code should compile with the provided test examples.

• You can not use dynamic data structures like a Vector, or any classes not mentioned in the lecture

• You can use a date class.

Example:

An example of a solution execution:

• You have to name your classes (Array|Flexible).java.

• You have to provide a description of your test cases. Submit this file with your code.

• You have to submit your test classes.

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

Base:

```
 1      import java.util.Date;
 2
 3      public class  Storage   {
 4              final String name = "Storage";
 5
 6              public Storage()        {
 7              }
 8              public String getClassName()    {
 9                      return name;
10              }
11              public String toString()        {
12                      return name;
13
14              }
15
16              public static void main(String args[])  {
17                      Storage aStorage = new Storage();
18                      System.out.println(aStorage);
19              }
20      }
21
22
```

Source Code: Src/25_2/Storage.java

Array:

```
 1      import java.util.Date;
 2
 3      public class  Array extends Storage     {
 4              int MAX = 4;
 5              int   soMany  = 0;
 6              int soManyNulls = 0;
 7              final String name = "Array";
 8              boolean unlimited = false;
 9              final Date creationTime = new Date();
10              Date modificationTime = null;
11              Object[] theStorage = new Object[MAX];
12
13              public Array()  {
14                      init();
15              }
16              protected void init()   {
17                      setMax(MAX);
18                      theStorage = new Object[MAX];
19                      for ( int index = 0; index < MAX; index ++ )
20                              theStorage[index] = null;
21              }
22              protected void setMax(int max)  {
23                      this.MAX = max;
24              }
25              protected Object[] getTheStorage()      {
26                      return theStorage;
27              }
28              private void setModificationTime()      {
29                      modificationTime = new Date();
30              }
31              private int findFreeSpace()     {
32                      for ( int index = 0; index < MAX; index ++ )    {
33                              if ( theStorage[index] == null )
34                                      return index;
35                      }
36                      return -1;
37              }
38              public boolean add(Object aObject)      {
39                      boolean rValue = false;
40                      if ( aObject == null )  {
41                              soManyNulls++;
42                              rValue = true;
43                      } else {
44                              if ( ! isFull () )      {
45                                      theStorage[findFreeSpace()] = aObject;
46                                      rValue = true;
47                                      soMany++;
48                              } else {
49                                      rValue = false;
50                              }
51                      }
52                      if ( rValue )
53                              setModificationTime();
54                      return rValue;
```

```
55              }
56              private int findObject(Object aObject)  {
57                      for ( int index = 0; index < MAX; index ++ )
58                              if ( ( theStorage[index] != null )       &&
59                                   ( theStorage[index] == aObject ) )
60                                      return index;
61                      for ( int index = 0; index < MAX; index ++ )
62                              if ( ( theStorage[index] != null )       &&
63                                   ( theStorage[index].equals(aObject) ) )
64                                      return index;
65                      return -1;
66              }
67              public boolean delete(Object aObject)   {
68                      boolean rValue = false;
69                      if ( aObject == null )  {
70                              rValue = soManyNulls > 0;
71                              soManyNulls --;
72                      } else {
73                              if ( ! isEmpty () )     {
74                                      rValue = false;
75                                      int index = findObject(aObject);
76                                      if ( index >= 0 )       {
77                                              theStorage[index] = null;
78                                              rValue = true;
79                                              soMany --;
80                                      }
81                              } else
82                                      rValue = false;
83                      }
84                      if ( rValue )
85                              setModificationTime();
86                      return rValue;
87              }
88              public int getMax()     {
89                      return MAX;
90              }
91              public int size()       {
92                      return soManyNulls > 0 ? soMany + 1 : soMany;
93              }
94              public boolean contains(Object aObject) {
95                      boolean rValue = true;
96                      if ( aObject == null )  {
97                              rValue = true;
98                      } else {
99                              if ( ! isEmpty () )     {
00                                      int index = findObject(aObject);
01                                      if ( index >= 0 )       {
02                                              rValue = true;
03                                      }
04                              } else
05                                      rValue = false;
06                      }
07                      return rValue;
08              }
```

```
09              public boolean isFull() {
10                      return soMany == MAX;
11              }
12              public boolean isEmpty()        {
13                      return soMany == 0;
14              }
15              public String getClassName()    {
16                      return name;
17              }
18              public String toString()        {
19                      String aString =
20                              "\n     name: " + getClassName()  +
21                              "\n     creationTime: " + creationTime  +
22                              "\n     unlimited: " + unlimited        +
23                              "\n     soMany = " + size()             +
24                              "\n     soManyNulls = " + soManyNulls   +
25                              "\n     modificationTime: " +
26                                      ( modificationTime == null ? "not set'
27                                      modificationTime )
28                              ;
29                      String content = "";
30                      for ( int index = 0; index < MAX; index ++ )    {
31                              if ( theStorage[index] != null )
32                                      content += "(" + index + ")" + theStor
33                      }
34                      return aString + "\n    " + content;
35              }
36
37              public static void testIdenticalString()        {
38                      Array aArray = new Array();
39                      aArray.add(new String("abc"));          // position 0
40                      aArray.add("abc");                      // position 1
41                      aArray.add(new String("abc"));          // position 0
42                      System.out.println(aArray);
43                      aArray.delete("abc");                   // should it c
44                      System.out.println(aArray);
45                      aArray.delete("abc");                   // should dele
46                      System.out.println(aArray);
47              }
48              public static void testNull()   {
49                      Array aArray = new Array();
50                      aArray.add(null);
51                      System.out.println(aArray);
52                      aArray.add(null);
53                      System.out.println(aArray);
54                      aArray.delete(null);
55                      System.out.println(aArray);
56              }
57              public static void main(String args[])  {
58                      // testNull();
59                      testIdenticalString();
60              }
61      }
62
```

63

Source Code: Src/25_2/Array.java

Flexible:

```
 1      import java.util.Date;
 2
 3      public class  Flexible extends Array    {
 4              private final String name = "Flexible";
 5
 6              private void copyAndIncrease() {
 7                      MAX += 2;
 8                      Object[] tmpStorage = new Object[getMax()];
 9                      Object[] storageSoFar = getTheStorage();
10                      for ( int index = 0; index < storageSoFar.length; inde
11                              tmpStorage[index] = storageSoFar[index];
12                      }
13                      setMax(getMax() * 2 );
14                      init();
15                      for ( int index = 0; index < storageSoFar.length; inde
16                              super.add(storageSoFar[index]);
17                      }
18
19              }
20              public boolean add(Object aObject)      {
21                      if ( isFull() )
22                              copyAndIncrease();
23                      return super.add(aObject);
24              }
25              public String getClassName()    {
26                      return name;
27              }
28              public String toString()        {
29                      return super.toString();
30              }
31
32              private void test()     {
33                      int theSize = getMax();
34                      System.out.println(this);
35                      String aString = "a";
36                      for ( int index = 0; index < theSize; index ++ ) {
37                              add(aString);
38                              aString += "a";
39                      }
40                      System.out.println(this);
41                      aString = "b";
42                      for ( int index = 0; index < theSize; index ++ ) {
43                              add(aString);
44                              aString += "b";
45                      }
46                      System.out.println(this);
47              }
48              public static void main(String args[])  {
```

```
49                      Flexible aFlexible = new Flexible();
50                      aFlexible.test();
51              }
52      }
53
54
```

Source Code: Src/25_2/Flexible.java

25.3. Homework 5.3 (10 Points)

Objective: Using of abstract class, interfaces, and inheritance.

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Given is a class hierarchy consisting of

• Interface

• Abstract class

• Inheritance

The goal of this home work is understanding the consequences of its use.

Explanation:

Given is the set of classes: *AbstractClass.java, C1.java, C2.java, C3.java, C4.java, C5.java, C6.java, I1.java, I2.java, TestAll.java.*

You have to answer the questions in the source files, and explain the output of *TestAll.java*.

First Interface:

```
1       public interface I1     {
2               public int i1variable           = 0;
3               public int i1and2variable       = 1;
4               void i1method();
5               void i1and2method();
6       }
```

Source Code: Src/25/I1.java

Second Interface:

```
1       public interface I2     {
2
3               public int i2variable           = 3333;
4               public int i1and2variable       = 4444;
5               void i2method();
6               void i1and2method();
7       }
```

Source Code: Src/25/I2.java

Abstract Class:

```
 1      abstract class  AbstractClass implements I1 {
 2              void abstractClass1()   {
 3                      System.out.println("AbstractClass: abstractClass1()");
 4              }
 5              public abstract int abstractClass2();
 6      }
```

Source Code: Src/25/AbstractClass.java

Class1:

```
 1      public class C1 extends AbstractClass {
 2              public void i1method()  {
 3                      System.out.println("C1: i1method()/i1variable = " + i1
 4              }
 5              public void i1and2method()      {
 6                      System.out.println("C1: i1and2method()/i1variable = "
 7                                      " i1and2variable = " + i1and2variab
 8              }
 9              void abstractClass1()   {
10                      System.out.println("C1: abstractClass1()");
11              }
12              public int abstractClass2()     {
13                      System.out.println("C1: abstractClass2()");
14                      return 2525;
15              }
16      }
```

Source Code: Src/25/C1.java

Class1:

```
 1      public class C2 extends AbstractClass {
 2              public void i1method()  {
 3                      System.out.println("C2: i1method()/i1variable = " + i1
 4              }
 5              public void i1and2method()      {
 6                      System.out.println("C2: i1and2method()/i1variable = "
 7                                      " i1and2variable = " + i1and2variab
 8              }
 9              void abstractClass1()   {
10                      System.out.println("C2: abstractClass1()");
11              }
12              public int abstractClass2()     {
13                      System.out.println("C2: abstractClass2()");
14                      return 2525;
15              }
16      }
```

Source Code: Src/25/C2.java

Class2:

```
1       public class C3 extends AbstractClass implements I2 {
2               int c3andC5 = 42;
3               public void c3andC5m()  {
4                       System.out.println("C3: c3andC5m()/c3andC5m: " + c3and
5               }
6               public int abstractClass2()     {
7                       System.out.println("C3: abstractClass2()");
8                       return 123456;
9               }
10              public void i1method()  {
11                      System.out.println("C3: i1method()/i2variable = " + i2
12              }
13              public void i2method()  {
14                      System.out.println("C3: i2method()");
15              }
16              public void i1and2method()      {
17                      System.out.println("C3: i1and2method()/i2variable = "
18                                      " I1.i1and2variable = " + I1.i1and2
19              }
20      }
```

Source Code: Src/25/C3.java

Class3:

```
1       public class C4 implements I1, I2 {
2               public int abstractClass2()     {
3                       System.out.println("C4: abstractClass2()");
4                       return 314;
5               }
6               public void i1method()  {
7                       System.out.println("C4: i1method()/i1variable = " + i1
8               }
9               public void i2method()  {
10                      System.out.println("C4: i2method()/i2variable = " + i2
11              }
12              public void i1and2method()      {
13                      System.out.println("C4: i1and2method()/i2variable = "
14                                      " I2.i1and2variable = " + I2.i1and2
15              }
16      }
```

Source Code: Src/25/C4.java

Class4:

```
1       public class C5 extends C3 {
2               int c3andC5 = 424242;
3               public void c3andC5m()  {
4                       System.out.println("C5: c3andC5m()/c3andC5m: " + c3and
5               }
6               public int abstractClass2()     {
```

```
 7                       System.out.println("C5: abstractClass2()");
 8                       return 123456;
 9               }
10               public void i1method()  {
11                       System.out.println("C5: i1method()/i2variable = " + i2
12               }
13               public void i2method()  {
14                       System.out.println("C5: i2method()");
15               }
16               public void i1and2method()      {
17                       System.out.println("C5: i1and2method()/i2variable = "
18                                               " I1.i1and2variable = " + I1.i1and2
19               }
20       }
```

Source Code: Src/25/C5.java

Class5:

```
 1       abstract class C6 extends C1 {
 2               public int abstractClass2()     {
 3                       System.out.println("C6: abstractClass1()");
 4                       return 1;
 5               }
 6       }
```

Source Code: Src/25/C6.java

Test Class:

```
 1       public class TestAll    {
 2               // draw the class diagram, including interface
 3               // why are these declararions legal?
 4               static I1       anI1    =       new C1();
 5               static I2       anI2    =       new C3();
 6               static I1       anI1a   =       new C2();
 7               static I1       anI1b   =       new C4();
 8               static I2       anI2a   =       new C4();
 9
10               // which methods will be called and why?
11               public static void test1()      {
12                       anI1.i1method();
13                       anI1b.i1method();
14                       anI1b.i1and2method();
15                       anI1b.i1method();
16                       anI2.i2method();
17                       anI2.i1and2method();
18               }
19               // which methods will be called and why?
20               public static void test2()      {
21                       C3 aC3 = new C3();
22                       C5 aC5 = new C5();
23                       C3 aaC3 = (C3)aC5;
```

```
24                      aC3.c3andC5m();
25                      aC5.c3andC5m();
26                      aaC3.c3andC5m();
27                      System.out.println("aaC3.c3andC5 = " + aaC3.c3andC5 );
28                      aaC3.c3andC5 = 99999;
29                      aaC3.c3andC5m();
30                      System.out.println("aaC3.c3andC5 = " + aaC3.c3andC5 );
31              }
32              public static void main(String[] args)  {
33                      test1();
34                      test2();
35              }
36              // give an example when you would use an abstract class but no
37              // give an example when you would use an interface but not an
38              // give an example when you have to use an interface
39      }
```

Source Code: Src/25/TestAll.java

The execution of *Test* produces the following output:

```
C1: i1method()/i1variable = 0
C4: i1method()/i1variable = 0
C4: i1and2method()/i2variable = 3333 I2.i1and2variable = 4444
C4: i1method()/i1variable = 0
C3: i2method()
C3: i1and2method()/i2variable = 3333 I1.i1and2variable = 1
C3: c3andC5m()/c3andC5m: 42
C5: c3andC5m()/c3andC5m: 424242
C5: c3andC5m()/c3andC5m: 424242
aaC3.c3andC5 = 42
C5: c3andC5m()/c3andC5m: 424242
aaC3.c3andC5 = 99999
```

Your Work:

You have to explain the output and answer the question asked in the files.

Requirements:

You have to submit all source files and you have to answer the question, including an explanation of the output in the source file.

Submission:

Submit your files via myCourses.

26. Homework 6

Posted: September/22/2020
Due: October/4/2020 24.00

The solutions for this homework are due October/4/2020 24.00. I recommend to submit at least one version of all homework solutions long before due date.

26.1. Homework 6.1 (10 Points)

Objective: Working with inheritance and generics

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

The objective is to write small examples using inheritance and generics in Java.

This home work is a modification of hw 5.1. You have to create classes which allows to store objets of: planets, binaries, and asteroids. The next part of this document describe instance variables and methods for each class. The methods listing do not include access modifiers. You have to come with constructors you think you need.

Planet instance variables:

```
String name;
double density;
double orbitalPeriod;
int numberOfMoons;
```

Planet methods:

```
public String getName()
public double getDensity()
public double getOrbitalPeriod()
public double getNumberOfMoons()
public void setName(String name)
public void setDensity(double density)
public void setOrbitalPeriod(double orbitalPeriod)
public void setNumberOfMoons(int numberOfMoons)
public String toString()
```

Binaries instance variables:

```
String name;
double  density;
String  discoverer;
int satellites;
```

Binaries methods:

```
public String getName()
public double getDensity()
public int getSatelites()
public void setSatelites(int satelites)
public String getDiscoverer()
public void setDiscoverer(String discoverer)
public void setName(String name)
public void setDensity(double density)
```

```
public String toString()
```

Asteroid instance variables:

```
String name;
double  density;
String  discoverer;
```

Asteroid methods:

```
public String getName()
public double getDensity()
public String getDiscoverer()
public void setDiscoverer(String discoverer)
public void setName(String name)
public void setDensity(double density)
public String toString()
```

You must be able to create a storage class, which allow you to store objects of the classes described above. The test method for planets might look like:

```
        public static void print(
                 StorageOfAstronomicalObject ... deleted ...  aStorageOfAstronc
                 System.out.println(aStorageOfAstronomicalObject.getAllNames())
                 System.out.println(aStorageOfAstronomicalObject);

         }
         public static void testPlanets()        {
                 StorageOfAstronomicalObject ... deleted ... aStorageOfAstronom
                         = new StorageOfAstronomicalObject ... deleted ...("Mil

                 aStorageOfAstronomicalObject.add(new Planet("Mercury", 5.427,

                 Planet aPlanet = new Planet("Saturn", 0.687, 10792, 82);
                 aStorageOfAstronomicalObject.add(aPlanet);;

                 aPlanet.setName("Earth");
                 aPlanet.setDensity(5.514);
                 aPlanet.setOrbitalPeriod(365.256363004);
                 aPlanet.setNumberOfMoons(1);
                 aStorageOfAstronomicalObject.add(aPlanet);

                 print(aStorageOfAstronomicalObject);
         }
```

The execution of the method produces:

```
Mercury, Earth, Earth,
0: Mercury/5.427/87.97/0
1: Earth/5.514/365.256363004/1
2: Earth/5.514/365.256363004/1

        average density: 5.484999999999999
```

As you have noticed "Saturn" is not in the output, but "Earth" is twice in the storage area. This is not a bug of the storage class. Assume a Vector is used to store the objects, and the *add(aPlanet);* is used. This would create the same output.

Your Work:

• You have to implement all classes needed. It is important to note that the compiler must compile all of your code without a warning.

• You have to use generics for your solution.

• You have to use inheritance wherever it is appropriate.

• You are allowed to use the Vector class.

Requirements:

• You have to name your classes in the same way this write up does.

• You might use more classes than this write up mentions.

• You have to test your solution.

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

Astroid:

```
 1      import java.util.Date;
 2
 3      public class  Asteroid  extends DiscovererOfStellarObject {
 4              String name;
 5              double  density;
 6              String  discoverer;
 7
 8              public Asteroid(String name, String discoverer, double density
 9                      super(name, discoverer, density);
10              }
11              public String toString()        {
12                      return "Asteroid: " + super.toString();
13              }
14              public static void main(String args[])  {
15                      Asteroid aAsteroid = new Asteroid("Ceres", "Giuseppe P
16                      Asteroid bAsteroid = new Asteroid("Pallas", "Heinrich
17                      Asteroid cAsteroid = new Asteroid("Juno", "Karl Ludwig
18
19                      System.out.println("aAsteroid = " + aAsteroid );
20                      System.out.println("bAsteroid = " + bAsteroid );
21                      System.out.println("cAsteroid = " + cAsteroid );
22
23              }
24      }
25
26
```

Source Code: Src/26_1/Asteroid.java

Binaries:

```
1       import java.util.Date;
2
3       public class  Binaries  extends DiscovererOfStellarObject {
4               int satellites;
5
6               public Binaries(String name, String discoverer, double density
7                       super(name, discoverer, density);
8                       this.satellites = satellites;
9               }
10              public int getSatelites()       {
11                      return satellites;
12              }
13              public void setSatelites(int satellites) {
14                      this.satellites = satellites;
15              }
16              public String toString()        {
17                      return "Binary: " + super.toString() + "/" + satellite
18              }
19              public static void main(String args[])  {
20                      Binaries aBinaries = new Binaries("Sylvis", "Norman Ro
21                      Binaries bBinaries = new Binaries("Elektra", "C. H. F.
22
23                      System.out.println("aBinaries = " + aBinaries );
24                      System.out.println("bBinaries = " + bBinaries );
25
26              }
27      }
28
29
```

Source Code: Src/26_1/Binaries.java

DiscovererOfStellarObject:

```
1
2       import java.util.Date;
3
4       public class  DiscovererOfStellarObject  extends StellarObject {
5               String  discoverer;
6
7               public DiscovererOfStellarObject(String name, String discovere
8                       super(name, density);
9                       this.discoverer =  discoverer;
10              }
11              public String getDiscoverer()   {
12                      return discoverer;
13              }
14              public void setDiscoverer(String discoverer)    {
15                      this.discoverer = discoverer;
```

```
16              }
17              public String toString()        {
18                      return super.toString() + "/" + discoverer;
19              }
20              public static void main(String args[])  {
21                      DiscovererOfStellarObject aDiscovererOfStellarObject =
22                              new DiscovererOfStellarObject("Sylvis", "Norma
23                      System.out.println("aDiscovererOfStellarObject = " + a
24
25              }
26      }
27
28
```

Source Code: Src/26_1/DiscovererOfStellarObject.java

DiscovererOfStellarObjectInterface:

```
 1
 2      public interface  DiscovererOfStellarObjectInterface  {
 3
 4              public String getDiscoverer();
 5              public void setDiscoverer(String discoverer);
 6      }
 7
 8
```

Source Code: Src/26_1/DiscovererOfStellarObjectInterface.java

Planet:

```
 1      import java.util.Date;
 2
 3      public class  Planet  extends StellarObject {
 4
 5              double orbitalPeriod;
 6              int numberOfMoons;
 7
 8              public Planet(String name, double density, double orbitalPeriod
 9                      super(name,density);
10                      this.orbitalPeriod =  orbitalPeriod;
11                      this.numberOfMoons =  numberOfMoons;
12              }
13              public double getOrbitalPeriod()        {
14                      return orbitalPeriod;
15              }
16              public double getNumberOfMoons()        {
17                      return numberOfMoons;
18              }
19              public void setOrbitalPeriod(double orbitalPeriod)      {
20                      this.orbitalPeriod =  orbitalPeriod;
21              }
```

```
22              public void setNumberOfMoons(int numberOfMoons) {
23                      this.numberOfMoons =  numberOfMoons;
24              }
25              public String toString()        {
26                      return "Planet: " + super.toString() + "/" +  orbitalP
27              }
28              public static void main(String args[])  {
29                      Planet aPlanet = new Planet("Mercury", 5.427, 87.97, 0
30                      System.out.println(aPlanet);
31                      aPlanet.setName("Saturn");
32                      aPlanet.setDensity(0.687);
33                      aPlanet.setOrbitalPeriod(10759.22);
34                      aPlanet.setNumberOfMoons(82);
35                      System.out.println(aPlanet);
36
37                      System.out.println("1: " + aPlanet.getName() );
38                      System.out.println("2: " + aPlanet.getDensity() );
39                      System.out.println("3: " + aPlanet.getOrbitalPeriod()
40                      System.out.println("4: " + aPlanet.getNumberOfMoons()
41
42              }
43      }
44
45
```

Source Code: Src/26_1/Planet.java

StellarObject:

```
 1      import java.util.Date;
 2
 3      public class  StellarObject  implements StellarObjectInterface {
 4              String name;
 5              double  density;
 6
 7              public StellarObject(String name, double density)       {
 8                      this.name =  name;
 9                      this.density =  density;
10              }
11              public String getName() {
12                      return name;
13              }
14              public double getDensity()      {
15                      return density;
16              }
17              public void setName(String name)        {
18                      this.name =  name;
19              }
20              public void setDensity(double density)  {
21                      this.density =  density;
22              }
23              public String toString()        {
24                      return name + "/" + density;
25              }
```

```
26              public static void main(String args[])  {
27                      StellarObject aStellarObject = new StellarObject("TheT
28                      System.out.println(aStellarObject);
29              }
30      }
31
32
```

Source Code: Src/26_1/StellarObject.java

StellarObjectInterface:

```
1
2       public interface  StellarObjectInterface {
3               public String getName();
4               public void setName(String name);
5               public double getDensity();
6               public void setDensity(double density);
7       }
```

Source Code: Src/26_1/StellarObjectInterface.java

StorageOfAstronomicalObject

```
1       import java.util.Vector;
2       import java.util.Date;
3
4       public class  StorageOfAstronomicalObject<T extends StellarObjectInter
5               String nameOfTheStorageOfAstronomicalObject = "StorageOfAstron
6               private double densityOnAverage = 0;
7               Vector<T> objectsInSpace = new Vector<T>();
8               int soManyPlanetsSet;
9               public StorageOfAstronomicalObject()    {
10              }
11              public StorageOfAstronomicalObject(String nameOfTheStorageOfAs
12                      this.nameOfTheStorageOfAstronomicalObject = nameOfTheS
13              }
14              public void add(T theObject)    {
15                      objectsInSpace.add(theObject);
16                      soManyPlanetsSet++;
17                      calculateDensityOnAverage();
18              }
19              private void calculateDensityOnAverage() {
20                      double allDensity = 0;
21                      for ( int index = 0; index < objectsInSpace.size(); in
22                              if ( objectsInSpace.elementAt(index) != null )
23                                      allDensity += objectsInSpace.elementAt
24                              }
25                      }
26                      densityOnAverage = allDensity / soManyPlanetsSet;
27              }
28              public String getAllNames()     {
29                      String theNames = "";
```

```
30                      for ( int index = 0; index < objectsInSpace.size(); in
31                              if ( objectsInSpace.elementAt(index) != null )
32                                      theNames +=  objectsInSpace.elementAt
33                              }
34                      }
35                      return theNames;
36              }
37              public String toString()        {
38                      String theStorageOfAstronomicalObject = "";
39                      for ( int index = 0; index < objectsInSpace.size(); in
40                              if ( objectsInSpace.elementAt(index) != null )
41                                      theStorageOfAstronomicalObject += inde
42                              }
43                      }
44                      return theStorageOfAstronomicalObject + "\n     averag
45              }
46      }
47
48
```

Source Code: Src/26_1/StorageOfAstronomicalObject.java

Test:

```
 1      public class  Test      {
 2
 3              public static void print(
 4                      StorageOfAstronomicalObject<StellarObjectInterface> aS
 5                      System.out.println(aStorageOfAstronomicalObject.getAll
 6                      System.out.println(aStorageOfAstronomicalObject);
 7
 8              }
 9              public static void testBinaries()       {
10                      StorageOfAstronomicalObject<StellarObjectInterface> aS
11                              =  new StorageOfAstronomicalObject<StellarObje
12                      Binaries aBinaries = new Binaries("Sylvis", "Norman Ro
13                      Binaries bBinaries = new Binaries("Elektra", "C. H. F.
14
15                      aStorageOfAstronomicalObject.add(aBinaries);
16                      aStorageOfAstronomicalObject.add(bBinaries);
17
18                      bBinaries = new Binaries("Hermione", "James Craig Wats
19                      aStorageOfAstronomicalObject.add(bBinaries);
20
21                      print(aStorageOfAstronomicalObject);
22              }
23              public static void testAstroids()       {
24                      StorageOfAstronomicalObject<StellarObjectInterface> aS
25                              =  new StorageOfAstronomicalObject<StellarObje
26                      Asteroid aAsteroid = new Asteroid("Ceres", "Giuseppe P
27                      Asteroid bAsteroid = new Asteroid("Pallas", "Heinrich
28                      Asteroid cAsteroid = new Asteroid("Juno", "Karl Ludwig
29
30                      aStorageOfAstronomicalObject.add(aAsteroid);
```

```
31                      aStorageOfAstronomicalObject.add(bAsteroid);
32                      aStorageOfAstronomicalObject.add(cAsteroid);
33
34                      cAsteroid = new Asteroid("Vesta", "Heinrich Wilhelm Ol
35                      aStorageOfAstronomicalObject.add(cAsteroid);
36
37                      print(aStorageOfAstronomicalObject);
38              }
39              public static void testPlanets()        {
40                      StorageOfAstronomicalObject<StellarObjectInterface> aS
41                              = new StorageOfAstronomicalObject<StellarObjec
42
43                      aStorageOfAstronomicalObject.add(new Planet("Mercury",
44
45                      Planet aPlanet = new Planet("Saturn", 0.687, 10792, 82
46                      aStorageOfAstronomicalObject.add(aPlanet);;
47
48                      aPlanet.setName("Earth");
49                      aPlanet.setDensity(5.514);
50                      aPlanet.setOrbitalPeriod(365.256363004);
51                      aPlanet.setNumberOfMoons(1);
52                      aStorageOfAstronomicalObject.add(aPlanet);
53
54                      print(aStorageOfAstronomicalObject);
55              }
56              public static void main(String args[])  {
57                      testPlanets();
58                      testAstroids();
59                      testBinaries();
60              }
61      }
62
63
```

Source Code: Src/26_1/Test.java

26.2. Homework 6.2 (10 Points)

Objective: Working with generics

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

The objective is to write small examples using generics defined in Java. This homework is a modification of 6.1.

Your Work:

Hw 6.1 omits "Saturn". Why? Please submit your answer in a ".txt" file.

You have to modify your classes from hw 6.1 accordingly so such "Saturn" stays in the storage. This method produces

```
        public static void testPlanets()        {
                StorageOfAstronomicalObject  ... deleted ... aStorageOfAstronc
                        = new StorageOfAstronomicalObject ... deleted ...("Mil

                aStorageOfAstronomicalObject.add(new Planet("Mercury", 5.427,

                Planet aPlanet = new Planet("Saturn", 0.687, 10792, 82);
                aStorageOfAstronomicalObject.add(aPlanet);;

                aPlanet.setName("Earth");
                aPlanet.setDensity(5.514);
                aPlanet.setOrbitalPeriod(365.256363004);
                aPlanet.setNumberOfMoons(1);
                aStorageOfAstronomicalObject.add(aPlanet);

                System.out.println(aStorageOfAstronomicalObject.getAllNames())
                System.out.println(aStorageOfAstronomicalObject);
        }
```

produces the following output:

```
Press ENTER or type command to continue
Mercury, Saturn, Earth,
0: Mercury/5.427/87.97/0
1: Saturn/0.687/10792.0/82
2: Earth/5.514/365.256363004/1

             average density: 3.876
```

Explanation:

Requirements:

• You have to name the classes in the same way this write up does.

• You have to fix the problem in your *StorageOfAstronomicalObject* class.

• You have to explain why your solton fixes the issue.

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

Asteroid:

```
1       public class  Asteroid  extends DiscovererOfStellarObject {
2
3               public Asteroid(String name, String discoverer, double density
4                       super(name, discoverer, density);
5               }
6               public String toString()        {
7                       return super.toString();
```

```
 8              }
 9              public Asteroid clone() {
10                      return new Asteroid(name, discoverer, density);
11              }
12
13              public static void main(String args[])  {
14                      Asteroid aAsteroid = new Asteroid("Sylvis", "Norman Ro
15                      Asteroid bAsteroid = new Asteroid("Elektra", "C. H. F.
16
17                      System.out.println("aAsteroid = " + aAsteroid );
18                      System.out.println("bAsteroid = " + bAsteroid );
19                      System.out.println("bAsteroid = " + bAsteroid.clone()
20
21              }
22      }
```

Source Code: Src/26_2/Asteroid.java

Binaries:

```
 1      import java.util.Date;
 2
 3      public class  Binaries  extends Asteroid {
 4              int satelites;
 5
 6              public Binaries(String name, String discoverer, double density
 7                      super(name, discoverer, density);
 8                      this.satelites = satelites;
 9              }
10              public int getSatelites()       {
11                      return satelites;
12              }
13              public void setSatelites(int satelites) {
14                      this.satelites = satelites;
15              }
16              public String toString()        {
17                      return super.toString() + "/" + satelites;
18              }
19              public Binaries clone() {
20                      return new Binaries(name, discoverer, density, satelit
21              }
22
23              public static void main(String args[])  {
24                      Binaries aBinaries = new Binaries("Sylvis", "Norman Ro
25                      Binaries bBinaries = new Binaries("Elektra", "C. H. F.
26
27                      System.out.println("aBinaries = " + aBinaries );
28                      System.out.println("bBinaries = " + bBinaries );
29
30              }
31      }
32
33
```

Source Code: Src/26_2/Binaries.java

DiscovererOfStellarObject:

```
 1
 2      public class  DiscovererOfStellarObject  extends ObjectInSpace {
 3
 4              String  discoverer;
 5
 6              public DiscovererOfStellarObject(String name, String discovere
 7                      super(name, density);
 8                      this.discoverer =  discoverer;
 9              }
10              public String getDiscoverer()   {
11                      return discoverer;
12              }
13              public void setDiscoverer(String discoverer)    {
14                      this.discoverer = discoverer;
15              }
16              public String toString()        {
17                      return super.toString() + "/" + discoverer;
18              }
19              public DiscovererOfStellarObject clone()        {
20                      return new DiscovererOfStellarObject(name, discoverer,
21              }
22              public static void main(String args[])  {
23                      DiscovererOfStellarObject aDiscovererOfStellarObject =
24                              new DiscovererOfStellarObject("Sylvis", "Norma
25                      System.out.println("aDiscovererOfStellarObject = " + a
26
27              }
28      }
```

Source Code: Src/26_2/DiscovererOfStellarObject.java

ObjectInSpace:

```
 1      import java.util.Date;
 2
 3      public class  ObjectInSpace  implements ObjectInSpaceInterface<ObjectI
 4              String name;
 5              double  density;
 6
 7              public ObjectInSpace(String name, double density)       {
 8                      this.name =  name;
 9                      this.density =  density;
10              }
11              public String getName() {
12                      return name;
13              }
14              public double getDensity()      {
15                      return density;
16              }
```

```
17              public void setName(String name)        {
18                      this.name =  name;
19              }
20              public void setDensity(double density)  {
21                      this.density =  density;
22              }
23              public String toString()        {
24                      return name + "/" + density;
25              }
26              public ObjectInSpace clone()    {
27                      return new ObjectInSpace(name, density);
28              }
29
30              public static void main(String args[])  {
31                      ObjectInSpace aObjectInSpace = new ObjectInSpace("Merc
32                      System.out.println(aObjectInSpace);
33              }
34      }
35
36
```

Source Code: Src/26_2/ObjectInSpace.java

ObjectInSpaceInterface:

```
1
2       public interface  ObjectInSpaceInterface<T> {
3               public String getName();
4               public double getDensity();
5               public T clone();
6       }
```

Source Code: Src/26_2/ObjectInSpaceInterface.java

Planet:

```
1       import java.util.Date;
2
3       public class  Planet  extends ObjectInSpace {
4               double  orbitalPeriod;
5               int    numberOfMoons;
6
7               public Planet(String name, double density, double orbitalPeric
8                       super(name, density);
9                       this.orbitalPeriod =  orbitalPeriod;
10                      this.numberOfMoons =  numberOfMoons;
11              }
12              public double getOrbitalPeriod()        {
13                      return orbitalPeriod;
14              }
15              public double getNumberOfMoons()        {
16                      return numberOfMoons;
17              }
```

```
18              public void setOrbitalPeriod(double orbitalPeriod)      {
19                      this.orbitalPeriod =  orbitalPeriod;
20              }
21              public void setNumberOfMoons(int numberOfMoons) {
22                      this.numberOfMoons =  numberOfMoons;
23              }
24              public String toString()        {
25                      return super.toString() +  "/" + orbitalPeriod + "/" +
26              }
27              public Planet clone()   {
28                      return new Planet(name, density, orbitalPeriod, number
29              }
30
31              public static void main(String args[])  {
32                      Planet aPlanet = new Planet("Mercury", 5.427, 87.97, 0
33                      System.out.println(aPlanet);
34                      aPlanet.setName("Saturn");
35                      aPlanet.setDensity(0.687);
36                      aPlanet.setOrbitalPeriod(10759.22);
37                      aPlanet.setNumberOfMoons(82);
38                      System.out.println(aPlanet);
39
40                      System.out.println("1: " + aPlanet.getName() );
41                      System.out.println("2: " + aPlanet.getDensity() );
42                      System.out.println("3: " + aPlanet.getOrbitalPeriod()
43                      System.out.println("4: " + aPlanet.getNumberOfMoons()
44
45              }
46      }
47
48
```

Source Code: Src/26_2/Planet.java

StorageOfAstronomicalObject:

```
 1      import java.util.Vector;
 2      import java.util.Date;
 3
 4      public class  StorageOfAstronomicalObject<T extends ObjectInSpaceInter
 5              String nameOfTheStorageOfAstronomicalObject = "StorageOfAstron
 6              private double densityOnAverage = 0;
 7              Vector<T> objectsInSpace = new Vector<T>();
 8              int soManyStellarObjects;
 9              public StorageOfAstronomicalObject()    {
10              }
11              public StorageOfAstronomicalObject(String nameOfTheStorageOfAs
12                      this.nameOfTheStorageOfAstronomicalObject = nameOfTheS
13              }
14              public void add(T theObject)    {
15                      //objectsInSpace.add(theObject.clone());
16                      objectsInSpace.add(theObject.clone());
17                      soManyStellarObjects++;
18                      calculateDensityOnAverage();
```

```
19              }
20              private void calculateDensityOnAverage() {
21                      double allDensity = 0;
22                      for ( int index = 0; index < objectsInSpace.size(); in
23                              if ( objectsInSpace.elementAt(index) != null )
24                                      allDensity += objectsInSpace.elementAt
25                              }
26                      }
27                      densityOnAverage = allDensity / soManyStellarObjects;
28              }
29              public String getAllNames()     {
30                      String theNames = "";
31                      for ( int index = 0; index < objectsInSpace.size(); in
32                              if ( objectsInSpace.elementAt(index) != null )
33                                      theNames +=  objectsInSpace.elementAt
34                              }
35                      }
36                      return theNames;
37              }
38              public String toString()        {
39                      String theStorageOfAstronomicalObject = "";
40                      for ( int index = 0; index < objectsInSpace.size(); in
41                              if ( objectsInSpace.elementAt(index) != null )
42                                      theStorageOfAstronomicalObject += inde
43                              }
44                      }
45                      return theStorageOfAstronomicalObject + "\n     averag
46              }
47      }
48
49
```

Source Code: Src/26_2/StorageOfAstronomicalObject.java

Test:

```
1       public class  Test      {
2
3               public static void testBinaries()       {
4                       StorageOfAstronomicalObject<ObjectInSpace> aStorageOfA
5                               =  new StorageOfAstronomicalObject<ObjectInSpa
6                       Binaries aBinaries = new Binaries("Sylvis", "Norman Ro
7                       Binaries bBinaries = new Binaries("Elektra", "C. H. F.
8
9                       aStorageOfAstronomicalObject.add(aBinaries);
10                      aStorageOfAstronomicalObject.add(bBinaries);
11
12                      bBinaries = new Binaries("Hermione", "James Craig Wats
13                      aStorageOfAstronomicalObject.add(bBinaries);
14
15                      System.out.println(aStorageOfAstronomicalObject.getAll
16                      System.out.println(aStorageOfAstronomicalObject);
17              }
18              public static void testAstroids()       {
```

```
19                      StorageOfAstronomicalObject<ObjectInSpace> aStorageOfA
20                              =  new StorageOfAstronomicalObject<ObjectInSpa
21                      Asteroid aAsteroid = new Asteroid("Ceres", "Giuseppe P
22                      Asteroid bAsteroid = new Asteroid("Pallas", "Heinrich
23                      Asteroid cAsteroid = new Asteroid("Juno", "Karl Ludwig
24
25                      aStorageOfAstronomicalObject.add(aAsteroid);
26                      aStorageOfAstronomicalObject.add(bAsteroid);
27                      aStorageOfAstronomicalObject.add(cAsteroid);
28
29                      cAsteroid = new Asteroid("Vesta", "Heinrich Wilhelm Ol
30                      aStorageOfAstronomicalObject.add(cAsteroid);
31
32                      System.out.println(aStorageOfAstronomicalObject.getAll
33                      System.out.println(aStorageOfAstronomicalObject);
34              }
35              public static void testPlanets()        {
36                      StorageOfAstronomicalObject<ObjectInSpace> aStorageOfA
37                              = new StorageOfAstronomicalObject<ObjectInSpac
38
39                      aStorageOfAstronomicalObject.add(new Planet("Mercury",
40
41                      Planet aPlanet = new Planet("Saturn", 0.687, 10792, 82
42                      aStorageOfAstronomicalObject.add(aPlanet);;
43
44                      aPlanet.setName("Earth");
45                      aPlanet.setDensity(5.514);
46                      aPlanet.setOrbitalPeriod(365.256363004);
47                      aPlanet.setNumberOfMoons(1);
48                      aStorageOfAstronomicalObject.add(aPlanet);
49
50                      System.out.println(aStorageOfAstronomicalObject.getAll
51                      System.out.println(aStorageOfAstronomicalObject);
52              }
53              public static void main(String args[])  {
54                      // testPlanets();
55                      // testAstroids();
56                      testBinaries();
57              }
58      }
59
60
```

Source Code: Src/26_2/Test.java

26.3. Homework 6.3 (10 Points)

Objective: Working with generics

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

The objective is to translate Generic expressions into english. generics defined in Java.

```
boolean addAll(Collection<? extends E> c)
boolean containsAll(Collection<?> c)
static <T> void fill(List<? super T> list, T obj)
static <T extends Comparable<? super T>> void sort(List<T> list)
static <T> void sort(List<T> list, Comparator<? super T> c)
static void swap(List<?> list, int i, int j)
static <K, V> Map<K,V> synchronizedMap(Map<K,V> m)
static void reverse(List<?> list)
```

Given are the following declarations. Describe each line in english.

Requirements:

You have to submit your solutions in a ".txt" file.

Submission:

Submit your files via myCourses.

27. Homework 7

Posted: August/21/2020
Due: October/11/2020 24.00

The solutions for this homework are due October/11/2020 24.00. I recommend to submit at least one version of all homework solutions long before due date.

27.1. Homework 7.1 (10 Points)

Objective: Working with generics

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

The objective is to write small examples for generics defined in Java.

Given is the following program:

```
 1      import java.util.Comparator ;
 2      import java.util.* ;
 3
 4      public class Test {
 5          public static void main(String args[]) {
 6              List<Address> aListOfAddresses  = new ArrayList<Address>();
 7              List<LP> aListOfLPs             = new ArrayList<LP>();
 8              aListOfAddresses.add(  new Address(1600, "Pennsylvania Avenue
 9              aListOfAddresses.add(  new Address(11, "Wall Street", "New Yor
10              aListOfAddresses.add(  new Address(102, "Lomb Memorial Drive",
11              aListOfAddresses.add(  new Address(1, "A", "B", "C", 1) );
12              aListOfAddresses.add(  new Address(2, "A", "B", "C", 1) );
```

```
13              aListOfAddresses.add(  new Address(3, "A", "B", "C", 1) );
14              aListOfAddresses.add(  new Address(4, "A", "B", "C", 1) );
15
16              aListOfLPs.add( new LP( 1960, "Deep Purple in Rock", "Deep Pur
17              aListOfLPs.add( new LP( 1973, "Dark Side of the Moon", "Pink F
18              aListOfLPs.add( new LP( 1, "A", "B ", (float)3, 4));
19              aListOfLPs.add( new LP( 2, "A", "B ", (float)3, 4));
20              aListOfLPs.add( new LP( 3, "A", "B ", (float)3, 4));
21              aListOfLPs.add( new LP( 0, "A", "B ", (float)3, 4));
22
23              Collections.sort(aListOfAddresses);
24              Collections.sort(aListOfLPs);
25              System.out.println(aListOfAddresses);
26              System.out.println(aListOfLPs);
27          }
28      }
29
```

Source Code: Src/27/Test.java

Design and implement the classes LP, and Address.

Your Work:

It might be a good idea to read the java doc pages for: ArrayList (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/ArrayList.html) and Collections (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/Collections.html)

The java doc page of these classes provide all the information you need.

The functionality of all the methods you design and write is up to you. It should be possible to compile your classes with my provided test program.

You must be able to execute Collections.sort(aListOfAddresses) in your test program.

Requirements:

• Your program should compile, without warnings, with the provided Test.java program.

• You have to name the source code LP.java and Address.java

• You have to use -Xlint to make sure your use of generics is correct.

• You must be able to sort the storage space used with Collections.sort(aListOfAddresses).

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

First:

```
1       public class Address implements Comparable<Address>     {
2
3               int houseNumber;
4               String streetName;
5               String nameOfTown;
```

```
 6              String state;
 7              int zipCode;
 8
 9              public Address()        {
10              }
11              public Address(int houseNumber, String streetName, String name
12                      this.houseNumber  = houseNumber;
13                      this.streetName   = streetName;
14                      this.nameOfTown   = nameOfTown;
15                      this.state        = state;
16                      this.zipCode      = zipCode;
17              }
18              public String toString()        {
19                      return houseNumber + "/" + streetName + "/" + nameOfTo
20              }
21              public int compareTo(Address addr)      {
22                      return toString().compareTo(addr.toString());
23              }
24      }
```

Source Code: Src/27_1/Address.java

Second:

```
 1      public class LP implements Comparable<LP>       {
 2
 3              int year;
 4              String artist;
 5              String title;
 6              float length;
 7              int tracks;
 8
 9              public LP()     {
10              }
11              public LP(int year, String title, String artist, float length,
12                      this.year   = year;
13                      this.title  = title;
14                      this.artist = artist;
15                      this.length = length;
16                      this.tracks = tracks;
17              }
18              public String toString()        {
19                      return year + "/" + title + "/" + artist + "/" + lengt
20              }
21              public int compareTo(LP addr)   {
22                      return toString().compareTo(addr.toString());
23              }
24      }
```

Source Code: Src/27_1/LP.java

27.2. Homework 7.2 (10 Points)

Objective: Working with generics

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

The objective is to write small examples for generics defined in Java.

Explanation:

The following, incomplete, interface:

```
public interface StorageInterface ... deleted ...
   void add(E x);                // true if it was successfully added, false ot
   boolean find(E x);            // true if x could be found, false otherwise
   boolean includesNull();       // true, if the storage include a null element
   boolean delete(E x);          // true if it was successfully deleted, false
}
```

describes what kind of objects can be inserted in classe(s) you have to implement.

One implementation of this interface is called *SortedStorage*. You can not use an existing java class, like an ArrayList etc,. You have to implement a binary search tree to store the not null objects. You have to test implementation with Strings and Integers and both class you designed and implemented for hw 27.1.

The requirements for *SortedStorage* are:

• It must be possible to add *null* to the storage; and it must be possible to determine if a *null* element has been added to the storage or not.

• Assume the following scenario:

```
aStorageInterfaceString.add("1");
aStorageInterfaceString.add("1");
aStorageInterfaceString.add(new String("1"));
aStorageInterfaceString.delete(new String("1"));
aStorageInterfaceString.delete("1");
```

which elements will be deleted when?

• It must be possible to add an object more than once.

• It must be possible to add a null element more than once, and delete as often as it was added.

• Your implementation of SortedStorage must keep the elements sorted at any given moment. This means you can not insert an element and then sort it because the elements are not sorted directly after you inserted the element.

Your Work:

You have to be able to explain why your solution meets the requirement regarding being sorted at any given moment in time.

You have to implemented a BST for your solution. This is part of my delete implementation minus the generics:

```
private Node ... deleted ... minimumElement(Node thisNode) {
        if (thisNode.left == null)
                return thisNode;
        else {
                return minimumElement(thisNode.left);
        }
}

private Node ... deleted ... deleteThisElementInTree(Node root, /* del
        if ( root == null )
                return null;
        if ( root.payLoad.compareTo(payLoad) > 0 ) {    // see insert
                root.left = deleteThisElementInTree(root.left, payLoad
        } else if ( root.payLoad.compareTo(payLoad) < 0 ) {     // see
                root.right = deleteThisElementInTree(root.right, payLo
        } else {
                if ( (root.left != null) && (root.right != null) ) {
                        Node tmp = root;
                        Node minimumNodeOnRight = minimumElement(tmp.r
                        root.payLoad = minimumNodeOnRight.payLoad;
                        root.counter = minimumNodeOnRight.counter;
                        root.right = deleteThisElementInTree(root.righ
                } else if (root.left != null) {
                        root = root.left;
                } else if (root.right != null) {
                        root = root.right;
                } else {
                        root = null;
                }
        }
        return root;
}
```

You should use interface definition when ever possible. The following example:

```
public static void testStringBuild( ... deleted ... aStorageInterfaceS

        String three = new String("3");

        aStorageInterfaceString.add(null);
        aStorageInterfaceString.add(null);
        aStorageInterfaceString.add("3");
        aStorageInterfaceString.add(three);
        aStorageInterfaceString.add("1");
        aStorageInterfaceString.add("2");
        System.out.println("1." + aStorageInterfaceString);
}
public static void testStringManipulate( ... deleted ... aStorageInter
        System.out.println("aStorageInterfaceString.find(null) " + aSt
        System.out.println("aStorageInterfaceString.delete(null) " + a
        System.out.println("aStorageInterfaceString.delete(1) " + aSto
        System.out.println("aStorageInterfaceString.delete(0) " + aSto
        System.out.println("2." + aStorageInterfaceString);
}
```

```
        public static void testInteger(StorageInterface ... deleted ... aStora

                aStorageInterfaceInteger.add(null);
                aStorageInterfaceInteger.add(null);
                aStorageInterfaceInteger.add(Integer.valueOf("55"));
                aStorageInterfaceInteger.add(Integer.valueOf("33"));
                aStorageInterfaceInteger.add(Integer.valueOf("66"));
                System.out.println("3: " + aStorageInterfaceInteger);
        }
        public static void testAddress( ... deleted ... aStorageInterfaceAddre
                aStorageInterfaceAddress.add(  new Address(1600, "Pennsylvania
                aStorageInterfaceAddress.add(  new Address(11, "Wall Street",
                aStorageInterfaceAddress.add(  new Address(102, "Lomb Memorial
                aStorageInterfaceAddress.add(  new Address(1, "A", "B", "C", 1
                System.out.println("4: " + aStorageInterfaceAddress);
       }
       public static void testLP( ... deleted ... aStorageInterfaceLP)     {
                aStorageInterfaceLP.add( new LP( 1960, "Deep Purple in Rock",
                aStorageInterfaceLP.add( new LP( 1973, "Dark Side of the Moon'
                System.out.println("aStorageInterfaceLP: " + aStorageInterface
        }
        public static void main(String args[] ) {
             ... deleted ...
                StorageInterface ... deleted  aStorageInterfaceInteger = new S
                StorageInterface ... deleted  aStorageInterfaceAddress = new S
                StorageInterface ... deleted  aStorageInterfaceLP = new Sorted
                testStringBuild(aStorageInterfaceString);
                testStringManipulate(aStorageInterfaceString);
                testInteger(aStorageInterfaceInteger);
                testAddress(aStorageInterfaceAddress);
                testLP(aStorageInterfaceLP);
        }
```

would produce something like:

```
1.
       includes so many null values = 2
       Values stored:  (  ( l: null 1/1 r:  ( l: null 2/1 r:  null  )   )  3/2
aStorageInterfaceString.find(null) true
aStorageInterfaceString.delete(null) true
aStorageInterfaceString.delete(1) true
aStorageInterfaceString.delete(0) false
2.
       includes so many null values = 1
       Values stored:  (  ( l: null 2/1 r:  null  )  3/2 r:  null  )
aStorageInterfaceInteger:
       includes so many null values = 2
       Values stored:  (  ( l: null 33/1 r:  null  )  55/1 r:  ( l: null 66/1
aStorageInterfaceAddress:
       includes so many null values = 0
       Values stored:  (  (  (  ( l: null 1/A/B/C/1
/1 r:  null  )  102/Lomb Memorial Drive/Rochester/NY/14623
/1 r:  null  )  11/Wall Street/New York/NY/10118
/1 r:  null  )  1600/Pennsylvania Avenue NW/Washington/DC/20500
```

```
/1 r:  null  )
aStorageInterfaceLP:
       includes so many null values = 0
       Values stored:  ( l: null 1960/Deep Purple in Rock/Deep Purple/43.3/7
/1 r:  ( l: null 1973/Dark Side of the Moon/Pink Floyd /43.09/10
/1 r:  null  )   )
```

You need to provide and submit a test environment

Requirements:

• You have to submit a reasonable test environment, i.e. code which tests your implementation.

• You might like to submit more class files, because of the your implementation might benefit from it.

• You can not use in your code: *@SuppressWarnings("unchecked")*.

• You have to use -Xlint to make sure your use of generics is correct.

• No warnings should be produced compiling your code.

• You may use only what we have covered in class.

• Your implementations must compile with the following test program.

```
 1
 2      public class TestInterfaceImplementation {
 3              public static void testStringSet(StorageInterface<String> aSto
 4
 5                      String three = new String("3");
 6
 7                      System.out.println("add(null) " + aStorageInterfaceStr
 8                      System.out.println("add(null) " + aStorageInterfaceStr
 9                      System.out.println("add(3) " + aStorageInterfaceString
10                      System.out.println("add(three) " + aStorageInterfaceSt
11                      System.out.println("add(1) " + aStorageInterfaceString
12                      System.out.println("add(2) " + aStorageInterfaceString
13                      System.out.println("add(2) " + aStorageInterfaceString
14                      System.out.println("1." + aStorageInterfaceString);
15              }
16              public static void testStringBuild(StorageInterface<String> aS
17
18                      String three = new String("3");
19
20                      aStorageInterfaceString.add(null);
21                      aStorageInterfaceString.add(null);
22                      aStorageInterfaceString.add("3");
23                      aStorageInterfaceString.add(three);
24                      aStorageInterfaceString.add("1");
25                      aStorageInterfaceString.add("2");
26                      System.out.println("1." + aStorageInterfaceString);
27              }
28              public static void testStringManipulate(StorageInterface<Strin
29                      System.out.println("aStorageInterfaceString.find(null)
30                      System.out.println("aStorageInterfaceString.delete(nul
31                      System.out.println("aStorageInterfaceString.delete(1)
32                      System.out.println("aStorageInterfaceString.delete(0)
```

```
33                      System.out.println("2." + aStorageInterfaceString);
34              }
35              public static void testInteger(StorageInterface<Integer> aStor
36
37                      aStorageInterfaceInteger.add(null);
38                      aStorageInterfaceInteger.add(null);
39                      aStorageInterfaceInteger.add(Integer.valueOf("55"));
40                      aStorageInterfaceInteger.add(Integer.valueOf("33"));
41                      aStorageInterfaceInteger.add(Integer.valueOf("66"));
42                      System.out.println("aStorageInterfaceInteger: " + aSto
43              }
44              public static void testAddress(StorageInterface<Address> aStor
45                      aStorageInterfaceAddress.add(  new Address(1600, "Penn
46                      aStorageInterfaceAddress.add(  new Address(11, "Wall S
47                      aStorageInterfaceAddress.add(  new Address(102, "Lomb
48                      aStorageInterfaceAddress.add(  new Address(1, "A", "B"
49                      System.out.println("aStorageInterfaceAddress: " + aSto
50              }
51              public static void main(String args[] ) {
52                      StorageInterface<String> aStorageInterfaceString = new
53                      StorageInterface<Integer> aStorageInterfaceInteger = n
54                      StorageInterface<Address> aStorageInterfaceAddress = n
55                      testStringBuild(aStorageInterfaceString);
56                      testStringManipulate(aStorageInterfaceString);
57                      testInteger(aStorageInterfaceInteger);
58                      testAddress(aStorageInterfaceAddress);
59              }
60      }
61
```

Source Code: Src/27/TestInterfaceImplementation.java

Submission:

Submit your files via myCourses.

Interface:

```
1       public interface StorageInterface<E extends Comparable<E>> {
2          boolean add(E x);
3          boolean find(E x);
4          boolean includesNull();
5          boolean delete(E x);
6       }
```

Source Code: Src/27_2/StorageInterface.java

Implementation:

```
1       import java.util.* ;
2
3
4       public class SortedStorage<E extends Comparable<E>>
5                                       implements StorageInterface<E>  {
```

```
 6
 7              private final int MAX = 10;
 8              private int size = 0;
 9
10              private class Node<E>   {
11                      Node()  {
12                      }
13                      Node(E element) {
14                              payLoad = element;
15                              counter = 1;
16                              left = right = null;
17                      }
18                      private Node<E> left;
19                      private Node<E> right;
20                      E payLoad= null;
21                      int counter = 0;
22              }
23
24              Node<E> root;
25              int soManyNulls = 0;
26
27              public SortedStorage( ) {
28                      root = null;
29              }
30              public int howManyNulls()       {
31                      return soManyNulls;
32              }
33              public boolean includesNull()   {
34                      return soManyNulls > 0;
35              }
36              public boolean delete( E element)       {
37                      boolean rValue = false;
38                      if ( element == null )  {
39                              if ( soManyNulls > 0 )  {
40                                      soManyNulls --;
41                                      rValue = true;
42                              } else
43                                      rValue = false;
44                      } else {
45                              rValue = deleteElementInTree(element);
46                      }
47                      return rValue;
48              }
49              private boolean deleteElementInTree(E element)  {
50                      boolean rValue = false;
51                      Node<E> aNode = null;
52                      if ( root == null )
53                              rValue = false;
54                      else {
55                              aNode = findThisElementInTree(root, element);
56                              if ( aNode == null )
57                                      rValue = false;
58                              else {
59                                      aNode.counter--;
```

```
60                                      rValue = true;
61                                      if ( aNode.counter == 0 )       {
62                                              root = deleteThisElementInTree
63                                      }
64                              }
65                      }
66                      return ( rValue );
67              }
68              private Node<E> minimumElement(Node<E> thisNode) {
69                      if (thisNode.left == null)
70                              return thisNode;
71                      else {
72                              return minimumElement(thisNode.left);
73                      }
74              }
75
76              private Node<E> deleteThisElementInTree(Node<E> root, E payLoa
77                      if ( root == null )
78                              return null;
79                      if ( root.payLoad.compareTo(payLoad) > 0 ) {    // see
80                              root.left = deleteThisElementInTree(root.left,
81                      } else if ( root.payLoad.compareTo(payLoad) < 0 ) {
82                              root.right = deleteThisElementInTree(root.righ
83                      } else {
84                              if ( (root.left != null) && (root.right != nul
85                                      Node<E> tmp = root;
86                                      Node<E> minimumNodeOnRight = minimumEl
87                                      root.payLoad = minimumNodeOnRight.payL
88                                      root.counter = minimumNodeOnRight.coun
89                                      root.right = deleteThisElementInTree(r
90                              } else if (root.left != null) {
91                                      root = root.left;
92                              } else if (root.right != null) {
93                                      root = root.right;
94                              } else {
95                                      root = null;
96                              }
97                      }
98                      return root;
99              }
00              public boolean find(E element)  {
01      /*
02                      return ( ( ( element == null ) && ( howManyNulls() > 0
03                      findElementInTree(root, element) );
04      */
05                      if ( ( element == null ) && ( howManyNulls() > 0 ) )
06                              return true;
07                      else
08                              return findElementInTree(root, element);
09              }
10              private boolean findElementInTree(Node<E> node, E element)
11                      if ( element == null )
12                              return false;
13                      Node<E> resultNode =  findThisElementInTree(node, elem
```

```
14                      return  ( resultNode != null );
15
16              }
17              private Node<E> findThisElementInTree(Node<E> node, E element)
18                      if ( node == null )     {
19                              return null;
20                      } else if ( node.payLoad.compareTo(element) == 0)
21                              return node;
22                      } else if ( node.payLoad.compareTo(element) > 0 ) {
23                                      return findThisElementInTree(node.left
24                      } else {
25                                      return findThisElementInTree(node.righ
26                      }
27
28              }
29              public boolean add(E element)   {
30                      boolean rValue;
31                      if (  find(element) )   {
32                              rValue = false;
33                      } else {
34                              if ( element == null )
35                                      soManyNulls++;
36                              else
37                                      addElementToTree(element);
38                              rValue = true;
39                      }
40                      return rValue;
41              }
42              private void addElementToTree(E element)  {
43                      if ( root == null )     {
44                              root = new Node<E>(element);
45                      } else  {
46                              addThisElementToTree(root, element);
47                      }
48              }
49              private void addThisElementToTree(Node<E> node, E element)  {
50                      if (  node.payLoad.compareTo(element) == 0 )
51                              node.counter ++;
52                      else if ( node.payLoad.compareTo(element) > 0 ) {  //
53                              if ( node.left == null )        {
54                                      node.left = new Node<E>(element);
55                              } else
56                                      addThisElementToTree(node.left, elemen
57                      } else {
58                              if ( node.right == null )       {
59                                      node.right = new Node<E>(element);
60                              } else
61                                      addThisElementToTree(node.right, eleme
62                      }
63              }
64              public List<E>getList()  {
65                      List<E> theResultsAsList = new ArrayList<E>();
66
67                      fillList(root, theResultsAsList);
```

```
68                      return theResultsAsList;
69              }
70              private void fillList(Node<E> node, List<E> theResultsAsList)
71                      if ( node != null ) {
72                              if ( node.left != null )
73                                      fillList(node.left, theResultsAsList);
74
75                              theResultsAsList.add(node.payLoad);
76
77                              if ( node.right != null )
78                                      fillList(node.right, theResultsAsList)
79                      }
80              }
81              private String parse(Node<E> node)  {
82                      String rValue = "";
83                      if ( node != null ) {
84                              rValue = " ( ";
85                              if ( node.left == null )
86                                      rValue += "l: null ";
87                              else
88                                      rValue += parse(node.left) + " ";
89
90                              rValue += node.payLoad + "/" + node.counter +
91
92                              if ( node.right == null )
93                                      rValue += "r:  null ";
94                              else
95                                      rValue += "r: " + parse(node.right) +
96                              rValue = rValue + " ) ";
97                      }
98                      return rValue;
99      // Values stored: +l: +l: null 3/1 r:  null  5/1 r: +l: null 6/1 r:  r
00              }
01              public String toString()  {
02                      String rValue = "\n     includes so many null values =
03                      "\n     Values stored: " +  parse(root);
04                      return rValue;
05              }
06
07      }
08
```

Source Code: Src/27_2/SortedStorage.java

27.3. Homework 7.3 (10 Points)

Objective: Working with Exceptions

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

The objective is to understanding a program with differeint kind of exceptions.

Given is the following program:

```
 1
 2      public class F {
 3
 4        private int noSystemExit()     {
 5                      try {
 6                              throw new Exception("1");
 7                      } catch (Exception e)   {
 8                              System.out.println("2");
 9                              return 0;
10                      } finally       {
11                              System.out.println("3 finally");
12                              return 1;
13                      }
14                      return 3;
15        }
16        private int noExeption()       {
17                      try {
18                              int x = 1 - 1;
19                              System.out.println("inside try: 1");
20                              return x;
21                      } catch (Exception e)   {
22                              System.out.println("inside catch: 2");
23                              return 1;
24                      } finally       {
25                              System.out.println("inside finally: 3 ");
26                              return 2;
27                      }
28        }
29        private int anExeption1()      {
30                      int[] anArray  = new int[1];
31                      try {
32                              anArray[2] = 1 / 0;
33                              System.out.println("inside try: 1");
34                              return 0;
35                      } catch (ArithmeticException e) {
36                              anArray[2] = 0;
37                              System.out.println("inside catch: 2");
38                              return 1;
39                      } finally       {
40                              System.out.println("inside finally: 3 ");
```

```
41                              return 2;
42                      }
43                      // return 3;
44      }
45      private int anExeption2()      {
46                      int[] anArray  = new int[1];
47                      try {
48                              anArray[2] = 0;
49                              anArray[2] = 1 / 0;
50                              System.out.println("inside try: 1");
51                              return 0;
52                      } catch (ArithmeticException e) {
53                              System.out.println("inside catch: 2");
54                              return 1;
55                      } finally       {
56                              System.out.println("inside finally: 3 ");
57                              return 2;
58                      }
59                      // return 3;
60      }
61      private void withSystemExit()  {
62                      try {
63                              throw new Exception("4");
64                      } catch (Exception e)   {
65                              System.out.println("5");
66                              System.exit(0);
67                      } finally       {
68                              System.out.println("6 finally");
69                      }
70                      System.out.println("exit(): you will not see this line
71      }
72
73      public static void main(String[] args) {
74            int r = new F().noSystemExit();
75            System.out.println(new F().noSystemExit());
76            new F().noExeption();
77            new F().anExeption1();
78            new F().anExeption2();
79            new F().withSystemExit();
80      }
81    }
```

Source Code: Src/27/F.java

Requirements:

• This program does not compile as is. Why?

• Explain the execution of each method in writing. You have to name the file 27_3.txt

Submission:

Submit your files via myCourses.

28. Homework 8

Posted: October/1/2020
Due: October/18/2020 24.00

The solutions for this homework are due October/18/2020 24.00. I recommend to submit at least one version of all homework solutions long before due date.

28.1. Homework 8.1 (10 Points)

Objective: Working with Files

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

The objective is to reading and processing a file.

Analyse the "NY Pick 10" number distribution.

Your Work:

The data can be found here:

```
 1      Draw Date,Winning Numbers
 2      09/30/2020,01 04 07 10 16 19 28 34 36 40 46 47 51 52 59 62 64 67 72 77
 3      09/29/2020,04 15 18 20 21 23 24 26 29 30 34 39 40 42 49 50 54 56 68 78
 4      09/28/2020,02 11 16 18 22 23 39 41 57 58 59 60 63 67 69 70 72 73 77 80
 5      09/27/2020,03 10 15 21 26 30 32 37 45 47 51 52 55 57 62 65 70 74 76 78
 6      09/26/2020,04 06 12 14 20 21 22 28 32 33 34 37 40 41 43 47 49 72 75 79
 7      09/25/2020,03 05 06 07 08 19 20 21 22 24 25 33 41 48 50 59 68 73 78 80
 8      09/24/2020,06 15 17 20 22 23 27 31 32 40 43 48 49 53 55 60 62 73 74 76
 9      09/23/2020,08 19 30 34 36 39 41 45 46 54 63 64 65 68 69 70 72 76 77 78
10      09/22/2020,01 02 03 05 16 24 25 28 30 33 35 52 54 57 60 65 68 73 74 78
11      09/21/2020,02 09 16 18 30 31 33 39 42 52 54 55 56 57 58 59 68 69 74 78
12      09/20/2020,06 09 12 16 18 19 23 28 31 35 38 48 49 57 63 65 69 72 74 76
13      09/19/2020,04 06 08 14 15 17 18 19 26 28 34 35 36 38 39 42 47 60 62 75
14      09/18/2020,02 07 12 17 18 22 23 24 26 28 31 32 37 40 46 57 61 66 67 72
15      09/17/2020,05 08 14 21 22 25 26 30 37 38 43 47 49 50 52 56 57 59 60 64
16      09/16/2020,01 04 06 14 24 27 32 38 40 42 45 49 52 56 57 63 68 69 71 78
17      09/15/2020,09 16 20 21 28 29 30 36 38 40 48 49 52 56 61 62 67 68 79 80
18      09/14/2020,05 09 12 13 14 15 16 17 18 25 32 46 53 63 64 67 71 74 76 80
19      09/13/2020,03 07 13 16 19 24 26 27 32 33 38 47 49 53 55 57 58 61 63 71
20      09/12/2020,01 05 14 16 19 25 28 31 34 40 43 47 48 51 52 53 54 66 70 73
21      09/11/2020,02 05 17 21 22 25 27 33 34 39 49 50 54 56 61 62 71 75 77 80
22      09/10/2020,06 08 15 19 21 23 24 26 30 36 44 45 47 49 57 58 61 65 68 74
23      09/09/2020,01 05 07 08 09 27 30 34 39 40 41 43 45 50 51 66 67 72 76 79
24      09/08/2020,02 05 14 17 19 20 26 27 32 38 44 52 55 60 63 67 72 73 75 78
25      09/07/2020,03 04 11 14 29 32 34 35 36 38 48 49 50 53 55 56 59 65 69 71
26      09/06/2020,01 03 07 08 17 20 26 29 37 40 42 43 44 46 49 53 60 66 72 75
27      09/05/2020,03 04 08 24 31 32 33 36 45 47 53 57 58 61 64 68 70 72 74 75
28      09/04/2020,06 10 12 15 19 20 23 26 33 42 46 47 48 49 50 51 57 62 63 72
29      09/03/2020,06 07 09 12 14 15 33 34 35 37 51 53 57 58 60 61 65 66 69 76
30      09/02/2020,03 07 10 22 23 27 31 32 34 35 36 44 50 52 56 61 65 68 69 70
```

```
09/01/2020,07 14 17 19 21 30 38 39 47 48 51 54 55 63 66 67 68 72 74 75
08/31/2020,05 06 11 19 20 22 23 31 32 36 37 40 41 54 60 63 75 76 77 78
08/30/2020,07 09 16 18 25 27 37 38 40 43 44 57 58 64 68 71 75 78 79 80
08/29/2020,02 03 09 11 14 17 22 23 24 27 34 35 36 38 43 45 49 58 62 64
08/28/2020,04 05 09 12 13 18 19 26 31 38 42 44 49 50 52 53 55 57 65 66
08/27/2020,07 11 12 13 15 19 28 31 35 36 44 45 54 56 65 66 71 72 74 80
08/26/2020,01 02 09 18 19 23 26 27 34 38 45 48 49 52 62 72 74 75 76 80
08/25/2020,09 10 11 16 17 18 29 33 43 44 47 49 54 56 60 62 65 71 72 73
08/24/2020,14 16 18 19 28 37 38 39 40 42 44 47 60 62 65 70 73 75 79 80
08/23/2020,02 05 08 12 15 16 17 18 19 20 22 27 37 54 56 61 66 67 72 77
08/22/2020,01 02 04 13 15 19 20 28 30 32 38 39 41 42 46 52 53 61 70 79
08/21/2020,03 06 07 09 17 25 26 31 33 41 42 44 45 48 58 60 66 67 72 73
08/20/2020,07 08 09 12 14 24 27 35 39 44 46 61 62 67 69 71 74 75 77 79
08/19/2020,11 13 15 16 18 20 23 26 30 39 48 51 57 58 62 67 69 71 72 73
08/18/2020,11 12 13 18 26 27 31 37 38 43 48 54 58 61 67 68 70 72 73 76
08/17/2020,04 07 09 12 13 16 20 27 28 29 43 47 50 51 53 57 59 61 65 69
08/16/2020,05 09 10 14 16 20 21 24 27 29 32 35 36 37 38 60 68 75 76 79
08/15/2020,02 07 15 16 20 21 30 31 32 33 38 43 47 51 61 66 67 68 75 77
08/14/2020,06 07 09 10 16 19 27 31 34 39 41 42 44 45 50 54 57 58 71 75
08/13/2020,01 03 12 23 25 29 31 36 42 44 49 53 56 58 59 63 68 69 70 71
08/12/2020,04 22 25 27 30 32 35 39 41 43 45 47 49 54 57 59 60 65 76 78
08/11/2020,04 07 08 10 17 22 27 37 39 41 45 46 61 66 68 70 72 74 75 76
08/10/2020,01 08 09 10 12 15 17 25 28 35 37 41 44 46 48 54 57 62 71 80
08/09/2020,18 20 21 22 31 33 36 38 41 48 50 53 58 67 72 73 74 75 76 77
08/08/2020,01 05 08 09 15 19 22 24 26 28 30 31 33 36 37 42 43 45 63 64
08/07/2020,13 16 18 26 28 34 36 40 41 42 44 45 46 48 49 55 58 71 73 75
08/06/2020,01 03 11 12 14 15 25 30 32 35 39 48 50 51 54 57 58 60 63 74
08/05/2020,06 11 16 20 21 23 26 29 31 32 33 46 51 54 58 66 69 74 75 78
08/04/2020,02 03 08 12 18 21 22 25 29 34 38 40 43 44 46 50 65 69 73 79
08/03/2020,03 09 14 18 22 23 30 31 37 44 46 47 50 53 54 63 67 68 72 74
08/02/2020,03 07 08 10 13 16 21 22 23 32 35 36 47 49 50 66 67 68 70 80
08/01/2020,03 04 08 11 15 26 28 29 30 40 41 43 44 46 54 56 57 63 66 72
07/31/2020,02 05 06 17 22 28 33 38 39 40 57 65 67 68 69 70 73 74 77 79
07/30/2020,02 05 06 08 09 15 16 21 29 31 33 34 37 42 49 53 58 63 69 77
07/29/2020,06 08 09 11 12 21 22 24 26 28 31 40 42 43 51 60 62 63 72 79
07/28/2020,03 05 06 17 23 26 27 29 31 34 35 36 45 48 53 59 61 66 67 75
07/27/2020,02 06 08 09 13 21 25 26 28 30 31 33 36 40 42 43 53 54 63 71
07/26/2020,04 06 10 11 12 15 18 26 27 28 32 33 34 36 50 53 56 75 78 79
07/25/2020,04 09 19 20 23 32 38 46 47 52 54 55 58 62 63 75 76 77 79 80
07/24/2020,12 14 17 19 20 25 33 34 43 48 52 56 61 62 63 65 69 75 76 77
07/23/2020,02 06 07 12 20 33 36 41 43 44 47 52 54 56 60 61 62 63 67 78
07/22/2020,01 10 11 26 29 32 37 38 40 53 54 57 59 62 72 73 74 75 76 80
07/21/2020,01 03 06 08 09 11 18 20 38 40 43 44 46 48 56 57 58 59 60 66
07/20/2020,06 13 16 18 19 20 24 28 29 33 34 44 50 51 59 63 69 71 77 78
07/19/2020,05 11 12 25 29 33 34 39 41 52 56 57 58 60 62 69 70 72 74 75
07/18/2020,01 03 05 06 15 22 23 34 45 52 59 63 65 66 69 72 75 77 79 80
07/17/2020,06 11 14 23 24 33 38 41 44 46 48 51 57 59 64 65 69 75 78 79
07/16/2020,08 12 15 20 23 28 31 33 37 48 51 54 55 56 57 59 71 73 75 77
07/15/2020,02 07 13 16 20 22 23 24 26 27 42 47 63 65 66 67 68 71 72 74
07/14/2020,01 10 12 13 15 16 19 25 26 34 41 43 48 50 54 57 65 71 76 77
07/13/2020,01 02 11 12 17 20 33 35 36 39 43 49 52 59 69 70 75 77 78 79
07/12/2020,03 04 09 10 12 18 20 48 49 50 51 60 61 62 65 66 67 69 74 75
07/11/2020,03 06 13 17 23 24 31 41 47 55 60 63 66 67 68 71 72 74 77 79
07/10/2020,02 18 23 29 31 44 48 51 52 56 58 59 60 61 70 71 73 75 76 79
```

```
07/09/2020,03 09 11 13 17 20 31 38 39 40 44 46 49 58 62 64 66 70 71 72
07/08/2020,02 04 05 07 08 11 13 15 17 18 32 33 41 53 55 61 62 66 74 78
07/07/2020,07 10 11 13 17 18 20 28 29 43 58 60 62 64 67 70 75 76 77 80
07/06/2020,01 02 03 05 08 09 18 19 33 40 46 47 48 51 56 59 62 63 67 72
07/05/2020,07 19 21 22 23 24 27 28 31 32 33 35 37 43 44 45 46 55 69 79
07/04/2020,01 02 07 14 15 16 24 37 38 40 42 43 44 47 48 61 63 64 71 78
07/03/2020,02 03 06 07 08 12 20 24 34 42 43 49 56 58 71 72 76 78 79 80
07/02/2020,01 04 08 09 13 15 18 19 30 32 42 46 56 59 61 62 65 70 72 80
07/01/2020,04 07 12 13 23 24 25 27 28 36 37 38 51 52 53 58 61 64 73 79
06/30/2020,01 05 09 10 11 13 15 26 27 30 35 42 47 49 52 55 57 74 75 77
06/29/2020,03 05 06 12 18 19 21 30 31 32 33 34 40 45 46 52 60 71 77 79
06/28/2020,03 08 13 19 20 22 24 28 29 32 33 34 37 39 41 43 47 50 73 75
06/27/2020,02 03 11 12 15 18 21 33 34 37 42 43 45 46 51 55 58 67 69 72
06/26/2020,01 04 07 09 23 24 28 33 34 41 51 59 60 61 64 66 67 76 77 79
06/25/2020,03 04 06 07 09 10 11 17 20 21 27 31 32 33 41 48 61 66 67 71
06/24/2020,01 12 20 21 27 47 48 51 54 58 60 61 62 69 70 73 74 77 78 79
06/23/2020,04 09 13 14 16 27 29 30 34 44 45 48 52 53 60 63 65 69 71 79
06/22/2020,06 10 14 20 21 24 29 31 35 38 40 42 53 55 56 57 68 72 74 75
06/21/2020,03 04 14 20 23 24 34 37 40 41 45 46 51 52 58 60 64 73 75 77
06/20/2020,03 04 05 09 18 23 24 27 33 39 47 48 54 58 61 62 69 72 73 77
06/19/2020,01 03 05 06 13 18 26 27 28 30 33 36 37 42 48 50 59 64 70 80
06/18/2020,01 02 05 10 15 16 18 26 28 30 35 36 43 46 49 61 67 68 75 79
06/17/2020,04 08 10 11 13 25 28 29 32 39 40 45 49 54 56 57 66 73 79 80
06/16/2020,05 07 20 22 25 30 32 33 34 38 41 42 43 46 49 52 55 57 64 71
06/15/2020,02 06 10 13 23 30 32 35 39 48 49 51 52 56 61 68 69 70 71 76
06/14/2020,02 03 05 12 23 24 28 31 33 35 37 43 46 55 56 60 69 74 77 78
06/13/2020,02 06 09 11 16 20 21 26 33 35 41 47 52 55 56 58 62 64 68 76
06/12/2020,03 07 09 15 18 29 31 33 38 46 54 57 61 66 67 69 74 75 76 80
06/11/2020,03 06 08 10 16 18 20 21 25 30 33 34 35 41 46 56 61 63 77 80
06/10/2020,01 04 16 28 29 34 37 38 39 42 44 47 49 50 54 56 62 63 68 71
06/09/2020,10 11 13 16 17 18 22 23 27 51 52 53 58 62 64 65 66 67 73 74
06/08/2020,03 04 08 09 17 20 31 32 35 44 52 55 57 59 60 61 64 66 77 80
06/07/2020,02 05 07 15 17 21 24 27 28 36 40 44 47 50 58 59 65 70 74 75
06/06/2020,01 02 04 06 07 08 14 15 19 20 22 23 25 27 31 42 51 66 71 75
06/05/2020,06 14 18 21 22 25 26 27 32 45 47 49 50 51 59 61 65 72 74 78
06/04/2020,02 16 24 26 28 29 31 37 43 46 50 52 53 54 56 61 68 70 73 74
06/03/2020,03 05 13 17 26 33 34 40 41 48 56 57 60 61 64 66 69 75 76 79
06/02/2020,06 09 10 17 18 20 24 32 39 41 44 47 50 51 53 62 65 68 71 77
06/01/2020,06 09 12 15 16 21 31 35 39 41 45 50 51 53 56 58 61 65 70 77
05/31/2020,08 09 10 11 18 22 27 29 31 34 35 38 39 42 46 55 62 63 67 78
05/30/2020,03 06 12 13 17 18 21 23 30 40 43 53 58 61 69 73 74 76 77 80
05/29/2020,05 08 10 17 19 21 25 28 39 43 54 56 59 61 66 67 75 76 77 80
05/28/2020,02 03 09 15 17 25 31 32 36 38 39 42 46 47 49 56 58 65 71 75
05/27/2020,01 12 21 23 29 32 37 42 45 46 52 53 55 56 61 63 68 70 77 78
05/26/2020,04 06 07 11 15 16 17 20 21 23 33 34 35 42 43 49 67 70 76 78
05/25/2020,08 12 14 16 20 24 27 28 29 30 32 34 37 40 44 45 63 70 73 80
05/24/2020,06 09 10 11 13 15 24 28 32 38 41 45 46 58 61 64 70 72 77 79
05/23/2020,02 09 12 13 22 25 26 27 28 39 40 42 43 44 50 51 57 67 75 76
05/22/2020,02 04 11 13 15 17 30 32 36 38 39 41 53 55 65 67 68 69 72 75
05/21/2020,06 10 12 13 15 18 22 25 29 32 36 38 41 45 60 62 66 67 68 71
05/20/2020,07 08 11 12 14 20 21 24 27 29 30 35 39 44 45 46 58 60 69 70
05/19/2020,02 05 08 13 21 24 29 30 31 35 38 40 45 48 58 64 67 69 79 80
05/18/2020,01 04 06 07 10 11 19 20 22 23 24 31 45 46 49 58 65 69 76 79
05/17/2020,03 04 07 10 16 23 24 29 30 39 40 41 46 48 49 55 61 63 71 72
```

```
05/16/2020,07 16 17 18 19 24 32 39 41 43 49 50 52 54 60 63 72 76 77 80
05/15/2020,01 05 06 13 18 29 31 42 44 45 47 48 58 59 60 66 69 74 77 79
05/14/2020,07 12 13 22 24 25 27 28 35 45 52 54 57 58 60 62 64 68 69 79
05/13/2020,01 03 09 14 19 31 33 35 43 49 52 53 54 55 57 62 64 66 71 74
05/12/2020,03 04 05 08 15 16 20 21 22 25 30 39 50 56 59 60 62 70 71 72
05/11/2020,03 05 06 17 21 29 33 34 37 43 53 54 58 59 61 65 71 72 75 77
05/10/2020,12 14 15 18 26 31 33 38 39 48 49 53 55 57 62 65 68 75 77 78
05/09/2020,01 03 06 14 15 21 23 29 31 41 45 49 55 60 62 65 72 74 75 76
05/08/2020,12 13 20 22 23 31 34 35 39 43 48 50 53 57 58 62 64 69 70 74
05/07/2020,02 05 07 12 14 19 23 28 34 38 39 42 47 50 55 56 68 74 76 79
05/06/2020,04 13 15 20 24 27 30 32 33 35 39 41 43 59 60 67 69 73 75 78
05/05/2020,07 12 13 18 22 26 32 35 42 44 46 48 50 53 55 60 64 71 73 78
05/04/2020,01 06 12 13 21 27 33 39 41 48 49 58 59 60 63 65 68 72 74 78
05/03/2020,01 09 12 13 16 27 30 34 39 42 43 44 45 52 56 57 58 67 72 74
05/02/2020,09 11 14 16 24 26 30 31 35 40 50 56 57 61 62 63 65 71 73 74
05/01/2020,08 09 10 16 17 21 29 30 32 33 41 42 46 52 53 63 64 68 74 77
04/30/2020,02 06 08 12 13 15 29 30 31 35 36 47 49 52 54 55 59 70 72 78
04/29/2020,16 24 25 28 32 35 37 41 48 50 57 60 65 66 67 68 69 75 77 80
04/28/2020,08 10 11 16 17 27 33 34 36 39 45 46 47 49 50 55 65 70 72 75
04/27/2020,05 12 13 15 16 19 25 28 30 32 36 41 47 49 54 56 58 62 64 65
04/26/2020,02 04 07 11 15 19 21 22 28 29 33 39 48 49 52 67 69 70 71 79
04/25/2020,06 12 14 34 36 43 44 45 58 59 60 61 64 65 71 72 73 74 75 77
04/24/2020,02 03 04 12 14 19 24 29 38 39 48 53 54 57 66 69 70 72 75 76
04/23/2020,03 04 15 23 25 28 32 36 38 45 46 50 54 60 61 64 67 69 76 80
04/22/2020,01 05 19 20 21 23 29 31 32 33 38 45 50 51 54 55 67 70 75 79
04/21/2020,01 08 11 14 16 17 22 26 29 43 48 50 52 55 61 62 65 67 70 75
04/20/2020,03 08 15 18 20 28 30 31 37 43 50 51 52 57 63 65 66 68 71 74
04/19/2020,01 08 13 15 16 21 25 31 34 37 42 46 50 53 55 57 70 74 77 79
04/18/2020,02 03 05 07 10 17 18 22 27 30 40 41 43 45 55 65 66 69 72 73
04/17/2020,01 02 04 05 06 07 10 11 20 22 35 39 40 45 59 62 63 71 74 78
04/16/2020,02 07 11 13 14 15 17 20 21 30 38 42 50 56 58 61 68 73 75 80
04/15/2020,01 06 09 12 13 16 18 22 25 31 34 43 44 46 50 58 64 74 76 80
04/14/2020,11 18 22 30 32 37 39 42 47 48 54 55 56 59 61 63 66 67 68 69
04/13/2020,02 11 21 24 33 34 36 38 39 43 46 51 52 53 59 61 70 72 74 79
04/12/2020,02 03 06 08 10 18 22 25 26 27 35 39 47 50 52 55 57 72 73 75
04/11/2020,05 06 12 16 18 29 33 39 40 43 46 47 48 50 51 52 63 66 76 77
04/10/2020,01 04 06 07 08 10 12 20 21 26 31 45 47 58 61 66 68 73 77 78
04/09/2020,06 17 19 21 27 29 30 48 50 51 52 53 56 57 65 71 72 73 76 77
04/08/2020,05 17 22 24 25 27 31 34 35 36 41 43 47 56 58 59 64 68 73 75
04/07/2020,02 06 09 12 13 15 17 25 29 36 44 47 52 56 62 63 64 69 70 74
04/06/2020,05 07 10 11 25 35 40 43 45 46 47 49 55 57 66 68 71 75 77 78
04/05/2020,01 06 07 08 10 20 21 26 27 28 40 43 45 46 47 60 62 65 68 72
04/04/2020,03 06 07 10 14 20 26 33 38 39 40 43 44 46 56 68 71 73 76 78
04/03/2020,03 06 07 15 17 20 22 25 32 33 36 38 44 45 46 53 57 65 76 78
04/02/2020,05 06 18 21 32 35 36 38 47 48 54 56 58 60 64 67 68 73 74 78
04/01/2020,02 04 06 08 09 11 19 24 28 32 33 40 53 54 57 60 61 64 70 74
03/31/2020,02 04 07 10 14 16 19 24 25 28 33 35 37 41 57 66 68 74 77 79
03/30/2020,01 07 09 10 11 12 18 21 23 25 26 32 41 47 48 54 59 64 66 67
03/29/2020,07 10 13 26 32 36 37 42 46 50 54 55 56 61 63 67 68 70 73 80
03/28/2020,02 04 08 10 12 17 18 20 31 32 33 34 38 47 49 50 51 56 59 78
03/27/2020,06 08 10 11 12 16 29 32 43 52 53 61 63 65 69 70 73 74 77 80
03/26/2020,08 10 16 20 24 27 30 31 33 37 40 44 45 47 49 56 58 64 74 75
03/25/2020,05 06 07 12 20 22 23 28 29 36 37 40 50 56 63 66 67 73 74 75
03/24/2020,02 03 04 23 27 28 31 45 47 48 49 51 59 62 63 67 69 70 75 76
```

```
93      03/23/2020,07 14 22 25 26 34 35 44 46 49 50 55 59 60 63 69 75 76 77 80
94      03/22/2020,05 14 16 20 26 28 29 38 39 41 42 44 48 64 72 74 75 76 77 79
95      03/21/2020,03 10 15 17 18 19 20 21 23 28 45 49 53 58 59 61 64 74 76 79
96      03/20/2020,01 02 05 07 08 10 14 23 24 27 29 41 42 46 52 58 61 70 72 74
97      03/19/2020,06 09 18 25 28 30 35 39 40 43 47 51 57 58 61 64 68 74 75 80
98      03/18/2020,04 06 11 14 16 18 24 28 30 31 35 41 47 54 57 59 69 71 78 79
99      03/17/2020,03 09 12 18 32 34 37 38 44 46 47 53 55 57 60 64 66 73 75 79
00      03/16/2020,13 17 21 37 38 39 40 44 47 48 49 53 56 61 66 70 71 72 77 78
01      03/15/2020,03 04 07 12 14 16 23 24 29 30 32 40 45 46 50 54 56 58 59 73
02      03/14/2020,01 02 03 04 06 09 11 13 14 16 28 36 42 43 46 53 62 64 74 79
03      03/13/2020,06 08 12 21 30 31 33 34 35 36 38 47 48 50 54 56 59 64 71 76
04      03/12/2020,12 18 21 23 27 29 30 36 39 42 43 47 49 53 58 59 62 63 68 79
05      03/11/2020,03 07 08 16 21 24 25 31 35 39 45 50 53 66 70 71 72 74 76 77
06      03/10/2020,03 05 06 08 18 19 20 23 24 28 30 31 33 35 39 45 57 72 73 77
07      03/09/2020,04 16 19 20 23 33 34 37 38 40 45 50 52 53 55 56 61 63 68 80
08      03/08/2020,02 12 15 18 20 28 33 41 44 47 50 52 56 59 60 63 66 70 72 77
09      03/07/2020,10 15 17 18 19 20 26 28 34 35 38 39 41 42 58 63 65 66 76 78
10      03/06/2020,01 11 14 17 18 25 30 33 41 44 50 52 59 64 67 70 71 76 78 79
11      03/05/2020,02 04 09 16 17 18 26 28 32 40 42 49 51 54 55 60 61 63 69 72
12      03/04/2020,01 04 07 08 09 15 16 18 23 29 32 42 46 51 52 63 64 65 76 79
13      03/03/2020,04 05 13 14 16 21 24 27 36 37 39 46 53 54 57 59 70 73 75 77
14      03/02/2020,02 13 20 21 26 29 39 41 49 52 55 57 60 61 63 65 68 71 72 73
15      03/01/2020,07 08 10 14 16 17 22 25 37 39 42 43 45 46 63 65 67 74 75 78
16      02/29/2020,02 05 11 14 16 21 22 25 28 35 41 44 51 52 56 58 63 66 70 79
17      02/28/2020,02 05 09 10 12 16 18 21 23 26 31 33 48 51 53 55 58 59 63 80
18      02/27/2020,01 02 08 16 21 22 25 35 36 38 43 51 52 54 56 58 63 64 68 80
19      02/26/2020,01 05 06 07 16 27 35 47 48 49 52 55 56 58 61 65 71 72 75 78
20      02/25/2020,04 05 10 15 19 42 50 51 52 54 55 56 57 61 62 65 68 70 72 73
21      02/24/2020,02 03 09 10 11 21 27 28 31 33 47 48 57 64 65 66 67 72 74 80
22      02/23/2020,06 07 20 25 26 27 30 40 44 52 54 55 59 61 62 64 65 68 72 76
23      02/22/2020,01 13 20 21 29 32 37 39 45 51 58 59 61 63 67 68 70 71 78 79
24      02/21/2020,03 06 09 14 15 26 27 28 33 35 37 40 46 48 57 66 68 70 71 76
25      02/20/2020,11 21 22 23 28 37 40 43 48 51 54 56 57 61 66 69 71 72 77 79
26      02/19/2020,15 17 21 23 25 27 28 31 34 35 43 45 46 50 51 62 65 71 72 76
27      02/18/2020,01 12 13 15 17 18 20 21 30 39 44 45 47 52 54 56 58 65 66 79
28      02/17/2020,03 10 16 23 26 27 29 37 40 41 45 50 51 52 55 59 64 71 74 80
29      02/16/2020,04 05 10 13 19 23 31 35 41 46 48 50 56 57 60 65 67 73 78 79
30      02/15/2020,03 05 15 16 17 20 23 30 31 34 35 42 46 55 60 62 64 67 68 71
31      02/14/2020,02 04 06 07 11 17 19 23 26 29 42 44 54 65 66 69 70 71 76 77
32      02/13/2020,11 15 22 23 24 26 28 29 42 44 48 51 60 63 67 70 71 72 77 80
33      02/12/2020,06 18 19 25 28 34 35 36 38 40 42 45 50 51 53 58 61 66 67 71
34      02/11/2020,02 05 14 15 24 25 29 30 36 40 42 44 46 51 53 57 64 73 74 80
35      02/10/2020,02 03 10 12 14 16 21 22 28 30 34 35 47 51 52 64 65 70 71 74
36      02/09/2020,05 09 12 16 18 22 27 34 39 40 41 52 53 54 62 63 64 66 68 78
37      02/08/2020,05 14 15 20 29 35 38 42 43 54 55 56 58 62 63 65 69 75 77 78
38      02/07/2020,04 09 13 14 20 21 24 34 42 48 49 56 57 60 61 62 66 67 71 79
39      02/06/2020,01 05 08 10 14 20 24 32 46 53 61 63 65 70 75 76 77 78 79 80
40      02/05/2020,02 06 08 11 16 17 26 27 40 41 51 53 63 65 66 67 72 76 77 80
41      02/04/2020,01 02 06 08 10 14 19 23 25 28 33 39 44 47 50 54 68 70 74 77
42      02/03/2020,01 03 04 10 15 17 25 32 36 42 47 50 51 52 59 64 68 73 74 77
43      02/02/2020,02 08 22 25 26 28 31 36 40 42 44 48 51 55 57 60 71 75 76 77
44      02/01/2020,02 15 16 23 25 29 32 35 39 45 46 50 55 56 60 68 70 74 75 79
45      01/31/2020,12 20 21 22 23 28 31 39 40 43 47 51 53 54 56 58 63 65 77 79
46      01/30/2020,02 04 05 08 10 11 16 23 24 30 38 40 47 48 52 59 64 65 75 76
```

```
47      01/29/2020,01 05 06 07 09 10 18 23 29 30 33 36 38 43 56 64 65 71 72 79
48      01/28/2020,01 02 03 06 07 12 14 16 23 24 27 29 35 41 48 53 58 67 72 76
49      01/27/2020,03 09 17 18 20 21 22 33 36 43 47 50 54 58 61 62 63 65 71 76
50      01/26/2020,03 09 10 12 14 27 29 34 38 44 46 47 49 58 59 65 67 69 75 76
51      01/25/2020,04 16 21 26 30 34 36 37 41 45 46 47 49 58 60 63 66 67 69 80
52      01/24/2020,03 10 11 13 14 20 26 32 38 39 41 53 58 59 61 63 66 69 75 77
53      01/23/2020,08 09 12 16 20 25 27 33 38 41 47 49 51 54 67 71 74 76 77 80
54      01/22/2020,03 08 17 22 24 28 31 34 37 42 44 46 47 57 60 61 63 64 65 71
55      01/21/2020,02 03 15 19 36 37 39 43 44 47 48 50 52 60 61 63 64 71 76 79
56      01/20/2020,02 03 05 09 10 11 15 25 27 30 31 32 40 41 48 55 59 70 73 76
57      01/19/2020,03 07 08 09 11 12 14 15 16 21 30 33 39 40 44 55 59 67 74 75
58      01/18/2020,01 04 13 14 15 16 19 32 35 38 42 48 49 54 69 72 74 75 78 80
59      01/17/2020,03 12 14 19 20 21 26 34 39 44 48 53 58 60 61 62 63 66 69 72
60      01/16/2020,03 05 07 08 13 15 17 19 20 25 28 31 46 49 50 55 59 60 64 74
61      01/15/2020,01 03 10 19 20 21 26 27 29 38 40 42 43 48 50 69 70 71 76 79
62      01/14/2020,01 05 07 13 24 28 30 32 45 48 51 59 62 63 64 65 71 76 77 78
63      01/13/2020,01 02 15 17 20 21 23 24 25 31 37 42 45 51 57 59 67 69 72 79
64      01/12/2020,02 05 07 08 13 19 23 29 31 34 36 39 47 49 50 56 65 71 76 78
65      01/11/2020,01 08 09 11 22 25 26 28 30 31 36 37 39 49 50 55 72 75 77 80
66      01/10/2020,08 13 14 18 25 31 32 38 41 44 46 51 52 54 61 65 66 74 77 79
67      01/09/2020,02 04 06 17 21 23 24 33 40 42 47 50 53 59 64 66 67 72 78 80
68      01/08/2020,05 09 11 20 23 24 27 28 30 36 43 47 49 56 59 61 64 72 75 76
69      01/07/2020,03 04 06 10 15 23 25 29 32 33 34 48 57 61 62 63 74 76 77 80
70      01/06/2020,01 05 06 12 13 18 20 32 40 46 47 49 53 54 56 58 60 61 66 71
71      01/05/2020,06 07 16 19 20 24 25 28 33 36 45 48 51 53 57 61 66 68 69 72
72      01/04/2020,01 07 08 09 12 15 22 23 26 32 38 47 52 55 63 67 71 74 77 78
73      01/03/2020,03 04 08 17 19 22 23 30 33 38 41 44 51 55 63 64 66 68 73 77
74      01/02/2020,04 08 14 20 21 23 27 29 33 35 36 37 42 50 52 55 57 63 69 73
75      01/01/2020,06 09 10 13 15 22 37 42 43 48 55 56 57 58 62 66 72 75 76 78
76      12/31/2019,01 06 07 11 12 13 24 27 29 30 31 44 48 49 50 56 57 62 72 78
77      12/30/2019,05 06 07 15 16 23 24 27 30 32 34 36 37 56 58 60 61 62 71 73
78      12/29/2019,02 10 11 14 20 22 30 32 34 37 42 48 49 54 55 59 64 72 79 80
79      12/28/2019,01 02 03 07 19 28 31 32 36 40 45 46 49 54 58 63 64 65 74 79
80      12/27/2019,13 15 17 18 34 38 39 42 44 45 46 51 55 58 61 64 69 70 78 80
81      12/26/2019,04 08 12 18 21 26 34 40 41 44 45 46 60 61 62 73 75 78 79 80
82      12/25/2019,01 11 12 15 21 22 27 33 34 35 36 37 46 53 63 65 68 71 73 79
83      12/24/2019,08 09 11 12 19 20 27 28 36 44 45 46 50 54 55 62 67 69 71 76
84      12/23/2019,08 09 11 16 22 23 24 40 42 46 50 54 60 64 65 70 74 75 76 77
85      12/22/2019,04 08 16 17 20 29 34 39 40 42 46 47 48 49 51 54 56 59 70 78
86      12/21/2019,12 15 18 22 23 27 30 34 44 45 49 56 57 58 60 64 72 76 78 80
87      12/20/2019,03 04 05 08 15 17 26 27 29 30 33 41 53 59 60 61 63 65 67 74
88      12/19/2019,15 16 17 23 25 29 31 38 40 41 48 51 52 53 57 58 70 75 79 80
89      12/18/2019,05 08 15 16 17 21 24 25 26 27 29 30 38 46 55 60 63 71 77 79
90      12/17/2019,02 08 09 13 22 23 24 25 29 30 32 45 46 49 50 53 57 59 68 74
91      12/16/2019,01 05 12 14 19 23 24 34 35 36 40 41 43 49 53 64 68 69 72 78
92      12/15/2019,02 12 16 18 20 23 24 27 29 34 38 40 41 43 48 64 66 69 73 79
93      12/14/2019,02 08 09 17 20 24 38 43 48 49 51 53 58 61 65 71 75 77 78 80
94      12/13/2019,11 12 17 19 20 25 32 38 41 45 48 49 50 53 62 66 68 76 79 80
95      12/12/2019,03 05 06 07 14 35 43 53 54 56 59 60 61 62 66 67 72 74 75 76
96      12/11/2019,02 04 15 19 20 25 26 27 30 31 34 45 51 52 54 56 57 63 75 77
97      12/10/2019,01 02 03 05 09 11 13 19 23 25 34 42 44 52 56 59 66 68 72 75
98      12/09/2019,04 07 09 13 16 18 20 27 28 34 36 39 40 46 52 58 64 65 73 76
99      12/08/2019,03 04 15 24 25 26 28 29 36 41 42 53 57 58 65 67 68 69 74 75
00      12/07/2019,02 03 04 09 11 16 19 26 29 32 40 49 51 54 58 61 66 70 72 80
```

01 12/06/2019,01 06 09 10 13 16 24 27 39 40 46 50 51 61 62 65 70 76 78 80
02 12/05/2019,06 10 11 20 29 31 45 46 52 57 58 61 62 63 68 74 76 77 78 80
03 12/04/2019,01 02 07 08 15 16 36 38 48 49 53 57 58 59 63 64 66 67 73 80
04 12/03/2019,02 10 11 12 15 17 19 22 27 32 33 41 45 57 58 59 60 64 70 80
05 12/02/2019,03 04 06 15 17 18 21 23 30 36 41 42 49 51 53 60 61 70 74 75
06 12/01/2019,07 11 13 14 15 18 22 29 30 35 36 42 44 51 53 68 75 76 78 79
07 11/30/2019,04 12 18 21 23 25 33 38 40 43 45 50 51 53 60 63 72 73 75 76
08 11/29/2019,03 06 08 11 19 23 29 30 31 34 36 46 49 50 51 53 55 60 64 80
09 11/28/2019,02 03 13 14 16 17 19 31 40 44 49 51 52 53 60 63 64 69 74 79
10 11/27/2019,09 12 14 15 22 24 29 31 39 42 52 53 62 65 68 71 73 74 79 80
11 11/26/2019,02 08 15 18 20 22 29 33 35 38 40 44 50 54 56 57 63 68 77 78
12 11/25/2019,05 06 16 18 19 31 35 36 37 42 47 49 62 63 64 66 67 69 70 77
13 11/24/2019,01 03 07 13 20 21 26 30 31 33 36 53 54 59 61 64 68 69 70 80
14 11/23/2019,03 04 05 17 21 27 30 32 42 44 50 51 56 60 64 66 68 69 78 79
15 11/22/2019,05 06 20 21 22 26 31 35 36 41 51 55 60 63 71 72 73 77 78 79
16 11/21/2019,11 12 16 18 19 20 23 24 32 33 34 35 42 45 58 63 66 69 70 78
17 11/20/2019,05 14 15 19 21 26 28 29 32 34 38 44 51 56 60 65 70 72 77 79
18 11/19/2019,01 04 05 11 13 14 18 23 25 32 33 34 39 41 48 57 64 65 75 78
19 11/18/2019,01 06 19 23 24 30 31 34 42 45 46 51 53 54 55 58 67 68 77 78
20 11/17/2019,03 08 11 13 21 26 27 32 37 45 47 48 54 56 58 60 62 64 66 70
21 11/16/2019,13 15 16 17 22 24 25 28 31 32 33 38 41 42 43 48 53 55 77 79
22 11/15/2019,06 10 18 19 21 23 24 35 38 39 40 47 48 49 57 60 63 71 77 80
23 11/14/2019,05 06 08 11 12 15 17 21 22 28 29 32 34 35 38 40 45 46 72 79
24 11/13/2019,01 03 04 07 18 25 29 30 31 32 40 41 45 58 62 64 72 73 79 80
25 11/12/2019,12 15 16 18 22 25 30 34 36 39 58 60 63 64 65 67 68 69 72 80
26 11/11/2019,02 05 11 16 19 20 21 33 36 37 39 44 49 55 57 59 66 68 75 79
27 11/10/2019,02 08 09 21 22 28 32 33 39 40 45 46 48 51 54 55 57 59 75 78
28 11/09/2019,01 03 04 08 12 13 22 26 29 30 33 39 50 51 53 61 67 72 74 80
29 11/08/2019,04 05 07 13 17 21 22 24 31 33 37 40 44 48 52 58 64 67 70 73
30 11/07/2019,03 09 12 13 16 30 32 37 38 44 53 54 59 60 65 67 69 74 76 78
31 11/06/2019,04 09 10 12 13 17 35 36 37 38 42 43 48 49 55 57 60 68 71 75
32 11/05/2019,01 04 05 09 11 16 20 25 32 36 37 38 41 43 59 61 62 68 70 71
33 11/04/2019,06 08 09 17 24 25 27 33 40 41 48 51 53 59 61 62 69 74 77 78
34 11/03/2019,04 06 08 09 12 13 24 26 27 28 31 39 46 47 49 57 61 66 67 73
35 11/02/2019,01 04 05 06 12 13 14 23 29 30 44 45 52 61 64 65 68 71 76 79
36 11/01/2019,02 04 16 18 21 23 25 28 29 31 37 39 45 47 51 59 63 67 68 70
37 10/31/2019,04 05 08 09 16 18 20 26 38 43 51 56 59 60 62 63 68 73 79 80
38 10/30/2019,03 05 06 08 19 24 27 30 32 33 45 46 47 49 51 57 64 66 76 80
39 10/29/2019,02 03 08 09 13 14 21 24 25 26 29 30 48 49 53 54 64 65 74 76
40 10/28/2019,03 05 06 07 08 15 18 19 22 28 31 37 41 49 53 55 57 61 62 63
41 10/27/2019,03 05 07 15 16 18 19 21 22 23 25 34 38 40 46 57 63 68 73 78
42 10/26/2019,02 07 12 15 19 20 37 42 43 51 52 58 61 64 65 66 71 74 75 78
43 10/25/2019,05 10 11 12 15 25 29 31 32 35 36 39 48 56 57 61 63 68 73 80
44 10/24/2019,01 02 03 14 17 20 21 31 35 39 42 56 62 63 69 70 72 75 78 79
45 10/23/2019,03 07 10 14 15 18 24 25 27 28 31 34 38 47 56 57 68 72 74 80
46 10/22/2019,11 14 24 26 27 30 36 40 44 45 46 54 57 58 61 67 72 76 77 79
47 10/21/2019,02 06 07 16 24 34 36 38 39 40 41 43 45 51 57 58 61 66 67 79
48 10/20/2019,01 10 15 18 19 20 23 24 25 26 28 30 32 33 53 56 60 64 70 72
49 10/19/2019,01 03 05 06 07 35 36 38 42 51 53 59 62 63 69 71 75 76 77 78
50 10/18/2019,02 10 16 17 19 21 31 33 37 39 42 44 46 47 52 54 55 67 71 77
51 10/17/2019,03 05 08 09 13 20 30 31 41 44 47 48 51 58 59 62 65 68 72 77
52 10/16/2019,01 03 08 16 25 26 33 34 37 43 50 52 58 61 63 64 67 69 71 78
53 10/15/2019,08 10 11 22 23 28 29 31 36 37 50 51 53 55 56 61 67 72 75 80
54 10/14/2019,02 08 19 26 28 30 33 47 48 54 56 58 61 62 63 68 69 70 72 78

```
10/13/2019,03 07 10 12 15 17 21 26 28 37 43 50 57 58 60 68 70 71 74 75
10/12/2019,02 04 05 09 12 13 23 29 33 34 37 52 56 60 63 70 71 76 79 80
10/11/2019,01 02 04 08 15 16 18 27 33 35 38 44 46 50 55 57 65 70 73 76
10/10/2019,10 12 20 21 23 24 31 32 33 34 37 38 48 49 52 53 58 65 71 79
10/09/2019,05 06 09 11 24 27 29 30 35 39 44 47 51 52 59 64 69 71 73 76
10/08/2019,09 10 14 15 22 23 24 27 33 36 37 52 57 59 64 71 72 74 75 77
10/07/2019,06 08 09 13 14 20 23 24 25 27 31 35 37 38 45 46 56 58 61 67
10/06/2019,09 10 14 28 31 33 35 43 46 51 60 61 62 63 68 70 76 78 79 80
10/05/2019,07 17 23 28 29 30 34 37 45 46 49 50 56 63 66 68 71 75 79 80
10/04/2019,12 20 23 27 36 38 41 44 46 48 49 53 55 56 58 60 67 71 75 77
10/03/2019,06 08 09 14 15 17 24 25 32 34 38 43 45 46 48 50 58 59 64 77
10/02/2019,05 07 09 13 14 16 21 26 30 33 37 52 57 58 61 64 66 68 72 74
10/01/2019,11 17 18 25 26 28 30 31 33 41 42 48 49 53 54 55 57 77 79 80
09/30/2019,02 03 09 25 27 34 37 39 40 45 48 57 59 61 62 63 65 67 73 80
09/29/2019,02 05 06 07 11 20 22 25 26 28 32 41 45 46 53 62 68 72 74 76
09/28/2019,01 09 11 22 23 24 25 26 31 32 34 36 40 46 47 48 62 65 68 72
09/27/2019,02 05 07 08 10 11 16 19 24 33 35 37 41 48 49 55 63 69 75 78
09/26/2019,06 07 11 18 22 28 30 34 36 38 45 46 47 51 58 64 65 76 77 78
09/25/2019,01 08 13 16 19 25 26 28 37 39 41 51 54 55 59 66 67 68 74 75
09/24/2019,06 09 10 14 18 20 24 33 44 48 56 61 62 64 67 70 71 75 76 79
09/23/2019,04 05 08 12 19 25 26 28 33 42 48 49 52 56 57 66 68 73 74 75
09/22/2019,09 12 19 24 26 32 33 40 41 42 50 52 55 58 63 66 68 69 72 76
09/21/2019,01 02 05 07 08 10 12 17 18 21 23 24 27 30 36 55 56 60 61 63
09/20/2019,04 09 14 15 19 20 25 26 32 41 42 49 53 58 61 63 67 69 73 78
09/19/2019,02 03 06 13 15 17 23 25 28 29 31 36 37 47 49 50 53 56 68 80
09/18/2019,02 05 16 18 19 20 32 38 40 43 44 47 48 54 64 65 67 68 75 78
09/17/2019,02 05 06 19 22 27 33 35 38 40 41 51 53 56 63 64 69 70 72 77
09/16/2019,05 11 15 16 28 30 45 49 50 52 53 57 59 65 67 70 72 74 77 78
09/15/2019,09 10 12 13 20 21 22 24 27 43 47 51 52 59 60 61 62 65 69 80
09/14/2019,02 07 09 10 11 13 17 29 33 35 37 38 42 47 50 54 59 69 72 76
09/13/2019,02 04 11 15 21 25 30 40 44 45 53 60 63 66 68 69 70 71 72 74
09/12/2019,01 10 12 15 26 28 38 47 51 53 61 63 65 69 70 71 72 75 78 80
09/11/2019,01 05 07 13 15 19 28 30 33 34 36 43 53 54 58 66 68 71 76 78
09/10/2019,14 18 21 29 30 36 39 41 45 49 51 57 58 61 68 71 74 75 76 80
09/09/2019,08 09 11 26 33 37 40 41 46 50 52 54 55 57 64 65 71 72 74 75
09/08/2019,04 10 15 19 24 26 27 28 32 33 37 51 53 54 60 62 66 67 70 75
09/07/2019,04 05 10 12 17 19 27 28 30 32 41 45 51 54 57 63 69 75 77 79
09/06/2019,03 04 06 09 14 17 20 27 32 34 35 36 44 52 54 56 66 69 78 80
09/05/2019,02 05 07 10 14 15 16 17 18 21 25 30 32 37 41 47 53 55 58 68
09/04/2019,08 11 18 20 22 26 28 32 34 37 42 43 47 55 56 62 63 67 70 73
09/03/2019,10 11 14 17 24 30 31 35 37 38 45 52 55 58 59 65 67 68 70 71
09/02/2019,02 11 18 22 37 39 42 49 57 59 61 62 65 67 71 73 76 77 78 80
09/01/2019,05 06 08 09 11 13 14 15 21 25 41 47 52 53 55 72 73 74 76 77
08/31/2019,05 06 08 12 15 20 21 22 24 26 29 30 35 37 47 50 57 65 75 76
08/30/2019,06 10 11 16 18 26 27 28 41 45 48 51 53 56 58 62 63 67 71 79
08/29/2019,02 03 04 05 06 08 09 11 12 20 34 36 37 38 47 53 61 64 67 80
08/28/2019,01 07 09 10 18 19 22 23 25 34 39 41 47 52 55 56 66 71 77 79
08/27/2019,05 11 12 17 18 39 41 47 51 53 55 56 58 62 71 74 75 76 79 80
08/26/2019,01 07 18 21 22 27 28 29 34 36 41 42 47 48 49 55 68 70 78 79
08/25/2019,05 06 09 10 11 24 27 30 35 43 49 51 60 63 65 66 72 75 77 78
08/23/2019,11 12 21 23 32 33 34 37 40 43 46 53 55 57 60 61 62 63 67 77
08/22/2019,08 11 13 14 17 19 20 23 27 36 43 44 50 52 55 59 68 72 73 76
08/21/2019,12 17 21 24 27 30 36 39 40 43 47 54 58 59 61 64 66 70 74 79
08/20/2019,03 04 13 19 27 28 30 32 36 37 44 50 55 64 66 70 74 75 77 79
```

```
09      08/19/2019,03 04 11 16 18 21 24 28 32 41 43 57 58 59 61 64 68 70 71 79
10      08/18/2019,03 07 12 14 20 23 35 36 38 39 45 47 49 51 52 53 56 59 67 74
11      08/17/2019,09 11 12 16 19 25 31 33 36 38 39 40 42 43 49 50 59 61 67 73
12      08/16/2019,06 08 09 14 16 17 18 20 23 25 42 47 51 55 56 58 59 67 72 76
13      08/15/2019,02 04 07 11 14 17 22 28 31 32 34 39 42 45 57 58 60 61 74 77
14      08/14/2019,01 10 13 20 33 38 42 43 49 50 51 52 54 60 64 66 70 75 79 80
15      08/13/2019,07 08 13 17 19 20 23 24 25 27 28 30 31 37 39 41 46 57 61 63
16      08/12/2019,01 04 05 10 19 22 24 26 27 31 34 48 50 57 60 65 66 72 76 80
17      08/11/2019,01 02 09 10 11 16 18 19 26 38 43 45 50 54 57 65 66 69 73 79
18      08/10/2019,01 10 12 13 15 17 18 19 21 24 29 30 33 41 46 55 57 63 64 68
19      08/09/2019,05 12 16 22 25 27 32 34 38 39 43 45 51 58 61 66 68 72 73 78
20      08/08/2019,09 13 17 21 23 35 37 40 44 46 55 56 58 59 63 69 71 72 75 76
21      08/07/2019,01 04 09 16 20 24 25 26 40 41 44 59 60 64 68 69 71 73 76 77
22      08/06/2019,08 13 18 20 26 27 33 37 39 45 48 50 56 59 61 67 69 70 74 79
23      08/05/2019,03 05 08 09 14 20 26 27 34 36 37 41 42 48 55 56 59 60 67 79
24      08/04/2019,01 10 12 13 18 21 22 24 32 36 40 50 54 56 57 58 61 63 67 70
25      08/03/2019,01 03 04 17 18 21 27 30 35 40 46 51 59 62 65 68 69 75 77 80
26      08/02/2019,01 04 05 07 08 11 14 18 32 37 42 44 45 50 53 56 67 68 73 76
27      08/01/2019,02 05 06 08 12 14 24 26 27 31 36 38 43 44 46 53 54 62 64 80
28      07/31/2019,03 06 14 15 16 17 25 33 38 42 43 49 52 54 62 63 67 69 76 79
29      07/30/2019,02 12 13 17 18 22 28 29 32 38 48 50 54 57 59 68 69 71 72 80
30      07/29/2019,08 19 27 33 35 42 43 45 49 53 55 56 61 62 68 69 70 71 73 79
31      07/28/2019,10 11 12 13 20 31 33 37 38 41 42 45 50 52 56 58 59 68 76 79
32      07/27/2019,01 03 09 10 16 19 20 21 24 26 29 32 35 50 52 54 56 72 77 78
33      07/26/2019,02 11 30 32 33 35 37 39 40 42 47 52 56 57 60 62 63 68 72 80
34      07/25/2019,03 08 09 21 23 25 32 33 35 38 40 44 48 50 53 58 60 63 75 78
35      07/24/2019,01 11 14 18 25 30 36 39 43 44 45 46 47 48 57 60 61 69 77 78
36      07/23/2019,02 06 12 16 24 35 38 44 46 55 56 57 60 61 62 66 67 68 74 80
37      07/22/2019,03 04 05 09 24 29 30 31 33 38 41 44 48 50 62 64 70 74 75 79
38      07/21/2019,03 15 20 21 22 23 24 25 29 32 37 41 45 49 51 57 59 61 70 71
39      07/20/2019,01 04 11 12 14 21 27 29 30 38 40 42 43 50 51 58 68 69 78 80
40      07/19/2019,11 12 14 16 17 23 24 31 39 49 53 57 58 60 61 63 68 76 77 79
41      07/18/2019,02 03 08 13 14 19 29 31 33 40 43 45 53 54 58 60 61 62 75 78
42      07/17/2019,02 05 08 11 14 17 20 22 23 26 32 36 43 52 53 62 68 70 72 73
43      07/16/2019,01 02 05 08 09 12 25 40 41 42 47 56 59 60 66 67 71 77 79 80
44      07/15/2019,01 04 13 18 20 23 27 37 42 45 47 50 57 61 63 64 66 75 76 78
45      07/14/2019,01 03 06 07 11 12 16 19 21 23 27 30 40 50 55 65 70 71 73 78
46      07/13/2019,06 08 10 13 15 24 25 30 34 48 49 54 59 64 65 68 70 73 78 80
47      07/12/2019,01 06 16 17 19 24 36 37 38 46 50 51 52 53 63 65 66 69 71 77
48      07/11/2019,05 06 11 17 18 25 28 30 35 42 44 51 58 60 63 64 71 72 76 78
49      07/10/2019,07 10 14 18 23 31 32 35 40 48 52 56 61 62 68 74 76 77 78 80
50      07/09/2019,02 03 05 11 13 15 25 26 33 38 39 41 43 51 56 68 69 76 78 80
51      07/08/2019,09 14 16 20 23 30 35 36 38 41 47 53 55 57 62 63 65 67 72 73
52      07/07/2019,03 05 06 09 10 23 27 31 37 44 53 54 56 58 60 61 67 68 70 78
53      07/06/2019,03 08 13 17 18 22 33 34 35 40 43 44 45 53 58 61 66 67 69 76
54      07/05/2019,01 03 06 11 14 18 24 30 33 44 45 48 51 58 63 65 68 69 71 74
55      07/04/2019,01 02 08 10 17 18 22 26 27 29 34 37 39 44 53 60 62 67 74 75
56      07/03/2019,03 07 14 16 17 22 24 26 27 30 33 37 41 45 53 59 65 74 76 80
57      07/02/2019,14 21 22 27 29 44 50 53 54 56 57 60 63 65 66 70 71 76 77 80
58      07/01/2019,02 04 17 18 22 26 27 30 31 36 43 44 49 62 64 65 71 78 79 80
59      06/30/2019,08 09 13 14 16 17 25 28 30 34 38 40 51 52 57 66 67 69 70 77
60      06/29/2019,07 09 21 25 29 36 41 44 45 46 49 52 53 60 62 65 69 70 77 79
61      06/28/2019,03 11 18 21 26 28 31 32 35 36 46 50 52 54 56 58 59 63 67 73
62      06/27/2019,01 03 05 09 11 12 13 14 27 32 39 42 43 58 59 60 61 62 74 79
```

```
63      06/26/2019,01 06 09 13 18 23 26 27 29 30 33 36 40 41 58 63 65 67 73 76
64      06/25/2019,01 10 19 20 24 25 36 38 40 43 45 52 53 55 63 64 66 67 74 76
65      06/24/2019,13 14 19 22 23 35 37 39 41 44 46 47 48 50 52 56 57 61 64 80
66      06/23/2019,06 09 15 20 32 34 36 41 46 47 50 51 52 53 55 60 64 66 74 77
67      06/22/2019,03 04 06 10 11 12 13 15 23 27 32 37 41 45 47 54 60 65 68 72
68      06/21/2019,02 05 18 20 21 26 29 32 33 38 39 40 44 45 60 65 66 69 74 75
69      06/20/2019,01 02 03 04 33 35 39 43 46 47 51 52 56 60 62 69 72 74 78 79
70      06/19/2019,07 09 10 14 17 26 31 34 37 40 42 44 47 50 52 53 73 75 76 80
71      06/18/2019,01 03 04 07 08 12 14 22 32 37 39 44 49 59 60 64 71 74 75 79
72      06/17/2019,02 03 04 07 18 21 22 30 31 32 35 39 43 44 49 52 67 69 74 76
73      06/16/2019,01 02 04 09 16 24 26 29 30 36 38 41 48 49 54 56 59 70 71 72
74      06/15/2019,01 06 08 14 24 27 33 39 40 46 49 51 52 54 59 62 63 69 72 75
75      06/14/2019,10 18 19 20 24 25 26 28 30 31 33 36 39 44 46 47 60 62 65 72
76      06/13/2019,18 21 22 25 28 30 32 33 34 37 40 41 42 47 51 56 59 69 71 80
77      06/12/2019,05 08 09 16 31 34 36 37 39 40 50 53 54 57 62 68 71 72 75 78
78      06/11/2019,07 10 11 14 18 21 35 37 40 42 51 53 55 59 70 71 72 76 79 80
79      06/10/2019,11 12 14 15 17 19 24 31 39 40 46 48 51 53 54 57 65 67 77 79
80      06/09/2019,01 12 17 19 23 28 35 36 37 38 45 48 54 61 64 66 67 72 74 75
81      06/08/2019,02 07 14 15 18 20 24 26 29 33 35 44 52 57 61 63 72 75 78 80
82      06/07/2019,02 08 13 15 16 18 24 25 28 33 37 40 45 49 52 55 66 68 78 79
83      06/06/2019,02 05 07 08 11 13 14 20 26 29 37 43 47 58 63 64 65 75 79 80
84      06/05/2019,01 02 03 14 19 26 27 35 40 46 47 51 53 54 58 72 73 74 78 79
85      06/04/2019,05 06 07 10 11 15 16 18 23 24 27 36 37 43 44 46 53 60 64 76
86      06/03/2019,01 10 18 22 28 32 41 44 49 57 59 60 61 63 64 67 69 71 75 78
87      06/02/2019,03 17 18 22 27 32 38 40 45 50 57 59 61 64 69 71 73 76 77 79
88      06/01/2019,06 10 16 29 33 35 36 46 48 49 50 51 53 55 61 63 67 75 77 80
89      05/31/2019,06 07 11 17 23 25 33 34 36 37 39 45 46 47 48 56 63 65 68 74
90      05/30/2019,04 05 08 14 18 19 20 23 25 28 36 43 44 53 56 59 61 68 70 72
91      05/29/2019,13 15 17 18 27 28 30 36 44 45 53 57 60 65 68 73 75 77 78 79
92      05/28/2019,02 06 13 16 23 26 27 29 38 39 40 41 50 54 59 69 70 71 73 75
93      05/27/2019,05 12 13 18 30 34 37 39 41 44 45 50 56 58 60 71 76 78 79 80
94      05/26/2019,01 04 09 12 17 20 24 25 32 33 40 41 45 50 52 64 66 70 77 80
95      05/25/2019,01 04 10 28 30 34 35 38 46 51 52 53 55 57 59 60 64 65 72 78
96      05/24/2019,13 15 18 26 27 35 39 42 47 51 54 60 61 64 65 66 69 78 79 80
97      05/23/2019,01 05 14 18 20 23 27 31 32 34 38 40 42 45 51 60 61 63 74 75
98      05/22/2019,06 08 13 15 16 17 19 20 27 33 34 50 51 53 54 60 64 68 73 78
99      05/21/2019,03 04 05 06 12 15 22 26 30 32 36 38 40 43 46 68 70 71 74 78
00      05/20/2019,07 09 16 20 24 30 32 35 37 38 40 44 59 60 62 64 66 72 75 76
01      05/19/2019,20 24 26 28 30 33 35 39 40 42 43 47 49 50 51 53 57 59 72 74
02      05/18/2019,05 06 07 09 11 16 19 21 23 30 36 39 52 61 64 65 68 70 71 77
03      05/17/2019,01 03 06 17 18 21 34 38 40 45 51 54 57 60 61 71 73 75 77 80
04      05/16/2019,02 06 10 12 13 14 23 30 33 38 42 44 46 49 51 57 61 62 65 77
05      05/15/2019,10 11 24 27 29 40 41 43 48 51 55 57 59 69 70 71 75 78 79 80
06      05/14/2019,01 07 15 17 21 23 27 30 38 41 42 43 44 45 50 51 54 73 74 79
07      05/13/2019,02 10 13 16 17 23 24 25 27 29 30 35 47 53 57 62 63 64 75 78
08      05/12/2019,06 12 15 16 17 20 28 29 32 35 41 44 47 48 51 52 59 64 76 79
09      05/11/2019,04 06 07 10 16 18 22 24 25 40 42 45 48 51 52 56 59 67 71 78
10      05/10/2019,11 12 15 17 19 24 25 30 31 32 33 37 38 43 54 58 60 61 69 77
11      05/09/2019,01 15 17 18 23 24 25 30 41 45 52 57 58 60 62 67 68 73 75 77
12      05/08/2019,01 03 06 08 14 15 23 25 28 30 34 47 53 54 55 57 63 72 75 80
13      05/07/2019,06 07 08 09 10 12 26 28 30 31 33 46 47 48 68 70 73 74 78 79
14      05/06/2019,12 23 24 26 28 30 32 33 35 41 43 45 54 56 58 59 67 69 75 80
15      05/05/2019,03 04 06 14 18 19 21 23 24 29 30 45 46 47 49 57 63 66 68 80
16      05/04/2019,14 17 21 22 23 24 32 35 37 39 40 41 49 61 62 63 68 69 71 78
```

```
17      05/03/2019,05 09 10 14 17 19 30 31 32 35 40 42 45 53 59 60 67 69 77 80
18      05/02/2019,02 05 09 13 17 20 21 23 24 29 31 39 42 44 45 56 57 64 72 78
19      05/01/2019,06 13 16 18 21 26 27 34 41 44 47 53 55 62 63 69 75 76 79 80
20      04/30/2019,01 08 13 16 21 25 29 30 37 43 44 45 46 52 61 66 68 69 70 79
21      04/29/2019,04 07 08 12 17 20 21 23 29 33 35 37 42 47 50 59 63 64 67 73
22      04/28/2019,01 04 05 07 10 16 18 28 30 31 43 46 48 55 62 64 65 69 76 79
23      04/27/2019,01 02 09 10 12 15 16 21 30 41 44 45 58 59 62 68 69 70 71 76
24      04/26/2019,03 06 07 10 12 13 19 24 27 31 33 40 43 46 48 70 75 76 77 80
25      04/25/2019,09 10 12 20 27 29 33 36 42 43 44 46 51 52 54 55 62 63 66 75
26      04/24/2019,02 03 04 06 07 08 10 19 23 32 34 38 44 50 51 54 58 64 70 71
27      04/23/2019,03 04 07 11 12 15 18 25 26 27 29 30 32 37 40 43 44 63 73 76
28      04/22/2019,01 02 06 15 20 25 39 41 43 56 57 58 59 61 64 66 68 70 74 80
29      04/21/2019,03 15 19 20 21 23 28 29 32 36 46 47 51 52 53 61 63 65 70 74
30      04/20/2019,01 08 09 10 13 14 17 19 20 24 31 32 35 46 60 61 62 64 65 80
31      04/19/2019,01 05 06 07 14 18 29 33 34 36 37 46 48 51 52 54 59 61 62 75
32      04/18/2019,12 14 15 18 23 26 30 33 43 51 52 53 54 62 63 64 69 71 72 74
33      04/17/2019,02 04 05 09 14 21 24 27 31 34 41 42 44 53 56 58 66 70 72 73
34      04/16/2019,05 07 14 17 20 27 41 42 45 46 49 53 55 57 58 64 65 69 73 79
35      04/15/2019,02 04 07 11 12 15 25 31 32 33 37 40 43 51 57 68 71 75 78 80
36      04/14/2019,03 04 09 10 13 15 22 25 26 29 36 45 58 59 63 67 71 74 75 80
37      04/13/2019,04 05 06 09 15 18 19 21 32 34 44 48 53 56 61 65 69 70 72 75
38      04/12/2019,04 06 08 26 27 28 29 31 43 46 48 49 57 60 63 69 71 72 76 80
39      04/11/2019,01 02 04 09 14 15 21 26 28 33 36 38 42 46 52 59 67 68 74 79
40      04/10/2019,06 08 10 17 18 27 33 38 39 43 46 47 51 53 54 58 62 65 71 74
41      04/09/2019,03 13 14 23 29 33 34 41 45 50 55 57 61 64 67 68 72 74 76 80
42      04/08/2019,05 06 11 17 26 29 30 33 35 41 43 46 55 63 65 66 68 71 74 78
43      04/07/2019,03 04 06 09 14 39 41 42 45 46 50 53 56 58 59 60 65 69 72 80
44      04/06/2019,04 08 15 16 17 20 28 29 30 32 39 47 49 52 53 54 56 59 77 80
45      04/05/2019,12 14 18 27 30 32 33 34 39 44 45 48 55 56 64 65 68 70 71 79
46      04/04/2019,01 04 08 11 18 19 22 23 25 30 32 44 62 66 67 68 70 75 76 80
47      04/03/2019,04 08 17 19 32 35 38 41 42 46 50 55 62 63 64 65 69 71 72 74
48      04/02/2019,02 04 08 14 16 21 29 31 41 45 51 53 57 58 61 67 71 74 78 79
49      04/01/2019,02 05 11 12 14 18 21 25 30 32 34 35 43 45 46 47 55 56 62 70
50      03/31/2019,04 13 24 25 30 33 37 38 43 45 48 50 51 57 60 61 64 72 76 77
51      03/30/2019,05 07 10 14 23 26 29 32 37 39 43 46 50 53 58 60 67 70 75 79
52      03/29/2019,02 05 12 13 16 20 22 25 30 32 41 52 53 54 64 65 70 72 75 76
53      03/28/2019,01 04 08 12 17 29 30 35 36 37 39 40 49 50 55 61 66 67 68 74
54      03/27/2019,01 02 07 15 20 21 25 36 38 40 42 46 50 51 52 63 64 68 74 80
55      03/26/2019,01 04 06 07 15 19 24 25 28 35 42 43 47 49 50 57 62 63 71 77
56      03/25/2019,08 09 13 19 20 27 32 38 39 45 59 60 61 63 70 71 73 76 77 78
57      03/24/2019,04 10 12 18 22 26 27 35 36 43 44 45 47 49 52 63 71 73 75 79
58      03/23/2019,01 09 12 16 27 32 35 37 38 40 41 49 53 55 57 65 72 74 76 78
59      03/22/2019,03 09 16 22 24 27 36 44 45 47 48 51 52 57 60 61 62 67 71 80
60      03/21/2019,02 07 10 11 21 29 30 31 32 40 43 47 60 65 67 68 71 73 74 76
61      03/20/2019,02 06 10 11 17 22 29 30 32 35 36 47 51 54 56 57 60 68 78 79
62      03/19/2019,03 09 11 13 22 23 25 35 40 41 44 45 49 51 54 59 61 65 68 78
63      03/18/2019,02 08 10 16 17 30 35 36 43 51 53 56 62 64 67 71 73 75 76 78
64      03/17/2019,18 26 33 34 35 36 40 41 43 44 45 49 50 53 58 59 62 69 73 78
65      03/16/2019,01 06 07 08 16 28 29 31 32 34 43 45 46 53 54 57 59 69 70 72
66      03/15/2019,02 04 07 12 15 24 34 36 44 51 52 53 59 61 62 63 64 65 71 80
67      03/14/2019,06 09 22 23 24 35 36 39 47 56 57 58 60 61 66 67 68 73 74 80
68      03/13/2019,01 04 07 17 24 26 33 35 37 39 45 47 50 55 56 60 63 69 71 72
69      03/12/2019,05 09 16 18 19 37 39 42 45 55 57 60 61 62 67 69 70 72 75 80
70      03/11/2019,03 04 06 10 11 12 18 22 24 37 40 41 48 54 58 60 67 68 75 76
```

```
71      03/10/2019,01 02 03 05 15 24 37 45 47 49 54 56 57 60 61 65 66 70 71 73
72      03/09/2019,09 13 16 19 20 26 28 30 33 43 45 46 47 53 55 61 62 66 67 76
73      03/08/2019,01 02 03 09 11 18 20 23 26 29 33 40 52 55 59 60 68 69 78 80
74      03/07/2019,02 07 12 13 19 23 26 28 33 36 48 56 59 63 65 70 71 75 78 79
75      03/06/2019,23 24 29 36 38 45 46 49 50 51 54 57 58 59 60 66 68 70 71 72
76      03/05/2019,05 08 15 18 22 27 28 31 32 35 41 46 49 54 55 56 63 64 68 72
77      03/04/2019,02 04 12 18 22 23 28 29 32 35 49 50 52 54 56 59 63 65 67 72
78      03/03/2019,05 06 09 12 19 22 44 45 48 50 53 54 56 62 65 71 73 75 77 80
79      03/02/2019,05 09 14 23 25 36 43 45 47 49 54 55 59 66 68 69 73 75 77 78
80      03/01/2019,01 02 03 06 07 09 21 28 30 39 41 47 49 51 57 58 65 70 72 74
81      02/28/2019,06 12 13 17 18 22 23 33 37 38 42 43 46 53 58 62 65 67 69 71
82      02/27/2019,01 02 22 24 26 31 44 45 46 48 55 56 57 58 59 60 62 73 74 77
83      02/26/2019,03 13 14 20 25 27 29 30 32 35 38 45 49 55 56 57 64 71 73 79
84      02/25/2019,07 18 19 20 21 26 27 28 34 36 43 52 55 60 67 69 70 74 77 80
85      02/24/2019,02 03 05 08 11 25 33 35 37 43 46 47 52 57 58 67 69 70 71 75
86      02/23/2019,03 08 12 22 28 34 35 36 39 40 41 42 49 53 55 63 68 70 72 77
87      02/22/2019,18 19 20 21 23 24 31 35 36 37 39 46 48 52 55 59 70 74 77 79
88      02/21/2019,11 13 22 23 24 26 33 36 37 42 45 54 56 57 58 60 63 69 75 76
89      02/20/2019,01 02 03 07 13 14 19 25 33 36 40 42 48 49 52 53 64 71 79 80
90      02/19/2019,05 07 09 17 22 33 35 36 39 40 41 44 46 51 57 61 71 72 73 74
91      02/18/2019,01 05 08 10 12 13 14 16 20 22 24 42 49 52 57 61 65 69 70 76
92      02/17/2019,08 10 13 14 16 17 20 22 26 30 32 33 37 47 48 71 72 73 74 77
93      02/16/2019,02 11 14 19 24 26 28 32 42 44 48 52 54 61 63 65 68 74 76 77
94      02/15/2019,03 11 22 32 35 37 40 41 42 49 51 52 55 58 59 68 73 74 75 79
95      02/14/2019,03 04 07 09 11 20 22 24 29 32 35 36 44 45 52 55 59 67 68 69
96      02/13/2019,08 15 16 17 21 28 36 44 45 46 50 53 54 56 58 65 66 72 76 79
97      02/12/2019,10 13 16 19 24 29 32 40 41 45 46 47 49 55 58 61 65 69 70 79
98      02/11/2019,02 09 12 13 18 20 27 29 34 35 40 45 50 52 57 58 61 68 74 76
99      02/10/2019,01 07 12 19 23 26 30 31 34 35 41 45 47 48 49 51 65 77 78 79
00      02/09/2019,01 02 03 06 07 12 21 27 35 39 46 50 51 55 59 60 62 63 76 77
01      02/08/2019,02 08 10 13 16 17 19 20 36 43 44 45 51 55 56 60 65 67 69 75
02      02/07/2019,01 07 19 29 30 31 32 35 36 45 53 54 57 68 70 73 74 75 76 79
03      02/06/2019,02 03 07 23 25 28 30 37 38 40 43 44 51 60 64 67 69 71 72 74
04      02/05/2019,03 04 13 17 18 25 36 42 43 48 49 50 54 56 58 60 64 66 74 78
05      02/04/2019,01 02 05 08 09 10 12 13 19 21 23 27 28 40 42 44 48 53 68 78
06      02/03/2019,11 12 21 22 23 25 29 31 35 39 45 47 48 56 57 61 66 70 79 80
07      02/02/2019,01 03 07 10 15 17 26 30 32 38 48 52 53 54 61 62 65 70 73 77
08      02/01/2019,01 02 03 04 11 12 13 22 24 26 49 51 54 55 59 60 65 69 76 80
09      01/31/2019,12 17 19 27 31 33 36 38 41 42 44 46 47 58 61 64 67 71 79 80
10      01/30/2019,03 07 11 13 20 23 35 37 40 44 48 50 52 56 57 60 63 69 72 75
11      01/29/2019,03 06 08 15 17 21 23 24 26 32 50 51 52 58 62 64 65 70 75 76
12      01/28/2019,02 09 11 12 16 22 24 25 33 34 46 49 56 59 64 70 71 74 76 80
13      01/27/2019,16 18 19 20 25 26 27 28 32 41 44 51 52 56 58 64 67 68 75 79
14      01/26/2019,01 05 07 12 16 18 19 29 34 39 43 44 46 49 52 57 70 71 72 74
15      01/25/2019,03 08 14 18 26 27 33 35 44 46 52 58 60 61 63 68 70 74 75 79
16      01/24/2019,02 08 22 24 27 29 30 32 34 38 45 46 52 55 59 62 64 66 73 77
17      01/23/2019,05 06 09 10 15 17 20 21 22 25 32 35 38 43 52 62 65 67 78 80
18      01/22/2019,03 04 11 13 14 16 17 21 22 36 38 39 43 44 46 47 51 61 66 70
19      01/21/2019,04 06 07 14 16 18 23 27 28 32 33 44 45 49 59 62 65 69 76 80
20      01/20/2019,04 06 07 09 15 16 17 19 27 34 36 41 43 46 52 60 69 70 73 74
21      01/19/2019,13 14 20 28 30 34 39 41 44 46 47 48 51 54 63 70 72 78 79 80
22      01/18/2019,03 07 11 13 16 26 33 35 43 49 52 53 58 61 62 64 66 69 73 77
23      01/17/2019,01 03 06 09 10 12 21 25 27 34 38 43 46 47 51 55 60 62 78 80
24      01/16/2019,05 08 09 14 15 16 17 21 26 28 34 35 49 62 63 64 71 72 76 77
```

```
01/15/2019,01 02 03 11 14 23 29 40 41 42 43 47 48 57 59 62 63 70 73 75
01/14/2019,08 15 16 22 24 25 31 37 41 53 55 62 63 64 69 73 74 75 77 80
01/13/2019,02 08 09 17 20 21 30 34 37 41 45 53 54 55 56 64 71 76 77 80
01/12/2019,04 05 11 15 28 37 38 42 44 45 48 52 57 59 62 67 68 72 78 80
01/11/2019,03 04 08 10 12 14 15 17 28 29 30 54 55 56 59 61 65 67 71 78
01/10/2019,14 16 26 28 29 35 36 39 40 42 44 45 49 57 58 61 64 66 70 79
01/09/2019,07 08 09 12 16 17 24 26 33 34 37 38 40 43 48 57 70 71 73 78
01/08/2019,06 11 19 21 22 28 30 31 38 40 46 47 52 53 59 65 71 75 77 80
01/07/2019,05 08 12 13 19 20 25 26 27 31 33 38 43 52 60 63 64 67 74 76
01/06/2019,01 04 08 13 15 31 33 36 42 47 52 53 57 58 62 68 70 75 78 80
01/05/2019,02 03 05 07 20 23 27 28 32 34 37 44 48 49 53 57 66 69 74 78
01/04/2019,03 06 09 11 19 32 38 39 40 41 46 47 52 55 59 65 66 73 75 77
01/03/2019,02 10 12 27 30 31 34 36 47 49 50 52 56 59 62 65 66 69 79 80
01/02/2019,02 03 06 10 14 16 25 26 52 53 54 56 62 65 70 71 74 77 79 80
01/01/2019,04 06 10 11 12 20 26 27 32 33 44 50 59 63 68 71 72 76 77 79
12/31/2018,10 19 20 21 27 29 37 40 49 55 59 62 65 68 70 71 75 76 77 79
12/30/2018,01 08 09 12 13 15 37 41 42 45 46 49 50 52 56 59 63 75 77 80
12/29/2018,01 04 09 11 15 18 20 23 24 41 43 45 46 48 55 56 59 63 65 78
12/28/2018,01 04 05 09 15 16 19 20 23 24 27 29 32 37 46 51 66 69 75 77
12/27/2018,02 16 19 21 22 23 26 28 31 33 36 42 43 51 54 55 63 66 72 77
12/26/2018,01 03 06 09 10 14 28 35 38 42 43 52 53 54 57 60 64 70 75 80
12/25/2018,06 08 12 15 18 21 30 31 34 38 50 51 55 59 63 69 70 71 75 79
12/24/2018,09 13 20 21 30 32 33 36 43 47 52 53 57 59 61 67 69 73 75 80
12/23/2018,09 10 18 22 23 29 32 36 37 43 45 46 50 59 61 63 65 68 75 79
12/22/2018,08 11 15 18 20 21 25 29 36 38 39 42 47 49 56 59 63 72 77 78
12/21/2018,01 05 07 08 16 18 26 27 38 42 48 49 51 52 53 55 57 60 72 80
12/20/2018,01 05 11 17 19 22 31 34 41 42 49 50 54 61 64 65 74 76 77 80
12/19/2018,14 16 17 19 31 36 39 41 43 49 50 51 57 58 61 68 70 77 78 79
12/18/2018,02 05 07 09 12 16 19 24 27 30 32 37 45 47 52 57 60 63 74 79
12/17/2018,07 15 17 19 28 31 32 34 37 40 42 43 54 59 60 61 65 68 70 78
12/16/2018,03 04 07 08 10 12 23 27 31 38 40 41 44 45 52 55 62 65 67 79
12/15/2018,04 06 09 14 19 23 24 26 33 34 36 37 45 48 52 68 70 74 75 78
12/14/2018,03 07 16 18 19 21 26 27 30 33 45 48 50 53 57 61 62 67 76 79
12/13/2018,05 12 13 15 17 23 28 40 44 51 56 59 61 62 64 69 74 76 77 78
12/12/2018,04 10 12 17 21 30 34 42 45 47 49 50 51 57 60 64 65 67 76 77
12/11/2018,01 04 06 07 13 20 25 40 47 50 52 59 61 62 67 69 70 71 73 77
12/10/2018,02 04 12 23 27 28 29 30 31 33 35 40 41 44 45 48 61 64 74 79
12/09/2018,06 08 10 12 13 15 19 26 32 48 50 52 55 58 60 64 66 69 76 79
12/08/2018,04 12 13 14 20 25 26 34 36 41 42 49 53 54 59 63 67 69 70 74
12/07/2018,07 10 18 21 29 31 35 37 38 41 53 56 57 59 64 66 69 73 76 80
12/06/2018,09 11 15 16 19 29 30 34 37 44 50 53 55 60 61 62 73 78 79 80
12/05/2018,03 06 15 16 22 24 26 28 34 38 41 42 44 45 49 54 60 67 74 80
12/04/2018,01 09 11 16 20 25 27 31 40 44 52 54 58 61 67 68 69 70 73 77
12/03/2018,10 13 14 15 16 21 22 23 31 38 45 49 51 53 59 62 63 66 74 77
12/02/2018,03 05 08 09 13 27 29 31 36 39 43 54 56 57 64 68 69 72 76 80
12/01/2018,05 09 10 14 16 17 19 21 23 25 28 33 42 59 61 63 69 71 72 77
11/30/2018,06 13 14 16 23 30 31 32 36 38 43 44 50 51 53 62 63 69 71 79
11/29/2018,04 05 11 12 19 21 23 33 34 36 51 52 55 58 61 62 64 70 72 73
11/28/2018,01 02 04 08 09 15 18 19 21 24 26 29 34 38 42 43 45 61 62 79
11/27/2018,03 05 20 21 27 30 32 37 42 46 48 50 51 56 60 66 68 69 72 79
11/26/2018,02 12 13 23 34 35 40 41 44 45 46 49 51 52 53 62 66 73 74 75
11/25/2018,01 03 06 08 10 11 15 27 35 37 39 42 46 47 50 58 59 74 77 80
11/24/2018,09 12 13 16 19 24 28 29 33 34 36 39 48 54 63 68 69 71 76 79
11/23/2018,01 02 04 08 09 19 20 23 25 28 34 38 40 42 46 51 52 55 65 80
```

```
11/22/2018,02 07 10 12 14 18 23 24 26 35 43 47 50 59 60 62 67 71 74 79
11/21/2018,02 03 04 06 07 08 11 15 19 22 26 29 32 41 43 46 48 52 56 74
11/20/2018,04 08 12 15 18 19 21 29 33 36 37 38 39 47 56 63 64 69 70 73
11/19/2018,01 09 12 13 16 22 31 32 38 40 42 43 45 46 51 54 60 63 70 75
11/18/2018,17 21 22 25 30 31 36 40 43 46 48 52 54 60 62 63 64 67 70 78
11/17/2018,02 08 15 17 20 24 29 31 32 34 37 39 44 53 56 57 63 65 67 69
11/16/2018,01 08 09 12 14 19 25 26 38 42 43 46 49 52 57 62 68 69 78 80
11/15/2018,01 05 15 16 22 24 25 26 30 32 33 35 39 47 61 63 64 67 73 77
11/14/2018,09 15 17 20 22 26 28 29 41 51 55 56 58 61 65 67 68 70 71 72
11/13/2018,02 05 10 13 18 19 26 33 41 43 45 46 47 52 54 61 71 77 79 80
11/12/2018,01 06 07 08 09 13 14 27 29 30 32 35 39 50 55 57 58 70 74 76
11/11/2018,06 15 23 29 33 36 37 40 48 50 53 56 61 62 63 66 67 69 71 80
11/10/2018,02 04 08 10 14 21 22 28 33 45 47 50 51 53 59 63 70 74 77 79
11/09/2018,17 19 27 29 41 42 49 54 58 60 61 62 63 64 65 66 69 70 75 77
11/08/2018,01 09 11 12 18 26 34 38 39 40 42 43 47 53 59 60 64 66 69 74
11/07/2018,05 10 13 19 22 23 24 30 31 32 33 37 39 41 44 47 53 57 72 77
11/06/2018,01 04 05 07 09 23 24 25 26 27 29 31 41 46 51 54 57 59 72 78
11/05/2018,02 06 09 13 19 24 26 28 37 38 51 52 55 57 62 63 71 73 75 77
11/04/2018,06 07 09 14 16 24 25 26 39 50 52 55 59 60 61 65 66 69 76 78
11/03/2018,01 06 08 09 13 21 23 28 29 33 40 41 42 45 46 50 64 69 73 80
11/02/2018,11 14 21 24 26 27 28 29 33 37 42 46 49 52 59 60 66 67 77 79
00 11/01/2018,01 08 09 20 23 24 29 30 32 39 40 42 46 47 54 58 68 77 78 80
01 10/31/2018,04 08 09 17 25 26 30 32 35 37 39 43 46 53 56 64 67 69 77 78
02 10/30/2018,01 04 08 09 12 16 22 25 28 31 39 43 48 53 55 58 64 72 75 77
03 10/29/2018,01 13 22 31 35 41 42 43 45 50 52 55 60 63 68 69 70 74 79 80
04 10/28/2018,03 05 06 15 16 17 21 22 32 36 37 44 45 49 55 63 66 71 72 74
05 10/27/2018,05 08 10 11 12 13 22 24 30 31 35 38 41 46 48 56 59 65 70 72
06 10/26/2018,02 03 06 08 09 11 12 14 22 39 42 50 52 67 72 74 76 77 78 79
07 10/25/2018,04 08 17 20 23 34 35 37 41 42 43 50 55 57 60 61 63 67 72 80
08 10/24/2018,02 09 14 15 16 17 20 26 29 32 39 46 49 52 58 61 69 72 73 77
09 10/23/2018,01 06 08 17 19 21 22 26 31 38 41 42 59 63 65 66 68 74 77 78
10 10/22/2018,03 04 09 12 14 36 40 44 45 47 53 54 62 64 69 73 74 75 76 77
11 10/21/2018,02 04 12 14 15 18 25 27 30 31 40 42 45 46 49 53 55 69 73 77
12 10/20/2018,06 08 09 10 16 20 21 25 36 39 43 49 58 61 66 72 74 76 77 79
13 10/19/2018,03 06 11 12 15 20 34 38 43 45 49 57 60 63 65 66 68 69 76 80
14 10/18/2018,03 04 05 11 17 23 24 30 33 37 38 39 43 49 50 53 62 70 73 77
15 10/17/2018,03 11 13 18 23 24 30 32 33 35 36 41 43 47 51 53 63 64 76 79
16 10/16/2018,02 04 06 11 21 24 26 31 36 37 46 57 59 62 65 66 67 69 71 76
17 10/15/2018,04 13 14 18 19 25 30 36 44 47 59 60 61 64 69 70 71 74 78 79
18 10/14/2018,04 05 18 22 23 24 45 47 50 51 57 58 59 66 69 71 74 77 78 79
19 10/13/2018,01 03 05 06 09 10 11 12 25 34 35 38 49 52 59 61 64 70 71 80
20 10/12/2018,03 04 08 14 15 18 19 28 31 33 38 41 42 43 47 50 53 59 65 75
21 10/11/2018,07 13 15 18 26 28 32 34 38 42 49 50 51 56 58 59 65 72 79 80
22 10/10/2018,02 04 06 07 14 15 23 30 31 33 38 45 48 49 51 53 54 55 56 69
23 10/09/2018,02 06 07 10 11 17 20 29 31 34 38 40 41 42 43 53 57 59 62 73
24 10/08/2018,03 09 14 16 23 29 34 44 45 48 50 51 54 59 61 69 72 75 76 80
25 10/07/2018,01 08 11 14 18 24 26 28 43 51 54 57 61 63 68 71 72 75 78 80
26 10/06/2018,01 02 18 20 22 27 30 46 47 50 52 56 59 60 65 67 69 70 75 76
27 10/05/2018,03 06 07 08 12 14 22 29 31 32 35 38 40 42 43 48 54 56 58 76
28 10/04/2018,01 07 09 14 19 21 30 32 33 35 37 39 41 48 52 54 66 69 71 73
29 10/03/2018,06 12 14 16 19 23 31 32 45 46 51 52 55 61 66 70 72 74 75 80
30 10/02/2018,02 03 04 07 09 16 36 38 39 46 47 49 52 54 66 70 72 73 76 79
31 10/01/2018,01 03 05 11 16 18 21 29 30 35 37 47 57 59 63 67 68 72 73 75
32 09/30/2018,12 13 15 16 20 22 26 31 33 38 41 45 58 61 62 63 67 68 70 76
```

```
09/29/2018,05 08 09 15 16 22 26 31 35 36 37 52 53 61 67 69 70 75 77 78
09/28/2018,06 12 13 16 22 26 28 31 42 45 53 57 58 61 65 66 68 77 78 79
09/27/2018,04 07 11 14 30 35 36 37 41 44 57 58 61 62 64 69 71 72 77 78
09/26/2018,02 05 07 08 09 14 22 23 24 25 30 35 41 47 54 58 64 65 76 78
09/25/2018,02 09 10 21 28 31 32 34 35 36 40 44 45 52 53 54 60 63 68 71
09/24/2018,01 02 03 07 09 10 11 28 32 33 41 42 44 49 50 54 67 77 78 80
09/23/2018,05 17 20 21 22 34 36 43 46 50 53 54 59 62 63 67 70 73 78 80
09/22/2018,04 06 08 09 15 19 21 23 33 37 41 43 45 47 60 62 71 73 78 79
09/21/2018,03 06 14 20 21 37 41 45 46 53 54 55 56 60 63 67 71 72 76 80
09/20/2018,03 08 13 24 26 29 32 34 36 39 40 43 44 47 48 55 62 67 68 79
09/19/2018,01 04 07 08 11 15 18 27 28 34 42 43 45 54 57 59 60 63 66 68
09/18/2018,01 09 12 25 26 28 29 30 31 35 44 45 49 54 62 63 67 74 79 80
09/17/2018,17 18 21 25 27 36 38 46 47 48 53 54 55 60 63 65 66 72 75 78
09/16/2018,07 10 16 19 24 27 29 32 33 35 40 49 53 54 58 60 61 71 72 75
09/15/2018,02 06 07 08 15 25 27 34 42 45 46 47 51 54 55 64 65 66 76 79
09/14/2018,04 18 19 27 31 35 37 38 42 43 47 48 50 52 54 60 64 68 74 78
09/13/2018,01 03 07 09 13 17 20 24 29 31 33 39 45 46 47 59 63 64 67 74
09/12/2018,02 03 04 05 12 15 19 20 31 37 38 39 42 43 44 49 57 62 67 77
09/11/2018,02 09 20 22 25 29 35 36 39 40 50 51 66 67 68 69 71 77 79 80
09/10/2018,05 06 13 15 20 25 26 30 38 43 45 51 55 56 67 69 72 74 76 77
09/09/2018,08 09 14 20 21 25 27 30 35 37 38 42 47 50 54 62 64 65 66 68
09/08/2018,08 09 13 14 23 25 26 28 35 39 41 43 46 49 51 54 57 62 67 80
09/07/2018,02 08 09 11 25 27 32 43 44 48 50 56 57 58 62 64 65 67 71 79
09/06/2018,01 03 09 14 16 18 20 26 32 34 37 48 54 58 61 62 63 75 77 79
09/05/2018,02 14 15 26 27 28 30 34 35 36 41 43 45 50 53 55 60 66 69 72
09/04/2018,03 05 06 08 18 20 24 27 34 36 38 55 62 63 67 69 70 71 72 80
09/03/2018,05 08 12 20 23 24 25 29 30 37 47 48 52 56 59 65 70 72 76 79
09/02/2018,04 08 09 11 13 18 20 24 26 31 39 40 50 53 55 60 67 70 72 78
09/01/2018,03 05 09 13 15 23 24 27 28 33 42 46 47 50 56 60 61 65 72 79
08/31/2018,14 18 21 23 32 36 38 40 44 45 48 51 58 59 60 62 65 67 76 79
08/30/2018,01 03 04 05 07 09 10 11 35 38 39 41 45 46 47 57 59 67 71 79
08/29/2018,09 13 19 21 24 29 30 36 37 39 42 44 47 55 60 62 67 68 69 80
08/28/2018,01 05 09 12 20 21 22 30 32 33 47 48 52 54 56 63 66 67 69 76
08/27/2018,06 08 09 14 17 18 22 23 28 41 50 51 59 60 61 63 65 73 77 78
08/26/2018,04 05 08 09 10 14 15 20 27 31 41 47 51 55 61 66 68 77 78 80
08/25/2018,06 08 13 15 20 22 23 33 34 36 37 45 47 52 55 63 64 66 72 75
08/24/2018,01 02 03 06 10 13 25 26 32 34 38 39 40 52 57 59 60 64 69 80
08/23/2018,02 03 06 07 08 09 30 32 37 43 45 47 50 54 57 58 73 74 76 78
08/22/2018,02 09 11 12 13 14 19 23 27 33 40 42 49 52 56 57 59 65 73 74
08/21/2018,02 04 05 11 13 15 16 20 21 22 23 24 25 35 37 39 41 66 70 71
08/20/2018,07 09 24 31 33 36 37 40 42 46 51 56 57 60 62 66 73 75 77 80
08/19/2018,01 04 06 15 24 27 31 34 41 48 52 58 59 63 65 68 73 74 75 78
08/18/2018,08 12 13 16 28 33 34 37 38 42 45 49 55 56 67 70 71 72 77 78
08/17/2018,05 06 07 09 10 17 24 25 29 33 34 36 43 50 53 62 65 66 75 78
08/16/2018,03 04 07 08 09 20 24 28 32 36 40 44 45 47 52 58 61 65 72 77
08/15/2018,08 09 18 19 21 25 29 31 35 42 43 47 50 51 57 61 62 66 75 76
08/14/2018,03 15 21 23 30 31 32 41 50 51 52 55 66 68 72 73 74 75 76 77
08/13/2018,03 07 08 09 12 19 22 28 30 31 33 37 41 43 44 46 49 63 64 79
08/12/2018,01 06 07 09 15 18 22 31 34 40 41 53 61 65 70 73 75 76 77 79
08/11/2018,01 02 03 08 13 17 19 23 27 28 33 50 52 53 55 56 59 60 65 76
08/10/2018,03 05 14 16 22 24 29 30 34 36 40 41 53 55 64 66 70 71 74 75
08/09/2018,01 03 04 08 39 41 42 47 52 57 59 60 61 65 67 69 70 72 73 79
08/08/2018,07 15 17 20 21 22 23 25 26 32 35 41 42 45 47 52 56 58 69 73
08/07/2018,02 03 10 16 17 19 22 23 25 26 36 37 49 50 53 54 56 61 68 73
```

```
87      08/06/2018,01 09 10 13 18 22 26 32 36 38 39 40 44 45 49 60 63 65 68 76
88      08/05/2018,05 09 21 26 31 32 36 39 41 42 46 50 53 60 62 64 65 72 75 78
89      08/04/2018,02 04 05 08 18 22 23 28 33 35 38 45 48 49 52 58 61 62 65 68
90      08/03/2018,02 05 06 10 16 24 30 41 43 44 48 52 56 57 60 64 69 72 74 78
91      08/02/2018,11 17 22 23 24 25 28 30 31 32 33 39 41 45 50 53 66 74 75 76
92      08/01/2018,02 03 05 07 12 27 28 31 33 45 46 54 56 61 63 66 67 73 75 76
93      07/31/2018,12 14 16 19 25 26 31 34 40 42 44 47 49 60 64 65 68 71 76 80
94      07/30/2018,02 06 07 09 16 18 19 23 30 36 43 45 60 61 62 65 67 70 73 75
95      07/29/2018,02 06 09 13 14 16 17 20 23 37 43 44 54 55 56 59 64 67 71 80
96      07/28/2018,03 04 07 09 12 13 16 17 20 22 27 33 38 52 58 59 62 63 66 69
97      07/27/2018,01 06 09 13 18 22 23 28 36 37 39 42 57 61 64 65 68 74 75 79
98      07/26/2018,07 14 19 24 27 32 37 39 40 49 57 61 65 68 71 72 73 74 76 80
99      07/25/2018,03 04 05 06 07 09 11 16 17 23 37 40 43 53 58 62 63 66 67 72
00      07/24/2018,05 06 09 17 18 19 23 29 30 31 39 46 56 58 62 66 73 75 77 80
01      07/23/2018,03 12 14 15 20 23 25 29 33 34 41 57 59 66 67 69 71 74 77 78
02      07/22/2018,02 03 04 08 11 14 17 18 24 26 27 30 35 36 48 60 62 67 74 77
03      07/21/2018,08 12 16 20 22 28 37 38 39 42 44 48 58 63 64 67 71 72 79 80
04      07/20/2018,03 04 08 10 14 15 17 25 27 31 33 36 37 45 50 51 59 65 69 72
05      07/19/2018,04 08 12 17 22 31 34 36 45 49 50 51 54 59 64 65 67 70 72 80
06      07/18/2018,02 18 20 25 27 30 31 33 35 36 44 47 52 53 62 63 67 72 73 75
07      07/17/2018,01 07 15 16 18 22 23 34 38 39 46 49 52 53 58 63 64 68 70 75
08      07/16/2018,03 05 18 31 37 39 40 42 46 47 48 51 55 56 57 61 67 68 69 70
09      07/15/2018,01 09 19 28 32 35 38 42 44 47 48 55 58 59 62 63 64 73 75 78
10      07/14/2018,03 09 16 24 26 30 31 33 38 46 51 57 59 61 62 72 75 77 78 79
11      07/13/2018,01 06 08 17 20 26 35 39 52 53 56 58 59 65 66 68 72 73 74 78
12      07/12/2018,02 05 06 18 19 20 24 27 28 29 31 40 42 46 49 52 60 64 69 76
13      07/11/2018,01 02 03 10 11 15 18 20 21 22 23 39 47 50 52 55 72 73 76 80
14      07/10/2018,02 04 08 15 21 22 38 39 41 42 43 45 46 54 59 65 66 68 76 78
15      07/09/2018,04 08 11 17 21 22 23 25 29 31 35 43 44 50 59 60 61 65 69 79
16      07/08/2018,04 09 17 26 27 28 30 40 47 48 50 51 53 55 63 72 73 75 79 80
17      07/07/2018,05 06 08 12 22 25 31 33 36 40 42 48 50 51 57 62 64 75 76 79
18      07/06/2018,02 06 09 10 13 17 27 30 33 38 42 43 48 52 56 59 61 64 71 77
19      07/05/2018,06 17 26 29 31 33 34 35 37 40 41 43 44 49 51 52 55 59 66 75
20      07/04/2018,01 02 05 10 11 12 15 17 19 20 23 29 30 37 39 42 44 56 65 73
21      07/03/2018,03 06 07 09 14 27 28 29 31 32 36 42 44 50 53 56 59 67 71 79
22      07/02/2018,05 07 13 14 21 23 28 34 42 43 45 46 49 54 57 63 65 71 72 80
23      07/01/2018,03 04 10 12 14 15 19 20 21 24 28 40 44 48 53 61 62 67 68 70
24      06/30/2018,02 03 05 06 07 10 16 18 20 29 36 37 38 41 42 48 57 59 68 71
25      06/29/2018,03 06 08 09 11 12 26 31 36 38 46 56 57 59 67 71 72 73 75 76
26      06/28/2018,11 17 18 23 29 32 36 38 41 43 44 46 47 49 53 56 62 70 75 80
27      06/27/2018,03 05 09 11 13 25 28 33 34 36 41 45 47 51 53 54 59 64 65 77
28      06/26/2018,02 10 27 28 29 30 35 39 40 45 50 54 57 58 59 67 72 74 75 77
29      06/25/2018,01 06 12 18 19 34 36 42 45 51 52 56 57 62 65 66 71 76 78 79
30      06/24/2018,02 04 07 09 10 18 20 26 27 34 39 50 51 55 59 64 68 74 76 77
31      06/23/2018,03 09 16 20 32 34 36 41 43 44 45 46 51 53 57 60 62 67 71 76
32      06/22/2018,01 03 08 09 10 18 20 23 24 27 31 33 35 38 47 57 59 60 65 75
33      06/21/2018,05 13 16 18 21 27 32 36 37 43 44 46 47 49 59 63 64 66 73 80
34      06/20/2018,02 09 12 16 17 18 23 26 27 29 38 39 47 56 61 65 67 77 78 80
35      06/19/2018,03 04 05 14 15 16 19 20 21 22 28 30 31 37 38 45 52 58 60 65
36      06/18/2018,01 02 05 06 09 10 15 19 36 39 50 57 59 60 64 68 69 70 75 77
37      06/17/2018,06 07 10 14 17 19 23 31 35 38 43 47 50 58 59 62 64 67 72 73
38      06/16/2018,05 13 19 20 26 35 40 42 43 51 56 57 62 70 71 73 74 76 78 80
39      06/15/2018,06 11 13 15 16 20 24 25 27 31 34 40 50 53 58 59 66 67 75 77
40      06/14/2018,03 08 19 20 26 28 33 49 54 55 58 60 61 64 69 74 75 76 77 80
```

```
06/13/2018,01 04 05 10 12 14 15 17 18 22 37 42 43 46 57 58 61 67 74 76
06/12/2018,04 06 09 22 23 29 31 39 42 43 45 50 52 58 62 63 64 68 72 75
06/11/2018,03 15 17 19 27 32 34 37 40 43 44 47 48 49 55 61 63 69 72 73
06/10/2018,02 03 04 06 21 23 26 28 31 40 43 44 49 53 56 58 61 67 70 80
06/09/2018,02 11 19 23 24 26 28 29 36 37 39 52 55 57 58 59 70 72 77 80
06/08/2018,06 10 14 22 27 34 35 41 47 49 52 54 56 57 60 63 71 76 77 80
06/07/2018,03 07 09 12 21 25 46 50 51 52 54 56 61 63 64 67 75 76 79 80
06/06/2018,06 07 08 14 23 26 32 36 38 39 43 44 47 50 51 54 65 70 72 75
06/05/2018,01 08 16 18 21 25 26 28 29 33 39 40 44 50 51 52 60 63 65 68
06/04/2018,04 08 09 11 13 16 19 27 30 34 35 36 38 39 43 49 50 60 64 73
06/03/2018,02 03 04 07 10 14 25 26 27 33 34 37 40 45 55 56 64 71 75 76
06/02/2018,09 10 13 17 21 23 26 32 35 36 46 48 49 56 64 68 73 74 76 80
06/01/2018,03 08 15 22 24 29 31 34 35 36 40 44 47 50 51 59 61 64 67 79
05/31/2018,01 04 08 13 19 36 38 40 41 45 47 57 58 59 62 66 68 75 76 78
05/30/2018,04 11 12 16 17 20 24 31 37 40 41 44 45 53 57 71 75 76 77 78
05/29/2018,07 09 11 15 17 18 26 29 31 46 47 49 53 57 59 64 67 70 72 78
05/28/2018,02 04 08 14 15 19 20 21 25 29 31 32 34 35 49 54 63 66 67 74
05/27/2018,17 21 22 26 27 45 46 47 48 49 50 52 53 54 56 58 63 65 71 74
05/26/2018,04 06 08 16 19 30 33 37 44 49 52 55 56 60 62 73 74 75 76 78
05/25/2018,02 07 09 10 11 13 16 19 20 22 26 35 36 42 46 50 56 58 66 68
05/24/2018,08 12 15 16 21 33 37 42 43 44 47 49 50 55 58 62 63 68 74 78
05/23/2018,04 08 15 18 27 28 32 36 41 43 53 54 63 65 66 67 68 69 76 79
05/22/2018,01 05 13 16 18 23 24 26 27 29 30 40 42 45 49 54 55 60 66 80
05/21/2018,02 05 13 16 19 23 25 26 29 30 32 38 46 48 52 53 62 63 74 80
05/20/2018,02 03 05 06 12 15 22 24 26 28 34 37 40 41 42 53 54 64 73 75
05/19/2018,07 08 09 11 16 18 20 23 50 58 60 62 65 66 67 68 70 76 77 79
05/18/2018,04 11 12 25 27 29 30 32 35 37 44 48 50 58 63 65 66 67 68 73
05/17/2018,10 12 14 16 25 27 32 34 37 39 44 50 57 65 69 70 72 73 74 77
05/16/2018,06 07 09 14 22 26 30 33 35 39 41 43 44 45 47 52 58 69 70 79
05/15/2018,05 06 14 27 29 32 34 37 43 46 49 53 58 61 65 71 72 73 74 75
05/14/2018,02 04 06 10 19 20 26 28 31 32 33 39 40 44 46 54 56 66 70 76
05/13/2018,02 06 07 08 11 18 21 25 26 30 44 47 59 63 66 68 69 73 77 80
05/12/2018,02 05 12 20 21 25 26 28 32 37 40 41 43 44 51 52 54 55 64 74
05/11/2018,02 08 12 14 15 26 31 37 41 43 49 51 54 56 59 62 64 69 76 80
05/10/2018,05 09 12 13 16 20 22 28 31 47 49 53 58 61 62 68 69 74 78 79
05/09/2018,01 05 16 18 19 20 26 31 32 33 45 50 51 53 60 61 62 64 68 77
05/08/2018,05 06 16 18 29 30 32 38 39 42 46 47 52 54 59 61 67 68 69 77
05/07/2018,04 08 15 21 23 28 37 40 42 43 44 46 47 48 49 55 57 59 61 68
05/06/2018,03 18 21 22 23 25 27 28 32 35 39 48 53 54 64 69 71 73 76 78
05/05/2018,08 15 16 17 18 19 24 25 27 32 34 38 41 42 49 52 56 61 63 78
05/04/2018,01 05 12 26 27 36 39 42 43 45 46 47 53 54 56 58 74 77 78 79
05/03/2018,03 04 07 11 14 17 19 24 30 32 42 51 56 60 63 65 66 68 72 79
05/02/2018,01 06 09 10 14 16 18 28 32 33 37 39 44 52 56 59 60 62 66 71
05/01/2018,16 25 28 29 30 31 34 36 38 40 44 45 46 50 58 59 62 64 66 77
04/30/2018,03 04 09 14 22 23 24 25 33 36 41 48 50 51 61 62 69 70 71 75
04/29/2018,01 03 15 19 23 34 40 43 48 51 55 56 63 66 67 69 70 71 75 79
04/28/2018,01 02 03 13 18 21 23 25 26 28 33 39 44 46 47 49 50 55 64 75
04/27/2018,06 11 16 18 27 28 30 31 39 41 62 64 65 67 69 70 72 73 74 80
04/26/2018,07 13 15 18 19 21 24 26 27 36 38 39 42 43 44 48 51 69 71 78
04/25/2018,03 09 12 20 23 26 29 30 31 34 39 41 43 44 45 62 63 64 68 75
04/24/2018,01 03 13 14 17 32 35 38 39 41 42 46 48 56 57 60 64 69 72 75
04/23/2018,11 12 13 30 32 36 37 39 40 41 47 48 53 54 56 60 62 65 75 78
04/22/2018,01 05 08 14 15 22 24 27 38 42 43 45 46 47 49 55 61 62 72 74
04/21/2018,05 12 15 21 23 27 29 31 33 42 45 51 52 53 56 57 61 63 74 78
```

```
95      04/20/2018,14 15 22 28 29 33 37 40 46 47 50 53 56 57 58 61 63 70 75 77
96      04/19/2018,05 06 10 18 22 27 28 30 32 35 51 58 60 63 67 69 70 75 77 80
97      04/18/2018,11 14 20 22 28 33 36 40 42 45 54 63 64 66 68 71 73 74 76 77
98      04/17/2018,08 09 12 17 20 27 34 36 41 46 47 55 58 67 68 71 75 76 77 78
99      04/16/2018,01 08 14 20 25 28 30 31 32 40 44 45 47 53 59 60 61 67 70 78
00      04/15/2018,05 07 12 13 21 25 31 32 33 34 37 46 48 49 53 55 56 63 69 80
01      04/14/2018,03 04 06 12 15 23 31 37 38 43 45 52 55 61 63 67 69 70 75 79
02      04/13/2018,03 09 13 19 21 33 34 37 45 49 50 55 56 60 64 71 73 74 75 78
03      04/12/2018,01 03 05 15 16 24 29 31 32 39 43 44 47 51 52 55 60 74 76 77
04      04/11/2018,03 06 09 13 15 17 18 25 29 38 40 42 43 50 52 55 63 71 75 78
05      04/10/2018,01 03 06 11 13 23 37 38 39 40 46 57 58 59 61 63 64 65 70 76
06      04/09/2018,06 13 14 15 26 27 35 38 40 43 48 49 53 54 58 63 69 73 77 79
07      04/08/2018,05 10 12 14 16 18 21 24 26 29 34 43 46 50 59 60 64 68 71 77
08      04/07/2018,02 06 07 10 24 31 32 38 39 40 41 42 46 47 50 56 57 59 61 69
09      04/06/2018,06 09 12 17 18 19 30 32 39 40 43 51 52 53 60 63 64 76 79 80
10      04/05/2018,04 07 09 11 18 20 24 29 30 39 44 46 47 56 63 69 72 73 74 75
11      04/04/2018,02 07 12 15 16 26 30 31 33 35 36 37 39 46 48 54 58 60 64 78
12      04/03/2018,02 05 11 14 16 29 30 31 33 34 36 40 48 55 57 67 69 71 78 80
13      04/02/2018,01 06 11 17 19 21 22 24 30 31 35 40 44 56 68 70 72 74 78 79
14      04/01/2018,02 04 17 25 27 29 30 32 37 42 44 49 58 60 61 62 66 69 79 80
15      03/31/2018,01 04 06 07 11 13 15 22 27 35 43 49 52 56 57 59 61 69 72 75
16      03/30/2018,07 11 12 17 20 23 27 31 48 49 52 53 55 63 65 70 71 72 74 79
17      03/29/2018,03 04 05 07 11 12 20 22 26 29 34 37 40 41 42 48 66 67 71 76
18      03/28/2018,03 04 12 15 25 33 46 47 53 59 60 62 64 65 67 69 74 77 79 80
19      03/27/2018,07 08 10 23 30 32 35 38 40 42 44 45 48 54 56 64 69 72 73 78
20      03/26/2018,01 08 12 13 14 17 27 32 37 38 39 41 50 53 54 61 67 71 72 74
21      03/25/2018,01 02 04 05 06 07 09 11 17 20 21 29 35 60 66 69 71 74 78 79
22      03/24/2018,04 13 15 16 21 23 30 36 40 43 45 46 53 61 62 64 65 68 70 74
23      03/23/2018,07 08 10 11 13 19 23 28 39 42 50 53 54 56 57 62 66 71 72 74
24      03/22/2018,07 13 14 15 20 21 24 26 27 30 33 34 45 52 56 59 68 70 75 76
25      03/21/2018,02 11 12 17 20 27 28 31 32 33 38 45 53 58 59 64 66 67 68 78
26      03/20/2018,01 09 12 13 14 19 22 23 25 33 35 39 50 51 54 56 64 75 76 78
27      03/19/2018,06 08 12 19 20 21 22 24 31 32 35 37 39 40 41 44 47 61 68 80
28      03/18/2018,03 05 06 07 13 14 15 17 21 32 33 35 39 40 42 57 65 70 72 80
29      03/17/2018,11 13 14 16 18 27 28 29 31 34 40 45 48 53 54 60 67 72 73 78
30      03/16/2018,02 07 09 16 17 19 29 34 35 38 39 44 49 50 51 62 64 69 70 80
31      03/15/2018,01 05 06 07 08 12 21 24 34 39 43 50 51 52 60 63 65 70 74 79
32      03/14/2018,04 05 16 20 21 27 28 29 33 35 38 39 40 44 53 55 57 58 71 72
33      03/13/2018,02 04 06 10 16 17 18 24 31 35 41 46 48 52 53 55 58 65 75 76
34      03/12/2018,01 03 05 08 24 29 31 34 35 38 41 42 45 46 49 52 62 67 70 80
35      03/11/2018,03 06 10 12 13 14 17 20 24 25 27 35 37 39 45 46 63 71 74 77
36      03/10/2018,02 10 14 15 16 18 28 38 42 48 51 54 55 56 62 67 69 70 73 80
37      03/09/2018,04 05 10 15 17 21 24 26 31 33 44 46 49 53 57 58 64 66 72 73
38      03/08/2018,03 04 05 13 15 20 33 35 37 41 44 45 52 54 57 60 71 75 76 80
39      03/07/2018,05 07 15 20 22 27 35 36 38 40 47 57 58 60 62 68 69 72 78 80
40      03/06/2018,03 12 13 24 26 30 36 42 44 45 50 52 56 57 60 61 63 68 76 80
41      03/05/2018,08 12 15 19 20 24 25 30 31 32 35 40 41 50 54 58 60 72 74 80
42      03/04/2018,05 07 08 09 10 14 19 22 23 28 32 33 42 45 60 63 70 72 76 77
43      03/03/2018,05 06 09 10 14 17 25 38 43 51 54 57 59 64 66 71 72 74 79 80
44      03/02/2018,03 04 07 10 11 23 25 27 28 35 36 45 46 65 68 70 71 73 77 80
45      03/01/2018,03 05 17 19 20 38 39 42 43 44 51 52 53 54 60 62 67 69 77 80
46      02/28/2018,01 06 07 11 13 14 17 27 30 32 36 37 49 53 61 63 70 73 77 78
47      02/27/2018,01 04 14 18 24 29 35 44 48 53 54 56 61 62 63 65 67 75 79 80
48      02/26/2018,03 04 08 09 12 15 19 22 26 28 35 40 46 54 56 57 59 62 68 80
```

```
49      02/25/2018,02 04 22 24 25 30 33 39 41 45 49 56 58 59 62 67 71 74 75 77
50      02/24/2018,07 08 15 17 19 23 25 29 33 37 38 42 43 46 54 55 56 66 69 71
51      02/23/2018,04 12 13 14 19 39 40 42 43 47 48 49 50 58 63 69 71 73 74 80
52      02/22/2018,02 06 09 10 11 15 16 21 22 23 39 41 49 54 55 59 60 66 76 80
53      02/21/2018,01 04 06 07 12 13 14 15 16 17 20 22 26 42 45 46 55 68 75 80
54      02/20/2018,01 02 04 05 06 10 15 20 23 26 29 30 31 32 36 45 50 58 67 74
55      02/19/2018,02 07 13 14 19 24 25 26 34 35 43 45 48 51 53 55 60 68 69 76
56      02/18/2018,02 03 06 11 12 13 14 19 20 26 28 31 33 42 49 51 52 76 77 80
57      02/17/2018,16 20 23 26 32 33 34 37 38 42 49 52 58 63 64 65 66 67 72 79
58      02/16/2018,05 06 15 25 28 38 39 41 42 51 56 59 62 63 69 72 74 75 76 78
59      02/15/2018,01 10 11 14 16 23 27 29 31 34 44 48 56 63 64 70 71 76 79 80
60      02/14/2018,07 24 26 27 29 30 34 38 41 43 49 53 56 58 60 65 69 73 75 78
61      02/13/2018,02 05 06 16 21 28 29 38 43 47 51 52 56 58 60 66 69 70 72 77
62      02/12/2018,01 04 05 19 22 23 26 29 33 38 40 46 48 54 58 61 64 66 71 74
63      02/11/2018,01 05 08 09 16 18 22 27 35 36 40 44 52 56 60 65 70 71 73 80
64      02/10/2018,06 14 21 24 25 33 34 42 44 47 51 56 58 64 70 72 75 77 79 80
65      02/09/2018,04 05 07 09 12 18 19 25 26 27 32 41 44 45 49 54 57 62 71 72
66      02/08/2018,01 05 08 11 22 26 28 35 37 43 45 50 54 60 64 66 68 69 71 75
67      02/07/2018,01 11 15 16 21 23 26 27 34 36 40 53 63 65 66 67 69 71 72 80
68      02/06/2018,02 09 11 18 24 27 30 33 35 36 43 48 52 53 56 60 64 66 68 77
69      02/05/2018,02 04 05 06 12 14 16 17 18 24 30 34 36 41 53 54 65 66 67 68
70      02/04/2018,03 04 05 08 10 12 15 18 19 33 38 44 49 51 53 59 66 68 73 75
71      02/03/2018,05 06 08 13 22 24 26 31 34 40 43 45 50 52 53 58 59 64 68 69
72      02/02/2018,03 07 08 13 31 33 35 38 39 40 49 51 61 64 65 66 70 75 78 79
73      02/01/2018,02 10 11 13 14 16 22 25 30 34 38 40 41 46 49 55 58 72 78 79
74      01/31/2018,02 03 06 11 18 21 23 28 29 34 38 45 47 48 57 59 61 71 73 75
75      01/30/2018,02 06 08 10 12 18 32 35 36 42 50 53 61 65 68 69 70 72 76 80
76      01/29/2018,05 07 09 13 16 18 21 24 30 36 49 51 54 56 59 60 70 73 74 77
77      01/28/2018,02 04 05 12 17 23 26 27 35 41 42 44 45 49 51 58 64 73 75 76
78      01/27/2018,01 04 16 19 25 27 28 31 39 42 50 56 60 61 63 66 69 72 75 78
79      01/26/2018,02 03 08 16 18 20 33 39 46 49 52 54 55 60 62 66 69 75 76 79
80      01/25/2018,04 05 07 09 13 19 30 31 34 35 50 51 52 55 57 60 62 66 68 77
81      01/24/2018,14 19 23 25 30 33 35 36 43 48 49 50 54 55 60 67 69 71 75 77
82      01/23/2018,07 08 11 15 23 31 34 35 40 49 51 54 55 56 60 64 74 75 77 78
83      01/22/2018,01 04 07 10 22 23 25 32 34 35 40 46 48 58 60 63 72 73 77 80
84      01/21/2018,02 03 04 10 12 16 21 24 25 34 44 51 53 64 65 70 71 73 75 78
85      01/20/2018,04 05 09 16 20 25 30 33 34 42 45 46 55 59 63 65 66 70 73 77
86      01/19/2018,05 08 10 14 16 18 24 26 38 40 50 51 53 55 56 59 60 63 68 79
87      01/18/2018,01 02 27 28 29 30 38 44 45 47 55 56 57 65 68 70 71 72 74 78
88      01/17/2018,05 11 17 24 27 29 30 35 38 42 52 53 59 60 61 63 66 73 76 78
89      01/16/2018,03 07 12 13 23 31 35 36 40 42 46 48 50 56 60 66 68 69 70 80
90      01/15/2018,01 07 15 16 17 32 36 38 43 50 53 56 57 59 70 71 72 73 74 76
91      01/14/2018,07 08 11 16 18 19 20 21 35 38 42 44 51 52 61 62 73 75 78 80
92      01/13/2018,02 04 14 21 22 23 32 37 44 47 48 50 52 58 60 62 65 68 76 77
93      01/12/2018,02 06 09 16 22 24 25 30 33 36 38 40 48 60 61 64 69 71 75 76
94      01/11/2018,02 03 07 10 11 13 14 15 23 27 35 50 51 55 56 57 61 68 76 79
95      01/10/2018,09 17 18 21 25 27 30 36 41 42 43 45 51 53 56 58 69 77 78 80
96      01/09/2018,06 07 09 12 14 18 21 25 28 33 35 38 48 57 65 66 67 68 69 72
97      01/08/2018,16 18 20 21 27 31 35 36 37 38 39 42 51 54 60 62 65 71 74 79
98      01/07/2018,06 21 24 27 29 30 31 32 33 35 36 40 46 55 59 61 67 71 78 79
99      01/06/2018,01 18 20 25 26 30 32 33 39 41 47 48 49 50 57 63 67 70 75 77
00      01/05/2018,01 09 10 11 13 18 26 27 29 31 33 34 37 44 57 59 61 62 69 76
01      01/04/2018,05 09 12 13 16 18 21 28 29 31 38 45 50 51 64 65 66 69 71 78
02      01/03/2018,01 05 08 13 18 20 26 30 36 38 42 45 47 49 51 53 60 62 63 64
```

```
03      01/02/2018,03 05 08 09 21 23 26 32 34 39 42 45 50 53 54 55 58 68 72 80
04      01/01/2018,02 04 09 11 12 25 31 36 39 49 56 57 60 64 67 69 70 72 74 76
05      12/31/2017,02 06 07 09 13 15 17 22 25 35 39 44 45 50 55 56 58 67 72 74
06      12/30/2017,01 04 06 08 19 20 26 28 30 31 32 34 36 39 52 56 57 59 62 71
07      12/29/2017,10 12 13 15 18 21 30 32 33 42 45 52 58 59 62 64 66 75 77 80
08      12/28/2017,05 09 13 14 15 16 19 25 32 36 41 47 49 50 52 59 62 67 73 76
09      12/27/2017,01 05 09 16 22 31 33 38 41 42 45 49 51 56 67 70 71 78 79 80
10      12/26/2017,05 06 08 12 15 18 27 30 39 49 51 53 56 58 59 63 66 73 78 79
11      12/25/2017,04 07 10 13 14 18 20 26 27 30 33 34 44 46 49 53 58 60 72 77
12      12/24/2017,04 15 18 20 22 23 24 30 34 36 38 50 51 58 61 64 73 75 76 79
13      12/23/2017,01 05 06 07 14 15 20 23 24 26 33 41 47 53 57 58 69 71 72 77
14      12/22/2017,01 07 10 16 23 29 34 37 39 42 46 48 51 52 59 63 66 68 75 78
15      12/21/2017,05 11 13 15 16 27 31 33 34 41 50 57 59 60 63 64 67 69 70 78
16      12/20/2017,03 06 10 13 15 25 27 43 45 47 50 51 53 57 59 63 67 72 73 76
17      12/19/2017,04 06 10 15 19 20 21 22 28 29 39 42 43 44 50 59 64 69 71 74
18      12/18/2017,05 07 08 09 12 17 21 24 34 36 37 49 50 51 57 63 64 67 68 72
19      12/17/2017,10 20 21 22 34 36 38 41 42 47 53 58 65 67 69 70 71 74 75 80
20      12/16/2017,02 06 10 13 20 22 24 29 36 39 43 45 47 48 56 68 69 72 76 78
21      12/15/2017,01 03 11 12 14 15 29 30 34 37 39 42 49 52 55 56 63 69 72 73
22      12/14/2017,13 16 18 19 22 24 35 38 43 46 48 49 50 56 60 62 70 77 78 80
23      12/13/2017,02 03 04 05 06 10 19 21 33 42 43 45 46 49 60 61 66 77 78 79
24      12/12/2017,01 03 10 11 13 14 18 22 36 45 46 54 56 59 63 64 72 76 77 78
25      12/11/2017,03 09 13 16 24 32 34 37 44 45 46 53 56 64 66 68 73 77 78 79
26      12/10/2017,02 03 06 11 14 16 17 19 22 23 26 45 63 64 65 67 68 73 74 76
27      12/09/2017,02 03 07 13 15 17 19 23 24 25 32 34 35 44 49 56 60 61 62 77
28      12/08/2017,01 13 19 21 24 25 28 29 34 35 43 44 50 51 59 67 68 71 76 77
29      12/07/2017,01 11 12 13 19 20 21 26 30 33 35 38 40 51 61 65 74 75 76 80
30      12/06/2017,01 13 14 15 16 17 18 21 24 38 39 58 59 60 61 63 67 68 71 76
31      12/05/2017,04 11 15 25 26 27 29 32 37 38 45 48 58 60 65 69 70 72 75 77
32      12/04/2017,07 08 11 12 14 20 26 34 37 47 49 50 51 64 68 70 73 74 76 80
33      12/03/2017,01 04 12 14 24 31 34 35 38 43 46 51 59 64 65 66 72 75 77 80
34      12/02/2017,02 08 09 22 29 32 34 37 38 46 50 53 57 58 60 63 68 69 77 80
35      12/01/2017,01 04 06 09 12 23 32 44 50 53 54 57 58 63 65 67 68 76 77 80
36      11/30/2017,02 12 13 21 25 27 28 30 37 45 53 57 59 61 63 64 69 71 73 80
37      11/29/2017,03 05 09 14 15 17 18 40 41 44 45 47 53 56 68 70 71 74 76 80
38      11/28/2017,07 10 15 18 20 26 30 33 42 48 51 52 56 57 65 71 73 75 78 80
39      11/27/2017,06 11 21 22 24 27 32 37 41 43 48 55 60 61 66 68 71 74 77 78
40      11/26/2017,03 04 05 09 10 15 18 19 23 29 48 49 52 55 59 69 72 73 75 76
41      11/25/2017,09 14 15 16 18 19 20 22 23 33 37 49 58 61 62 63 66 73 77 79
42      11/24/2017,03 08 09 11 16 18 20 23 24 34 35 48 52 57 65 66 72 73 76 77
43      11/23/2017,01 05 08 13 15 16 39 48 49 51 57 60 61 63 66 69 70 76 78 79
44      11/22/2017,01 04 07 12 13 23 24 26 34 37 40 43 57 58 64 65 67 70 75 80
45      11/21/2017,04 11 25 26 31 34 35 37 40 42 45 50 55 57 64 66 69 70 72 79
46      11/20/2017,03 04 11 12 19 20 23 24 26 27 34 41 46 47 50 51 57 64 70 73
47      11/19/2017,03 04 12 20 21 28 30 32 33 35 48 50 52 56 60 63 68 70 71 76
48      11/18/2017,13 17 19 25 30 31 33 36 44 48 50 53 57 58 61 63 64 67 72 75
49      11/17/2017,06 11 13 20 24 25 26 29 31 34 38 42 58 60 61 63 66 69 73 77
50      11/16/2017,09 14 18 20 22 28 29 30 33 39 40 47 51 52 54 56 64 65 67 76
51      11/15/2017,01 03 04 07 09 16 17 18 25 33 40 45 48 51 52 54 56 68 69 74
52      11/14/2017,07 13 14 17 23 24 25 26 29 33 41 42 44 47 48 49 59 63 64 71
53      11/13/2017,01 07 09 17 19 20 23 28 30 32 35 39 42 57 61 62 67 70 76 77
54      11/12/2017,01 02 06 08 11 12 13 27 29 38 40 41 43 45 56 57 62 64 74 77
55      11/11/2017,01 12 14 18 19 20 23 24 27 28 33 37 42 56 57 61 65 66 76 78
56      11/10/2017,06 14 17 18 23 27 28 34 41 42 45 49 53 57 61 66 74 76 77 80
```

```
11/09/2017,01 04 08 10 13 14 25 43 48 49 54 56 62 67 68 73 74 77 79 80
11/08/2017,01 05 09 14 15 20 22 29 31 42 44 48 52 57 62 63 68 71 74 80
11/07/2017,06 10 11 30 32 34 38 41 42 43 45 47 49 52 56 63 73 74 76 80
11/06/2017,04 06 14 15 21 22 32 36 39 45 46 47 52 59 64 67 74 75 76 78
11/05/2017,07 11 16 20 21 28 35 36 37 41 46 47 48 52 62 64 65 67 73 74
11/04/2017,02 16 21 24 27 38 42 47 48 53 54 55 56 57 60 62 63 70 73 74
11/03/2017,02 06 09 11 15 24 25 26 30 31 37 38 41 45 48 49 50 54 55 62
11/02/2017,09 14 16 17 20 22 24 33 38 42 43 47 52 54 57 63 67 68 69 73
11/01/2017,01 02 05 06 14 16 27 40 47 54 57 58 59 60 66 68 72 74 75 80
10/31/2017,02 03 05 11 14 15 17 24 30 31 33 46 50 52 54 59 71 74 75 77
10/30/2017,07 10 17 19 21 25 30 34 35 43 50 51 52 59 60 62 67 69 70 77
10/29/2017,01 08 17 20 28 31 32 38 42 43 45 48 50 54 55 58 62 69 74 75
10/28/2017,02 04 10 14 22 29 30 32 33 34 36 39 40 47 49 55 57 59 73 75
10/27/2017,02 04 10 12 17 21 27 40 41 42 43 44 45 52 56 61 68 70 73 78
10/26/2017,07 09 13 16 21 22 23 24 29 30 33 38 39 42 47 52 66 68 71 73
10/25/2017,02 13 15 18 21 28 36 38 43 45 49 55 57 58 61 64 66 71 72 80
10/24/2017,02 03 04 06 14 16 20 22 25 30 40 42 47 50 52 55 63 70 72 80
10/23/2017,04 05 09 15 17 18 21 24 27 34 44 48 60 61 65 66 69 70 75 78
10/22/2017,04 08 09 12 13 16 24 25 26 33 34 38 47 49 67 68 72 78 79 80
10/21/2017,01 05 12 16 18 20 21 22 26 31 33 34 47 49 59 60 65 67 73 80
10/20/2017,01 03 06 10 12 20 21 23 24 25 28 30 36 39 42 44 52 62 68 72
10/19/2017,01 02 05 10 11 19 20 28 34 41 50 53 57 58 59 71 73 78 79 80
10/18/2017,13 14 15 19 20 21 23 24 25 35 41 44 49 50 54 57 69 76 78 80
10/17/2017,07 11 16 18 19 22 28 32 41 44 47 48 52 55 64 65 70 76 77 79
10/16/2017,03 04 14 15 17 18 25 26 30 35 37 43 45 50 53 63 65 66 67 72
10/15/2017,06 07 13 20 25 27 29 34 38 39 42 46 49 53 54 56 64 70 75 80
10/14/2017,01 02 03 06 09 11 16 17 19 25 31 32 35 39 45 52 56 68 72 73
10/13/2017,04 10 11 13 24 37 39 41 42 43 44 53 54 56 60 67 68 71 72 74
10/12/2017,10 11 16 17 25 27 30 34 37 38 40 43 45 46 50 61 65 67 68 73
10/11/2017,01 04 05 06 08 14 15 25 31 33 35 49 59 61 63 65 67 71 72 78
10/10/2017,03 05 17 22 24 30 31 32 35 36 43 45 46 50 51 61 62 65 69 73
10/09/2017,04 05 10 12 15 25 27 28 34 39 46 56 58 66 71 73 74 77 78 79
10/08/2017,01 06 09 10 20 21 28 32 33 34 47 49 52 54 55 58 66 70 71 77
10/07/2017,01 08 10 13 17 19 20 22 27 28 32 34 35 44 45 57 70 71 78 80
10/06/2017,03 04 11 12 13 15 17 19 25 27 29 39 40 41 46 56 65 70 71 77
10/05/2017,07 08 12 13 24 35 37 45 47 49 51 56 57 59 61 65 76 78 79 80
10/04/2017,03 04 06 07 17 20 27 29 34 35 38 48 54 60 62 64 65 68 71 72
10/03/2017,06 20 23 25 26 27 28 31 32 36 46 48 50 52 53 55 60 66 67 77
10/02/2017,01 02 05 08 10 16 21 23 25 28 33 38 40 46 50 51 54 65 72 77
10/01/2017,06 11 16 18 19 21 22 23 28 36 40 41 46 57 60 63 64 73 74 79
09/30/2017,07 09 13 19 21 29 32 33 39 40 41 48 50 52 56 57 58 62 75 77
09/29/2017,01 02 07 08 13 15 22 32 37 43 44 45 48 54 62 70 71 73 74 78
09/28/2017,04 08 11 13 17 23 24 27 29 33 36 46 50 54 57 60 62 63 75 78
09/27/2017,02 03 08 10 17 20 26 29 33 38 43 44 45 48 56 57 67 69 79 80
09/26/2017,03 06 07 24 29 31 33 36 37 44 46 48 49 51 58 60 61 67 70 77
09/25/2017,01 03 08 22 24 25 27 29 37 39 45 50 56 59 63 65 69 75 77 79
09/24/2017,04 07 10 11 14 21 23 25 28 31 35 36 40 59 62 66 68 75 76 77
09/23/2017,03 05 06 11 12 13 14 18 28 33 44 46 51 53 54 65 68 74 77 79
09/22/2017,14 15 21 26 29 30 35 37 44 45 61 62 64 67 68 69 70 72 76 79
09/21/2017,04 06 07 08 14 17 19 20 31 42 46 51 54 56 57 58 60 61 64 65
09/20/2017,02 03 12 17 18 19 24 28 31 39 40 43 44 46 70 71 73 75 76 80
09/19/2017,06 11 13 14 17 21 24 38 40 45 49 54 57 59 64 66 68 69 73 75
09/18/2017,01 04 05 07 10 15 16 19 25 36 37 50 51 54 61 63 69 71 72 79
09/17/2017,10 13 17 21 30 31 34 48 51 59 60 63 64 65 67 68 69 71 74 79
```

```
09/16/2017,06 18 19 23 24 26 32 34 35 38 43 46 47 48 58 62 65 71 72 76
09/15/2017,08 09 12 14 16 24 29 33 41 45 47 51 52 58 59 62 64 68 78 80
09/14/2017,04 07 08 14 16 24 26 27 29 33 35 47 48 51 52 55 56 62 67 75
09/13/2017,06 11 14 19 24 26 29 34 43 50 52 54 58 59 63 69 71 72 78 80
09/12/2017,08 12 15 18 22 25 33 35 43 46 47 51 54 55 56 57 67 70 76 79
09/11/2017,03 09 12 14 15 21 24 27 30 31 32 36 38 39 44 48 50 60 64 78
09/10/2017,01 02 04 10 15 20 26 27 31 44 50 54 55 62 65 66 67 69 70 74
09/09/2017,03 04 05 06 09 11 17 29 31 37 49 52 55 60 61 67 73 74 75 77
09/08/2017,05 08 09 14 15 23 27 29 30 35 36 40 46 50 59 60 76 78 79 80
09/07/2017,01 03 04 21 23 25 26 31 32 33 34 36 42 49 53 56 65 67 72 73
09/06/2017,15 21 22 29 31 33 38 41 44 50 51 52 56 59 62 63 70 74 76 77
09/05/2017,03 08 13 15 23 27 29 37 40 46 47 49 50 53 60 62 67 69 78 79
09/04/2017,04 19 20 23 27 28 29 38 42 47 51 53 55 59 66 67 70 74 78 79
09/03/2017,03 08 17 18 20 26 27 33 37 38 46 49 52 58 69 73 74 77 78 80
09/02/2017,07 10 15 21 22 24 32 34 40 41 43 45 63 68 69 71 73 76 77 78
09/01/2017,01 12 13 16 29 31 32 37 38 39 44 45 47 50 52 58 60 63 73 79
08/31/2017,03 05 06 08 09 21 24 25 34 39 44 52 61 65 68 75 77 78 79 80
08/30/2017,02 03 08 14 16 17 22 26 34 36 42 43 44 49 53 55 58 63 73 80
08/29/2017,03 04 06 08 09 11 12 16 21 24 26 28 33 34 37 38 50 65 69 75
08/28/2017,06 12 22 24 25 33 36 38 39 43 50 51 53 54 56 57 61 72 76 80
08/27/2017,01 02 05 11 16 19 20 24 27 29 30 37 43 50 53 61 62 69 74 80
08/26/2017,02 04 06 16 22 26 27 30 31 32 40 42 44 51 53 61 65 72 73 74
08/25/2017,05 12 13 14 19 20 23 24 30 34 41 44 45 49 51 60 61 72 76 80
08/24/2017,04 08 10 13 17 19 23 33 35 38 40 47 49 50 61 69 70 72 76 80
08/23/2017,05 14 22 27 29 38 40 46 47 49 50 61 64 65 69 72 74 75 76 78
08/22/2017,07 08 10 16 20 25 29 38 39 40 41 43 50 56 63 68 70 76 78 80
08/21/2017,01 02 07 08 10 13 14 16 27 30 32 33 43 48 52 54 56 63 66 68
08/20/2017,03 10 12 14 21 23 29 30 31 39 48 49 54 57 58 62 64 66 67 72
08/19/2017,04 05 06 10 11 18 19 20 31 37 41 47 48 58 63 65 67 69 78 79
08/18/2017,03 05 08 10 11 24 34 40 48 51 53 54 60 61 62 63 64 70 76 80
08/17/2017,12 14 17 19 25 27 28 34 36 42 46 49 51 55 59 69 72 74 78 80
08/16/2017,08 10 20 23 29 31 32 35 41 46 49 55 57 61 66 67 69 70 73 79
08/15/2017,01 03 06 09 12 24 25 27 28 32 36 38 40 42 48 51 55 56 68 75
08/14/2017,05 08 14 20 27 29 33 41 43 44 45 46 48 50 53 58 64 70 73 74
08/13/2017,04 07 08 12 19 23 32 36 37 40 45 53 54 63 64 66 68 69 70 74
08/12/2017,16 17 19 23 24 30 39 42 44 45 46 48 52 54 58 60 66 71 74 76
08/11/2017,06 07 14 22 27 29 31 41 42 45 49 52 55 58 61 63 66 71 75 80
08/10/2017,01 10 14 16 21 23 24 28 29 32 33 35 41 43 44 45 46 59 64 67
08/09/2017,11 15 16 17 22 25 31 34 44 48 51 53 57 58 61 65 67 72 77 80
08/08/2017,05 06 15 18 19 24 26 28 39 44 46 47 48 53 56 57 59 75 76 79
08/07/2017,01 16 21 22 28 32 34 37 46 48 50 51 53 56 57 59 64 67 79 80
08/06/2017,01 07 08 11 15 17 19 22 26 29 30 39 40 45 46 57 58 63 69 77
08/05/2017,03 09 12 24 27 28 29 31 36 42 47 56 57 63 64 65 67 69 71 72
08/04/2017,07 10 12 26 28 30 40 43 44 48 51 52 58 59 65 67 72 73 75 80
08/03/2017,02 03 04 07 09 10 18 29 30 37 40 42 47 58 59 60 65 68 69 80
08/02/2017,01 07 09 15 18 24 28 31 35 40 42 46 50 57 58 69 72 75 78 79
08/01/2017,10 14 17 21 26 28 31 32 37 40 41 43 46 50 53 63 66 69 72 79
07/31/2017,05 10 11 14 16 18 25 30 31 32 40 43 47 54 57 58 60 63 70 74
07/30/2017,08 09 13 16 17 27 29 33 42 43 48 49 52 55 59 65 66 68 69 75
07/29/2017,03 05 08 12 16 19 31 32 43 47 48 50 52 53 61 62 63 65 67 76
07/28/2017,06 10 11 12 18 19 24 34 43 52 53 57 59 61 62 65 71 73 78 79
07/27/2017,03 08 10 12 15 17 22 24 25 39 41 43 44 46 59 63 68 71 72 73
07/26/2017,03 12 13 14 15 21 23 25 33 37 38 41 42 46 47 58 60 63 67 68
07/25/2017,01 09 11 17 22 24 25 28 37 47 54 55 56 59 62 63 64 65 68 74
```

```
65      07/24/2017,01 06 08 10 18 25 29 32 38 40 44 50 51 57 59 69 72 77 79 80
66      07/23/2017,01 04 09 12 25 32 39 40 41 43 47 48 52 53 57 58 61 66 67 68
67      07/22/2017,01 03 23 25 30 38 42 43 44 45 46 47 49 50 56 63 68 69 71 76
68      07/21/2017,01 03 07 13 15 22 30 38 39 44 55 61 65 66 68 70 75 78 79 80
69      07/20/2017,05 06 11 17 24 30 33 36 37 41 49 50 51 55 61 63 67 68 71 75
70      07/19/2017,01 02 08 09 11 13 15 16 19 21 28 35 42 43 45 47 49 61 62 73
71      07/18/2017,05 07 10 13 15 20 30 33 35 44 50 56 65 68 70 73 74 77 78 79
72      07/17/2017,05 06 23 27 29 42 44 51 52 59 60 61 63 66 68 69 71 73 79 80
73      07/16/2017,01 02 11 13 16 17 19 26 29 32 33 41 45 46 53 61 64 65 68 74
74      07/15/2017,02 03 12 13 19 24 29 34 35 41 44 45 46 54 57 60 61 68 79 80
75      07/14/2017,05 10 14 15 16 21 34 40 41 43 54 56 65 68 70 74 77 78 79 80
76      07/13/2017,01 05 08 09 11 12 15 25 28 34 37 38 40 47 49 51 53 56 66 74
77      07/12/2017,02 09 10 13 20 42 44 50 51 52 53 56 60 61 62 67 68 73 74 78
78      07/11/2017,05 10 13 25 27 29 30 43 44 46 47 52 55 56 58 63 67 72 74 76
79      07/10/2017,01 02 05 14 18 19 20 24 27 30 32 39 41 43 52 54 58 59 65 69
80      07/09/2017,06 07 11 14 15 17 21 32 37 38 42 46 48 50 53 65 66 68 78 79
81      07/08/2017,07 09 13 15 19 20 29 34 37 40 41 45 49 56 57 62 64 75 77 78
82      07/07/2017,05 09 10 17 18 22 28 29 31 32 33 42 51 53 55 56 58 64 70 79
83      07/06/2017,02 06 07 16 18 20 21 22 26 33 37 41 52 53 56 60 65 77 78 80
84      07/05/2017,10 11 14 15 19 24 29 33 38 44 46 50 52 55 58 62 64 70 71 78
85      07/04/2017,06 07 11 12 14 18 27 30 36 39 44 45 48 51 54 57 58 61 69 77
86      07/03/2017,05 07 12 15 19 25 27 28 36 38 41 42 43 49 51 53 55 59 62 73
87      07/02/2017,01 04 05 10 14 15 20 25 29 30 40 43 47 57 60 62 64 65 76 80
88      07/01/2017,01 03 04 14 16 22 24 29 33 42 45 58 59 63 66 69 75 77 78 80
89      06/30/2017,05 06 12 14 15 25 26 28 35 40 44 47 49 50 60 65 66 68 69 71
90      06/29/2017,01 02 10 13 17 21 22 23 30 32 35 38 53 63 64 65 67 68 69 79
91      06/28/2017,02 08 12 14 18 23 36 41 52 53 61 64 66 68 72 73 74 76 78 79
92      06/27/2017,12 15 18 19 21 22 24 27 29 41 44 47 48 51 55 58 60 63 78 80
93      06/26/2017,06 15 18 26 27 30 34 35 36 38 39 42 47 54 61 63 65 66 75 77
94      06/25/2017,02 05 07 10 11 13 15 27 30 35 38 42 46 54 59 60 63 67 73 76
95      06/24/2017,01 07 08 12 13 17 23 26 28 31 37 44 49 51 60 64 72 75 78 79
96      06/23/2017,10 15 16 26 27 29 30 31 40 50 57 64 65 66 71 72 74 75 77 78
97      06/22/2017,03 12 22 23 28 31 33 42 46 51 54 56 60 62 63 65 71 75 76 79
98      06/21/2017,03 09 14 15 16 23 24 34 37 43 44 47 51 54 59 62 65 68 76 77
99      06/20/2017,01 10 14 15 19 20 22 24 25 27 34 35 36 38 44 45 47 52 63 80
00      06/19/2017,01 12 13 25 26 27 31 41 42 44 45 49 53 61 64 65 67 68 72 74
01      06/18/2017,03 04 17 20 22 23 34 36 38 39 45 47 52 55 59 63 66 67 71 73
02      06/17/2017,04 09 10 14 21 22 24 25 31 38 39 43 47 59 60 63 64 65 70 71
03      06/16/2017,04 05 07 14 18 23 26 29 35 38 40 43 46 51 54 58 66 69 74 78
04      06/15/2017,08 11 16 17 21 23 27 28 35 38 42 43 44 45 46 49 50 55 74 80
05      06/14/2017,04 09 11 14 18 29 31 36 41 42 44 48 49 52 53 55 56 72 77 79
06      06/13/2017,06 10 16 20 23 29 32 41 44 45 52 53 55 60 62 63 74 75 76 80
07      06/12/2017,08 16 18 23 30 33 36 37 40 41 42 43 55 56 68 73 74 75 76 80
08      06/11/2017,02 07 09 10 11 13 20 25 27 28 39 40 51 53 58 59 64 69 79 80
09      06/09/2017,02 05 18 28 33 34 36 39 47 52 54 57 60 66 67 70 71 74 78 79
10      06/08/2017,02 03 04 07 11 16 21 29 30 32 38 42 45 47 50 58 59 62 66 73
11      06/07/2017,09 12 14 16 20 23 25 44 46 47 48 51 52 55 56 62 63 66 70 75
12      06/06/2017,06 07 09 12 15 16 18 24 28 35 36 46 47 48 49 51 64 66 72 80
13      06/05/2017,06 10 11 13 19 21 27 37 38 41 46 47 55 57 61 66 72 73 75 77
14      06/04/2017,02 04 09 18 23 25 27 30 41 43 44 48 50 54 56 57 60 66 68 79
15      06/03/2017,03 04 06 08 12 13 14 17 20 22 28 29 43 44 49 50 58 65 72 80
16      06/02/2017,03 05 14 18 22 23 24 27 32 34 39 44 48 51 54 60 63 72 77 79
17      06/01/2017,01 10 13 14 18 24 25 26 28 33 34 36 39 48 50 51 53 60 67 72
18      05/31/2017,06 11 21 22 25 27 28 37 39 40 44 50 55 56 66 67 74 76 77 80
```

```
19      05/30/2017,03 04 06 18 24 31 32 47 50 51 59 60 63 68 69 72 77 78 79 80
20      05/29/2017,06 07 10 14 26 30 36 37 43 48 49 54 58 63 64 65 72 78 79 80
21      05/28/2017,02 07 08 14 19 22 23 27 29 30 31 35 38 47 53 55 58 68 73 78
22      05/27/2017,01 12 16 18 22 30 41 43 46 52 58 62 65 66 67 68 72 75 77 79
23      05/26/2017,04 17 21 22 23 26 27 28 29 30 31 41 46 47 50 53 55 59 67 80
24      05/25/2017,03 05 12 16 18 19 27 28 32 35 38 44 45 51 56 62 64 76 79 80
25      05/24/2017,01 07 13 20 25 27 29 34 41 45 55 57 61 64 66 69 71 75 77 78
26      05/23/2017,02 10 15 16 28 30 33 35 43 50 51 52 53 55 61 64 65 66 68 72
27      05/22/2017,03 15 17 19 20 24 27 31 39 51 53 62 64 70 72 74 75 76 77 79
28      05/21/2017,05 18 19 20 31 33 38 42 44 45 46 49 54 62 63 65 67 68 74 76
29      05/20/2017,01 02 03 04 09 25 27 29 34 35 38 39 43 49 55 56 67 73 77 78
30      05/19/2017,01 08 13 15 30 34 35 36 39 40 59 63 64 66 69 70 74 75 78 80
31      05/18/2017,03 05 09 13 14 15 23 28 29 32 34 35 39 43 55 57 62 68 78 80
32      05/17/2017,02 05 06 14 15 18 21 24 25 27 31 35 37 41 48 49 65 69 71 76
33      05/16/2017,02 14 16 18 21 24 39 42 47 52 54 59 60 62 65 67 68 69 73 79
34      05/15/2017,05 13 14 16 18 23 25 28 34 35 37 39 41 43 47 51 53 59 67 72
35      05/14/2017,08 12 23 26 34 37 41 42 45 50 52 57 58 59 70 71 76 78 79 80
36      05/13/2017,13 16 23 25 26 33 35 43 46 48 49 50 52 58 61 62 64 65 67 72
37      05/12/2017,01 06 08 10 12 20 23 26 35 41 42 43 48 50 56 60 64 72 74 79
38      05/11/2017,01 02 07 09 11 14 19 28 30 38 40 50 51 52 53 59 61 63 67 76
39      05/10/2017,04 06 10 17 20 21 23 26 29 32 33 39 43 55 62 71 72 74 75 77
40      05/09/2017,04 06 10 12 16 17 33 48 49 51 52 60 65 66 70 71 73 74 76 79
41      05/08/2017,04 07 08 09 10 11 13 20 21 22 23 25 33 35 37 44 45 53 60 79
42      05/07/2017,10 15 21 22 23 25 26 30 31 32 35 40 41 42 47 55 70 73 75 78
43      05/06/2017,01 03 04 06 09 10 16 43 44 45 48 49 51 56 59 63 69 71 73 74
44      05/05/2017,02 16 18 19 22 31 35 39 40 44 46 48 49 52 54 59 65 67 75 79
45      05/04/2017,02 03 09 14 23 27 32 38 39 43 52 53 58 63 66 68 70 73 74 80
46      05/03/2017,06 16 17 19 21 22 24 25 31 34 35 39 51 53 62 63 65 71 74 79
47      05/02/2017,02 05 11 15 22 25 28 29 31 33 40 48 55 59 60 67 68 71 74 78
48      05/01/2017,06 09 12 13 16 18 20 23 34 40 41 42 51 57 59 65 72 73 78 80
49      04/30/2017,01 04 08 12 13 14 17 24 31 35 37 41 46 52 58 59 62 72 74 77
50      04/29/2017,03 04 18 20 22 33 34 37 48 49 51 52 53 54 58 59 60 64 67 78
51      04/28/2017,06 08 09 10 14 17 20 21 25 27 36 44 45 46 48 50 52 65 67 72
52      04/27/2017,05 13 16 21 28 33 34 37 38 44 45 47 48 49 50 58 64 66 67 74
53      04/26/2017,01 03 06 11 19 21 26 28 32 33 44 57 59 62 63 65 66 70 77 79
54      04/25/2017,07 12 14 16 17 22 26 33 34 37 49 51 54 56 57 62 64 67 71 73
55      04/24/2017,08 11 12 13 16 24 26 29 32 36 40 45 47 48 54 56 59 69 74 75
56      04/23/2017,01 02 04 07 12 23 24 30 35 36 37 39 44 47 51 53 56 63 68 73
57      04/22/2017,04 11 15 18 21 23 27 30 34 38 39 40 46 52 53 57 69 75 77 78
58      04/21/2017,04 07 11 13 20 31 32 33 39 41 45 48 49 55 58 60 63 65 74 78
59      04/20/2017,02 04 12 16 23 27 28 30 32 39 44 45 47 48 63 64 66 69 74 76
60      04/19/2017,12 15 22 23 28 32 36 41 49 50 52 54 55 56 61 66 69 71 74 79
61      04/18/2017,04 06 14 17 28 32 34 35 43 47 50 53 61 63 67 71 73 75 76 80
62      04/17/2017,01 07 15 22 28 29 30 33 42 46 47 51 59 60 62 63 64 67 68 79
63      04/16/2017,01 03 23 24 28 29 30 32 34 37 46 47 56 58 59 62 63 64 73 78
64      04/15/2017,02 06 08 09 14 18 31 32 43 44 46 47 55 56 60 61 62 65 66 71
65      04/14/2017,01 15 16 24 25 28 40 45 51 53 54 55 58 59 60 63 67 71 73 78
66      04/13/2017,13 20 21 22 26 27 29 32 33 40 43 48 53 58 63 64 65 68 78 79
67      04/12/2017,03 05 06 08 14 16 22 27 35 38 41 42 46 47 48 55 61 65 76 80
68      04/11/2017,02 20 22 27 28 37 44 49 54 58 67 69 72 73 75 76 77 78 79 80
69      04/10/2017,04 05 06 07 13 21 32 34 38 46 50 52 57 62 64 67 70 71 74 77
70      04/09/2017,01 03 04 14 15 27 28 30 31 38 41 45 49 55 64 65 67 68 70 79
71      04/08/2017,05 12 14 15 17 19 26 28 30 33 35 36 38 46 51 56 67 70 77 80
72      04/07/2017,01 10 11 12 14 26 29 31 32 35 40 43 53 55 62 64 69 76 78 80
```

```
04/06/2017,05 08 10 16 20 21 23 24 25 26 27 31 34 36 40 42 44 52 56 78
04/05/2017,02 06 08 10 18 22 24 27 30 34 44 47 54 57 61 65 71 74 79 80
04/04/2017,06 10 11 12 17 19 21 26 35 36 47 50 51 54 58 60 68 70 71 75
04/03/2017,02 04 05 09 10 15 22 24 33 38 39 45 56 57 58 61 64 68 72 75
04/02/2017,07 08 09 10 12 14 16 24 36 41 44 52 54 55 58 59 61 63 67 80
04/01/2017,05 10 11 12 17 21 25 26 27 28 35 37 42 49 53 64 65 71 72 74
03/31/2017,02 03 07 11 16 24 26 27 29 38 39 50 51 58 61 65 69 74 75 78
03/30/2017,02 05 12 15 18 22 24 26 47 48 52 55 57 60 63 65 66 68 69 79
03/29/2017,04 13 16 18 20 21 22 26 28 33 34 36 41 44 59 60 63 74 78 79
03/28/2017,04 05 09 11 17 18 20 25 27 30 40 46 47 52 54 60 64 69 74 77
03/27/2017,02 04 07 12 15 17 18 19 21 22 25 43 48 51 62 66 69 73 76 79
03/26/2017,05 06 07 10 13 16 17 22 23 35 55 56 59 64 65 67 68 74 75 78
03/25/2017,04 09 12 13 15 21 26 28 37 38 42 48 49 50 56 59 62 64 69 78
03/24/2017,05 10 18 19 24 30 31 32 35 36 41 43 47 50 56 64 65 67 69 78
03/23/2017,01 04 06 14 16 18 26 27 29 31 33 41 43 50 60 63 67 69 73 75
03/22/2017,07 08 10 11 12 13 15 17 26 34 36 40 48 64 67 68 71 72 73 74
03/21/2017,09 14 15 16 21 22 24 31 32 43 54 59 60 62 64 72 74 75 77 78
03/20/2017,06 07 09 10 18 20 23 27 30 31 37 39 40 44 46 51 52 62 67 75
03/19/2017,03 04 06 07 12 13 29 31 35 43 50 54 57 59 66 67 68 75 79 80
03/18/2017,01 03 10 11 18 20 24 27 29 36 47 48 49 52 55 58 60 62 64 71
03/17/2017,04 05 11 12 13 25 30 35 40 47 48 55 56 57 58 62 64 70 73 79
03/16/2017,01 02 04 09 15 19 20 24 25 26 31 44 45 49 55 58 65 66 67 80
03/15/2017,04 14 18 21 23 24 25 27 29 33 35 44 50 55 63 64 65 66 73 77
03/14/2017,02 09 11 12 21 23 29 36 39 40 42 44 45 47 50 52 56 62 68 77
03/13/2017,04 19 21 22 23 27 30 33 34 36 43 51 54 61 65 70 74 76 78 80
03/12/2017,04 06 08 10 19 20 22 32 35 41 42 46 50 51 59 68 70 72 74 77
03/11/2017,13 20 26 29 30 33 34 38 39 45 46 51 52 55 57 60 61 64 67 80
03/10/2017,03 07 15 16 23 26 28 30 33 35 37 38 40 45 64 65 68 69 73 74
03/09/2017,11 13 23 24 25 29 32 35 36 41 42 43 48 54 56 60 66 75 76 77
03/08/2017,05 07 09 11 13 14 20 36 39 52 53 55 56 59 60 61 64 71 76 80
03/07/2017,01 02 03 04 09 14 18 19 20 21 24 25 28 47 48 59 65 68 72 76
03/06/2017,03 11 13 14 15 19 21 22 29 36 38 39 41 42 51 52 53 58 67 77
03/05/2017,02 03 04 08 11 12 18 22 37 39 42 44 48 49 57 58 63 66 72 77
03/04/2017,05 06 08 12 16 29 30 35 41 43 45 46 56 59 67 70 71 73 76 80
03/03/2017,04 06 08 09 12 16 23 30 31 33 40 41 42 50 51 55 56 59 61 72
03/02/2017,04 06 12 14 18 24 27 44 46 47 48 52 56 69 71 74 75 76 78 80
03/01/2017,01 02 03 17 25 27 28 31 39 41 42 51 54 61 63 65 67 72 75 79
02/28/2017,03 07 11 16 17 20 21 22 24 26 27 45 47 49 57 65 66 76 77 78
02/27/2017,01 03 08 09 14 23 24 30 33 37 42 43 45 48 51 68 74 77 78 79
02/26/2017,04 17 18 19 21 22 33 35 37 39 41 42 43 46 47 64 66 72 73 76
02/25/2017,03 07 13 15 18 27 38 42 43 47 49 51 52 59 60 61 66 67 73 76
02/24/2017,02 03 19 23 24 29 36 37 38 41 43 47 48 56 60 62 63 64 74 75
02/23/2017,05 07 10 15 20 22 31 32 33 42 45 46 49 50 52 54 56 57 69 71
02/22/2017,02 07 08 09 13 18 24 25 26 27 28 29 32 36 42 44 50 57 63 73
02/21/2017,03 14 17 21 23 26 32 33 35 41 54 56 58 59 61 64 65 70 72 80
02/20/2017,01 03 05 15 22 26 28 29 34 35 37 38 41 42 45 53 57 72 74 79
02/19/2017,04 11 13 15 17 21 22 31 37 38 41 52 60 61 66 67 71 74 75 76
02/18/2017,01 13 16 17 22 24 26 27 28 30 37 40 42 45 47 52 56 57 70 73
02/17/2017,03 05 08 19 24 26 27 29 37 38 46 50 57 60 61 64 65 74 78 80
02/16/2017,01 02 03 04 09 12 20 35 39 41 45 47 54 64 71 72 73 74 77 79
02/15/2017,01 06 09 11 13 15 16 19 21 22 33 38 41 44 63 68 69 71 73 74
02/14/2017,02 03 04 09 10 14 19 21 22 23 33 46 50 54 61 62 69 70 72 80
02/13/2017,10 12 15 17 32 42 43 45 48 52 54 57 60 64 72 74 76 77 78 80
02/12/2017,08 09 11 13 20 22 30 41 45 46 49 52 57 58 62 67 71 73 74 76
```

```
02/11/2017,01 05 09 13 15 23 29 33 34 40 44 46 50 52 57 61 62 68 72 78
02/10/2017,01 05 07 11 17 23 26 27 35 38 42 45 55 62 63 65 67 69 70 73
02/09/2017,10 15 18 21 26 27 28 29 31 35 37 39 43 46 51 53 63 66 69 73
02/08/2017,03 07 14 21 28 30 32 39 43 50 54 55 58 62 64 65 71 74 76 78
02/07/2017,01 11 20 21 25 26 27 30 32 33 36 37 44 52 53 54 61 62 63 80
02/06/2017,07 10 11 13 15 21 23 25 26 28 32 33 45 49 52 61 62 63 73 79
02/05/2017,03 04 05 07 08 13 16 23 30 35 39 53 59 60 63 67 69 77 79 80
02/04/2017,02 10 11 12 14 23 24 29 30 31 33 47 49 50 58 67 68 69 72 76
02/03/2017,04 06 10 14 15 21 23 27 28 31 35 40 43 47 52 57 61 64 69 70
02/02/2017,01 03 04 08 13 19 25 35 39 45 49 57 62 63 64 71 72 74 78 79
02/01/2017,01 03 06 08 09 11 20 24 26 27 33 38 42 45 48 52 69 72 75 78
01/31/2017,11 19 23 25 31 32 33 39 47 52 54 56 61 64 68 71 73 75 79 80
01/30/2017,04 06 07 08 10 11 12 19 22 23 24 35 42 47 50 55 62 66 68 74
01/29/2017,02 08 11 15 16 19 22 28 34 48 50 51 53 56 59 61 62 65 68 74
01/28/2017,04 15 17 20 21 23 27 40 43 44 48 50 61 62 67 68 72 75 79 80
01/27/2017,06 07 10 12 20 22 25 30 33 34 36 41 44 51 57 61 73 77 79 80
01/26/2017,03 08 10 13 19 23 28 30 38 41 49 51 53 55 57 61 67 77 79 80
01/25/2017,03 06 12 13 14 16 17 22 23 26 38 39 40 42 60 62 69 74 75 80
01/24/2017,01 02 05 07 09 15 16 21 26 27 33 34 37 47 50 56 62 73 76 79
01/23/2017,01 13 19 21 23 28 35 36 38 41 49 53 54 57 60 64 69 77 78 79
01/22/2017,03 04 06 07 09 11 17 20 21 23 25 28 31 32 34 45 60 64 72 80
01/20/2017,01 08 09 11 17 18 20 22 24 25 26 28 32 33 34 53 59 65 72 80
01/19/2017,06 07 12 16 27 29 31 38 40 41 46 49 54 55 56 57 60 65 69 73
01/18/2017,01 04 06 09 20 22 24 29 31 41 45 50 53 57 60 66 72 78 79 80
01/17/2017,05 07 09 12 16 17 18 20 21 28 37 39 45 49 53 59 60 61 72 73
01/16/2017,06 07 11 12 15 20 22 27 28 30 32 41 49 57 62 64 69 71 73 78
01/15/2017,07 16 22 24 26 28 30 31 32 34 37 39 52 57 60 64 70 72 77 79
01/14/2017,01 04 09 10 13 14 18 21 24 25 30 32 38 39 57 61 68 74 75 78
01/13/2017,03 07 08 09 10 12 17 20 26 35 37 41 48 60 63 64 69 72 74 76
01/12/2017,02 04 09 10 11 14 16 21 25 37 39 45 53 58 61 63 68 74 75 76
01/11/2017,02 03 05 08 09 17 21 24 29 42 47 48 54 56 58 64 69 75 78 79
01/10/2017,01 06 09 16 20 21 23 28 30 33 35 39 44 48 51 54 56 58 72 75
01/09/2017,05 07 09 10 14 27 32 34 39 42 43 44 45 46 47 57 59 68 69 74
01/08/2017,02 08 09 19 23 29 30 32 42 44 47 50 51 53 54 55 60 65 68 72
01/07/2017,02 04 05 06 13 16 23 28 30 31 33 41 45 46 49 60 69 71 73 76
01/06/2017,01 03 09 14 28 30 35 36 37 41 44 47 48 55 58 60 62 67 69 80
01/05/2017,04 11 13 20 24 34 36 42 47 53 54 55 56 58 65 66 67 71 77 80
01/04/2017,06 10 13 17 21 23 26 28 30 38 44 51 56 57 58 60 64 70 72 79
01/03/2017,02 09 12 16 21 27 38 39 40 44 46 53 55 56 57 58 59 65 71 74
01/02/2017,20 22 23 27 31 35 37 40 45 47 48 49 51 66 69 71 73 75 76 80
01/01/2017,01 08 11 13 15 17 20 23 24 27 30 31 35 36 45 48 52 67 69 72
12/31/2016,09 10 12 18 27 31 34 37 39 47 53 55 62 64 66 68 70 72 74 76
12/30/2016,02 06 08 13 18 20 22 24 29 37 46 48 50 54 55 58 62 73 74 75
12/29/2016,01 08 13 15 17 20 24 26 33 34 42 47 48 54 55 56 60 64 78 80
12/28/2016,02 15 20 21 29 32 33 34 37 39 41 50 52 53 54 56 63 66 68 77
12/27/2016,09 12 13 15 16 19 22 26 29 35 37 39 42 49 55 58 59 60 66 71
12/26/2016,04 11 12 13 20 23 31 35 41 44 48 51 55 58 67 70 73 74 75 80
12/25/2016,12 14 21 22 29 33 37 39 40 49 50 54 56 66 67 69 74 75 77 80
12/24/2016,02 06 09 13 26 27 28 30 32 33 36 37 39 41 48 58 59 61 63 73
12/23/2016,03 06 08 17 19 25 28 30 31 35 46 49 65 67 68 69 72 73 74 75
12/22/2016,09 14 17 18 20 22 25 27 32 43 44 46 48 52 55 56 63 64 77 80
12/21/2016,01 06 09 19 20 21 27 28 44 46 53 54 65 68 69 70 71 75 76 77
12/20/2016,05 06 07 18 24 29 32 35 37 40 45 50 52 57 60 70 73 74 75 79
12/19/2016,01 03 04 10 18 25 29 30 32 33 35 42 43 44 55 56 59 69 70 71
```

```
81      12/18/2016,01 04 16 22 23 30 34 36 40 41 51 54 58 60 61 64 66 73 76 77
82      12/17/2016,10 12 15 17 20 24 27 30 32 34 38 41 45 46 50 51 52 54 68 77
83      12/16/2016,03 07 11 12 13 14 15 18 33 37 41 45 53 55 57 65 66 69 70 72
84      12/15/2016,01 04 05 07 09 11 14 23 27 28 29 31 32 34 43 44 53 63 71 80
85      12/14/2016,02 03 04 07 14 16 23 24 27 28 32 46 47 52 56 57 62 64 71 75
86      12/13/2016,07 12 14 16 19 27 28 30 37 44 50 52 54 55 56 57 58 67 76 78
87      12/12/2016,03 06 07 09 16 21 23 29 41 43 44 46 50 51 54 59 66 71 73 79
88      12/11/2016,04 06 13 14 19 23 27 28 29 31 49 51 52 61 63 66 69 75 79 80
89      12/10/2016,01 04 12 14 15 17 27 28 30 40 45 49 51 60 61 62 71 73 78 79
90      12/09/2016,01 10 14 15 18 21 30 31 34 37 39 44 46 50 52 54 55 64 72 76
91      12/08/2016,03 05 14 19 25 26 27 28 29 31 32 36 42 45 60 61 63 68 74 76
92      12/07/2016,01 03 13 23 31 35 38 43 47 48 52 60 64 65 68 72 77 78 79 80
93      12/06/2016,10 16 19 20 21 22 24 29 31 41 44 59 60 63 64 66 68 69 71 77
94      12/05/2016,03 07 08 09 21 25 26 38 46 47 53 60 64 66 69 70 74 75 78 80
95      12/04/2016,02 05 07 14 15 16 22 28 34 45 47 51 56 65 69 70 71 72 74 76
96      12/03/2016,03 05 06 13 17 40 41 42 43 51 52 53 58 63 67 68 71 75 76 79
97      12/02/2016,03 07 10 12 14 15 25 35 36 40 41 55 58 59 61 62 66 70 74 75
98      12/01/2016,07 13 19 23 30 34 35 38 46 49 51 54 58 61 62 67 68 69 76 77
99      11/30/2016,04 05 16 20 22 24 30 32 42 44 46 47 48 51 57 58 61 63 70 71
00      11/29/2016,02 03 05 08 09 17 21 22 23 24 27 29 34 42 47 55 66 67 70 76
01      11/28/2016,03 07 08 12 18 20 22 24 26 31 32 33 34 41 55 59 61 69 73 76
02      11/27/2016,10 11 12 16 18 24 33 36 37 38 39 40 48 60 65 73 75 76 77 78
03      11/26/2016,02 03 04 05 14 20 21 25 26 27 30 50 52 59 62 67 69 73 75 79
04      11/25/2016,09 10 15 19 20 23 27 29 35 40 42 43 46 50 54 57 62 63 74 77
05      11/24/2016,06 07 11 19 31 32 36 43 44 46 47 49 55 59 61 64 66 71 78 79
06      11/23/2016,01 03 09 17 19 24 27 40 41 42 44 50 58 61 62 65 68 69 70 78
07      11/22/2016,06 07 08 13 14 18 25 29 35 36 39 45 46 52 55 56 57 58 60 77
08      11/21/2016,05 08 12 13 17 19 20 26 33 37 40 42 47 50 55 60 62 64 67 72
09      11/20/2016,08 17 18 23 24 27 28 30 32 36 44 45 49 55 56 60 66 70 71 72
10      11/19/2016,01 03 06 07 19 22 24 36 38 42 45 54 55 56 60 63 72 74 78 80
11      11/18/2016,10 16 17 21 22 24 26 28 39 40 43 46 47 51 61 63 73 75 78 80
12      11/17/2016,01 04 05 13 20 21 24 26 28 30 35 40 41 47 55 58 59 72 74 79
13      11/16/2016,01 05 17 25 26 31 34 39 41 42 43 45 50 58 64 67 69 76 78 79
14      11/15/2016,02 04 10 12 14 15 18 25 29 31 34 41 48 49 56 64 69 70 71 74
15      11/14/2016,04 07 09 10 12 15 29 31 37 41 43 45 46 48 53 59 67 68 75 76
16      11/13/2016,02 06 12 13 17 24 25 29 34 36 37 43 46 50 60 64 70 71 75 76
17      11/12/2016,04 06 07 09 15 21 23 24 31 32 36 38 42 46 51 53 64 76 78 79
18      11/11/2016,01 02 06 12 16 19 24 30 38 44 47 52 55 59 61 62 63 64 73 74
19      11/10/2016,05 08 09 11 14 15 18 20 21 23 29 37 40 41 50 59 62 66 67 79
20      11/09/2016,04 05 15 18 19 21 28 32 34 42 43 45 48 49 55 59 70 74 75 76
21      11/08/2016,12 16 17 23 30 32 33 34 36 42 44 46 50 54 56 57 60 70 71 80
22      11/07/2016,01 04 05 07 09 10 19 22 28 29 30 35 42 47 48 55 61 69 70 80
23      11/06/2016,01 03 05 09 13 16 23 27 38 41 43 44 49 56 57 58 66 71 74 77
24      11/05/2016,03 04 05 10 14 18 26 34 38 39 42 44 47 51 54 63 71 76 78 80
25      11/04/2016,03 09 11 13 14 17 20 21 22 26 28 45 46 48 56 60 69 72 74 75
26      11/03/2016,01 07 10 11 12 13 18 25 30 32 34 35 40 41 43 50 54 55 63 68
27      11/02/2016,05 06 12 14 21 22 24 26 28 30 33 43 44 55 62 63 67 71 74 77
28      11/01/2016,08 09 14 15 19 20 25 28 31 36 38 40 41 43 44 55 71 74 75 80
29      10/31/2016,04 11 12 14 16 17 18 34 35 37 40 44 47 49 56 58 63 70 71 77
30      10/30/2016,01 15 21 24 26 28 31 32 34 39 41 45 49 50 53 57 60 62 64 72
31      10/29/2016,07 08 11 17 21 23 24 27 30 38 45 46 49 55 56 60 70 74 76 78
32      10/28/2016,02 04 07 16 25 27 28 32 33 51 53 54 55 57 61 62 67 68 70 75
33      10/27/2016,05 08 10 17 24 28 29 36 40 41 45 52 55 59 62 64 65 72 76 79
34      10/26/2016,03 05 16 18 19 20 21 23 24 32 35 39 40 46 50 57 64 69 72 80
```

```
10/25/2016,04 06 08 18 19 20 22 26 28 35 41 43 48 51 52 61 67 72 75 80
10/24/2016,03 04 10 14 17 19 22 24 29 31 34 37 39 40 47 48 57 66 73 75
10/23/2016,03 05 06 08 10 15 18 19 23 24 34 39 43 45 58 60 61 64 65 79
10/22/2016,01 03 09 10 12 25 38 40 44 45 46 53 56 59 62 63 64 69 73 77
10/21/2016,03 04 07 11 17 20 26 27 28 31 32 36 43 44 46 56 61 70 75 80
10/20/2016,04 06 10 17 18 22 29 34 37 47 48 49 53 65 66 67 70 72 77 79
10/19/2016,02 11 23 28 33 35 37 42 46 50 54 57 62 67 69 70 73 74 76 77
10/18/2016,04 14 15 21 27 29 36 39 46 49 51 54 58 61 63 65 70 74 75 77
10/17/2016,09 10 11 21 22 28 31 32 37 39 42 43 44 47 50 56 59 70 71 75
10/16/2016,02 05 08 29 33 41 44 45 47 50 53 59 60 62 70 72 73 75 76 77
10/15/2016,11 12 18 22 24 27 30 36 40 41 48 49 51 57 58 59 64 65 68 69
10/14/2016,05 08 15 16 31 39 43 46 49 54 55 58 62 64 66 68 69 72 73 79
10/13/2016,01 03 06 12 16 19 24 27 36 37 38 51 52 54 57 63 66 68 73 80
10/12/2016,17 23 24 25 26 32 40 45 49 50 54 58 61 65 68 70 71 73 76 79
10/11/2016,07 09 10 12 15 27 33 34 37 39 45 46 48 49 52 53 55 61 64 77
10/10/2016,02 06 14 16 19 33 39 41 43 44 46 52 55 60 61 63 64 65 72 73
10/09/2016,13 17 18 20 27 31 33 39 41 43 44 46 54 56 62 67 70 71 75 78
10/08/2016,02 04 12 13 16 19 21 28 31 45 55 60 61 63 65 66 69 73 76 79
10/07/2016,10 18 24 27 33 35 37 38 39 42 43 48 52 57 62 72 73 75 78 79
10/06/2016,09 10 16 21 22 23 27 36 37 38 39 40 50 52 53 57 62 65 68 71
10/05/2016,02 03 12 15 17 18 19 21 25 31 32 45 48 51 54 55 65 70 73 74
10/04/2016,05 08 11 19 20 21 22 23 24 38 40 46 49 54 55 57 58 62 67 75
10/03/2016,01 08 09 13 21 22 23 29 30 33 36 42 43 49 52 56 57 63 64 79
10/02/2016,01 04 06 08 09 15 18 19 25 38 41 42 45 46 47 49 50 51 65 70
10/01/2016,09 16 18 21 22 31 34 35 36 39 40 42 43 44 45 52 55 56 62 66
09/30/2016,09 11 14 15 16 20 23 30 33 37 40 44 49 53 54 55 56 61 69 78
09/29/2016,04 13 15 24 25 31 37 38 46 48 56 57 62 63 65 67 69 73 79 80
09/28/2016,01 06 09 15 16 24 32 34 36 44 55 57 59 63 66 67 71 72 74 76
09/27/2016,03 10 13 15 17 19 27 29 33 35 41 42 44 59 60 63 68 70 73 77
09/26/2016,03 11 13 15 16 20 22 32 35 37 47 55 60 61 63 68 73 74 77 79
09/25/2016,01 05 11 17 22 23 33 34 38 41 42 45 46 50 53 54 69 70 76 78
09/24/2016,01 06 08 14 18 26 32 33 36 37 45 46 51 54 57 59 62 65 71 77
09/23/2016,02 03 05 08 09 16 22 28 29 31 32 42 43 53 55 58 64 69 70 71
09/22/2016,01 03 05 11 13 23 27 31 32 35 38 46 50 52 56 59 60 62 72 80
09/21/2016,01 12 23 25 27 29 31 35 37 39 44 45 46 47 50 57 58 62 68 72
09/20/2016,04 08 13 14 16 24 31 36 38 46 47 48 53 60 61 62 63 69 76 79
09/19/2016,04 07 13 16 17 18 22 27 28 33 36 39 43 46 53 55 58 67 77 78
09/18/2016,01 04 13 14 20 26 28 36 38 42 45 47 50 53 57 64 68 69 73 74
09/17/2016,04 11 13 17 27 31 33 37 38 39 49 50 56 58 59 68 71 72 74 79
09/16/2016,05 10 11 12 20 22 25 31 33 43 46 47 51 57 64 65 67 73 77 79
09/15/2016,05 12 13 14 15 21 25 32 42 43 45 49 55 56 60 61 64 67 78 80
09/14/2016,11 16 18 25 28 30 31 32 33 34 47 48 56 59 62 66 69 70 76 79
09/13/2016,10 15 16 18 21 22 27 44 47 51 55 57 59 60 65 66 68 72 75 77
09/12/2016,03 07 10 11 15 19 21 25 28 30 34 37 43 48 51 57 58 70 71 80
09/11/2016,03 06 08 10 18 19 20 22 28 31 37 39 49 54 62 63 68 73 77 79
09/10/2016,03 07 11 17 23 26 31 32 40 41 50 53 63 64 66 67 75 76 78 79
09/09/2016,02 05 07 08 09 10 16 17 25 38 44 46 60 62 63 66 73 74 78 79
09/08/2016,06 09 10 11 16 31 34 40 41 43 45 46 49 51 58 59 64 70 71 72
09/07/2016,01 07 08 09 13 15 28 34 35 37 39 45 50 51 56 58 65 72 76 77
09/06/2016,04 05 11 18 19 24 28 31 32 42 43 47 54 58 65 66 67 71 72 79
09/05/2016,03 05 06 08 10 11 14 15 16 21 29 30 41 55 59 60 62 64 71 78
09/04/2016,07 08 12 17 30 31 35 36 38 44 50 55 59 62 63 64 71 73 77 80
09/03/2016,01 05 07 11 14 17 19 35 38 42 43 49 50 55 57 58 68 74 78 79
09/02/2016,08 16 19 21 24 27 33 38 40 42 43 45 47 48 53 61 66 72 74 79
```

```
89      09/01/2016,06 11 17 26 30 36 38 39 45 46 48 49 54 58 63 72 73 75 77 79
90      08/31/2016,01 08 09 12 16 17 22 29 31 32 33 37 38 43 46 47 50 58 73 74
91      08/30/2016,08 15 16 19 20 21 28 29 31 44 47 51 58 61 63 64 65 74 78 80
92      08/29/2016,02 06 07 11 13 15 18 25 27 34 37 40 41 44 46 48 49 50 52 77
93      08/28/2016,04 05 06 08 12 13 17 28 35 38 40 47 48 49 52 56 62 66 71 80
94      08/27/2016,03 08 12 16 20 21 22 23 26 31 41 42 47 51 58 62 68 73 75 78
95      08/26/2016,04 07 08 12 13 14 16 27 34 37 38 39 41 44 45 49 54 56 67 74
96      08/25/2016,01 07 10 14 16 26 27 29 32 36 40 41 47 53 65 67 72 73 78 79
97      08/24/2016,02 05 11 15 18 20 22 26 30 32 35 40 45 50 52 53 74 76 78 79
98      08/23/2016,02 09 11 15 16 21 28 33 34 35 41 43 44 52 54 55 56 71 76 78
99      08/22/2016,01 08 09 10 11 12 16 17 30 35 36 37 39 40 46 56 62 66 69 76
00      08/21/2016,09 14 15 16 17 21 23 27 45 46 51 54 55 60 62 66 68 71 77 80
01      08/20/2016,06 07 08 11 18 21 25 27 28 31 32 36 38 47 57 59 62 64 69 72
02      08/19/2016,06 08 09 10 11 12 15 19 22 24 32 33 40 44 49 50 53 61 62 69
03      08/18/2016,09 12 16 20 30 41 42 43 46 48 49 50 53 56 58 60 68 70 76 80
04      08/17/2016,08 09 20 23 24 27 30 33 34 41 46 57 58 59 62 63 66 71 73 80
05      08/16/2016,01 06 07 09 12 13 27 29 38 42 43 45 48 53 57 61 63 66 67 72
06      08/15/2016,03 05 15 21 24 26 30 32 33 38 39 40 44 45 48 60 61 68 69 80
07      08/14/2016,07 12 13 14 15 18 20 22 24 26 27 29 31 35 42 58 63 64 71 78
08      08/13/2016,03 18 19 25 36 37 38 45 49 50 51 54 61 65 66 67 69 70 72 79
09      08/12/2016,02 03 05 09 12 13 21 24 31 34 37 40 45 48 53 60 62 63 67 78
10      08/11/2016,01 03 06 08 15 18 21 22 23 27 28 35 40 48 59 61 66 70 76 80
11      08/10/2016,01 03 08 13 14 17 20 32 43 47 48 52 55 62 67 72 75 76 78 80
12      08/09/2016,04 05 07 08 17 22 24 29 31 33 34 37 44 47 50 51 57 61 63 68
13      08/08/2016,03 04 07 11 14 26 31 32 36 43 44 66 67 69 70 71 72 75 78 79
14      08/07/2016,04 11 13 14 20 21 24 27 30 33 36 40 45 50 56 58 59 66 69 76
15      08/06/2016,07 08 16 17 19 22 29 34 35 45 46 47 52 54 57 58 61 68 70 72
16      08/05/2016,03 07 12 19 21 27 28 31 33 39 41 43 46 47 48 53 58 61 71 75
17      08/04/2016,05 10 19 23 26 28 29 45 46 52 54 58 59 62 66 67 71 73 78 79
18      08/03/2016,07 09 10 12 22 25 29 32 34 39 41 42 44 50 51 63 66 71 73 79
19      08/02/2016,01 11 15 17 21 23 25 26 27 31 40 41 42 44 52 61 63 70 76 79
20      08/01/2016,09 11 14 17 19 22 24 27 33 41 43 44 49 53 54 63 66 69 71 77
21      07/31/2016,03 04 05 08 10 18 22 23 25 34 42 50 57 61 63 66 69 73 77 79
22      07/30/2016,04 07 12 24 29 32 33 37 38 40 48 51 54 57 62 64 65 68 69 71
23      07/29/2016,04 05 06 07 11 16 17 21 29 36 38 39 46 47 57 58 59 60 66 71
24      07/28/2016,01 04 06 09 11 14 17 22 29 31 33 34 51 53 55 64 68 69 70 74
25      07/27/2016,08 09 22 25 28 29 34 35 36 40 42 44 46 47 50 52 54 57 72 78
26      07/26/2016,01 14 18 19 21 27 29 32 33 35 36 39 43 45 47 57 61 66 72 80
27      07/25/2016,02 03 07 08 11 16 21 30 37 38 43 45 47 49 57 60 66 68 69 74
28      07/24/2016,01 04 05 08 13 19 24 26 30 32 36 37 47 55 57 60 61 73 76 79
29      07/23/2016,06 11 19 20 22 30 34 35 47 52 56 58 59 61 64 65 70 71 77 80
30      07/22/2016,14 19 22 23 24 31 36 38 42 45 46 47 52 53 58 59 70 74 75 78
31      07/21/2016,10 13 16 20 23 24 29 30 31 33 36 40 45 48 51 66 69 73 78 79
32      07/20/2016,01 03 11 22 23 24 30 33 40 41 43 45 49 50 53 57 61 70 74 78
33      07/19/2016,01 04 07 09 10 12 16 26 30 36 41 42 46 56 63 67 69 70 73 75
34      07/18/2016,02 10 14 17 19 25 31 36 43 46 54 56 58 59 60 67 69 70 72 78
35      07/17/2016,10 14 20 21 24 28 30 33 37 43 44 45 48 53 58 59 61 70 73 74
36      07/16/2016,02 08 11 18 22 30 33 35 36 44 46 50 52 59 61 62 69 70 72 73
37      07/15/2016,01 03 07 10 11 12 13 14 19 21 32 36 48 51 53 56 58 60 62 63
38      07/14/2016,03 10 19 23 24 28 36 38 42 46 51 53 57 58 61 65 72 73 75 77
39      07/13/2016,01 03 07 09 11 12 15 16 26 27 32 38 48 55 65 68 70 71 73 74
40      07/12/2016,03 12 18 19 23 31 34 38 42 48 49 50 52 53 61 65 72 75 76 79
41      07/11/2016,02 08 20 21 24 26 34 38 43 45 54 55 62 65 66 70 72 74 78 79
42      07/10/2016,02 07 09 11 18 23 32 35 37 43 46 47 57 58 61 62 71 75 76 78
```

```
07/09/2016,01 04 05 13 19 27 28 29 32 38 40 49 52 54 58 60 64 68 71 77
07/08/2016,01 03 13 16 28 30 33 37 43 49 50 56 57 58 61 68 69 70 71 78
07/07/2016,02 07 08 09 17 21 26 33 36 40 41 44 46 54 56 69 70 72 75 77
07/06/2016,02 03 05 12 29 35 39 46 51 52 54 55 60 65 67 68 76 77 78 79
07/05/2016,07 09 18 28 29 31 33 35 38 43 44 45 46 47 48 49 53 63 70 80
07/04/2016,01 02 03 08 09 15 18 23 28 32 35 43 44 54 60 61 65 71 79 80
07/03/2016,09 12 15 25 26 29 31 34 38 39 51 58 63 64 67 70 71 74 78 80
07/02/2016,03 04 11 17 21 25 27 32 37 39 40 45 47 50 58 64 68 75 78 79
07/01/2016,17 26 37 40 41 43 46 48 52 54 55 56 60 61 66 69 70 73 75 79
06/30/2016,01 05 12 14 18 19 23 27 31 43 45 46 48 49 50 52 59 74 79 80
06/29/2016,02 05 06 07 22 25 26 29 34 40 47 48 49 54 56 59 63 66 76 80
06/28/2016,06 13 15 20 21 23 24 26 32 36 45 47 49 57 61 62 68 73 74 79
06/27/2016,02 06 08 09 12 19 23 28 38 42 43 47 51 52 65 66 67 71 75 78
06/26/2016,04 10 20 21 23 24 27 30 38 41 47 53 61 62 63 64 71 72 74 79
06/25/2016,10 12 15 19 23 28 32 37 44 48 58 61 64 65 68 71 73 76 77 80
06/24/2016,04 20 25 31 34 35 37 39 42 47 49 52 55 60 63 71 73 77 78 80
06/23/2016,01 04 06 16 17 20 31 32 36 39 40 42 47 58 60 61 63 66 68 69
06/22/2016,02 04 09 18 22 26 32 38 47 49 50 57 59 61 62 63 67 74 77 80
06/21/2016,01 02 06 13 17 20 30 31 36 41 43 49 51 55 60 64 71 75 77 78
06/20/2016,03 05 09 10 16 29 31 34 35 36 37 38 49 50 56 66 71 73 75 80
06/19/2016,03 05 16 19 27 37 42 44 45 46 48 49 51 54 57 60 62 72 73 78
06/18/2016,02 03 05 17 18 27 32 37 39 40 54 55 60 64 67 68 69 77 78 79
06/17/2016,02 11 13 17 21 23 25 31 33 40 47 50 52 53 62 68 69 74 75 80
06/16/2016,02 04 22 23 32 33 38 40 42 43 44 46 51 57 61 67 68 71 73 76
06/15/2016,06 12 15 23 32 35 38 39 44 46 47 53 54 59 62 64 66 70 73 78
06/14/2016,01 09 10 13 17 19 21 23 30 34 38 39 46 49 53 57 59 60 66 74
06/13/2016,03 04 07 13 18 20 24 35 36 43 47 52 53 55 60 63 65 68 74 80
06/12/2016,05 06 10 11 16 18 22 30 35 39 50 58 61 63 68 70 72 74 79 80
06/11/2016,03 05 11 18 22 28 29 36 38 48 53 54 57 60 62 63 69 74 77 80
06/10/2016,02 04 09 10 13 14 16 20 22 26 33 38 41 45 60 62 64 67 74 80
06/09/2016,07 13 14 16 17 18 21 22 23 25 28 31 32 46 52 55 56 64 66 79
06/08/2016,04 05 08 10 11 13 18 20 24 31 54 59 63 65 68 69 72 73 75 79
06/07/2016,06 08 12 13 14 19 20 25 32 36 39 40 44 45 60 61 63 64 65 71
06/06/2016,02 04 11 16 21 23 24 28 35 39 40 42 44 45 48 51 70 74 75 78
06/05/2016,01 04 05 10 11 20 21 24 28 34 38 40 41 54 60 64 68 71 77 80
06/04/2016,02 21 23 30 31 34 36 42 50 57 60 61 62 63 70 71 73 75 78 79
06/03/2016,03 07 13 14 25 31 38 40 48 49 54 56 57 64 70 71 73 74 76 78
06/02/2016,01 04 07 10 16 21 23 24 26 28 30 37 51 53 55 63 64 67 69 80
06/01/2016,06 11 14 19 20 31 32 40 42 44 52 53 54 57 60 62 65 66 69 74
05/31/2016,04 06 08 15 36 38 39 42 46 48 51 55 58 60 62 63 64 66 75 78
05/30/2016,01 03 19 22 24 28 33 40 41 46 50 53 54 58 59 60 64 67 75 80
05/29/2016,07 10 12 13 14 24 33 34 36 37 40 44 46 47 64 66 68 71 72 80
05/28/2016,05 08 09 13 14 15 23 24 29 33 37 38 41 43 45 47 51 61 76 78
05/27/2016,06 07 15 16 21 22 23 24 28 36 38 40 42 52 58 70 72 73 76 79
05/26/2016,02 03 05 06 08 14 17 19 21 25 41 42 43 45 54 57 67 68 71 73
05/25/2016,03 05 06 07 15 16 18 24 28 37 40 41 53 59 63 74 75 76 77 79
05/24/2016,01 07 09 13 24 27 30 31 32 33 36 42 43 52 60 64 67 68 72 78
05/23/2016,03 12 14 23 25 28 29 32 35 43 46 52 55 56 60 64 69 71 76 77
05/22/2016,01 02 04 12 21 24 25 28 38 43 47 54 56 58 62 64 71 76 77 80
05/21/2016,01 11 15 17 23 24 30 32 36 39 41 43 48 49 52 56 57 62 68 71
05/20/2016,02 05 09 11 12 18 19 22 34 38 41 46 48 60 62 66 68 69 74 77
05/19/2016,02 03 07 11 13 14 25 27 33 46 48 50 56 58 63 66 67 76 77 79
05/18/2016,01 03 26 29 31 34 38 44 46 49 50 52 53 54 61 65 66 71 78 80
05/17/2016,11 15 16 20 34 35 38 39 41 42 43 46 48 49 55 58 70 73 75 77
```

```
97      05/16/2016,02 05 12 22 27 28 32 42 48 49 52 54 59 61 64 71 72 77 79 80
98      05/15/2016,03 06 09 18 22 27 31 32 43 46 53 61 64 65 71 74 75 77 78 80
99      05/14/2016,07 13 16 21 24 25 31 32 35 42 51 52 62 63 66 67 71 73 74 78
00      05/13/2016,01 03 05 07 18 19 25 34 41 42 50 54 56 57 60 61 66 77 78 80
01      05/12/2016,01 04 09 13 16 23 24 26 30 39 41 43 50 52 60 64 70 71 72 76
02      05/11/2016,04 07 12 13 19 21 23 24 26 28 31 32 33 41 44 47 51 57 71 78
03      05/10/2016,02 04 06 11 13 18 22 29 36 37 38 39 44 46 66 69 70 75 76 80
04      05/09/2016,02 07 09 11 12 14 19 25 30 47 48 55 57 61 64 66 69 70 75 76
05      05/08/2016,05 07 08 12 13 15 16 30 39 42 44 50 51 52 55 61 69 75 76 78
06      05/07/2016,07 10 12 13 21 24 33 36 37 41 46 50 57 58 60 61 68 69 72 74
07      05/06/2016,03 15 21 22 24 26 29 32 33 35 37 42 43 44 45 52 54 58 75 76
08      05/05/2016,02 03 05 16 17 19 23 26 31 40 42 45 47 56 58 60 71 72 73 77
09      05/04/2016,05 07 11 13 14 16 24 25 26 33 34 38 41 42 45 50 57 63 71 79
10      05/03/2016,04 12 25 26 29 34 36 37 39 42 44 45 50 53 56 63 67 68 69 71
11      05/02/2016,01 03 04 19 20 26 29 30 31 34 43 45 59 61 62 65 68 74 76 79
12      05/01/2016,03 08 09 10 11 15 18 22 30 43 45 49 51 52 57 60 62 64 68 74
13      04/30/2016,02 05 06 07 08 12 17 18 31 35 41 42 43 44 58 60 61 66 75 77
14      04/29/2016,08 09 10 17 18 19 21 23 24 27 28 32 36 43 59 67 71 76 78 79
15      04/28/2016,07 09 10 13 15 16 18 23 25 31 32 36 39 45 51 53 59 61 67 72
16      04/27/2016,04 06 09 11 13 15 18 19 21 23 25 27 36 38 51 54 57 67 75 76
17      04/26/2016,04 13 26 28 32 33 39 41 47 48 52 53 57 58 60 61 68 69 76 78
18      04/25/2016,03 12 15 17 18 22 25 38 39 45 47 51 52 54 64 67 69 74 75 76
19      04/24/2016,07 10 11 12 13 14 15 16 27 37 38 43 44 47 53 54 59 67 69 73
20      04/23/2016,02 04 07 08 14 23 27 49 50 51 52 58 59 60 64 69 76 77 79 80
21      04/22/2016,01 02 08 19 25 28 31 32 33 34 44 49 52 53 54 62 64 74 78 79
22      04/21/2016,06 07 12 17 18 20 23 29 31 33 34 42 43 48 55 57 60 61 64 77
23      04/20/2016,02 06 08 10 13 17 19 30 31 32 34 36 43 46 50 53 62 65 69 80
24      04/19/2016,01 02 04 05 07 09 10 14 21 26 27 34 51 52 53 58 65 72 74 79
25      04/18/2016,01 08 09 18 27 29 35 38 39 42 43 46 49 52 56 59 69 70 72 75
26      04/17/2016,01 04 07 09 25 30 31 35 38 40 41 43 51 56 63 64 68 69 70 76
27      04/16/2016,04 08 17 19 20 25 27 35 41 46 49 52 56 58 60 62 63 68 73 80
28      04/15/2016,04 08 11 14 19 20 21 26 28 34 42 43 45 54 56 63 73 78 79 80
29      04/14/2016,01 06 13 14 17 18 19 22 24 36 37 38 41 45 50 51 58 74 75 76
30      04/13/2016,02 09 19 24 25 26 30 32 34 40 43 44 51 52 53 66 67 72 73 79
31      04/12/2016,01 05 06 10 12 20 24 26 31 32 39 40 42 45 49 54 56 57 69 70
32      04/11/2016,02 05 06 12 16 17 19 22 23 30 31 35 36 38 52 55 57 62 73 80
33      04/10/2016,03 11 14 21 23 29 31 39 43 50 51 54 55 62 68 70 72 76 78 80
34      04/09/2016,02 07 10 13 15 27 33 39 44 46 50 51 55 56 61 62 63 65 67 73
35      04/08/2016,02 07 10 12 13 17 21 22 25 26 29 35 42 45 49 50 53 68 69 73
36      04/07/2016,01 15 22 26 30 31 35 42 44 50 51 52 53 60 61 66 71 72 78 79
37      04/06/2016,03 08 12 13 18 19 30 37 38 42 45 48 53 55 56 57 58 69 71 79
38      04/05/2016,01 05 06 07 11 14 22 30 32 34 35 37 44 45 54 55 60 61 66 68
39      04/04/2016,03 05 06 11 14 22 27 29 30 36 41 47 58 62 64 65 68 70 72 77
40      04/03/2016,02 09 19 21 36 39 40 42 53 56 57 61 63 64 69 72 73 75 77 79
41      04/02/2016,02 08 20 22 24 26 28 29 33 39 40 42 43 44 60 62 66 67 78 79
42      04/01/2016,09 11 12 14 15 20 21 22 23 27 36 43 46 53 61 73 76 77 78 79
43      03/31/2016,07 11 14 19 23 25 28 30 31 33 38 39 42 47 53 59 62 68 73 76
44      03/30/2016,03 13 15 18 21 27 29 35 38 44 45 47 48 49 55 56 58 63 71 74
45      03/29/2016,01 03 04 11 24 27 35 38 41 45 47 48 52 53 60 63 64 66 67 68
46      03/28/2016,02 03 07 10 11 24 30 31 32 37 47 48 52 57 62 65 66 70 72 79
47      03/27/2016,02 04 08 09 11 15 24 32 33 40 44 45 50 55 62 63 65 67 74 77
48      03/26/2016,04 05 06 07 09 16 19 20 24 28 31 33 38 44 49 59 62 63 64 79
49      03/25/2016,02 05 08 13 14 16 18 20 35 39 43 45 47 50 56 58 65 69 70 76
50      03/24/2016,01 02 03 05 11 12 20 27 29 31 33 36 39 51 52 58 60 61 76 77
```

```
51      03/23/2016,01 03 07 09 10 15 16 19 23 24 30 45 50 53 57 61 64 65 68 80
52      03/22/2016,08 13 15 21 22 25 30 31 33 38 39 42 55 60 62 63 64 66 76 79
53      03/21/2016,09 10 13 14 26 30 34 35 40 42 45 47 50 51 52 53 68 70 76 80
54      03/20/2016,06 12 17 22 26 30 38 40 45 47 53 56 60 62 67 68 69 73 76 80
55      03/19/2016,02 03 15 17 19 21 23 30 36 41 43 45 46 49 50 51 64 67 68 77
56      03/18/2016,03 05 06 11 22 23 39 41 43 44 45 52 54 56 61 66 67 68 70 74
57      03/17/2016,04 16 19 20 22 29 35 37 38 40 42 48 55 59 61 63 66 67 72 73
58      03/16/2016,09 11 20 22 27 28 29 31 38 40 44 50 56 62 63 65 67 71 73 79
59      03/15/2016,02 19 27 28 29 33 37 40 41 43 45 47 62 63 64 68 70 74 79 80
60      03/14/2016,04 08 09 17 18 19 21 23 27 32 38 46 47 52 53 57 61 73 76 77
61      03/13/2016,02 04 17 18 19 23 25 27 31 34 44 49 51 55 57 63 65 71 76 79
62      03/12/2016,02 09 17 25 27 33 37 41 42 43 48 49 55 56 64 65 68 72 74 75
63      03/11/2016,06 11 13 17 30 31 37 40 41 44 50 53 55 59 61 65 67 75 78 80
64      03/10/2016,01 03 05 08 11 12 14 21 25 32 35 42 45 46 50 51 52 55 67 69
65      03/09/2016,03 05 08 09 12 17 23 27 28 34 36 40 51 55 63 64 69 71 74 77
66      03/08/2016,01 06 07 09 10 21 27 28 29 30 34 35 36 40 50 51 63 70 74 79
67      03/07/2016,01 09 18 22 23 25 27 29 32 37 42 55 56 59 61 62 65 66 76 78
68      03/06/2016,07 09 14 17 19 30 35 37 48 50 51 54 56 57 60 63 64 66 69 77
69      03/05/2016,08 11 14 23 32 38 39 40 42 44 48 56 59 60 63 65 66 76 77 78
70      03/04/2016,01 09 10 13 20 25 38 49 50 53 55 56 58 62 65 67 69 71 72 74
71      03/03/2016,02 03 09 13 14 18 20 27 28 30 31 33 38 43 46 53 59 63 67 69
72      03/02/2016,05 07 08 10 17 25 28 31 37 38 41 50 51 56 57 59 66 67 68 75
73      03/01/2016,02 08 09 12 20 21 22 23 24 30 32 33 34 35 39 42 44 52 63 79
74      02/29/2016,03 06 09 12 13 21 24 38 40 48 49 52 53 54 62 63 65 69 76 78
75      02/28/2016,01 03 10 17 19 21 24 29 36 41 49 54 56 65 66 71 72 74 77 80
76      02/27/2016,03 06 10 12 15 20 25 30 33 43 48 50 58 60 65 69 71 72 73 78
77      02/26/2016,09 11 12 15 18 20 23 24 30 33 36 40 46 47 56 59 61 62 67 74
78      02/25/2016,06 14 15 16 17 21 23 24 34 38 40 49 56 58 59 62 63 65 76 77
79      02/24/2016,11 25 26 31 32 33 35 38 42 43 44 46 49 50 54 59 63 65 68 80
80      02/23/2016,05 09 10 21 23 30 46 50 52 55 58 60 61 62 63 65 68 71 78 79
81      02/22/2016,01 05 14 15 18 19 22 25 28 34 42 52 54 58 61 62 67 68 71 78
82      02/21/2016,01 06 10 13 14 18 23 27 34 36 38 44 47 57 63 64 65 70 73 80
83      02/20/2016,03 16 21 25 35 36 39 41 45 48 50 51 55 59 62 66 70 72 76 79
84      02/19/2016,01 02 03 05 08 15 17 22 31 32 34 36 44 53 57 61 64 66 73 80
85      02/18/2016,01 05 07 09 14 16 20 22 29 38 49 53 59 60 62 64 68 70 77 78
86      02/17/2016,05 07 11 15 20 22 24 25 26 28 29 30 43 47 51 58 65 69 72 73
87      02/16/2016,10 12 21 25 28 30 32 35 39 41 47 48 49 58 60 71 73 74 78 80
88      02/15/2016,02 03 12 20 21 28 30 35 40 43 48 53 57 63 64 67 73 77 78 80
89      02/14/2016,03 05 14 15 17 18 19 24 27 29 30 35 38 40 55 61 65 69 76 80
90      02/13/2016,03 05 09 15 16 18 34 35 40 48 57 59 64 66 69 70 71 75 77 80
91      02/12/2016,05 11 18 33 34 37 39 45 49 50 51 57 65 66 71 72 75 77 78 79
92      02/11/2016,08 10 13 15 18 21 23 25 29 30 35 36 56 58 62 64 66 69 75 76
93      02/10/2016,01 04 08 10 11 14 15 22 32 34 40 42 43 49 55 62 67 69 74 79
94      02/09/2016,04 06 07 10 12 14 15 19 22 40 47 52 55 57 63 66 67 68 71 75
95      02/08/2016,01 02 03 04 05 08 12 23 25 28 34 39 42 43 45 50 51 57 59 60
96      02/07/2016,04 12 17 18 26 30 32 33 35 40 45 46 55 58 59 63 67 70 72 77
97      02/06/2016,04 11 12 16 21 27 28 30 32 35 38 44 46 47 54 62 68 71 75 79
98      02/05/2016,04 09 10 12 13 15 25 31 46 48 56 59 60 64 66 67 73 77 78 80
99      02/04/2016,01 02 05 06 07 12 14 20 24 25 28 29 32 33 40 50 51 52 53 69
00      02/03/2016,01 06 07 12 14 18 19 20 21 26 28 40 44 56 60 64 68 74 77 79
01      02/02/2016,02 07 13 17 23 26 27 29 32 37 38 39 46 47 61 66 71 72 75 77
02      02/01/2016,02 06 07 08 11 15 16 20 21 25 31 35 38 41 45 47 49 51 58 70
03      01/31/2016,01 04 06 10 15 19 27 32 33 34 36 40 50 59 60 61 65 67 70 72
04      01/30/2016,03 09 15 16 21 23 26 28 30 34 40 46 50 51 53 55 58 66 67 80
```

```
05      01/29/2016,02 05 08 13 16 36 39 40 43 44 46 52 54 55 60 63 64 69 73 80
06      01/28/2016,01 02 06 11 18 19 31 32 36 37 45 47 49 50 55 56 60 63 72 77
07      01/27/2016,08 09 15 20 23 24 25 43 60 63 64 65 66 68 70 71 72 73 77 78
08      01/26/2016,02 03 11 13 22 25 35 41 42 43 46 48 49 53 56 63 65 71 74 75
09      01/25/2016,16 17 20 22 25 28 29 33 35 47 48 49 51 52 53 56 61 66 73 74
10      01/24/2016,04 06 09 11 12 14 16 19 21 28 32 33 36 40 43 57 64 68 69 78
11      01/23/2016,01 03 05 17 18 20 25 30 42 46 49 51 58 60 61 62 63 64 68 74
12      01/22/2016,03 13 14 16 17 18 31 32 37 47 48 49 51 57 59 63 65 76 77 78
13      01/21/2016,11 12 14 16 22 24 26 28 35 38 40 49 51 60 61 69 73 77 78 80
14      01/20/2016,02 10 12 17 19 23 24 34 39 40 42 45 48 49 53 55 62 67 76 80
15      01/19/2016,06 10 15 17 19 30 32 33 35 38 40 42 43 44 50 58 62 74 76 78
16      01/18/2016,03 14 16 25 27 28 31 32 33 35 38 40 45 49 59 63 67 70 73 75
17      01/17/2016,04 10 12 21 22 23 25 27 34 40 42 49 51 55 59 68 73 76 78 80
18      01/16/2016,02 12 16 20 25 26 27 29 30 31 36 41 47 53 56 63 66 67 74 78
19      01/15/2016,13 15 19 21 25 27 29 34 36 37 40 44 46 51 54 59 62 74 76 79
20      01/14/2016,12 15 16 30 31 35 36 37 40 43 44 45 46 48 49 58 63 65 71 73
21      01/13/2016,04 05 10 11 17 18 19 24 28 34 35 38 46 55 60 64 67 73 78 79
22      01/12/2016,03 07 08 09 10 11 14 15 18 28 34 51 52 62 63 67 73 75 76 80
23      01/11/2016,02 04 05 20 21 23 27 28 31 36 39 41 48 49 50 60 61 63 65 74
24      01/10/2016,08 10 13 14 16 20 21 22 24 25 26 28 29 35 37 45 56 57 60 67
25      01/09/2016,01 06 08 14 20 23 26 29 31 35 45 46 48 50 53 54 68 72 75 76
26      01/08/2016,04 07 18 20 23 24 25 28 29 34 38 40 42 44 58 59 64 67 68 75
27      01/07/2016,06 07 09 14 15 17 21 22 31 42 43 47 53 68 70 71 72 73 77 79
28      01/06/2016,04 09 11 13 21 31 34 41 46 49 50 51 52 55 56 58 63 64 65 69
29      01/05/2016,10 20 22 23 25 35 39 42 45 46 50 52 53 57 61 62 67 72 74 77
30      01/04/2016,02 03 04 06 07 08 14 18 20 24 32 33 38 44 48 50 56 59 69 74
31      01/03/2016,04 08 10 12 15 33 38 39 40 45 46 49 50 60 64 69 71 72 77 79
32      01/02/2016,03 09 11 13 14 28 30 34 35 43 45 52 57 59 63 65 69 70 77 78
33      01/01/2016,05 08 12 14 15 29 38 40 44 46 47 50 57 67 69 70 72 76 78 79
34      12/31/2015,04 07 09 12 13 24 33 38 43 44 52 54 55 59 65 67 73 74 77 78
35      12/30/2015,05 06 09 10 11 18 24 25 37 43 44 51 57 58 61 71 73 74 75 76
36      12/29/2015,12 15 17 22 23 27 29 30 31 33 35 42 44 58 66 69 75 76 77 80
37      12/28/2015,02 15 18 19 23 26 34 36 37 38 39 44 51 57 59 61 70 72 73 78
38      12/27/2015,03 06 10 14 20 24 29 31 35 43 49 54 60 62 63 67 68 74 77 79
39      12/26/2015,02 09 11 13 14 15 28 29 33 35 43 45 50 54 58 66 71 75 77 79
40      12/25/2015,03 04 05 13 16 27 29 31 32 33 34 35 37 40 45 47 70 73 75 77
41      12/24/2015,01 02 11 16 19 20 22 27 28 30 34 37 45 56 59 60 66 68 71 72
42      12/23/2015,01 05 18 20 22 26 31 32 33 35 42 44 48 54 55 58 63 66 74 78
43      12/22/2015,06 09 10 13 24 28 32 35 45 52 56 58 59 62 64 65 70 72 77 80
44      12/21/2015,02 10 12 15 18 27 34 43 46 49 53 55 59 64 65 66 68 74 75 79
45      12/20/2015,03 05 15 21 23 25 26 28 29 32 46 48 53 55 56 58 60 64 69 79
46      12/19/2015,12 14 15 23 28 29 32 39 43 48 49 53 59 60 61 64 66 69 75 77
47      12/18/2015,05 09 11 12 14 16 24 28 31 32 36 38 42 45 46 48 66 73 74 79
48      12/17/2015,01 02 03 04 06 11 15 19 27 42 51 54 59 60 61 62 65 67 68 77
49      12/16/2015,06 09 10 11 20 21 23 45 51 54 55 59 62 66 68 71 72 73 77 78
50      12/15/2015,04 07 08 09 12 16 20 21 22 24 39 41 43 44 45 61 63 65 66 75
51      12/14/2015,05 07 15 19 20 23 24 26 31 33 40 43 49 63 64 73 75 78 79 80
52      12/13/2015,06 10 12 15 20 25 29 38 50 51 56 59 61 66 68 73 75 77 78 80
53      12/12/2015,05 06 10 11 17 19 20 24 33 43 47 49 58 60 66 67 70 74 76 77
54      12/11/2015,04 06 10 12 17 18 19 26 29 35 41 43 44 45 49 50 57 63 68 77
55      12/10/2015,06 10 11 14 19 24 33 37 39 44 47 48 50 56 63 69 71 75 77 78
56      12/09/2015,08 17 19 23 26 29 30 31 37 40 43 53 54 56 58 61 66 68 72 78
57      12/08/2015,06 08 09 14 16 19 20 23 27 30 41 43 53 54 56 58 71 72 74 79
58      12/07/2015,01 02 07 13 18 20 23 24 32 35 37 38 40 41 42 47 59 60 62 72
```

```
59      12/06/2015,01 03 14 20 24 25 33 36 40 48 51 52 55 56 58 62 63 72 74 80
60      12/05/2015,02 03 08 10 13 16 33 38 41 46 49 50 56 64 68 74 75 76 77 80
61      12/04/2015,07 15 20 23 24 25 31 37 45 54 56 58 62 65 71 72 76 77 78 79
62      12/03/2015,01 03 04 08 09 10 15 18 24 28 29 32 33 34 37 38 48 51 54 61
63      12/02/2015,01 07 09 16 20 27 30 33 35 36 45 48 49 51 59 60 66 70 77 80
64      12/01/2015,01 03 04 06 09 10 11 15 27 29 50 51 52 53 54 56 61 64 65 75
65      11/30/2015,02 04 11 12 15 23 25 33 35 50 51 55 57 58 60 61 65 68 72 78
66      11/29/2015,01 02 03 06 15 22 23 32 37 40 45 50 54 62 63 68 70 74 77 80
67      11/28/2015,02 06 07 16 17 22 25 35 37 40 43 52 54 60 64 65 67 68 69 70
68      11/27/2015,06 11 16 18 20 22 27 31 35 36 39 41 44 45 49 51 55 64 67 74
69      11/26/2015,01 03 07 12 13 16 17 19 23 28 34 36 37 40 47 51 55 72 74 75
70      11/25/2015,02 05 07 11 17 18 20 29 31 36 48 51 56 60 65 68 69 73 74 75
71      11/24/2015,04 06 09 11 13 19 20 22 23 26 32 39 48 52 53 56 59 63 69 80
72      11/23/2015,03 10 18 22 26 28 29 35 37 40 41 47 51 52 55 56 62 66 67 78
73      11/22/2015,01 03 08 10 11 18 22 24 32 36 41 45 47 50 52 55 64 67 69 78
74      11/21/2015,02 08 14 18 22 26 28 35 37 38 40 42 44 46 49 54 60 68 74 77
75      11/20/2015,09 14 24 32 39 41 43 46 47 48 50 51 53 55 58 66 67 68 69 71
76      11/19/2015,02 08 10 11 13 14 21 27 28 39 42 45 54 55 63 64 66 68 69 79
77      11/18/2015,04 06 09 12 17 28 29 30 34 38 40 42 45 54 59 60 66 67 71 80
78      11/17/2015,03 05 07 11 14 17 22 31 37 40 46 52 55 56 59 62 66 67 72 75
79      11/16/2015,01 07 17 19 22 23 26 36 40 44 46 49 51 57 61 67 68 75 78 79
80      11/15/2015,01 02 03 04 06 08 11 13 14 28 34 38 40 41 48 56 63 72 74 77
81      11/14/2015,05 12 13 19 31 34 35 37 41 42 44 45 46 49 51 52 68 72 75 80
82      11/13/2015,04 05 07 17 20 24 29 30 32 33 34 38 42 50 58 66 71 75 76 78
83      11/12/2015,01 02 04 09 19 24 26 45 46 48 49 51 54 55 62 64 70 71 75 78
84      11/11/2015,09 13 21 23 25 26 30 31 33 37 41 43 53 55 60 66 67 72 75 80
85      11/10/2015,01 08 14 16 26 30 34 35 36 41 45 50 51 55 57 65 66 68 70 74
86      11/09/2015,13 19 20 21 24 25 28 33 34 40 45 54 55 56 58 60 65 66 73 78
87      11/08/2015,04 06 08 09 11 13 15 21 24 29 33 44 46 48 54 60 65 66 72 80
88      11/07/2015,05 07 08 09 15 21 22 27 29 31 39 40 41 50 56 57 62 69 74 80
89      11/06/2015,02 04 06 11 14 18 31 32 35 38 43 51 54 55 57 59 60 61 71 72
90      11/05/2015,14 15 19 33 34 35 36 37 39 42 44 46 51 55 59 61 63 69 71 73
91      11/04/2015,01 06 14 15 17 21 22 30 31 35 37 40 41 54 55 57 60 64 73 75
92      11/03/2015,02 05 08 10 12 13 17 18 21 34 39 40 44 46 56 57 64 72 73 75
93      11/02/2015,04 09 14 19 25 27 30 34 36 41 45 48 50 53 57 64 67 68 70 72
94      11/01/2015,01 03 04 05 10 22 23 25 33 34 41 42 43 51 52 54 57 59 62 74
95      10/31/2015,01 03 08 11 22 26 34 35 47 48 50 53 56 57 62 65 73 74 76 78
96      10/30/2015,04 11 19 23 24 25 36 39 40 41 44 50 51 54 55 58 66 68 77 78
97      10/29/2015,01 03 04 11 13 16 18 21 22 23 29 32 34 42 45 61 62 66 73 74
98      10/28/2015,04 08 11 23 30 32 37 39 41 44 45 47 48 50 56 59 67 71 74 80
99      10/27/2015,01 06 09 23 24 25 26 32 35 40 42 45 54 65 67 68 70 77 78 80
00      10/26/2015,04 06 08 12 13 22 23 32 33 36 37 38 40 46 53 56 59 66 71 78
01      10/25/2015,06 09 10 11 15 20 22 32 35 37 47 51 52 54 56 63 71 72 78 80
02      10/24/2015,01 02 16 19 21 35 39 41 47 48 50 53 55 59 63 67 68 76 78 79
03      10/23/2015,06 12 14 16 18 25 28 30 34 40 42 43 48 50 54 55 56 60 71 80
04      10/22/2015,01 03 09 11 13 20 29 34 35 37 38 39 40 43 48 52 55 64 67 76
05      10/21/2015,02 07 09 15 16 20 21 22 38 43 46 51 53 55 65 68 73 75 77 79
06      10/20/2015,01 03 11 15 17 18 19 25 34 35 36 43 47 49 54 56 62 69 72 76
07      10/19/2015,04 06 09 10 16 22 37 38 41 44 47 49 50 54 58 60 64 66 70 76
08      10/18/2015,02 04 09 11 13 15 16 17 25 28 40 42 50 52 58 59 60 68 72 73
09      10/17/2015,02 04 05 08 11 15 23 24 25 27 39 44 48 56 62 66 70 74 77 80
10      10/16/2015,07 12 16 17 19 20 23 24 28 29 31 34 37 45 50 60 65 70 78 80
11      10/15/2015,01 10 13 21 22 23 28 30 33 35 38 47 52 54 55 59 62 63 65 66
12      10/14/2015,06 15 17 20 21 22 33 34 37 41 42 45 46 53 54 56 66 69 71 79
```

```
13      10/13/2015,02 06 08 14 25 29 34 36 42 43 44 45 49 56 58 60 72 75 77 80
14      10/12/2015,03 06 08 11 14 19 23 24 33 42 52 54 56 64 66 68 69 78 79 80
15      10/11/2015,02 03 05 08 20 23 25 26 29 31 32 36 37 40 47 54 58 60 63 76
16      10/10/2015,01 04 05 07 15 16 19 31 42 46 47 53 57 58 59 67 68 73 74 79
17      10/09/2015,03 06 08 11 15 16 17 23 25 29 30 33 41 49 52 55 59 67 74 75
18      10/08/2015,08 11 12 14 17 18 24 28 29 30 31 38 43 51 53 62 64 65 75 78
19      10/07/2015,02 04 08 11 15 20 21 22 31 33 34 40 41 45 49 50 51 57 65 73
20      10/06/2015,01 09 16 17 19 21 34 37 40 49 52 54 57 58 60 63 71 75 78 80
21      10/05/2015,01 03 10 14 17 19 20 22 24 36 37 38 42 52 55 56 60 72 79 80
22      10/04/2015,01 08 17 20 21 22 24 33 39 45 50 52 57 58 63 69 72 76 79 80
23      10/03/2015,01 02 03 12 14 17 24 25 26 30 32 34 35 38 44 48 49 53 57 67
24      10/02/2015,13 16 20 23 25 37 43 44 46 49 51 54 56 57 58 62 64 66 69 73
25      10/01/2015,01 03 09 10 11 22 26 29 35 38 39 42 46 50 53 62 64 67 69 78
26      09/30/2015,01 05 06 07 11 18 19 23 24 26 29 32 35 36 37 50 77 78 79 80
27      09/29/2015,02 12 17 24 25 30 31 39 41 45 48 49 56 57 58 62 65 66 71 72
28      09/28/2015,04 09 11 12 23 25 27 30 32 34 35 45 46 54 55 68 70 73 74 76
29      09/27/2015,03 05 06 07 08 09 11 25 30 34 37 39 41 42 57 63 67 74 77 79
30      09/26/2015,03 09 12 15 24 25 28 31 33 37 40 44 46 47 48 58 59 65 68 79
31      09/25/2015,03 04 12 13 20 21 22 26 28 34 36 38 42 44 45 48 53 57 67 77
32      09/24/2015,01 02 04 10 13 14 15 16 21 31 32 35 42 46 57 62 72 74 77 79
33      09/23/2015,04 12 15 26 28 31 36 42 45 46 50 53 55 58 63 70 71 72 75 76
34      09/22/2015,07 08 10 15 19 23 25 27 29 32 33 45 49 53 58 60 69 71 75 79
35      09/21/2015,01 02 03 15 23 27 32 41 42 43 44 46 51 53 55 61 64 65 70 78
36      09/20/2015,02 04 05 07 10 13 14 19 21 23 28 31 39 46 48 51 54 59 63 64
37      09/19/2015,03 04 05 06 09 10 16 21 22 37 38 39 42 47 50 56 60 62 70 79
38      09/18/2015,10 12 15 19 21 25 27 31 33 34 37 38 42 45 47 52 56 69 71 73
39      09/17/2015,01 02 07 09 18 25 27 33 36 43 44 46 47 56 59 62 67 71 74 79
40      09/16/2015,01 09 15 21 23 25 35 47 50 52 57 59 60 61 63 64 67 70 74 78
41      09/15/2015,16 20 23 27 31 36 46 47 48 51 53 54 57 58 60 64 68 74 78 80
42      09/14/2015,01 02 03 04 06 10 22 28 32 39 42 49 51 52 66 68 71 75 79 80
43      09/13/2015,01 02 03 04 08 13 16 17 20 25 27 28 34 41 44 45 52 68 73 77
44      09/12/2015,06 13 15 16 20 25 31 35 39 43 44 45 47 56 57 60 61 65 74 76
45      09/11/2015,07 08 09 11 13 16 19 23 26 30 35 41 44 50 56 59 64 69 73 79
46      09/10/2015,04 12 17 31 32 37 40 43 44 45 47 57 60 62 65 69 70 72 77 79
47      09/09/2015,04 05 07 16 24 26 28 30 36 42 47 48 50 58 59 62 64 68 74 79
48      09/08/2015,01 02 15 16 22 23 24 26 29 35 41 42 43 49 50 53 62 67 79 80
49      09/07/2015,12 13 15 18 19 22 23 24 30 32 34 43 44 47 49 53 65 66 69 73
50      09/06/2015,03 10 14 18 19 21 26 36 38 39 46 49 50 52 61 62 64 66 73 74
51      09/05/2015,03 07 13 21 25 33 38 39 40 43 49 51 56 59 62 65 70 72 74 79
52      09/04/2015,03 05 06 09 12 13 22 27 31 44 50 51 57 58 60 61 62 73 75 77
53      09/03/2015,02 05 10 16 20 24 28 30 38 48 49 50 52 55 59 61 62 63 67 77
54      09/02/2015,03 04 11 21 23 25 32 33 34 36 46 47 58 59 65 69 71 74 76 78
55      09/01/2015,05 08 12 14 15 19 23 24 29 36 38 46 47 53 55 62 63 65 75 77
56      08/31/2015,03 04 11 12 17 24 25 27 32 37 39 43 44 46 51 54 55 59 64 68
57      08/30/2015,08 09 11 12 14 17 21 31 33 34 39 43 51 58 60 61 65 69 74 78
58      08/29/2015,07 12 21 22 26 29 30 37 39 40 42 49 52 55 56 60 68 72 75 78
59      08/28/2015,05 07 09 17 21 30 34 35 41 49 53 54 55 57 61 67 69 70 75 78
60      08/27/2015,04 06 10 11 19 26 32 34 36 40 41 43 54 55 59 64 72 75 76 77
61      08/26/2015,01 02 04 08 12 13 14 23 34 38 39 42 44 45 48 51 57 74 75 78
62      08/25/2015,03 07 10 11 13 15 22 25 26 30 31 32 33 36 40 43 55 58 64 75
63      08/24/2015,01 08 11 12 18 23 29 30 33 34 35 42 45 53 55 57 58 59 76 79
64      08/23/2015,01 17 18 20 21 30 34 37 40 50 60 61 63 66 67 68 71 72 78 80
65      08/22/2015,01 04 09 20 27 35 36 41 42 43 45 47 48 51 58 60 65 71 72 78
66      08/21/2015,09 11 12 16 26 27 29 42 47 48 52 53 54 59 63 66 69 75 77 79
```

```
67      08/20/2015,08 11 12 16 22 24 32 34 42 43 49 53 54 60 66 67 71 73 75 78
68      08/19/2015,05 07 08 09 10 12 18 24 26 35 40 46 49 50 61 62 70 71 74 75
69      08/18/2015,05 06 15 16 17 25 28 29 35 40 48 49 53 65 69 70 71 72 73 79
70      08/17/2015,04 08 09 14 17 20 24 31 33 34 36 37 40 41 51 63 67 71 72 79
71      08/16/2015,06 11 19 20 23 26 28 29 34 36 47 54 55 57 58 61 69 71 73 77
72      08/15/2015,03 10 11 23 29 31 37 42 47 50 55 56 58 65 66 67 68 75 76 80
73      08/14/2015,03 04 11 12 25 26 36 49 50 58 59 60 64 66 70 71 73 74 77 80
74      08/13/2015,05 08 10 14 17 21 23 26 42 44 46 56 57 59 65 70 71 73 76 79
75      08/12/2015,01 02 06 09 15 23 26 30 33 35 38 40 44 47 56 62 65 67 69 77
76      08/11/2015,02 04 06 07 12 15 19 20 22 23 25 26 27 37 43 45 52 57 58 64
77      08/10/2015,01 02 03 09 15 21 24 28 33 34 40 43 54 59 66 73 74 76 77 78
78      08/09/2015,07 09 13 18 24 28 34 35 36 43 46 51 58 65 66 68 69 73 76 78
79      08/08/2015,08 10 11 13 14 16 25 26 32 34 37 38 43 47 50 61 67 75 79 80
80      08/07/2015,01 07 16 28 29 34 35 38 43 44 45 49 52 61 63 66 73 75 77 80
81      08/06/2015,04 06 13 16 25 33 39 42 47 51 52 53 56 58 59 61 63 64 66 68
82      08/05/2015,03 17 22 25 26 32 33 37 38 42 44 45 46 48 49 53 55 57 60 66
83      08/04/2015,02 06 08 09 15 17 25 30 32 33 37 40 42 44 45 63 68 70 75 79
84      08/03/2015,01 09 10 15 18 19 22 26 27 30 38 41 42 47 53 58 59 66 73 74
85      08/02/2015,03 05 08 14 15 29 30 31 32 34 46 47 48 55 59 62 69 71 73 76
86      08/01/2015,01 02 10 12 17 20 21 22 28 29 33 36 39 42 45 47 52 63 66 79
87      07/31/2015,06 07 08 10 11 14 15 16 26 30 31 34 35 44 50 56 63 65 66 72
88      07/30/2015,01 04 06 12 16 17 19 21 22 23 25 33 41 47 49 54 58 67 69 78
89      07/29/2015,01 06 10 14 19 20 23 26 29 32 36 40 45 46 51 52 55 66 68 70
90      07/28/2015,01 03 08 11 12 13 15 18 24 25 26 28 32 33 36 38 43 55 71 76
91      07/27/2015,02 03 05 09 11 22 23 33 40 48 49 50 54 57 63 64 66 77 78 80
92      07/26/2015,04 05 07 11 15 17 18 20 24 31 34 44 46 49 50 62 64 66 67 68
93      07/25/2015,06 08 12 19 38 39 45 47 50 52 53 56 61 66 67 69 70 72 73 76
94      07/24/2015,04 05 06 21 24 30 31 38 44 47 52 60 67 69 70 74 75 78 79 80
95      07/23/2015,01 03 11 19 21 30 35 36 37 39 40 42 43 51 65 72 73 74 75 77
96      07/22/2015,03 05 11 15 26 27 34 39 49 51 52 56 61 62 67 72 75 76 77 80
97      07/21/2015,01 04 10 11 12 13 14 16 20 24 26 29 37 42 43 46 61 65 68 75
98      07/20/2015,14 19 26 27 28 29 31 33 35 36 38 41 42 45 49 51 55 59 60 70
99      07/19/2015,01 02 03 06 09 17 20 21 24 33 37 42 47 52 56 63 67 75 77 79
00      07/18/2015,01 02 04 05 06 08 09 10 21 27 30 31 32 43 60 70 72 73 77 79
01      07/17/2015,05 08 12 18 19 27 35 38 43 44 46 47 51 55 58 59 60 64 66 74
02      07/16/2015,02 10 11 12 25 35 36 39 41 44 45 51 52 54 66 70 73 75 78 80
03      07/15/2015,01 05 06 07 11 14 19 23 37 38 40 41 42 43 46 50 53 56 59 73
04      07/14/2015,02 04 11 13 17 19 22 24 27 39 40 41 43 49 63 69 71 74 76 78
05      07/13/2015,06 07 10 12 13 14 21 26 27 28 35 37 39 42 53 56 58 59 72 77
06      07/12/2015,06 07 08 12 13 16 28 30 31 39 41 43 47 51 52 55 58 59 60 63
07      07/11/2015,04 06 08 10 11 14 22 23 31 32 35 37 44 49 54 58 67 68 72 78
08      07/10/2015,06 09 10 13 25 27 30 31 35 36 37 40 41 44 45 50 56 63 71 78
09      07/09/2015,01 09 12 16 23 24 26 28 34 36 41 43 44 51 54 57 66 68 70 77
10      07/08/2015,01 02 08 12 14 16 17 20 24 29 33 38 43 44 48 53 59 62 67 72
11      07/07/2015,02 04 10 22 26 29 30 39 41 51 52 53 56 60 61 62 65 68 70 79
12      07/06/2015,03 04 08 15 16 21 25 27 28 40 51 56 58 59 62 67 74 76 78 79
13      07/05/2015,07 09 13 17 19 20 25 31 39 41 42 49 51 52 57 60 70 71 78 79
14      07/04/2015,04 06 09 12 13 25 29 30 33 40 44 46 47 55 62 69 71 73 75 78
15      07/03/2015,02 05 08 21 28 32 33 37 38 40 41 42 44 45 54 59 76 77 79 80
16      07/02/2015,05 06 07 08 13 16 23 24 26 30 31 41 43 47 51 58 61 64 68 69
17      07/01/2015,01 16 21 23 24 26 31 35 36 45 46 48 51 52 53 54 62 66 67 79
18      06/30/2015,03 07 08 09 16 20 27 30 35 51 53 55 58 60 62 65 66 74 77 80
19      06/29/2015,02 03 09 15 17 23 26 33 34 36 43 44 53 57 61 64 70 75 76 79
20      06/28/2015,06 10 17 21 23 33 35 36 37 38 39 43 45 49 58 63 65 66 75 79
```

```
06/27/2015,01 02 13 15 20 22 32 33 35 36 37 39 42 53 56 66 68 69 78 79
06/26/2015,02 03 09 25 27 28 33 34 38 42 47 49 50 56 59 60 62 63 64 75
06/25/2015,10 20 23 26 28 32 33 36 38 47 48 49 52 54 64 70 74 75 77 78
06/24/2015,03 04 05 09 13 15 26 27 32 40 42 49 51 53 56 59 60 66 76 78
06/23/2015,08 09 11 13 17 18 21 22 23 29 31 32 34 36 41 43 46 67 69 73
06/22/2015,01 07 14 18 26 28 35 37 38 39 41 42 44 47 56 61 62 70 76 80
06/21/2015,02 03 05 10 15 18 21 22 24 28 40 42 47 49 59 69 70 74 76 80
06/20/2015,03 10 15 17 21 23 27 28 29 30 32 34 35 58 65 66 68 71 74 76
06/19/2015,01 03 07 10 14 18 25 34 35 37 44 49 57 58 65 69 71 72 78 79
06/18/2015,04 06 07 08 15 19 23 32 36 43 44 46 54 62 64 66 67 69 75 80
06/17/2015,02 03 04 05 06 07 15 19 26 28 33 39 41 46 52 53 54 64 65 77
06/16/2015,03 04 05 10 11 17 26 29 39 41 48 49 53 56 58 64 68 72 73 79
06/15/2015,05 16 27 30 36 37 39 41 42 50 51 62 63 64 65 66 70 77 78 79
06/14/2015,01 05 06 07 11 12 21 28 33 41 50 54 57 59 64 69 73 74 75 77
06/13/2015,03 06 08 12 14 16 19 22 24 28 30 32 39 44 55 56 59 62 64 71
06/12/2015,04 05 06 07 09 12 13 21 25 31 32 37 47 50 54 59 60 66 70 73
06/11/2015,05 10 11 15 17 18 31 34 36 45 49 50 54 60 61 62 72 74 76 78
06/10/2015,03 04 05 06 14 17 21 23 26 31 42 44 48 49 54 62 65 74 76 79
06/09/2015,03 05 06 07 11 13 20 34 35 38 39 43 46 48 49 53 54 66 69 71
06/08/2015,18 20 21 23 24 25 31 32 33 38 51 52 55 58 60 67 68 73 75 79
06/07/2015,05 07 09 15 16 18 27 28 33 37 41 44 46 48 51 57 65 68 72 74
06/06/2015,03 04 07 08 13 17 21 25 26 27 31 36 51 54 64 66 67 75 76 78
06/05/2015,02 04 08 10 11 12 27 30 39 43 46 47 59 65 69 70 76 77 78 80
06/04/2015,03 04 13 18 22 28 30 31 42 43 45 54 57 62 67 70 72 73 75 79
06/03/2015,01 16 19 20 21 23 26 30 32 33 35 40 41 53 57 66 71 74 79 80
06/02/2015,06 08 14 15 18 20 22 28 29 31 33 37 43 52 57 71 72 74 76 80
06/01/2015,01 03 05 12 19 21 22 24 25 26 27 33 36 37 39 44 46 57 72 77
05/31/2015,01 02 03 17 20 25 28 29 36 37 38 47 51 55 59 62 64 66 73 76
05/30/2015,10 15 18 20 22 25 26 28 30 36 41 42 51 53 58 59 63 65 71 75
05/29/2015,01 02 05 06 08 25 33 37 41 43 44 48 49 52 54 58 65 66 74 76
05/28/2015,05 14 16 22 26 30 31 38 44 47 50 54 58 60 62 63 68 75 77 79
05/27/2015,01 02 12 23 26 28 29 32 34 35 38 42 45 47 48 52 55 56 70 78
05/26/2015,02 05 08 09 12 15 19 21 22 25 32 33 40 56 59 61 64 69 73 80
05/25/2015,03 06 11 12 16 18 21 22 31 32 33 44 49 51 55 59 70 71 79 80
05/24/2015,01 03 15 16 19 21 24 27 35 36 37 42 45 49 51 54 59 74 75 76
05/22/2015,03 04 06 09 12 18 22 23 24 28 36 42 43 44 45 48 63 64 75 77
05/21/2015,01 02 07 20 21 22 23 37 39 41 44 48 54 57 59 64 69 71 75 77
05/20/2015,04 05 09 14 16 18 25 29 32 40 43 46 47 50 52 58 59 71 75 76
05/19/2015,02 03 10 12 13 22 33 36 41 49 50 52 60 65 67 70 71 77 78 80
05/18/2015,02 09 10 14 15 18 20 26 36 37 39 40 49 50 53 62 70 76 77 80
05/17/2015,02 03 08 09 18 21 27 30 31 39 51 53 54 55 58 60 62 69 75 80
05/16/2015,02 10 15 16 23 24 28 29 33 36 37 48 50 56 60 64 68 75 77 80
05/15/2015,09 10 14 20 24 29 31 40 43 46 49 52 55 57 63 66 67 75 78 79
05/14/2015,01 06 13 21 30 32 33 35 36 38 42 46 52 53 55 58 66 68 75 77
05/13/2015,03 08 13 14 15 24 27 34 39 48 52 56 61 63 65 70 71 72 73 75
05/12/2015,04 14 19 21 29 30 36 48 49 54 57 58 59 61 63 64 65 66 69 79
05/11/2015,08 09 10 17 23 26 28 30 31 41 53 59 60 61 62 63 67 68 72 76
05/10/2015,08 12 15 16 17 19 24 25 26 27 32 55 60 64 68 69 72 74 79 80
05/09/2015,04 05 06 10 11 15 22 24 30 34 39 43 45 50 60 64 67 71 77 80
05/08/2015,02 03 04 12 25 27 28 32 37 38 42 44 48 52 55 57 67 68 71 74
05/07/2015,07 08 12 15 19 23 25 27 36 38 43 51 53 60 66 67 69 72 74 78
05/06/2015,05 11 13 20 21 22 27 35 38 39 41 44 50 56 57 58 60 64 70 76
05/05/2015,08 14 15 16 18 20 22 28 29 30 31 41 47 54 57 61 68 69 76 78
05/04/2015,02 05 08 10 11 13 17 27 28 42 44 57 61 63 67 68 73 76 77 80
```

```
75      05/03/2015,09 19 21 25 26 33 36 37 40 43 47 56 58 64 68 71 73 75 76 77
76      05/02/2015,02 03 12 14 15 27 30 37 43 50 52 53 63 66 67 72 74 77 78 80
77      05/01/2015,01 05 06 08 10 11 13 20 26 35 44 48 49 51 53 55 65 68 72 74
78      04/30/2015,06 08 16 20 24 25 26 33 34 36 37 41 49 53 54 55 63 67 72 77
79      04/29/2015,08 14 15 16 19 25 27 38 41 44 49 52 55 57 60 62 70 71 76 80
80      04/28/2015,04 07 22 23 27 33 37 40 41 42 44 50 53 54 60 62 67 72 76 79
81      04/27/2015,02 04 12 16 18 19 21 24 26 32 43 44 48 49 61 68 71 73 74 79
82      04/26/2015,01 08 12 15 16 18 19 24 33 37 38 47 48 58 62 63 64 67 68 73
83      04/25/2015,09 11 20 24 26 35 37 38 41 43 44 48 54 57 58 59 60 64 71 75
84      04/23/2015,01 03 06 09 11 14 17 19 24 26 28 30 33 41 44 54 58 62 66 73
85      04/22/2015,05 06 17 19 24 25 27 36 37 40 51 53 54 62 63 64 65 66 68 78
86      04/21/2015,02 13 15 28 29 32 36 42 44 49 53 55 57 58 61 62 63 64 72 78
87      04/20/2015,04 11 18 22 24 27 29 30 33 40 43 45 46 50 51 60 61 77 78 79
88      04/19/2015,01 04 08 12 14 16 21 30 33 39 40 45 47 53 55 57 59 62 66 77
89      04/18/2015,06 11 12 17 18 25 32 35 36 37 39 43 44 45 54 63 65 70 74 80
90      04/17/2015,02 06 18 19 24 25 28 29 30 38 41 43 49 52 57 60 61 65 66 77
91      04/16/2015,03 08 12 15 23 26 30 32 40 41 43 44 53 55 57 64 65 69 74 80
92      04/15/2015,01 13 15 22 32 38 39 41 42 54 55 56 59 64 66 68 75 78 79 80
93      04/14/2015,01 07 08 09 15 17 20 21 26 38 47 52 56 57 58 65 67 70 72 80
94      04/13/2015,01 03 04 17 19 23 24 25 26 29 46 48 59 61 67 69 75 77 79 80
95      04/12/2015,01 02 08 16 18 19 20 21 25 29 32 33 41 42 53 60 65 71 77 80
96      04/11/2015,01 07 08 10 13 19 23 27 28 31 34 37 39 45 46 50 53 57 58 64
97      04/10/2015,08 11 12 19 20 32 33 35 50 53 62 63 68 69 73 74 76 77 78 79
98      04/09/2015,01 03 04 05 07 08 11 14 23 28 29 32 42 44 58 67 72 73 76 78
99      04/08/2015,02 03 04 05 08 14 17 18 20 26 27 30 34 35 49 53 60 63 77 79
00      04/07/2015,01 02 08 15 18 20 21 32 40 42 44 55 59 67 71 72 74 75 77 79
01      04/06/2015,07 14 16 17 23 24 25 28 34 43 55 57 62 64 66 67 68 73 74 75
02      04/05/2015,09 26 28 29 35 42 43 44 47 50 51 52 53 55 57 62 65 73 76 77
03      04/04/2015,02 04 07 10 12 14 16 22 28 32 33 37 41 49 50 53 54 59 66 67
04      04/03/2015,01 14 20 21 22 23 28 31 34 38 43 44 49 50 59 63 64 67 75 78
05      04/02/2015,02 04 07 15 17 18 20 25 26 29 32 34 35 45 51 57 59 63 72 77
06      04/01/2015,02 12 13 17 18 22 27 33 34 37 40 41 45 50 60 64 66 71 73 74
07      03/31/2015,01 02 04 08 10 16 17 23 24 31 34 39 43 44 46 52 58 66 78 80
08      03/30/2015,10 12 16 22 24 30 31 37 38 39 40 41 45 46 50 55 56 63 68 76
09      03/29/2015,02 08 16 20 25 30 32 35 36 39 49 50 52 54 55 57 67 68 74 77
10      03/28/2015,02 05 06 09 10 11 14 15 18 21 25 27 29 34 38 39 53 58 62 74
11      03/27/2015,04 06 09 10 15 16 17 19 20 34 38 39 48 53 56 67 70 76 77 79
12      03/26/2015,04 07 12 17 28 29 30 39 40 44 49 54 55 56 64 69 70 71 72 75
13      03/25/2015,01 03 08 09 10 19 22 26 30 38 41 45 47 48 63 66 68 72 74 79
14      03/24/2015,01 10 14 17 20 25 28 36 42 53 55 57 58 60 68 69 73 75 78 80
15      03/23/2015,03 04 05 18 20 27 29 33 38 45 46 50 52 57 65 66 67 69 73 76
16      03/22/2015,03 10 13 14 19 21 34 36 38 41 42 44 45 54 57 61 62 64 70 71
17      03/21/2015,04 08 14 15 22 23 32 36 40 51 55 59 61 63 65 71 73 76 77 79
18      03/20/2015,04 05 06 10 11 36 37 39 41 42 48 59 63 64 65 66 68 70 74 75
19      03/19/2015,04 06 08 09 10 13 14 21 24 27 35 37 38 42 44 46 49 53 58 76
20      03/18/2015,01 06 08 09 13 17 22 23 24 28 29 30 40 48 50 59 62 69 76 78
21      03/17/2015,17 19 21 22 25 30 31 32 35 36 37 38 40 53 60 67 74 75 77 80
22      03/16/2015,01 06 10 11 15 20 21 23 33 36 46 48 49 51 57 61 66 68 72 75
23      03/15/2015,06 14 18 19 21 25 28 30 31 32 34 35 36 39 45 46 49 60 67 78
24      03/14/2015,01 03 07 09 17 19 22 23 24 28 32 35 53 54 58 61 62 63 75 79
25      03/13/2015,05 16 23 24 28 33 39 40 42 48 50 52 54 56 58 60 64 69 71 73
26      03/12/2015,02 14 18 20 23 24 30 34 37 43 44 52 59 61 68 69 72 74 78 79
27      03/11/2015,05 11 12 14 16 17 21 22 24 29 34 36 38 45 56 60 69 70 73 78
28      03/10/2015,07 08 09 12 18 19 20 35 38 39 46 47 48 49 55 62 65 69 70 71
```

```
03/09/2015,12 13 17 20 24 34 37 39 46 54 57 62 69 71 72 74 77 78 79 80
03/08/2015,06 09 11 12 13 19 21 24 26 31 36 42 43 55 56 57 63 71 78 79
03/07/2015,03 04 05 12 17 18 19 21 23 26 32 33 38 41 43 44 46 53 63 79
03/06/2015,01 10 12 22 25 34 39 43 45 46 58 59 61 63 69 73 74 77 78 79
03/05/2015,04 06 11 13 14 21 23 24 28 35 36 38 40 44 48 54 56 58 61 67
03/04/2015,01 02 06 07 08 11 15 17 22 31 56 57 58 60 65 66 70 71 73 77
03/03/2015,03 06 15 16 20 24 26 27 29 30 41 42 44 51 56 57 59 63 69 71
03/02/2015,03 05 06 07 08 10 12 17 28 33 37 40 42 45 54 55 64 70 73 79
03/01/2015,02 04 07 11 13 20 21 23 27 29 32 36 42 50 51 56 60 63 76 79
02/28/2015,02 05 08 11 21 23 25 29 32 39 41 46 50 56 57 62 66 68 71 80
02/27/2015,13 16 17 19 20 26 29 36 38 42 44 50 53 54 55 58 61 74 75 77
02/26/2015,01 04 12 13 15 26 27 28 29 30 38 39 40 44 47 60 62 70 72 73
02/25/2015,05 20 22 25 26 28 30 33 34 35 36 37 46 54 60 61 65 70 75 78
02/24/2015,05 06 08 16 18 21 28 37 43 52 55 57 59 61 67 71 72 73 79 80
02/23/2015,13 16 19 25 27 30 34 35 38 40 41 44 49 58 61 73 74 76 78 79
02/22/2015,07 08 10 12 24 25 29 34 35 36 39 40 43 45 48 55 65 68 69 78
02/21/2015,01 02 03 04 06 07 09 12 21 35 37 43 46 47 56 60 65 74 76 80
02/20/2015,11 14 18 24 25 29 33 35 37 41 47 50 54 57 58 59 65 70 73 76
02/19/2015,11 17 20 22 23 28 30 33 34 37 40 41 45 55 57 61 64 65 68 70
02/18/2015,02 03 05 10 13 15 17 26 38 39 41 43 45 53 54 58 63 65 66 73
02/17/2015,01 05 10 12 15 23 31 36 40 41 45 51 53 54 60 61 64 66 72 79
02/16/2015,03 06 09 10 13 18 19 25 27 32 43 44 50 53 56 62 66 74 76 80
02/15/2015,01 06 13 19 21 23 27 31 34 45 48 53 57 62 63 67 76 78 79 80
02/14/2015,03 04 12 22 28 29 31 41 42 46 50 54 55 56 57 62 66 67 71 74
02/13/2015,05 09 10 13 17 19 21 23 25 27 30 37 44 46 55 56 62 68 73 77
02/12/2015,03 06 13 18 21 22 29 31 36 42 46 53 55 60 63 66 68 74 75 79
02/11/2015,01 02 13 17 23 24 26 34 36 40 43 50 51 53 57 60 61 73 74 77
02/10/2015,01 02 05 07 11 20 28 40 45 46 52 56 58 60 63 64 68 74 75 77
02/09/2015,01 08 10 11 13 14 20 29 30 33 37 39 44 47 49 53 56 57 78 79
02/08/2015,04 06 07 08 15 21 25 28 32 39 45 46 53 55 59 61 63 69 75 80
02/07/2015,04 09 10 11 12 17 19 27 30 37 40 43 44 52 53 54 57 66 68 79
02/06/2015,01 04 05 13 19 20 21 22 24 47 49 50 63 64 65 66 67 69 73 79
02/05/2015,02 05 11 16 21 23 27 32 38 46 48 57 59 60 64 70 73 76 79 80
02/04/2015,01 04 08 16 21 23 25 30 36 39 40 42 45 56 58 61 66 76 79 80
02/03/2015,07 10 15 25 26 27 30 32 39 40 44 49 60 67 68 70 72 73 75 76
02/02/2015,05 10 16 18 21 23 25 28 35 36 39 41 44 51 62 63 65 68 72 77
02/01/2015,01 02 04 06 12 13 17 20 21 22 28 35 45 49 59 64 66 67 70 78
01/31/2015,04 07 15 20 21 28 32 39 43 48 50 53 54 61 63 65 68 70 71 73
01/30/2015,03 05 08 12 13 15 18 23 26 29 32 42 56 58 59 61 69 74 75 77
01/29/2015,03 08 11 16 18 20 34 36 37 45 46 48 51 55 57 60 64 66 75 78
01/28/2015,04 11 15 17 28 34 39 40 41 46 50 53 58 61 62 65 69 73 76 78
01/27/2015,02 03 11 12 13 16 19 25 36 37 39 40 43 46 48 52 56 58 77 80
01/26/2015,01 02 03 10 12 14 15 23 33 47 51 56 60 61 69 71 75 77 78 79
01/25/2015,07 11 13 18 21 22 23 26 27 28 31 34 47 49 54 65 72 74 76 77
01/24/2015,06 07 08 11 16 18 26 31 33 36 40 43 49 52 57 59 60 69 73 74
01/23/2015,17 20 25 28 29 34 35 41 43 52 54 60 63 66 67 72 73 74 76 77
01/22/2015,03 07 11 12 16 17 24 25 34 38 52 55 57 60 65 66 69 70 76 77
01/21/2015,01 05 06 09 17 26 30 38 40 43 51 52 54 59 62 71 73 76 78 80
01/20/2015,12 13 26 32 35 37 39 41 48 49 52 54 55 56 64 65 70 71 72 77
01/19/2015,03 09 10 12 18 24 26 33 43 45 49 56 57 59 60 62 71 75 79 80
01/18/2015,02 04 13 17 20 25 31 34 38 42 46 50 56 57 65 68 71 74 75 77
01/17/2015,01 04 09 10 20 24 25 34 46 51 60 62 63 64 69 70 71 74 75 77
01/16/2015,10 16 25 28 30 38 40 41 43 45 46 47 49 55 57 62 63 69 72 79
01/15/2015,01 03 11 12 13 14 20 21 23 25 29 31 40 41 46 52 58 68 70 73
```

```
01/14/2015,03 07 11 12 14 18 26 29 30 32 35 37 42 43 51 53 56 57 58 63
01/13/2015,04 13 18 21 23 31 38 42 44 45 46 51 52 57 58 67 72 74 75 79
01/12/2015,01 07 09 12 16 17 25 33 36 37 39 45 47 51 59 61 63 66 72 75
01/11/2015,01 13 17 19 23 31 38 39 48 55 57 59 61 64 69 70 71 73 78 80
01/10/2015,15 16 20 21 22 31 33 35 37 39 40 42 43 55 59 61 63 70 72 80
01/09/2015,02 03 04 06 08 11 17 20 21 25 27 30 38 46 53 61 66 67 78 80
01/08/2015,02 14 20 26 28 30 35 37 41 46 51 53 55 59 60 61 68 72 73 76
01/07/2015,02 09 14 16 18 27 34 35 38 46 49 50 52 53 55 65 71 72 75 80
01/06/2015,01 03 04 09 11 14 17 18 24 27 34 40 41 47 52 57 60 64 66 70
01/05/2015,09 12 21 23 27 30 37 41 44 46 58 59 69 70 71 72 74 75 76 78
01/04/2015,09 10 12 20 30 31 35 39 44 45 46 54 55 58 64 69 72 76 79 80
01/03/2015,03 07 16 19 20 22 24 29 34 40 45 51 53 57 59 63 69 70 74 78
01/02/2015,02 04 05 06 15 20 22 23 29 31 37 47 49 50 53 55 59 60 70 77
01/01/2015,04 05 06 09 10 14 20 24 36 37 39 46 47 60 62 67 68 69 73 78
12/31/2014,04 06 07 15 20 21 22 38 48 49 50 53 57 60 61 62 63 68 71 73
12/30/2014,06 09 10 21 23 34 37 44 49 51 53 58 62 63 65 66 73 75 76 78
12/29/2014,03 04 10 18 20 26 27 28 31 35 39 44 51 58 61 66 71 72 73 80
12/28/2014,08 15 24 25 29 30 36 41 44 48 59 60 63 68 70 72 76 77 78 80
12/27/2014,01 06 11 12 13 19 29 32 34 36 38 56 58 59 64 67 70 71 76 78
12/26/2014,05 11 12 19 20 24 25 27 29 34 36 38 43 52 54 57 71 73 76 77
12/25/2014,09 10 13 14 17 24 29 32 36 38 41 47 48 54 57 60 61 63 70 78
12/24/2014,02 06 08 12 16 23 24 29 30 33 36 37 53 57 64 66 67 70 78 80
12/23/2014,01 06 15 17 20 23 25 32 38 41 42 50 54 55 62 68 70 74 75 80
12/22/2014,01 02 05 11 22 23 25 27 31 45 46 51 56 58 61 63 69 71 75 80
12/21/2014,03 06 07 08 09 10 15 24 26 28 30 31 39 42 53 57 62 67 69 78
12/20/2014,01 04 05 07 08 18 19 22 26 28 41 46 47 50 56 61 65 67 76 78
12/19/2014,01 12 14 16 25 26 31 36 39 41 45 47 48 53 57 58 62 72 73 75
12/18/2014,05 06 09 12 14 15 17 28 34 36 44 50 52 57 58 62 65 69 71 80
12/17/2014,01 08 11 12 15 16 28 36 38 44 47 50 58 59 60 65 75 76 79 80
12/16/2014,04 06 14 19 20 27 28 29 33 41 43 46 47 51 59 61 62 69 73 78
12/15/2014,05 13 18 20 22 25 31 33 35 39 41 47 49 50 54 60 62 68 71 77
12/14/2014,01 07 15 25 26 27 28 31 32 39 40 46 47 52 58 65 67 71 75 79
12/13/2014,09 13 16 18 20 23 26 27 34 35 38 40 51 58 60 63 68 73 76 80
12/12/2014,01 03 10 20 27 31 40 44 48 51 52 56 61 62 64 65 67 69 70 71
12/11/2014,07 10 14 17 23 25 26 29 30 37 38 41 45 48 51 54 59 63 65 68
12/10/2014,05 14 16 27 30 31 33 38 40 43 45 46 54 55 60 64 69 73 75 76
12/09/2014,01 05 09 19 24 27 42 44 49 52 55 61 62 65 66 67 70 74 77 80
12/08/2014,04 11 18 23 25 30 38 40 42 43 47 48 51 55 57 62 63 64 70 71
12/07/2014,04 09 14 22 27 29 30 32 35 40 47 50 51 53 54 62 64 68 72 75
12/06/2014,03 07 11 12 16 20 21 22 28 34 39 40 45 52 56 61 63 67 70 72
12/05/2014,05 08 15 18 20 25 26 27 30 33 38 47 50 51 59 60 68 70 74 76
12/04/2014,01 04 11 13 26 27 29 31 34 35 39 41 43 45 49 50 52 69 71 77
12/03/2014,08 12 22 27 33 35 36 37 39 48 51 52 55 56 67 68 69 73 78 80
12/02/2014,01 03 04 05 07 09 12 30 36 40 42 52 53 58 61 66 69 71 72 73
12/01/2014,07 09 11 12 17 28 30 36 38 42 44 46 63 64 69 71 72 74 77 80
11/30/2014,06 16 19 20 21 24 25 35 41 44 46 48 49 56 57 59 68 71 74 80
11/29/2014,09 10 11 12 18 25 27 31 34 43 50 51 52 54 59 63 66 68 70 77
11/28/2014,04 07 08 11 14 22 26 35 37 38 41 44 45 52 56 62 68 77 78 79
11/27/2014,04 05 07 09 20 21 24 25 26 29 31 32 41 50 59 60 61 64 70 77
11/26/2014,02 03 06 09 12 17 25 26 27 32 35 42 44 48 60 67 68 71 73 76
11/25/2014,01 06 14 17 19 29 33 41 44 45 46 49 52 55 56 57 59 68 73 77
11/24/2014,01 06 10 13 18 21 27 45 46 52 57 60 62 64 69 70 75 76 78 80
11/23/2014,03 07 13 19 20 21 23 24 33 36 38 39 47 48 49 54 63 71 72 77
11/22/2014,01 06 08 09 17 19 24 26 27 30 32 40 50 53 66 67 68 71 72 79
```

```
11/21/2014,06 07 15 20 24 29 30 33 40 44 51 55 58 59 63 66 68 72 73 75
11/20/2014,01 05 09 12 14 25 30 31 32 33 37 39 45 51 57 60 61 64 68 76
11/19/2014,02 04 19 21 30 34 35 37 38 39 41 46 51 56 58 60 66 70 73 75
11/18/2014,01 03 04 08 14 20 24 27 29 31 32 34 39 47 48 50 56 65 71 74
11/17/2014,11 12 18 19 21 30 31 35 37 39 44 45 49 50 56 70 73 74 75 80
11/16/2014,01 07 14 16 17 21 24 25 28 29 32 38 48 51 54 57 67 70 74 76
11/15/2014,02 05 08 14 21 26 28 32 35 36 39 46 47 51 55 66 67 74 78 80
11/14/2014,08 14 17 18 20 21 28 31 43 45 49 52 57 61 64 66 70 72 79 80
11/13/2014,01 05 06 08 25 36 42 44 52 54 56 57 59 61 64 66 68 72 73 77
11/12/2014,05 08 10 12 15 29 31 40 47 50 53 58 61 64 67 69 77 78 79 80
11/11/2014,01 05 06 14 17 21 24 30 32 41 46 50 52 53 61 67 69 71 73 78
11/10/2014,04 12 16 18 20 23 24 26 30 31 33 40 44 45 47 58 65 66 71 74
11/09/2014,05 06 14 16 17 22 24 26 32 34 35 37 47 51 53 57 69 71 74 76
11/08/2014,02 09 14 16 27 34 45 47 48 49 51 53 54 57 69 71 72 73 74 75
11/07/2014,02 04 10 13 14 19 21 23 26 30 32 48 51 53 54 59 61 69 73 74
11/06/2014,05 08 11 14 16 20 22 24 30 31 34 39 44 48 54 61 63 71 77 80
11/05/2014,04 06 08 09 17 19 23 28 30 32 37 38 39 51 52 56 65 67 70 73
11/04/2014,07 08 11 16 18 20 22 26 32 39 43 46 48 58 59 63 64 69 73 76
11/03/2014,01 05 10 12 18 20 26 30 35 36 48 51 53 55 56 61 67 69 71 79
11/02/2014,01 04 10 12 13 17 19 23 37 42 46 48 57 60 63 64 65 67 72 75
11/01/2014,03 04 05 07 13 18 19 20 22 24 31 35 48 50 53 55 65 68 70 76
10/31/2014,02 19 21 27 28 32 33 47 56 60 61 63 64 68 69 70 71 72 75 77
10/30/2014,02 04 05 06 15 20 21 34 39 43 44 47 57 59 65 67 68 71 73 79
10/29/2014,01 05 08 11 18 20 27 28 35 44 45 47 51 53 54 55 63 67 69 75
10/28/2014,02 03 05 06 08 10 19 22 26 30 34 37 40 42 44 45 46 58 61 71
10/27/2014,02 04 11 18 22 27 33 37 38 39 42 43 47 49 51 55 56 65 66 77
10/26/2014,02 07 16 17 18 19 26 27 28 32 35 36 44 47 60 61 68 71 76 79
10/25/2014,02 04 09 12 17 18 20 34 36 38 39 42 49 51 56 59 60 70 73 77
10/24/2014,06 09 10 12 14 15 16 17 18 19 27 35 37 48 57 67 74 75 76 77
10/23/2014,02 10 13 19 28 29 34 37 39 43 47 49 50 52 54 57 60 69 74 80
10/22/2014,07 11 12 16 17 18 23 28 33 36 39 40 54 59 61 66 69 71 73 80
10/21/2014,02 06 13 16 18 27 28 32 39 41 44 49 52 57 61 62 68 69 72 78
10/20/2014,01 05 19 20 24 29 31 33 35 38 40 41 46 52 55 60 68 72 76 79
10/19/2014,03 13 14 16 17 18 19 26 33 38 40 42 55 59 61 63 69 72 79 80
10/18/2014,03 06 11 13 20 21 24 28 33 38 39 41 43 44 59 65 68 69 76 78
10/17/2014,03 04 18 21 22 26 27 31 38 50 51 53 54 56 57 58 59 71 73 78
10/16/2014,03 04 06 08 17 30 31 32 33 36 37 46 47 48 56 63 64 65 71 77
10/15/2014,03 06 09 10 14 18 20 24 28 30 44 46 48 50 58 60 64 68 77 80
10/14/2014,06 07 09 14 22 23 29 36 42 43 45 47 48 52 54 60 67 68 71 77
10/13/2014,03 06 07 08 13 17 20 24 25 27 28 54 58 59 61 62 68 69 72 73
10/12/2014,01 03 16 21 22 23 25 27 33 35 40 41 43 53 54 59 60 63 68 73
10/11/2014,08 09 10 13 16 19 23 28 34 35 37 49 50 55 56 62 65 66 70 78
10/10/2014,05 08 16 23 25 26 32 41 45 46 53 54 60 62 63 67 70 74 78 80
10/09/2014,02 06 10 11 20 23 36 40 42 44 46 49 57 58 65 70 73 76 78 80
10/08/2014,03 05 09 21 25 27 29 30 36 41 43 45 48 51 58 62 65 69 72 76
10/07/2014,02 10 12 14 15 20 23 36 46 50 51 56 58 59 62 64 67 68 74 80
10/06/2014,01 08 09 10 18 19 38 40 45 46 50 53 59 63 64 68 69 70 73 80
10/05/2014,02 05 14 15 16 17 22 33 38 40 45 48 53 56 61 64 67 70 76 78
10/04/2014,08 13 15 20 26 31 33 41 42 45 46 47 48 51 55 60 61 63 66 67
10/03/2014,12 16 20 28 31 32 33 34 36 48 53 55 63 66 68 70 73 74 77 80
10/02/2014,03 11 12 15 17 24 31 32 38 39 43 44 47 53 55 59 62 69 72 80
10/01/2014,02 13 16 18 20 28 32 35 38 42 46 53 55 61 62 67 73 74 76 78
09/30/2014,14 15 18 26 34 40 42 43 49 51 56 57 60 61 66 68 70 72 76 80
09/29/2014,03 04 08 10 17 31 35 37 40 45 55 58 59 60 62 65 71 72 73 80
```

```
91      09/28/2014,01 02 03 10 14 16 21 27 29 31 39 44 53 56 60 64 65 66 67 77
92      09/27/2014,03 15 17 19 23 28 29 31 34 37 39 54 58 61 68 69 71 75 76 78
93      09/26/2014,07 09 10 16 17 22 24 25 28 33 35 38 41 55 60 62 68 73 74 78
94      09/25/2014,04 08 12 14 22 23 24 25 28 32 35 36 41 44 48 49 53 54 66 79
95      09/24/2014,03 04 19 20 21 23 31 32 33 34 35 36 37 42 50 51 55 57 59 62
96      09/23/2014,01 02 03 06 12 15 21 22 27 29 38 40 41 42 51 57 58 61 69 76
97      09/22/2014,03 04 08 11 24 25 30 32 33 35 42 45 46 50 52 64 67 70 75 79
98      09/21/2014,04 21 22 32 35 36 37 38 39 40 42 53 57 58 59 64 65 68 69 73
99      09/20/2014,03 04 06 08 15 18 24 31 32 42 47 48 52 55 58 60 65 72 73 79
00      09/19/2014,02 06 17 18 21 24 30 31 38 48 49 54 58 61 63 64 65 75 77 79
01      09/18/2014,04 07 10 13 14 15 17 29 32 33 35 37 41 42 48 49 50 66 68 70
02      09/17/2014,05 09 10 12 13 14 16 17 18 23 25 29 32 38 43 44 67 69 72 80
03      09/16/2014,01 02 07 12 13 16 20 21 29 30 31 32 42 45 46 66 68 69 71 77
04      09/15/2014,01 02 03 05 07 17 22 26 33 34 35 39 44 45 49 52 55 71 77 80
05      09/14/2014,04 07 15 16 17 25 26 27 39 42 43 46 47 53 62 67 69 73 76 78
06      09/13/2014,05 06 13 14 15 20 22 26 28 30 38 41 43 56 58 63 65 69 71 72
07      09/12/2014,11 14 16 23 27 28 31 37 40 42 45 47 49 52 55 56 62 68 72 80
08      09/11/2014,01 06 17 20 22 24 25 34 36 37 41 56 60 61 64 69 71 73 76 77
09      09/10/2014,03 05 08 10 13 14 16 24 35 38 42 46 48 49 54 58 67 70 72 73
10      09/09/2014,03 06 08 10 11 12 20 25 32 33 36 38 41 45 46 51 64 67 69 78
11      09/08/2014,05 08 09 12 18 22 31 34 37 42 50 58 64 66 67 71 75 76 79 80
12      09/07/2014,05 12 16 31 33 38 39 42 45 49 55 56 57 62 63 65 68 76 77 78
13      09/06/2014,02 04 05 06 09 17 19 22 30 34 36 41 48 53 54 55 59 71 75 78
14      09/05/2014,03 10 11 24 26 27 29 32 39 45 47 51 66 70 72 74 75 76 78 79
15      09/04/2014,06 08 13 18 24 27 28 30 32 35 37 42 44 49 57 61 68 76 79 80
16      09/03/2014,11 14 15 16 25 26 28 31 37 41 47 48 49 54 60 65 68 70 74 80
17      09/02/2014,14 16 17 19 24 26 28 34 35 36 37 38 45 46 56 58 59 61 62 74
18      09/01/2014,08 10 11 12 13 18 25 32 34 35 37 42 43 44 46 49 62 64 69 76
19      08/31/2014,04 16 17 19 20 22 38 47 50 51 52 53 59 64 65 71 72 75 79 80
20      08/30/2014,06 09 14 19 21 24 30 37 38 39 50 51 56 60 65 66 67 71 74 80
21      08/29/2014,03 05 11 17 18 22 23 25 27 32 35 36 41 42 45 46 52 62 76 79
22      08/28/2014,03 04 09 11 16 17 19 20 24 33 40 43 45 49 50 54 68 69 71 74
23      08/27/2014,02 04 19 22 24 39 42 46 47 50 51 58 59 60 61 63 66 69 74 75
24      08/26/2014,05 07 08 10 20 26 27 29 30 32 34 36 38 46 47 50 53 66 72 77
25      08/25/2014,01 03 14 24 25 26 29 33 37 41 45 48 50 51 56 58 66 71 72 78
26      08/24/2014,05 06 08 10 18 19 22 23 26 31 36 45 46 47 62 66 71 74 75 79
27      08/23/2014,01 08 10 13 17 20 26 34 40 48 50 54 57 58 59 65 69 77 78 80
28      08/22/2014,01 10 11 15 16 21 31 39 45 47 50 52 53 55 57 58 59 69 70 71
29      08/21/2014,04 05 09 12 15 24 28 29 32 35 37 38 40 57 60 61 66 67 68 74
30      08/20/2014,01 05 07 08 09 10 13 14 24 36 39 41 44 48 55 63 68 76 78 80
31      08/19/2014,10 13 16 20 21 22 23 28 32 34 36 44 46 61 66 69 71 74 79 80
32      08/18/2014,06 18 19 23 24 27 28 34 42 44 45 52 54 58 60 61 73 77 79 80
33      08/17/2014,03 11 18 22 27 29 31 32 35 37 38 39 41 45 52 58 59 64 73 75
34      08/16/2014,01 07 08 21 23 26 28 31 36 37 39 42 44 54 59 61 62 63 77 78
35      08/15/2014,09 13 19 20 22 23 27 28 30 33 35 39 49 51 59 60 65 66 70 73
36      08/14/2014,03 04 08 21 26 27 32 33 34 35 36 37 43 44 51 53 61 65 75 79
37      08/13/2014,04 07 12 13 14 19 23 26 38 41 43 45 57 61 62 66 68 71 73 79
38      08/12/2014,05 09 12 18 20 23 28 32 34 38 41 45 46 49 50 51 53 59 69 77
39      08/11/2014,01 06 12 13 17 22 25 26 31 34 36 37 43 44 48 61 62 63 71 80
40      08/10/2014,01 04 10 13 14 17 22 25 33 34 36 37 44 51 53 55 63 64 69 74
41      08/09/2014,01 04 06 08 10 16 23 26 28 29 34 35 50 54 62 66 70 71 74 75
42      08/08/2014,05 07 09 12 18 20 30 31 33 34 39 48 53 54 59 61 67 68 78 80
43      08/07/2014,01 05 07 14 20 23 26 31 36 42 48 51 53 55 61 63 65 71 75 76
44      08/06/2014,03 25 26 30 32 34 40 45 46 48 50 51 58 59 63 68 72 73 74 77
```

```
08/05/2014,07 09 10 11 16 19 24 32 47 49 50 51 52 54 70 72 73 74 75 77
08/04/2014,10 13 14 18 25 31 32 37 39 41 44 47 51 52 54 57 60 63 71 73
08/03/2014,04 13 15 22 27 35 40 44 47 49 51 64 66 68 69 70 73 77 78 80
08/02/2014,03 06 08 12 13 29 31 33 34 39 40 43 46 54 58 61 66 68 69 71
08/01/2014,02 08 11 15 18 20 24 28 37 41 42 46 57 58 59 65 66 68 74 75
07/31/2014,05 06 08 15 18 20 22 29 41 42 45 47 51 53 55 56 62 64 68 80
07/30/2014,02 03 10 11 12 14 18 20 26 30 31 32 41 43 58 63 66 67 69 78
07/29/2014,02 05 06 08 09 20 22 27 28 37 40 41 42 51 58 64 66 71 73 80
07/28/2014,05 08 12 15 19 23 25 27 29 37 44 45 51 56 61 62 64 68 70 75
07/27/2014,08 10 15 19 20 21 24 30 33 40 45 48 54 55 57 66 73 74 75 77
07/26/2014,08 10 12 15 16 20 21 29 30 37 44 45 57 58 59 61 67 76 77 80
07/25/2014,07 21 24 25 27 29 30 37 38 40 46 52 54 56 58 59 69 70 72 76
07/24/2014,03 08 11 17 31 35 36 40 43 49 51 61 64 67 68 72 75 77 79 80
07/23/2014,02 04 09 10 12 16 17 25 37 40 41 44 52 58 59 60 61 62 64 72
07/22/2014,02 04 05 16 18 19 20 22 23 27 28 33 42 54 56 60 61 69 78 80
07/21/2014,02 04 09 18 22 26 28 32 33 37 38 39 56 57 62 65 69 76 77 80
07/20/2014,08 23 25 26 34 36 39 41 45 46 47 54 56 57 61 62 63 66 71 73
07/19/2014,02 03 04 13 14 36 38 39 41 42 44 48 51 57 60 64 68 71 73 80
07/18/2014,03 07 19 26 29 30 33 36 41 43 48 49 51 55 57 58 60 70 77 79
07/17/2014,02 08 15 16 17 19 20 23 27 28 30 36 40 50 51 53 62 66 72 74
07/16/2014,02 04 09 11 15 17 29 32 34 41 43 46 47 52 59 63 66 74 75 77
07/15/2014,04 14 16 17 18 30 37 38 42 45 49 50 57 58 62 67 69 70 71 79
07/14/2014,02 10 19 22 25 28 29 32 35 36 43 47 50 51 52 55 59 74 75 77
07/13/2014,03 04 08 10 18 21 25 29 31 32 36 38 43 45 46 48 55 66 69 75
07/12/2014,03 04 05 07 08 09 20 26 30 32 38 41 44 46 55 62 65 67 68 70
07/11/2014,02 06 09 10 17 23 26 27 28 34 35 40 41 44 46 47 53 63 65 73
07/10/2014,03 07 12 14 15 17 30 32 33 35 42 45 55 56 59 60 62 72 75 79
07/09/2014,02 05 14 24 30 31 33 39 41 45 48 52 54 55 61 64 66 74 77 80
07/08/2014,01 05 06 13 20 23 25 29 35 39 43 48 53 55 56 60 62 69 75 78
07/07/2014,04 05 07 10 14 16 18 22 24 25 29 38 44 46 48 52 53 56 64 79
07/06/2014,03 06 11 12 13 16 17 21 26 29 51 55 57 60 61 63 66 67 69 73
07/05/2014,03 05 06 09 10 13 14 22 24 26 28 29 37 45 62 66 77 78 79 80
07/04/2014,02 04 05 14 15 16 22 31 36 37 39 43 45 52 60 62 63 69 70 74
07/03/2014,05 14 18 19 21 26 28 30 34 35 36 40 48 50 51 57 66 69 76 78
07/02/2014,01 04 05 07 09 13 14 18 23 29 35 38 39 42 48 57 61 64 66 68
07/01/2014,04 13 15 24 32 34 37 40 42 52 55 62 64 65 67 69 70 72 75 78
06/30/2014,01 02 09 13 16 20 26 29 32 33 39 44 45 48 52 53 59 65 72 75
06/29/2014,04 09 15 21 22 27 30 46 56 61 62 63 64 67 72 73 75 76 79 80
06/28/2014,01 05 08 11 12 26 32 33 38 40 41 44 45 52 56 57 58 70 73 79
06/27/2014,11 16 23 25 27 29 32 33 36 38 42 43 45 48 52 55 63 67 68 76
06/26/2014,01 02 07 10 14 15 17 25 26 30 33 34 35 43 46 47 50 58 67 74
06/25/2014,12 14 22 23 27 28 39 47 49 50 51 52 53 59 60 68 69 72 76 79
06/24/2014,06 07 08 16 17 25 26 30 35 36 44 51 52 54 55 58 63 65 77 79
06/23/2014,04 08 11 18 20 22 36 38 44 48 49 50 53 54 57 64 67 73 75 77
06/22/2014,08 09 10 11 12 21 23 27 31 39 40 41 53 54 58 60 65 70 73 74
06/21/2014,06 08 15 19 20 21 27 36 38 39 46 47 51 55 61 67 68 71 72 75
06/20/2014,03 07 08 13 18 20 21 25 27 29 31 38 39 40 46 47 57 72 75 80
06/19/2014,07 08 09 15 20 34 35 40 42 44 48 50 52 54 57 62 65 66 70 71
06/18/2014,02 09 12 16 20 21 29 39 41 44 50 56 66 67 69 72 74 75 78 80
06/17/2014,02 04 07 08 20 22 35 37 39 43 48 54 56 59 61 65 68 71 74 75
06/16/2014,08 14 20 26 27 29 30 33 34 35 40 43 44 46 51 57 65 68 77 79
06/15/2014,03 07 09 10 12 13 21 22 29 31 35 36 40 42 58 59 61 65 66 74
06/14/2014,09 10 12 14 17 18 32 33 36 37 50 59 60 62 66 67 72 74 76 79
06/13/2014,01 04 06 07 08 16 18 20 25 26 28 29 39 43 51 56 62 68 75 77
```

```
99      06/12/2014,02 03 14 16 20 26 27 29 32 37 42 49 51 53 56 59 62 68 72 76
00      06/11/2014,05 07 09 11 14 15 20 22 25 36 39 40 49 59 62 66 71 72 73 75
01      06/10/2014,01 06 07 18 24 26 27 33 36 39 46 49 60 62 64 66 68 75 77 79
02      06/09/2014,01 05 07 13 14 18 32 33 34 37 38 46 52 54 58 61 68 73 75 80
03      06/08/2014,04 06 09 10 12 13 20 24 27 34 35 39 45 49 50 55 56 58 60 65
04      06/07/2014,03 06 09 10 16 18 20 24 31 38 44 52 54 57 60 65 66 70 72 78
05      06/06/2014,01 04 09 12 13 18 22 23 27 30 35 37 47 49 54 64 67 69 76 79
06      06/05/2014,01 06 07 14 15 16 28 29 30 31 40 41 43 57 62 68 70 72 76 78
07      06/04/2014,06 11 15 19 20 27 30 35 40 41 42 48 49 60 64 66 67 68 70 78
08      06/03/2014,05 06 08 09 14 21 26 32 35 36 38 40 47 55 63 64 67 70 72 77
09      06/02/2014,10 12 15 19 21 23 24 29 30 46 57 58 59 61 64 69 71 72 73 77
10      06/01/2014,03 09 10 11 15 20 25 29 39 40 45 46 48 50 52 58 70 71 75 76
11      05/31/2014,06 10 16 22 27 30 38 40 42 46 50 51 53 63 64 67 69 75 76 80
12      05/30/2014,09 11 13 20 23 24 26 28 29 30 37 46 49 52 53 55 60 71 72 76
13      05/29/2014,02 05 06 08 09 10 18 20 24 26 29 31 32 33 50 52 58 59 65 78
14      05/28/2014,01 02 03 05 08 09 11 12 16 18 20 26 34 35 39 50 55 60 64 74
15      05/27/2014,15 19 21 22 27 37 40 46 51 53 56 59 60 63 64 66 68 71 74 78
16      05/26/2014,04 06 08 14 18 19 28 30 33 35 36 42 44 49 52 53 59 66 75 77
17      05/25/2014,10 13 16 24 30 31 34 36 44 49 53 55 59 70 73 74 77 78 79 80
18      05/24/2014,02 08 11 16 18 19 20 21 23 27 35 36 38 43 48 62 63 66 75 77
19      05/23/2014,02 05 10 11 15 17 21 25 26 28 29 38 41 42 45 54 55 66 77 78
20      05/22/2014,10 11 12 13 24 25 27 39 45 50 51 56 58 59 62 65 72 74 75 80
21      05/21/2014,01 04 05 07 09 17 18 19 26 29 31 33 43 48 49 56 58 60 64 74
22      05/20/2014,01 04 15 17 24 26 30 31 33 37 45 46 52 59 66 69 72 74 75 79
23      05/19/2014,01 10 13 24 31 32 33 38 40 42 46 47 49 54 62 69 74 76 77 80
24      05/18/2014,05 09 10 15 25 27 28 31 34 39 40 42 51 60 66 67 69 72 75 80
25      05/17/2014,08 11 15 19 23 24 25 34 36 39 41 44 46 47 52 56 59 65 75 77
26      05/16/2014,03 13 18 20 26 31 33 34 40 41 45 46 49 50 56 57 61 66 67 75
27      05/15/2014,04 07 09 11 13 22 29 33 37 40 49 54 58 61 64 68 70 72 79 80
28      05/14/2014,03 05 07 10 14 18 19 23 26 28 29 36 46 48 52 57 58 59 73 80
29      05/13/2014,06 11 12 18 21 23 33 42 44 45 49 52 54 55 56 58 68 72 76 79
30      05/12/2014,02 03 05 06 21 23 33 34 39 41 48 50 53 54 58 60 61 69 76 78
31      05/11/2014,10 11 13 14 19 20 29 30 33 37 38 40 47 52 55 61 67 68 69 76
32      05/10/2014,01 04 05 08 09 12 13 19 23 27 30 35 37 42 43 53 54 62 74 75
33      05/09/2014,01 08 10 11 14 15 17 21 26 28 34 35 38 40 44 61 64 67 73 79
34      05/08/2014,07 11 14 18 20 24 29 30 31 32 33 34 36 41 52 54 66 70 75 79
35      05/07/2014,03 04 07 24 28 34 35 36 45 50 53 57 59 60 63 64 65 76 79 80
36      05/06/2014,02 08 09 10 16 19 27 34 35 38 39 50 54 60 63 65 68 70 71 74
37      05/05/2014,04 09 12 18 21 25 27 42 43 44 47 48 51 56 57 58 59 62 72 79
38      05/04/2014,02 03 05 07 08 13 17 19 22 25 30 34 38 49 50 53 62 64 75 78
39      05/03/2014,15 16 19 21 24 31 33 39 41 45 54 57 58 60 62 66 67 74 76 79
40      05/02/2014,12 17 22 25 28 32 33 38 47 48 51 53 54 56 58 60 61 62 70 75
41      05/01/2014,02 07 08 12 13 16 17 18 19 20 23 33 35 39 41 55 65 68 72 75
42      04/30/2014,03 08 09 22 31 35 40 45 48 52 54 57 60 61 64 65 67 76 77 80
43      04/29/2014,04 05 13 19 25 28 33 34 35 41 53 57 59 60 63 64 67 75 77 79
44      04/28/2014,04 06 09 15 18 22 26 40 41 45 48 55 57 61 63 68 71 75 76 78
45      04/27/2014,03 06 07 10 11 20 22 23 27 28 30 54 55 56 58 59 64 72 73 77
46      04/26/2014,05 11 13 27 31 44 47 50 56 58 59 60 62 63 64 67 69 75 76 79
47      04/25/2014,05 17 18 20 23 28 34 37 38 45 46 47 50 56 62 63 68 69 70 74
48      04/24/2014,02 12 15 17 19 23 25 26 27 31 34 35 50 57 59 61 63 66 67 73
49      04/23/2014,04 08 14 18 19 30 34 35 36 39 40 41 51 55 56 57 69 71 73 80
50      04/22/2014,03 05 11 21 25 29 35 42 44 47 55 56 57 58 60 64 65 68 74 79
51      04/21/2014,01 08 11 16 17 18 22 24 27 29 33 39 47 50 60 62 66 71 74 80
52      04/20/2014,09 10 12 15 17 19 20 24 35 42 47 49 52 59 64 69 70 72 76 79
```

```
53      04/19/2014,01 02 09 10 11 13 14 19 21 26 28 30 42 47 55 62 65 66 71 76
54      04/18/2014,01 12 13 15 16 24 30 36 40 41 46 47 51 53 54 55 59 60 61 75
55      04/17/2014,01 02 04 06 10 12 20 24 26 37 42 47 56 62 64 67 68 70 76 80
56      04/16/2014,03 06 07 10 12 13 20 22 23 24 27 42 47 56 58 64 65 66 69 70
57      04/15/2014,01 11 13 14 16 21 24 25 26 30 37 39 41 44 55 59 65 68 74 77
58      04/14/2014,02 03 04 13 18 20 24 26 40 42 48 53 56 60 64 65 67 68 72 78
59      04/13/2014,02 03 10 12 18 19 21 23 32 36 40 41 42 49 56 63 64 69 75 80
60      04/12/2014,16 17 23 24 26 28 32 34 38 41 46 47 48 53 54 60 61 62 63 65
61      04/11/2014,03 04 05 14 19 25 29 31 36 38 40 41 44 59 62 65 67 68 74 78
62      04/10/2014,03 08 11 15 21 23 25 31 33 34 44 46 47 48 50 53 55 67 69 76
63      04/09/2014,06 12 17 26 31 35 43 49 52 53 56 57 61 62 63 66 69 73 78 79
64      04/08/2014,05 08 11 14 15 21 26 30 38 40 44 47 48 51 54 62 63 65 72 75
65      04/07/2014,02 03 04 05 14 17 25 29 30 37 38 42 45 47 50 55 61 64 71 72
66      04/06/2014,05 07 10 14 21 25 37 42 43 48 52 55 61 63 64 65 67 69 71 72
67      04/05/2014,05 08 11 15 16 20 23 24 27 34 35 36 38 40 41 48 60 69 70 78
68      04/04/2014,01 03 08 10 11 14 18 26 35 39 40 44 45 48 49 55 56 65 76 80
69      04/03/2014,01 05 13 17 21 28 35 37 40 41 43 46 50 59 65 67 68 72 78 79
70      04/02/2014,02 05 12 20 22 23 26 31 35 38 41 45 52 56 57 62 63 66 67 80
71      04/01/2014,02 10 14 15 25 26 28 32 35 38 42 44 45 50 52 53 59 63 74 80
72      03/31/2014,01 04 14 16 24 30 38 39 44 50 51 52 56 57 58 59 60 62 72 73
73      03/30/2014,05 06 08 10 11 14 17 19 37 39 42 49 52 56 57 62 72 74 75 80
74      03/29/2014,03 04 10 12 14 18 21 29 30 33 36 46 56 57 58 59 76 77 79 80
75      03/28/2014,03 04 10 13 17 19 23 28 31 38 43 48 53 60 68 71 72 76 79 80
76      03/27/2014,04 14 16 19 20 22 29 34 44 45 47 62 68 69 70 71 76 77 78 80
77      03/26/2014,02 06 11 17 22 24 31 33 36 43 56 59 62 64 68 71 72 73 77 80
78      03/25/2014,18 23 26 28 29 31 33 36 37 43 48 54 58 62 64 65 72 74 77 79
79      03/24/2014,01 04 06 12 22 25 26 27 29 38 41 47 52 61 63 65 66 67 70 75
80      03/23/2014,05 07 09 12 16 19 23 30 32 33 34 50 51 55 56 59 62 65 66 72
81      03/22/2014,01 05 06 08 10 11 13 18 21 23 24 30 32 34 37 40 42 44 46 64
82      03/21/2014,02 03 04 05 06 09 14 26 35 38 41 48 53 55 56 57 71 73 75 78
83      03/20/2014,03 06 09 11 12 14 20 21 26 29 32 34 40 47 56 68 70 71 75 79
84      03/19/2014,03 07 09 14 20 24 28 29 32 36 39 41 53 59 60 62 63 64 66 67
85      03/18/2014,05 11 14 17 18 26 30 33 37 43 47 52 54 60 64 68 69 73 79 80
86      03/17/2014,01 05 07 10 11 13 16 21 24 26 28 41 44 45 47 57 59 61 68 72
87      03/16/2014,10 14 16 18 20 21 22 23 24 27 29 33 35 40 47 51 62 69 71 72
88      03/15/2014,01 07 10 12 18 22 26 29 31 32 41 45 54 63 64 66 73 74 77 78
89      03/14/2014,02 04 05 07 09 11 15 19 24 25 26 38 40 49 55 57 67 73 77 79
90      03/13/2014,02 04 07 11 13 15 29 31 34 35 36 41 46 47 49 55 56 61 71 77
91      03/12/2014,01 03 04 07 12 15 23 25 26 30 47 51 53 57 58 62 64 65 74 76
92      03/11/2014,03 04 08 11 16 19 20 26 33 36 37 42 48 50 54 66 67 72 77 80
93      03/10/2014,01 02 08 18 20 23 24 31 35 38 39 44 47 56 64 66 70 73 75 80
94      03/09/2014,02 05 13 14 17 23 30 34 50 52 55 57 58 59 66 70 76 78 79 80
95      03/08/2014,01 13 17 23 36 39 41 42 44 46 49 50 54 57 60 63 69 71 72 78
96      03/07/2014,06 07 10 12 20 22 25 31 33 39 40 42 46 50 54 62 67 69 74 77
97      03/06/2014,05 09 12 16 22 23 33 41 44 46 54 58 60 63 64 65 73 76 77 79
98      03/05/2014,03 04 05 07 08 09 15 19 24 26 30 34 37 44 56 62 65 66 73 77
99      03/04/2014,11 12 14 17 21 26 27 28 32 37 42 46 56 62 63 68 72 73 75 78
00      03/03/2014,05 10 17 18 29 31 33 34 38 40 47 56 58 62 67 68 69 71 73 78
01      03/02/2014,02 04 07 13 19 24 28 36 38 49 54 57 59 67 68 71 73 74 78 79
02      03/01/2014,02 07 12 20 22 24 30 31 35 41 45 46 48 55 56 60 65 70 73 80
03      02/28/2014,01 08 12 15 17 19 22 28 29 31 36 37 41 43 52 56 63 67 68 79
04      02/27/2014,02 04 09 25 26 34 35 36 38 40 41 44 46 50 51 57 58 67 70 77
05      02/26/2014,03 04 12 16 19 31 33 35 37 38 41 43 44 45 49 50 52 68 72 74
06      02/25/2014,02 05 06 08 09 13 18 23 24 25 35 37 38 39 44 46 49 52 53 68
```

07 02/24/2014,05 08 12 16 17 23 25 26 28 29 30 35 42 43 56 64 75 76 77 80
08 02/23/2014,01 09 14 17 19 30 40 44 49 53 56 61 62 66 67 70 73 75 78 79
09 02/22/2014,04 06 13 19 22 28 47 50 52 55 56 59 66 67 68 69 70 71 76 77
10 02/21/2014,10 12 14 20 22 23 24 27 32 34 40 43 50 53 57 64 67 68 69 71
11 02/20/2014,01 02 07 11 13 21 28 32 39 40 42 44 58 63 64 67 71 73 74 79
12 02/19/2014,04 05 14 18 28 34 35 38 41 45 48 50 52 54 55 59 63 69 73 78
13 02/18/2014,02 03 04 11 14 15 22 24 28 29 30 37 39 52 58 62 67 68 71 77
14 02/17/2014,01 03 04 07 17 18 26 29 30 38 39 45 52 54 56 59 63 64 75 80
15 02/16/2014,04 09 13 17 19 21 25 34 40 44 46 49 53 55 58 62 66 67 71 73
16 02/15/2014,03 05 12 13 19 24 25 26 28 32 35 36 43 49 52 54 58 61 69 77
17 02/14/2014,09 11 12 14 15 16 18 24 26 30 31 35 38 45 54 61 62 66 75 76
18 02/13/2014,01 05 08 09 11 15 19 22 27 29 38 46 51 56 59 64 67 68 70 73
19 02/12/2014,02 04 08 15 17 22 24 25 32 33 34 36 37 38 41 43 62 63 66 68
20 02/11/2014,01 06 08 09 10 12 24 30 34 35 37 38 44 52 57 58 65 69 70 80
21 02/10/2014,08 10 19 21 25 27 29 33 39 41 43 48 49 54 62 63 64 66 71 79
22 02/09/2014,02 08 09 21 26 27 33 35 40 41 46 47 53 55 56 61 64 74 78 80
23 02/08/2014,03 08 10 13 19 22 25 29 32 34 43 48 56 60 65 72 73 75 79 80
24 02/07/2014,02 10 14 16 18 25 26 32 33 34 36 45 52 54 61 65 66 69 73 78
25 02/05/2014,09 11 13 14 15 22 25 33 35 39 44 52 56 68 71 72 73 76 77 79
26 02/04/2014,04 05 06 08 09 10 17 24 29 34 36 41 49 51 58 66 71 76 78 80
27 02/03/2014,01 05 12 16 19 23 28 31 34 40 43 44 46 47 48 49 50 53 68 78
28 02/02/2014,02 03 08 12 14 19 21 25 42 43 45 50 52 53 55 61 62 68 76 77
29 02/01/2014,03 04 06 07 11 16 20 22 27 31 33 38 41 44 46 50 56 60 71 79
30 01/31/2014,03 04 09 10 24 25 27 28 33 40 42 43 44 46 50 57 68 71 75 79
31 01/30/2014,05 07 11 14 15 20 24 28 29 31 36 39 45 50 52 53 68 73 75 79
32 01/29/2014,06 08 11 14 24 29 39 46 48 52 53 54 64 66 71 74 76 77 79 80
33 01/28/2014,04 06 07 09 11 19 24 30 33 35 37 40 43 53 54 58 59 63 71 77
34 01/27/2014,02 08 09 11 23 26 31 33 39 43 44 50 52 56 57 59 61 66 68 73
35 01/26/2014,01 05 18 29 31 34 38 40 46 50 53 55 59 60 69 71 72 73 74 79
36 01/25/2014,05 11 15 16 17 21 23 29 38 44 51 57 60 62 63 68 69 70 73 79
37 01/24/2014,01 07 12 24 26 29 42 45 47 48 49 54 57 58 62 63 68 71 77 78
38 01/23/2014,05 08 10 14 24 26 35 36 48 54 55 59 64 65 66 67 69 72 76 79
39 01/22/2014,03 11 23 31 35 37 38 39 40 45 49 56 59 62 63 65 66 72 78 79
40 01/21/2014,06 08 09 14 23 26 28 30 31 37 38 41 53 56 57 60 70 75 76 77
41 01/20/2014,02 08 10 12 16 20 25 26 31 32 39 45 47 48 52 62 64 72 76 80
42 01/19/2014,02 09 10 20 21 25 26 29 34 35 39 43 46 51 55 57 68 73 76 79
43 01/18/2014,07 09 18 22 29 30 31 35 37 40 43 45 50 51 53 62 64 76 78 80
44 01/16/2014,01 02 10 11 13 24 26 29 34 39 45 55 63 65 67 72 74 75 77 80
45 01/15/2014,01 08 11 13 24 26 31 36 38 40 52 53 55 59 60 62 63 74 76 77
46 01/13/2014,08 15 17 24 30 32 43 44 45 52 54 57 58 62 63 66 74 75 78 80
47 01/12/2014,05 09 12 14 18 20 22 28 32 36 43 55 57 59 62 63 65 71 74 80
48 01/10/2014,08 09 15 18 19 24 30 36 38 40 42 44 48 50 54 65 66 67 70 76
49 01/09/2014,03 04 05 14 17 22 23 25 26 32 37 40 46 49 57 58 61 65 67 80
50 01/08/2014,02 04 10 20 21 25 29 30 34 39 44 45 46 54 57 61 62 64 66 68
51 01/07/2014,10 14 18 21 22 25 26 31 35 44 51 59 60 70 72 73 75 78 79 80
52 01/06/2014,03 12 14 16 19 20 27 31 32 33 35 38 43 45 46 61 64 68 77 79
53 01/05/2014,01 03 09 14 15 18 19 21 31 32 35 45 48 53 54 57 58 60 71 78
54 01/04/2014,01 03 09 14 15 16 31 38 44 45 50 54 55 63 67 72 73 74 77 78
55 01/03/2014,02 04 07 08 09 10 14 22 23 27 31 35 38 49 52 57 59 71 74 78
56 01/02/2014,01 03 13 14 15 17 24 29 35 40 41 46 48 53 57 62 63 66 68 77
57 01/01/2014,06 07 15 24 35 41 42 43 48 49 50 53 55 58 62 64 70 71 74 78
58 12/31/2013,03 04 05 08 10 23 29 39 42 43 46 50 54 55 59 64 72 75 78 80
59 12/30/2013,02 03 11 13 17 23 28 31 34 36 37 46 50 66 68 70 71 72 74 78
60 12/29/2013,03 06 08 20 21 27 29 37 39 42 47 61 63 67 69 70 72 74 77 80

```
61      12/28/2013,02 11 12 13 14 23 24 30 35 39 40 47 56 60 61 62 67 70 73 80
62      12/27/2013,05 06 15 18 22 25 26 30 31 35 39 41 43 44 61 64 67 69 78 80
63      12/26/2013,15 17 18 24 28 31 32 34 39 42 45 56 57 59 60 61 63 66 74 80
64      12/25/2013,03 07 12 14 16 20 30 32 40 41 44 45 47 50 51 52 58 62 75 80
65      12/24/2013,02 03 04 10 12 23 30 34 36 44 49 57 62 67 69 72 75 77 79 80
66      12/23/2013,16 18 23 26 27 38 46 52 56 57 59 60 62 64 68 70 72 73 74 79
67      12/22/2013,05 12 15 16 17 19 21 29 34 36 39 42 47 49 57 65 67 72 75 77
68      12/21/2013,01 02 03 09 10 11 14 24 32 34 35 37 38 47 48 58 59 62 69 75
69      12/20/2013,06 13 14 20 24 26 28 33 35 37 48 51 53 54 56 58 60 69 74 77
70      12/18/2013,08 11 14 17 19 23 27 30 41 43 46 51 52 56 59 70 73 74 76 79
71      12/17/2013,02 04 06 07 08 15 22 23 26 37 38 40 41 42 46 48 49 52 64 79
72      12/16/2013,04 05 10 14 16 24 36 39 43 52 54 55 58 63 64 70 71 72 79 80
73      12/15/2013,04 12 19 28 31 35 38 41 54 55 61 63 64 65 67 69 71 75 78 80
74      12/13/2013,04 06 07 17 20 21 22 27 31 34 38 43 45 49 50 54 62 66 71 75
75      12/12/2013,03 08 11 12 13 15 16 22 24 27 30 33 39 45 50 52 54 56 62 65
76      12/11/2013,03 07 09 13 14 20 24 30 35 36 50 58 61 62 63 64 65 72 73 74
77      12/10/2013,09 17 19 30 31 36 37 38 42 43 44 45 49 62 67 69 71 73 75 78
78      12/08/2013,11 12 18 19 20 37 42 43 51 52 55 58 59 62 64 65 66 75 78 80
79      12/07/2013,01 04 08 12 13 19 21 24 27 30 31 41 42 46 50 51 52 64 69 78
80      12/06/2013,04 09 11 14 22 24 28 31 36 42 45 49 52 66 69 70 71 72 73 80
81      12/05/2013,03 06 13 14 18 32 38 41 43 44 52 53 55 57 61 63 65 67 71 80
82      12/04/2013,01 05 06 09 11 12 14 15 22 24 27 29 37 38 47 48 51 58 59 66
83      12/03/2013,06 08 13 22 23 27 29 32 33 34 37 41 48 50 55 56 57 61 64 71
84      12/02/2013,07 09 12 13 14 18 26 27 33 35 39 40 42 54 57 61 62 65 66 69
85      12/01/2013,01 06 08 21 22 29 31 32 36 42 45 49 58 59 62 63 64 66 70 80
86      11/30/2013,02 03 07 13 15 17 22 27 29 41 42 43 48 62 63 64 66 69 71 72
87      11/29/2013,04 07 08 10 13 15 16 17 25 28 30 31 33 34 41 50 62 70 76 78
88      11/28/2013,04 10 11 12 17 19 27 29 32 33 50 58 59 61 65 66 71 72 75 80
89      11/27/2013,02 03 07 11 13 14 19 21 23 25 27 40 45 48 49 55 66 67 68 77
90      11/26/2013,03 04 05 11 13 15 17 18 19 23 27 29 31 39 43 60 63 65 72 74
91      11/25/2013,02 03 05 07 09 10 21 23 30 34 35 42 43 45 46 48 49 52 68 70
92      11/24/2013,01 07 11 22 26 32 37 41 45 46 48 51 52 61 65 69 74 75 76 78
93      11/23/2013,05 07 09 12 20 22 25 26 29 33 38 39 44 45 47 60 61 62 64 69
94      11/22/2013,06 16 23 26 27 34 35 36 41 46 54 59 60 62 64 70 71 72 78 79
95      11/21/2013,07 13 17 27 30 33 35 43 55 58 63 64 66 68 70 71 72 74 75 79
96      11/20/2013,05 09 13 16 21 27 29 49 50 54 55 56 60 64 66 67 72 74 75 76
97      11/19/2013,03 13 14 21 36 37 38 40 43 46 50 51 66 71 72 73 75 76 78 80
98      11/18/2013,06 07 10 12 19 33 34 35 36 38 41 48 55 58 62 68 75 76 79 80
99      11/17/2013,02 04 12 14 19 20 25 27 34 35 37 40 41 42 44 59 65 66 72 76
00      11/16/2013,01 02 09 11 12 15 29 32 40 41 42 53 57 65 66 67 70 71 73 78
01      11/15/2013,08 10 11 24 25 26 27 32 39 46 49 52 53 55 56 60 62 66 76 80
02      11/14/2013,05 06 15 17 21 23 27 29 31 40 41 45 52 54 55 57 64 71 76 77
03      11/13/2013,09 11 12 16 19 21 25 33 35 36 43 46 47 58 59 66 67 69 70 74
04      11/12/2013,01 05 08 09 13 17 21 24 30 31 32 33 34 38 39 55 61 62 63 73
05      11/11/2013,01 06 10 11 12 13 17 18 21 27 30 31 35 36 43 52 53 55 65 67
06      11/10/2013,01 03 06 07 08 17 24 37 39 41 45 46 51 55 56 57 59 72 77 80
07      11/09/2013,03 08 18 20 25 27 28 33 34 40 41 43 48 58 59 62 63 71 72 75
08      11/08/2013,04 05 07 12 16 17 22 23 30 35 39 42 45 46 49 51 62 64 73 79
09      11/07/2013,03 09 10 13 20 27 28 31 32 40 50 52 54 57 58 65 71 73 77 79
10      11/06/2013,04 05 11 13 14 16 18 21 28 35 40 46 57 58 59 64 66 67 71 77
11      11/05/2013,04 09 11 12 17 18 20 21 23 27 29 37 48 60 61 65 66 76 78 79
12      11/04/2013,01 02 03 05 09 10 11 17 18 23 26 27 30 47 49 55 59 61 68 71
13      11/03/2013,01 03 08 11 16 26 32 39 42 45 49 50 51 56 60 63 66 68 71 77
14      11/02/2013,03 07 08 12 15 16 21 29 33 34 36 42 57 60 64 65 69 74 78 80
```

```
11/01/2013,03 12 14 19 28 32 37 40 43 48 53 61 64 67 68 71 72 75 79 80
10/31/2013,01 03 06 09 20 39 43 46 51 52 55 59 61 64 70 71 76 77 78 79
10/30/2013,03 04 10 13 14 16 19 27 28 30 41 42 49 53 54 66 67 72 75 77
10/29/2013,04 06 08 11 14 19 24 25 27 36 40 47 51 55 58 60 64 69 74 76
10/28/2013,01 04 06 16 17 22 25 29 33 34 36 47 50 55 56 57 66 74 76 80
10/27/2013,01 02 04 13 18 23 27 28 31 37 39 56 59 61 62 64 68 70 73 77
10/26/2013,02 05 09 19 24 26 30 37 42 52 54 55 60 63 64 67 68 71 73 76
10/25/2013,09 15 16 20 26 27 29 38 39 40 43 45 46 57 61 68 75 76 79 80
10/24/2013,06 07 09 14 19 23 29 34 40 44 48 56 58 59 64 65 70 71 73 74
10/23/2013,05 14 19 26 28 29 36 44 46 47 48 53 56 60 61 63 64 73 79 80
10/22/2013,04 13 15 17 22 24 27 32 38 40 44 47 48 63 65 68 71 72 75 79
10/21/2013,03 04 10 12 15 16 20 21 25 27 34 36 39 46 51 56 60 69 74 77
10/20/2013,07 11 13 24 29 35 37 42 43 47 48 52 53 57 61 62 64 66 69 73
10/19/2013,04 09 10 12 19 20 22 27 36 37 39 40 44 54 55 57 61 66 68 69
10/18/2013,03 08 12 15 17 20 22 24 29 30 36 41 46 67 69 73 74 78 79 80
10/17/2013,01 02 16 18 27 29 33 41 45 47 52 55 63 64 65 66 69 76 79 80
10/16/2013,02 03 07 13 16 18 19 22 30 36 41 43 57 63 65 69 72 73 74 79
10/15/2013,01 05 08 11 14 16 17 26 31 40 42 44 47 50 55 56 69 71 74 80
10/14/2013,12 14 16 20 28 30 33 35 37 49 50 53 57 58 61 62 63 70 79 80
10/13/2013,08 18 29 32 33 34 43 44 45 46 49 53 55 58 63 69 70 74 78 79
10/12/2013,02 04 07 08 09 11 13 14 18 24 25 29 30 35 39 54 58 64 65 79
10/11/2013,08 22 23 25 27 30 33 35 42 44 46 49 53 56 62 63 68 75 79 80
10/10/2013,03 06 10 18 23 28 29 32 35 37 38 39 41 43 46 53 57 77 79 80
10/09/2013,04 15 16 18 22 30 31 32 37 41 43 45 46 47 50 51 57 67 70 71
10/08/2013,01 03 05 13 19 20 28 32 36 41 45 47 53 55 65 66 70 71 74 77
10/07/2013,02 05 12 13 17 23 24 27 28 29 31 32 35 37 43 46 52 53 74 80
10/06/2013,01 07 12 36 37 38 39 40 41 42 44 48 49 54 56 63 70 76 78 80
10/05/2013,09 12 19 31 32 35 43 49 54 57 58 60 64 65 67 68 69 70 73 76
10/04/2013,05 16 17 18 20 25 33 34 37 42 44 48 49 54 59 61 67 73 75 79
10/03/2013,05 10 11 14 27 29 30 39 44 45 46 49 50 60 61 62 63 72 79 80
10/02/2013,03 07 14 16 22 28 36 40 43 52 54 57 60 61 68 70 71 72 73 78
10/01/2013,02 10 15 18 23 25 29 33 40 41 42 45 56 59 63 64 71 74 76 77
09/30/2013,04 12 17 20 26 31 33 34 45 59 63 65 67 68 69 70 72 73 79 80
09/29/2013,06 09 18 20 21 22 24 28 30 31 34 36 38 51 52 63 64 67 74 75
09/28/2013,05 07 08 14 17 27 32 33 35 38 48 51 53 57 58 60 70 71 77 79
09/27/2013,08 11 13 15 23 25 35 37 39 41 42 46 48 49 50 52 57 65 67 73
09/26/2013,02 04 06 12 16 17 18 19 20 27 28 30 42 44 46 47 52 65 71 76
09/25/2013,01 14 25 27 28 34 37 41 44 46 48 49 51 55 65 67 70 72 74 75
09/24/2013,03 05 06 11 13 15 21 27 28 34 41 44 47 58 61 63 68 70 77 80
09/23/2013,12 14 19 22 24 25 27 28 35 37 40 43 44 54 61 65 67 70 72 76
09/22/2013,02 07 11 12 14 15 17 22 36 45 47 49 55 57 72 73 74 76 78 80
09/21/2013,05 06 08 15 18 21 24 26 27 33 38 40 42 43 53 56 58 60 79 80
09/20/2013,01 12 14 22 26 27 32 37 49 50 55 61 62 63 65 68 72 73 74 78
09/19/2013,02 06 08 16 18 22 25 28 29 31 37 42 43 44 49 51 62 68 74 78
09/18/2013,01 06 09 11 15 16 18 20 23 27 30 41 43 46 48 62 69 72 76 80
09/17/2013,05 08 09 16 20 21 28 33 34 35 41 42 43 47 48 56 70 71 72 73
09/16/2013,01 02 03 08 11 17 18 24 26 33 34 53 57 66 71 73 75 78 79 80
09/15/2013,03 09 10 12 20 24 25 26 29 32 38 42 53 58 62 64 67 70 76 77
09/14/2013,04 06 21 23 26 27 40 42 46 49 51 52 53 54 57 69 70 72 78 80
09/13/2013,02 05 09 13 20 21 27 28 32 34 40 45 46 52 55 61 62 66 76 79
09/12/2013,04 05 06 09 16 18 28 33 34 40 41 44 50 58 59 60 65 66 72 73
09/11/2013,01 04 05 10 13 14 23 25 28 29 30 38 39 44 46 61 63 73 74 77
09/10/2013,05 09 10 14 16 27 28 29 35 36 37 40 49 52 59 61 66 74 75 78
09/09/2013,17 18 21 22 23 28 29 35 40 41 43 45 48 52 58 60 63 70 72 77
```

```
69      09/08/2013,08 09 10 11 19 21 25 27 34 36 39 44 52 59 60 63 66 71 72 75
70      09/07/2013,04 14 21 24 25 27 31 33 36 37 39 42 45 53 61 63 64 66 73 74
71      09/06/2013,05 07 11 19 24 28 32 35 36 38 41 52 56 57 61 62 66 67 70 74
72      09/05/2013,01 05 07 10 11 21 23 30 31 32 33 34 39 40 43 49 55 57 61 64
73      09/04/2013,04 05 08 12 14 24 30 35 36 46 52 56 57 59 60 67 69 72 73 76
74      09/03/2013,01 03 10 11 13 18 20 25 27 33 36 40 50 54 58 62 68 76 79 80
75      09/02/2013,06 09 11 13 17 20 24 26 32 38 39 46 49 52 55 59 61 68 74 79
76      09/01/2013,03 04 05 06 11 13 15 17 19 20 21 26 39 41 42 47 54 58 77 79
77      08/31/2013,02 04 07 15 20 25 28 30 35 40 41 48 53 54 57 58 60 62 71 77
78      08/30/2013,02 11 13 14 16 18 21 24 25 36 41 43 56 59 60 65 71 72 73 75
79      08/29/2013,03 09 10 11 14 15 18 19 33 45 46 50 51 53 56 58 67 68 72 74
80      08/28/2013,01 04 11 14 15 16 18 27 29 31 35 36 37 38 40 46 53 64 70 74
81      08/27/2013,05 08 10 11 30 32 37 39 40 41 45 49 51 55 60 64 67 68 72 75
82      08/26/2013,05 06 09 10 12 13 14 27 35 36 38 43 47 51 55 64 67 71 72 74
83      08/25/2013,02 03 05 07 20 21 22 25 28 33 35 39 40 41 44 51 62 64 68 79
84      08/24/2013,01 06 07 09 10 11 14 18 20 25 30 39 45 47 53 60 61 74 75 76
85      08/23/2013,06 08 17 19 21 31 32 36 37 40 52 54 55 59 60 61 69 70 72 74
86      08/22/2013,01 04 15 17 19 24 26 30 38 42 44 56 62 65 68 69 70 71 76 78
87      08/21/2013,04 06 08 12 14 20 22 23 26 28 34 45 49 50 59 63 64 65 70 80
88      08/20/2013,01 04 09 10 14 15 18 21 23 26 38 39 40 52 54 56 65 70 77 78
89      08/19/2013,03 08 10 14 15 20 25 28 31 34 35 36 60 62 63 66 68 71 73 78
90      08/18/2013,13 16 27 31 32 37 38 43 44 45 52 59 61 63 64 67 69 70 71 75
91      08/17/2013,01 04 05 16 17 21 22 28 31 32 48 49 51 57 61 65 72 73 78 79
92      08/16/2013,02 03 06 08 10 22 24 26 28 36 38 41 42 44 55 57 61 71 73 77
93      08/15/2013,04 18 19 25 28 30 36 43 46 49 50 52 55 56 58 66 71 72 78 80
94      08/14/2013,06 07 18 19 21 23 31 40 43 45 48 56 68 71 72 73 75 76 77 80
95      08/13/2013,05 06 09 16 20 22 25 27 28 31 32 34 51 54 59 62 67 70 71 73
96      08/12/2013,02 03 06 09 17 20 22 23 25 29 31 37 40 41 43 49 52 54 72 74
97      08/11/2013,04 05 06 19 20 25 34 35 38 45 49 50 52 55 56 62 65 69 70 78
98      08/10/2013,02 03 04 16 18 20 21 22 30 31 33 40 43 47 54 56 65 66 70 73
99      08/09/2013,10 14 20 21 23 28 29 33 35 37 38 40 44 49 59 68 69 75 77 79
00      08/08/2013,05 07 14 17 19 20 25 30 32 37 39 41 46 53 54 56 59 70 71 80
01      08/07/2013,04 09 25 28 29 30 33 38 42 43 44 49 50 51 54 57 60 66 73 76
02      08/06/2013,01 02 03 05 06 09 12 26 32 38 39 40 44 55 60 72 73 74 75 76
03      08/05/2013,02 08 20 22 23 24 26 27 32 38 39 43 48 66 68 69 71 76 77 79
04      08/04/2013,01 02 04 11 15 18 19 23 30 32 41 44 45 49 52 56 61 75 78 79
05      08/03/2013,01 09 10 11 15 18 22 24 29 32 36 44 45 46 48 57 62 63 64 69
06      08/02/2013,04 06 11 14 17 19 23 25 26 33 36 40 41 44 46 49 60 71 74 75
07      08/01/2013,04 08 11 13 16 17 21 26 28 29 33 47 50 54 62 65 68 72 78 80
08      07/31/2013,03 04 11 13 15 16 18 22 23 32 35 40 43 45 52 56 59 60 70 79
09      07/30/2013,01 03 06 07 08 15 16 22 26 41 42 43 48 49 54 56 58 68 71 79
10      07/29/2013,06 08 14 17 21 22 33 38 39 40 41 45 51 53 61 65 70 71 75 78
11      07/28/2013,02 06 09 11 13 17 19 36 37 38 41 49 50 58 63 64 65 71 75 76
12      07/27/2013,03 04 11 16 20 23 26 28 30 31 37 44 53 63 68 69 70 74 78 79
13      07/26/2013,02 04 06 09 18 20 25 26 31 37 40 55 60 62 65 69 71 72 73 77
14      07/25/2013,12 17 20 24 27 29 32 35 40 46 47 52 55 58 60 69 73 74 78 80
15      07/24/2013,13 15 25 27 29 32 33 37 39 41 43 52 55 59 69 74 75 77 78 79
16      07/23/2013,02 05 16 22 26 33 35 36 38 43 46 49 57 58 67 69 70 76 77 78
17      07/22/2013,02 07 09 10 11 13 17 20 21 24 27 30 31 46 48 50 52 62 72 77
18      07/21/2013,01 05 06 08 10 11 12 13 17 27 38 40 42 49 53 58 61 64 72 75
19      07/20/2013,06 07 11 12 16 19 23 29 39 40 41 44 56 67 69 70 73 74 77 79
20      07/19/2013,01 04 08 10 16 29 31 33 34 37 43 46 47 48 50 54 57 62 68 72
21      07/18/2013,06 07 09 23 24 30 34 41 43 46 48 53 60 65 66 68 70 75 76 77
22      07/17/2013,07 20 24 25 26 28 39 40 46 48 49 57 60 67 68 70 73 74 75 78
```

```
07/16/2013,06 07 15 18 22 24 27 28 29 31 35 43 45 46 56 57 58 60 64 72
07/15/2013,05 08 22 27 29 37 41 42 44 51 54 55 58 62 64 69 70 72 73 75
07/14/2013,05 07 14 18 27 29 32 35 39 41 45 47 49 60 62 65 67 70 75 80
07/13/2013,01 03 08 10 18 21 24 25 38 42 46 47 49 51 56 58 59 73 76 77
07/12/2013,07 11 12 14 15 17 19 21 22 27 29 37 42 45 47 50 52 64 74 76
07/11/2013,02 07 09 10 11 12 15 17 19 25 32 37 45 57 58 59 61 64 69 77
07/10/2013,01 03 06 11 17 18 21 24 32 41 45 49 55 58 63 71 72 73 74 80
07/09/2013,02 04 05 10 14 18 22 28 29 31 32 36 39 40 48 53 55 57 58 78
07/08/2013,09 11 16 26 34 37 38 39 45 46 47 52 56 58 60 62 65 70 75 77
07/07/2013,02 03 07 09 13 15 16 20 21 23 29 48 50 51 57 67 68 71 78 79
07/06/2013,03 08 09 16 17 25 26 29 33 42 49 51 52 56 57 60 64 67 73 75
07/05/2013,04 08 09 11 13 15 18 24 31 37 40 47 50 56 66 67 73 77 78 80
07/04/2013,02 09 13 16 18 32 33 36 38 42 43 44 55 56 57 68 69 70 73 74
07/03/2013,01 05 12 16 21 31 32 33 38 49 50 51 53 56 58 69 75 77 78 79
07/02/2013,05 08 13 14 17 19 24 32 35 38 41 46 51 57 64 65 71 74 76 79
07/01/2013,02 04 05 09 11 14 16 17 22 28 30 38 50 54 60 64 66 67 74 77
06/30/2013,03 04 07 09 11 18 19 22 25 30 31 35 39 50 57 62 65 75 79 80
06/29/2013,09 11 14 21 26 30 32 33 37 42 43 44 48 54 59 60 66 68 70 78
06/28/2013,04 06 14 20 23 24 28 29 30 31 37 40 51 57 58 59 64 67 68 73
06/27/2013,01 15 16 18 20 31 34 41 44 47 51 60 62 64 65 67 70 71 72 74
06/26/2013,03 14 15 23 26 28 31 32 43 45 56 59 60 62 63 72 73 75 79 80
06/25/2013,05 11 13 21 23 26 27 33 41 44 48 50 64 66 67 70 73 74 75 76
06/24/2013,01 04 07 11 21 27 32 37 40 43 45 46 49 60 63 72 73 74 79 80
06/23/2013,03 05 08 14 17 21 23 25 26 28 33 37 40 45 49 57 61 69 71 79
06/22/2013,08 09 11 14 19 21 25 32 33 34 35 40 46 47 64 68 70 75 76 80
06/21/2013,04 06 07 13 14 15 17 22 23 27 29 35 42 47 57 58 62 63 72 80
06/20/2013,05 07 09 24 29 30 31 34 35 49 53 58 61 63 66 70 71 73 77 79
06/19/2013,01 03 08 11 13 15 16 19 25 27 28 29 32 35 63 68 71 77 79 80
06/18/2013,05 12 13 14 18 19 21 22 23 26 31 33 42 45 48 53 57 60 69 71
06/17/2013,02 06 14 15 19 20 24 27 31 34 35 39 40 54 55 62 63 67 68 74
06/16/2013,02 04 13 19 28 29 30 31 34 43 44 47 49 53 59 63 70 71 79 80
06/15/2013,02 08 09 14 15 16 17 25 29 36 39 40 48 54 58 60 65 67 77 80
06/14/2013,05 12 14 20 27 28 31 33 36 40 45 46 53 57 62 65 66 68 77 78
06/13/2013,07 09 11 19 24 29 32 35 38 45 47 48 54 58 59 66 67 73 76 78
06/12/2013,03 08 12 15 23 28 33 42 48 51 52 59 60 61 63 65 69 72 73 77
06/11/2013,01 05 07 08 12 19 27 33 35 41 48 50 54 57 66 69 71 73 74 75
06/10/2013,01 03 09 10 12 16 21 23 24 26 27 31 33 47 57 68 70 71 79 80
06/09/2013,09 11 12 13 14 19 23 32 36 39 40 47 56 57 59 61 67 69 76 77
06/08/2013,03 11 16 17 22 23 28 34 35 37 38 41 47 50 57 60 64 72 75 77
06/07/2013,01 03 07 10 14 19 26 30 41 42 46 56 57 62 65 68 71 74 75 78
06/06/2013,03 10 11 28 31 34 38 39 45 46 48 50 51 54 55 57 60 69 72 80
06/05/2013,07 11 12 14 25 27 35 38 44 45 48 49 54 58 59 60 66 67 75 76
06/04/2013,03 06 08 12 13 14 27 30 45 49 53 56 57 58 60 61 65 71 73 80
06/03/2013,03 05 18 22 25 26 31 33 34 36 40 42 44 47 52 59 60 65 67 79
06/02/2013,02 03 06 11 15 17 38 43 44 46 48 52 53 55 64 66 70 71 73 74
06/01/2013,02 03 09 11 17 18 20 22 27 30 32 40 51 53 59 67 69 72 77 78
05/31/2013,01 05 06 08 10 13 18 19 29 35 36 42 45 51 59 64 66 68 71 73
05/30/2013,02 05 09 17 21 23 24 30 39 41 42 44 45 53 56 62 65 68 73 78
05/29/2013,01 03 08 17 20 21 22 25 26 37 46 48 51 56 59 61 62 73 74 76
05/28/2013,01 05 07 13 17 20 21 30 33 38 44 47 53 54 62 67 68 76 77 79
05/27/2013,02 05 11 15 23 26 32 37 38 39 57 58 59 60 61 64 66 67 72 73
05/26/2013,05 16 22 23 25 27 30 34 39 45 50 58 60 61 64 68 70 72 74 75
05/25/2013,02 06 09 12 13 19 20 22 25 27 31 33 34 42 48 55 58 62 72 77
05/24/2013,04 06 10 15 22 23 25 26 27 29 36 37 39 48 56 62 69 72 78 80
```

```
77      05/23/2013,01 04 10 13 16 39 41 42 43 44 45 49 53 54 55 59 63 69 74 79
78      05/22/2013,02 05 08 12 22 23 29 31 33 47 48 49 60 63 66 70 71 74 78 79
79      05/21/2013,06 07 16 22 24 25 28 36 38 41 42 45 48 50 55 59 66 70 71 76
80      05/20/2013,05 07 17 18 25 26 37 42 43 44 47 51 53 61 69 71 72 74 78 79
81      05/19/2013,04 06 10 14 15 24 25 28 43 49 51 53 54 59 62 68 69 75 76 80
82      05/18/2013,01 03 13 14 17 26 27 28 37 43 45 54 57 58 61 66 69 74 76 77
83      05/17/2013,03 04 07 09 10 23 36 37 41 45 48 52 58 64 68 69 70 72 73 76
84      05/16/2013,11 12 15 18 24 26 28 30 32 36 37 38 42 43 53 54 62 68 75 77
85      05/15/2013,04 06 09 13 23 28 29 30 31 32 35 38 46 57 62 67 69 71 74 80
86      05/14/2013,01 04 07 18 21 24 33 37 39 47 48 49 61 66 68 71 72 75 77 80
87      05/13/2013,02 03 10 12 15 19 29 31 35 39 43 47 48 50 62 67 74 77 78 80
88      05/12/2013,02 09 10 11 17 21 22 25 27 29 31 32 36 41 64 66 68 75 79 80
89      05/11/2013,03 07 08 14 15 16 18 23 26 30 37 38 43 53 55 65 71 75 78 79
90      05/10/2013,03 04 08 09 11 18 22 26 28 32 39 42 44 48 51 54 58 60 75 79
91      05/09/2013,04 05 08 10 11 13 15 23 28 29 42 43 45 49 50 53 60 62 63 66
92      05/08/2013,01 05 09 10 16 17 19 25 26 33 34 37 39 43 47 48 53 71 75 79
93      05/07/2013,02 12 14 18 22 27 33 34 35 39 49 50 51 52 61 67 69 71 72 77
94      05/06/2013,02 08 11 13 15 19 23 28 33 35 42 44 45 47 48 55 66 74 78 79
95      05/05/2013,03 04 05 06 23 31 33 35 38 43 45 48 50 54 60 63 65 66 79 80
96      05/04/2013,11 12 21 27 29 30 31 33 39 42 47 50 56 57 64 68 70 76 77 79
97      05/03/2013,04 07 12 16 19 20 23 28 29 38 40 48 52 53 58 62 63 71 72 78
98      05/02/2013,02 04 06 08 13 17 18 22 23 34 36 40 43 48 50 55 62 64 68 75
99      05/01/2013,04 05 06 08 16 17 23 25 30 31 35 43 45 53 58 61 62 63 69 75
00      04/30/2013,03 06 10 11 19 20 24 35 36 41 42 47 58 60 66 68 72 73 74 78
01      04/29/2013,22 24 28 29 33 34 38 42 49 51 52 57 63 64 66 67 70 72 77 79
02      04/28/2013,03 05 07 10 13 18 21 26 33 35 36 41 42 44 48 51 59 64 70 73
03      04/27/2013,04 05 08 14 19 25 26 29 37 45 47 59 63 66 68 69 70 71 72 74
04      04/26/2013,03 04 11 13 15 22 24 26 29 30 39 40 44 45 47 57 70 72 73 75
05      04/25/2013,03 08 11 21 24 27 37 45 49 51 54 56 57 58 62 64 66 67 76 77
06      04/24/2013,03 04 06 16 17 24 26 31 32 33 34 42 48 49 50 57 58 59 62 80
07      04/23/2013,06 11 16 19 20 22 36 45 46 47 50 52 53 54 58 59 61 62 69 74
08      04/22/2013,01 03 06 11 12 13 16 17 22 25 26 29 31 43 44 48 50 58 70 79
09      04/21/2013,02 04 12 13 20 23 26 32 43 46 50 57 58 60 61 63 66 69 70 72
10      04/20/2013,01 04 06 07 11 13 18 28 35 37 39 40 42 47 52 62 67 72 73 76
11      04/19/2013,01 03 05 06 07 17 22 26 29 36 41 43 48 58 60 62 65 66 78 80
12      04/18/2013,06 08 12 14 19 20 23 25 30 31 33 35 36 37 42 57 71 72 74 77
13      04/17/2013,01 05 06 10 16 19 30 36 44 46 48 50 52 56 58 62 65 69 73 76
14      04/16/2013,01 03 04 09 11 13 15 17 21 25 28 29 36 40 54 61 62 65 69 79
15      04/15/2013,08 13 14 18 24 28 29 31 35 43 46 47 48 54 59 60 63 70 78 79
16      04/14/2013,04 07 09 10 21 26 29 44 45 48 50 56 57 60 66 67 74 75 76 77
17      04/13/2013,01 07 13 18 19 22 30 31 32 34 46 47 56 58 61 62 63 64 73 76
18      04/12/2013,05 07 11 26 27 28 33 34 36 44 45 46 49 56 59 60 73 75 76 79
19      04/11/2013,01 07 09 10 39 40 44 47 49 51 53 55 58 62 63 65 66 67 76 79
20      04/10/2013,02 04 06 10 12 13 23 26 27 30 33 45 47 49 53 57 59 61 74 76
21      04/09/2013,01 02 07 08 11 29 30 33 36 38 39 41 48 49 52 55 64 65 69 74
22      04/08/2013,03 04 08 09 14 15 22 24 31 34 38 39 49 51 61 62 65 66 68 69
23      04/07/2013,05 06 07 12 18 31 43 44 48 50 51 52 55 57 58 61 66 68 72 80
24      04/06/2013,05 07 09 12 14 20 25 34 35 43 47 48 50 51 62 64 69 70 78 79
25      04/05/2013,02 03 06 11 13 15 17 28 32 35 37 38 49 53 60 68 76 77 79 80
26      04/04/2013,02 03 21 23 26 29 30 35 47 48 55 57 60 62 65 68 73 77 78 79
27      04/03/2013,04 06 09 14 25 27 28 29 31 33 35 36 39 41 43 53 55 69 70 80
28      04/02/2013,05 08 13 16 23 30 31 33 34 35 36 37 39 40 52 53 63 68 70 77
29      04/01/2013,01 02 03 04 06 12 13 17 23 26 38 39 42 54 56 57 59 60 66 71
30      03/31/2013,03 04 06 17 18 20 22 30 32 43 48 52 56 58 61 62 64 68 76 80
```

```
03/30/2013,02 03 05 06 08 09 10 11 21 27 39 42 48 52 55 56 60 64 72 80
03/29/2013,05 16 22 24 26 31 33 38 42 44 47 49 50 64 65 66 67 73 74 76
03/28/2013,05 09 21 22 25 29 30 32 36 40 46 51 53 54 57 60 61 68 75 80
03/27/2013,01 09 14 16 18 19 23 29 35 36 44 47 52 53 57 58 59 61 63 76
03/26/2013,03 07 13 14 21 25 28 29 43 44 51 54 57 62 65 67 69 72 78 80
03/25/2013,02 07 11 12 14 19 27 28 30 33 49 50 54 60 65 70 74 76 77 80
03/24/2013,01 10 17 22 25 33 40 41 45 46 49 51 52 57 59 67 70 75 77 78
03/23/2013,02 09 12 13 17 19 24 26 34 37 38 43 55 56 57 60 61 69 73 74
03/22/2013,01 02 03 06 08 11 12 20 29 32 35 36 39 41 45 57 65 66 71 76
03/21/2013,02 10 11 17 19 22 24 26 28 35 39 47 50 57 60 61 63 64 66 77
03/20/2013,03 09 10 13 15 20 29 41 45 46 47 50 54 56 65 66 67 70 73 78
03/19/2013,01 13 18 20 25 26 35 36 39 40 41 43 47 52 53 54 59 68 71 73
03/18/2013,04 10 12 13 15 19 20 22 23 25 34 36 38 41 43 47 48 50 67 74
03/17/2013,02 04 06 08 09 10 16 20 21 42 47 57 59 60 64 68 75 76 79 80
03/16/2013,04 06 10 15 21 23 28 29 32 33 36 46 48 59 64 65 66 69 70 71
03/15/2013,08 20 21 22 35 40 41 44 48 50 51 55 58 61 68 74 75 76 78 80
03/14/2013,04 06 14 16 20 24 26 27 28 35 44 45 53 57 60 62 65 70 72 78
03/13/2013,06 17 18 19 20 25 26 42 43 47 49 57 60 62 64 65 67 68 76 80
03/12/2013,11 17 19 21 22 26 31 40 43 50 56 59 60 62 63 70 72 75 79 80
03/11/2013,02 06 09 15 18 21 23 28 30 32 35 36 50 52 61 62 68 76 78 79
03/10/2013,04 09 10 11 14 22 26 27 28 30 31 33 47 49 54 61 62 65 72 78
03/09/2013,06 12 13 24 27 31 33 35 37 43 50 52 53 59 60 63 64 65 69 80
03/08/2013,12 18 24 25 28 36 41 42 47 48 49 50 54 56 61 62 64 71 73 79
03/07/2013,01 06 13 22 26 30 36 38 39 40 47 49 50 52 56 64 65 66 69 77
03/06/2013,03 06 08 13 16 27 30 31 34 43 51 55 59 61 64 72 75 76 78 79
03/05/2013,01 03 04 06 12 16 30 32 36 37 42 45 48 49 50 55 56 57 61 62
03/04/2013,02 04 06 07 10 14 16 17 24 26 28 41 56 58 59 63 64 69 73 74
03/03/2013,06 08 17 20 23 27 28 30 37 39 47 49 54 62 68 69 74 75 78 79
03/02/2013,18 25 37 38 40 45 50 52 54 55 56 66 67 68 69 70 71 75 76 78
03/01/2013,02 10 11 15 18 22 27 32 33 35 41 46 60 61 68 69 74 76 77 78
02/28/2013,04 06 09 12 13 20 22 27 28 29 44 54 62 65 66 69 72 73 74 77
02/27/2013,05 10 12 16 18 27 30 32 33 36 37 38 39 40 42 43 52 58 62 78
02/26/2013,14 17 20 27 33 40 45 47 48 49 52 53 60 61 62 65 67 70 74 79
02/25/2013,01 07 10 22 25 29 30 35 38 44 47 52 53 54 58 63 71 74 77 80
02/24/2013,01 02 11 23 32 39 41 42 45 47 49 51 53 54 58 66 68 70 72 73
02/23/2013,03 05 06 08 17 36 40 43 47 48 50 61 64 67 69 70 71 72 75 78
02/22/2013,02 09 14 21 23 24 37 39 45 49 51 52 54 57 60 66 68 69 74 76
02/21/2013,01 06 10 18 20 24 28 31 34 43 45 46 51 62 64 70 74 75 77 80
02/20/2013,03 08 09 14 18 24 32 38 39 46 49 50 52 58 59 60 63 66 73 78
02/19/2013,05 08 14 15 21 26 27 38 39 40 44 48 52 53 56 57 66 67 70 73
02/18/2013,02 05 06 13 14 18 20 25 28 29 30 33 53 57 60 66 67 69 79 80
02/17/2013,02 03 11 14 15 21 23 26 28 40 46 50 53 54 60 67 68 74 76 78
02/16/2013,03 04 12 19 21 25 28 32 36 38 48 53 54 55 59 63 66 67 68 73
02/15/2013,04 11 15 20 24 28 29 30 33 35 38 47 48 50 53 55 62 67 68 76
02/14/2013,02 13 14 15 20 22 23 27 30 35 36 39 43 50 53 59 61 69 73 74
02/13/2013,01 07 09 13 16 17 25 26 28 30 33 40 48 49 53 60 61 65 67 76
02/12/2013,04 05 06 11 13 15 23 24 25 37 39 42 46 47 48 53 57 59 73 77
02/11/2013,05 06 09 11 17 24 30 32 33 36 43 47 54 56 57 58 59 66 69 76
02/10/2013,05 20 22 24 26 27 33 38 42 43 44 46 48 52 53 68 74 76 78 79
02/09/2013,04 05 07 13 19 24 30 38 42 43 49 54 55 57 59 60 62 75 77 79
02/08/2013,02 09 13 17 18 23 26 29 33 36 37 39 45 46 49 52 61 67 70 74
02/07/2013,03 08 09 15 16 24 25 26 27 40 42 46 49 55 58 59 61 64 75 77
02/06/2013,02 04 05 07 09 12 20 23 30 31 40 55 58 63 64 69 70 77 79 80
02/05/2013,07 09 12 16 17 18 29 35 38 39 44 47 50 51 54 58 60 62 64 67
```

```
02/04/2013,01 02 03 08 14 17 21 24 26 32 34 35 47 49 52 55 57 73 74 76
02/03/2013,01 03 06 07 12 17 18 20 25 33 34 37 51 53 54 57 58 59 64 69
02/02/2013,04 08 11 14 16 21 25 33 35 38 40 41 45 47 48 56 57 67 75 78
02/01/2013,02 12 14 15 19 23 27 40 42 44 47 49 51 54 55 59 62 64 68 73
01/31/2013,02 08 09 10 15 19 21 23 24 34 41 42 49 51 54 61 66 71 74 77
01/30/2013,10 11 21 24 26 30 35 36 40 42 47 50 51 56 58 61 63 77 78 80
01/29/2013,02 06 08 12 18 20 22 32 34 36 48 49 51 60 62 68 69 70 75 78
01/28/2013,03 04 08 12 15 21 29 33 34 37 40 43 56 61 62 75 76 77 78 79
01/27/2013,01 04 05 16 24 25 27 31 35 37 38 40 41 43 48 59 60 61 65 78
01/26/2013,02 09 14 15 16 21 25 27 29 41 50 57 59 60 69 70 72 73 75 76
01/25/2013,01 03 05 06 07 09 14 17 28 30 33 36 46 51 52 56 61 64 69 79
01/24/2013,07 09 10 22 23 24 25 26 39 42 43 48 55 59 60 61 63 67 70 75
01/23/2013,06 07 15 24 27 29 32 34 42 46 49 52 55 56 59 60 68 76 77 79
01/22/2013,02 05 07 10 19 22 28 34 35 36 37 38 45 49 50 54 55 60 61 74
01/21/2013,02 03 07 15 16 17 25 32 34 41 48 50 55 57 59 62 64 75 77 80
01/20/2013,02 05 07 15 16 27 39 40 41 46 47 50 55 57 60 66 67 70 72 80
01/19/2013,02 07 09 14 17 19 34 35 36 37 40 43 53 54 55 57 68 72 75 79
01/18/2013,08 11 14 16 29 31 32 35 37 40 44 48 52 53 58 61 64 66 70 72
01/17/2013,03 04 08 09 10 12 19 23 25 28 31 37 42 44 50 53 67 71 76 80
01/16/2013,03 07 08 14 19 24 25 27 28 32 36 53 55 63 69 75 76 77 78 80
01/15/2013,11 13 20 27 30 36 38 39 48 52 56 60 65 68 69 70 72 73 74 80
01/14/2013,02 03 06 08 10 11 16 18 21 27 29 30 42 44 45 50 52 58 60 61
01/13/2013,01 02 05 07 09 11 12 14 25 26 28 31 32 34 36 56 60 67 72 74
01/12/2013,02 03 10 12 19 21 22 23 24 26 30 33 39 45 52 62 63 64 72 74
01/11/2013,02 07 08 14 16 17 43 47 48 50 52 55 56 60 61 64 73 74 76 78
01/10/2013,13 15 21 24 27 31 38 41 43 48 52 53 61 63 65 67 72 75 77 79
01/09/2013,03 04 06 16 20 21 22 25 28 30 45 52 53 54 56 62 63 66 77 80
01/08/2013,02 03 09 12 16 20 23 28 31 33 35 36 44 49 53 60 71 72 76 77
01/07/2013,04 07 13 15 19 24 25 30 36 40 41 43 48 52 56 58 63 69 78 80
01/06/2013,01 05 12 15 26 29 33 36 41 44 46 48 51 52 58 60 64 66 74 78
01/05/2013,06 09 16 17 18 25 27 29 32 33 46 47 48 49 59 60 62 68 70 80
01/04/2013,03 12 13 16 19 21 24 28 33 35 38 39 46 54 56 62 63 65 68 78
01/03/2013,02 04 05 10 13 20 28 30 34 38 40 52 55 56 57 58 64 66 72 75
01/02/2013,03 05 08 16 17 19 24 26 30 34 41 42 45 46 52 55 68 72 74 79
01/01/2013,06 11 17 20 21 22 23 30 36 45 46 48 53 55 57 63 65 66 69 77
12/31/2012,08 12 22 24 26 32 36 40 50 53 57 59 60 65 68 69 71 73 75 77
12/30/2012,03 06 07 19 22 23 25 27 28 34 35 41 50 56 59 67 72 75 76 78
12/29/2012,05 06 08 09 12 16 18 22 26 27 32 40 48 51 56 58 61 67 73 79
12/28/2012,03 04 07 16 22 24 27 33 41 42 45 49 51 54 58 60 62 64 67 68
12/27/2012,02 04 05 08 09 13 17 18 21 24 30 31 34 46 54 59 64 73 75 79
12/26/2012,14 18 19 22 23 25 26 27 30 36 39 40 46 47 48 50 63 65 72 80
12/25/2012,01 02 03 09 12 22 24 25 28 38 41 51 53 56 57 61 65 69 73 77
12/24/2012,01 03 04 05 09 16 18 19 20 28 30 39 46 48 50 61 62 76 77 78
12/23/2012,02 06 11 12 14 16 20 25 32 35 36 41 48 57 58 61 64 71 78 80
12/22/2012,04 10 16 17 24 26 27 34 43 44 46 50 52 58 59 63 68 72 74 77
12/21/2012,04 05 07 14 18 22 23 27 33 34 39 43 48 49 57 59 62 75 76 79
12/20/2012,03 20 21 26 28 30 34 39 40 42 49 58 61 62 64 71 72 73 75 80
12/19/2012,07 09 13 15 20 25 26 27 29 34 39 46 47 53 55 56 59 62 68 76
12/18/2012,04 08 13 14 19 29 38 44 45 53 57 58 60 64 66 69 71 75 78 80
12/17/2012,06 09 11 19 21 23 30 31 32 33 35 36 38 39 47 52 53 57 65 69
12/16/2012,16 17 18 19 20 22 27 29 33 37 41 44 51 54 56 58 59 71 72 78
12/15/2012,07 21 22 25 27 30 31 34 37 46 49 53 57 61 64 71 72 76 78 79
12/14/2012,03 04 08 19 24 26 34 43 44 45 46 49 53 57 63 64 65 66 74 79
12/13/2012,01 03 05 08 10 13 14 19 23 28 33 41 47 49 50 57 61 63 65 69
```

```
12/12/2012,04 05 07 14 18 26 31 37 43 44 46 47 54 55 59 60 62 67 69 73
12/11/2012,10 16 20 25 30 42 44 45 46 48 56 58 59 60 61 67 68 70 71 73
12/10/2012,04 05 06 08 09 24 28 29 34 36 53 56 58 60 62 63 66 74 76 78
12/09/2012,01 02 05 08 12 15 19 21 22 23 31 32 39 43 51 53 58 65 73 75
12/08/2012,01 02 06 08 09 10 11 12 17 18 23 24 25 26 30 36 62 71 73 76
12/07/2012,08 13 20 21 25 32 36 37 39 47 57 62 63 65 71 75 76 77 79 80
12/06/2012,13 15 17 18 20 26 30 34 43 49 50 51 54 56 57 62 72 76 77 78
12/05/2012,05 06 12 13 21 24 25 26 27 32 33 51 54 55 58 59 63 64 72 75
12/04/2012,02 10 12 13 19 21 27 31 33 36 39 42 44 48 52 62 67 68 74 77
12/03/2012,11 18 23 39 40 41 43 45 46 48 50 51 56 59 61 69 71 76 79 80
12/02/2012,02 13 17 20 21 24 27 28 30 31 33 36 37 40 41 43 63 64 71 79
12/01/2012,02 08 13 15 29 39 42 53 57 58 59 62 67 68 69 70 71 74 77 79
11/30/2012,04 10 12 18 19 20 28 33 40 43 45 52 63 65 66 67 69 70 76 79
11/29/2012,02 03 04 07 11 15 32 35 38 40 45 49 53 59 65 67 71 75 77 79
11/28/2012,02 05 07 12 13 19 24 30 34 41 42 45 47 53 56 62 69 72 74 77
11/27/2012,01 10 14 15 16 26 28 30 44 48 50 55 56 61 62 63 66 68 69 79
11/26/2012,08 12 13 15 16 21 23 28 31 33 36 40 41 45 47 50 58 61 75 76
11/25/2012,02 05 09 16 21 24 25 27 30 33 34 41 42 50 54 56 61 66 70 72
11/24/2012,01 06 09 16 20 28 31 32 33 35 41 43 44 51 65 68 70 71 72 77
11/23/2012,01 06 17 18 20 22 28 29 30 35 36 37 39 41 45 48 51 65 69 70
11/22/2012,01 05 06 09 16 20 22 32 33 40 50 51 54 57 59 66 67 74 78 79
11/21/2012,01 02 13 15 19 24 27 30 32 37 38 40 41 45 49 54 55 61 74 75
11/20/2012,06 09 10 11 23 26 27 29 43 49 52 54 55 56 60 61 62 65 69 72
11/19/2012,02 03 05 08 09 11 14 16 24 25 26 28 37 45 49 52 58 73 75 78
11/18/2012,02 06 07 09 15 18 23 27 29 32 33 35 51 55 58 66 68 70 79 80
11/17/2012,02 04 08 09 20 27 28 37 39 40 42 49 61 63 64 67 71 73 74 80
11/16/2012,02 04 15 24 31 35 36 37 49 50 52 59 61 63 67 70 71 75 77 78
11/15/2012,11 12 13 14 17 21 23 26 32 38 52 57 59 60 62 66 67 68 75 76
11/14/2012,10 11 13 15 18 20 22 24 26 27 33 38 44 47 48 53 56 64 65 67
11/13/2012,01 06 13 17 22 23 32 35 36 42 44 48 49 52 53 58 63 67 69 75
11/12/2012,04 14 15 16 24 29 32 34 35 36 42 43 45 49 54 56 59 64 73 75
11/11/2012,01 04 05 07 12 13 15 18 21 22 32 34 35 36 50 58 60 63 76 77
11/10/2012,08 09 11 13 15 25 30 31 32 37 40 41 45 48 50 54 66 67 68 79
11/09/2012,01 11 12 15 16 24 25 27 31 33 40 41 48 51 53 56 59 63 68 78
11/08/2012,04 06 07 09 13 17 20 29 35 36 37 41 43 56 59 61 65 70 72 77
11/07/2012,01 04 10 16 18 25 28 30 36 40 45 46 48 63 67 70 74 77 78 80
11/06/2012,01 04 08 13 16 26 33 34 40 46 47 49 50 53 55 60 61 62 63 76
11/05/2012,08 10 16 23 26 27 29 32 34 36 37 46 47 51 52 53 54 61 69 78
11/04/2012,03 05 07 14 18 27 30 33 35 44 50 60 61 62 64 66 67 68 71 76
11/03/2012,02 03 06 16 26 33 35 37 44 51 55 56 57 60 63 67 70 71 74 77
11/02/2012,06 10 11 16 29 33 34 35 39 43 46 50 53 56 64 69 75 76 79 80
11/01/2012,01 05 13 22 33 39 41 47 50 51 52 53 56 61 66 71 72 73 75 80
10/31/2012,01 09 13 19 27 32 35 36 38 39 50 56 59 60 63 65 74 77 78 79
10/30/2012,07 11 12 14 15 17 24 34 36 38 40 44 45 46 49 57 58 59 63 70
10/29/2012,04 11 12 17 19 26 29 31 33 35 37 40 42 45 47 54 58 62 70 76
10/28/2012,02 03 09 15 22 23 26 29 31 36 40 41 46 50 57 58 69 73 78 79
10/27/2012,03 08 10 18 21 24 28 31 35 36 41 46 49 55 59 61 74 77 78 79
10/26/2012,01 02 03 06 08 14 22 28 29 40 42 43 50 53 56 63 66 68 76 77
10/25/2012,04 22 26 30 32 33 38 39 46 50 51 54 55 62 63 65 67 68 69 77
10/24/2012,01 03 05 14 15 17 24 29 30 40 45 51 54 58 65 68 69 71 77 80
10/23/2012,01 05 09 28 30 33 37 45 49 50 57 61 63 65 66 67 68 76 77 79
10/22/2012,04 05 07 08 12 19 25 26 29 34 38 43 44 45 50 51 53 58 59 77
10/21/2012,01 05 07 09 15 16 22 29 30 36 38 41 43 45 50 55 56 60 61 72
10/20/2012,01 08 17 21 30 31 32 33 34 35 45 48 54 59 65 66 69 73 78 80
```

```
93      10/19/2012,09 11 17 20 21 22 23 28 30 39 45 51 54 59 60 65 68 69 70 75
94      10/18/2012,11 14 15 18 21 23 25 27 28 30 32 38 41 46 47 53 59 63 72 78
95      10/17/2012,02 03 08 21 22 24 36 37 42 44 46 52 59 60 62 67 68 69 72 79
96      10/16/2012,04 09 11 16 17 20 23 24 33 34 41 42 52 57 61 68 69 76 78 80
97      10/15/2012,01 03 08 11 13 18 19 20 27 33 38 44 49 52 59 62 63 70 72 75
98      10/14/2012,01 03 14 15 18 23 24 30 31 33 34 38 46 50 54 58 61 67 72 80
99      10/13/2012,02 08 09 11 15 26 27 40 41 43 44 49 52 57 62 66 67 72 79 80
00      10/12/2012,05 10 11 15 19 25 30 31 32 33 36 40 42 52 59 61 70 72 77 80
01      10/11/2012,05 06 08 11 12 13 16 20 23 25 26 32 33 36 47 50 59 72 75 79
02      10/10/2012,16 19 20 22 23 37 39 43 45 47 48 57 58 60 65 67 69 71 72 80
03      10/09/2012,01 03 05 17 23 24 27 28 29 34 41 52 53 61 63 70 71 72 73 79
04      10/08/2012,01 03 05 15 21 22 23 26 30 34 35 40 47 49 52 55 60 62 64 66
05      10/07/2012,04 16 17 21 23 27 33 40 43 44 45 49 52 53 54 55 60 63 70 79
06      10/06/2012,02 03 11 13 15 26 35 41 42 45 46 47 52 54 59 60 62 66 69 72
07      10/05/2012,05 19 20 21 24 28 35 36 38 44 46 48 49 52 54 55 62 63 69 70
08      10/04/2012,06 10 12 18 21 22 24 28 31 32 37 59 61 63 64 71 75 77 79 80
09      10/03/2012,05 07 08 22 23 25 27 30 37 38 40 46 51 54 60 61 62 70 76 78
10      10/02/2012,06 09 11 13 14 24 25 27 29 32 35 41 48 54 60 61 63 64 70 77
11      10/01/2012,13 18 28 29 31 39 40 41 47 49 53 56 60 62 63 64 67 72 73 77
12      09/30/2012,03 05 22 26 28 31 38 47 52 53 56 57 62 63 67 72 74 75 76 78
13      09/29/2012,04 06 10 14 15 17 20 25 26 28 37 39 40 48 52 53 59 63 66 80
14      09/28/2012,01 03 09 15 19 24 27 29 39 40 42 44 47 48 64 67 71 74 75 78
15      09/27/2012,03 06 08 14 16 21 24 25 26 32 33 41 42 44 48 52 56 69 75 76
16      09/26/2012,02 03 08 12 18 23 24 26 27 35 37 38 42 45 50 52 58 65 76 80
17      09/25/2012,08 12 18 24 29 36 37 38 39 50 52 53 56 57 58 63 65 72 77 78
18      09/24/2012,02 06 11 13 14 17 19 31 36 37 39 50 51 54 56 57 67 69 72 78
19      09/23/2012,04 05 07 12 18 20 27 36 37 41 42 45 47 50 54 55 61 65 71 80
20      09/22/2012,03 07 10 11 19 22 24 29 30 34 38 50 59 61 64 69 73 77 78 80
21      09/21/2012,02 04 09 17 19 22 24 29 43 44 53 56 57 66 71 72 76 77 78 80
22      09/20/2012,04 05 10 11 12 13 16 24 32 33 39 41 44 46 54 66 67 70 76 77
23      09/19/2012,03 05 06 08 21 23 26 32 37 39 41 42 44 45 50 53 60 63 71 78
24      09/18/2012,05 14 15 21 24 27 28 29 31 34 37 42 47 49 52 53 61 62 66 73
25      09/17/2012,02 03 10 15 18 19 22 29 30 33 34 36 47 51 60 63 64 66 70 76
26      09/16/2012,03 10 11 15 16 18 20 28 31 34 38 43 46 51 54 65 70 71 78 80
27      09/15/2012,01 03 04 05 09 17 19 24 27 32 35 36 52 58 59 60 65 67 78 80
28      09/14/2012,02 11 14 19 21 22 26 33 39 40 41 43 53 54 55 56 60 63 66 76
29      09/13/2012,02 11 12 22 30 32 34 36 40 41 42 44 48 56 58 60 64 75 76 79
30      09/12/2012,01 02 04 05 07 12 16 24 26 27 38 50 51 53 63 67 73 74 75 77
31      09/11/2012,01 06 17 19 23 25 26 27 30 39 41 44 46 53 60 63 65 75 77 80
32      09/10/2012,16 23 43 44 47 48 52 53 55 56 58 62 63 65 66 71 72 76 78 79
33      09/09/2012,01 02 04 11 18 24 26 29 37 38 50 52 54 55 59 62 63 71 74 77
34      09/08/2012,03 06 07 09 10 23 26 27 28 35 39 44 47 48 55 56 60 65 71 80
35      09/07/2012,06 12 18 19 20 24 28 32 38 48 50 51 54 56 64 68 69 75 78 79
36      09/06/2012,03 08 10 12 16 20 24 25 29 32 34 35 36 40 47 53 63 66 70 76
37      09/05/2012,02 04 09 14 15 20 21 23 25 28 32 39 43 49 60 63 68 72 74 78
38      09/04/2012,01 08 10 11 12 21 22 24 26 31 33 35 40 47 49 52 58 60 63 74
39      09/03/2012,01 09 10 12 15 16 18 19 34 39 41 44 47 50 54 55 64 69 73 77
40      09/02/2012,02 03 07 08 10 12 13 17 29 35 38 43 45 48 60 61 63 74 75 79
41      09/01/2012,03 11 15 18 20 22 29 38 42 44 50 51 56 57 60 61 75 77 79 80
42      08/31/2012,04 13 17 18 23 24 27 28 29 32 34 41 42 46 60 63 68 75 79 80
43      08/30/2012,06 15 22 23 26 30 31 34 41 47 52 55 57 59 61 62 65 70 73 77
44      08/29/2012,01 10 17 18 21 25 27 34 35 37 38 39 41 43 49 50 54 62 69 74
45      08/28/2012,05 12 14 15 16 23 30 32 39 41 43 45 53 55 56 63 65 69 75 77
46      08/27/2012,05 08 10 12 15 17 26 30 34 40 42 46 49 52 59 69 70 72 75 80
```

```
47      08/26/2012,01 09 12 17 18 19 21 30 35 40 41 45 48 49 56 61 64 65 69 75
48      08/25/2012,02 07 16 21 22 25 30 37 39 43 46 55 59 66 67 70 71 73 74 77
49      08/24/2012,01 02 04 05 06 07 08 09 14 21 26 37 38 41 43 50 59 60 62 71
50      08/23/2012,03 07 09 12 14 19 24 25 27 33 44 48 49 50 51 52 55 62 68 79
51      08/22/2012,01 02 06 18 40 42 45 48 52 53 56 57 59 62 67 68 72 77 78 80
52      08/21/2012,04 05 10 13 16 18 19 26 27 33 40 42 45 55 57 58 66 70 72 80
53      08/20/2012,01 21 27 33 34 38 39 40 46 49 50 52 53 58 59 60 61 64 65 80
54      08/19/2012,07 08 09 11 12 15 17 19 21 28 32 40 46 50 54 64 67 70 72 76
55      08/18/2012,01 04 10 14 20 25 30 32 45 46 47 49 51 53 55 59 63 64 74 80
56      08/17/2012,01 04 05 06 13 15 16 18 21 26 31 33 37 42 44 50 60 68 69 72
57      08/16/2012,01 04 05 07 09 13 19 21 22 23 39 43 49 56 60 61 66 68 75 80
58      08/15/2012,01 10 11 12 15 18 21 38 39 42 45 47 48 49 52 57 61 66 74 76
59      08/14/2012,01 06 09 10 13 14 20 22 24 30 35 42 44 60 62 64 66 68 73 80
60      08/13/2012,03 21 24 25 26 29 30 31 32 33 36 40 42 62 63 65 67 69 78 80
61      08/12/2012,04 08 09 14 16 33 34 35 42 43 47 57 59 60 62 63 65 66 72 80
62      08/11/2012,05 08 11 14 25 29 31 35 36 40 44 53 55 60 66 67 68 69 70 79
63      08/10/2012,04 05 08 13 16 19 20 22 27 32 33 35 37 48 49 50 54 56 65 78
64      08/09/2012,03 11 13 21 24 25 34 36 40 42 43 44 48 51 54 59 61 69 72 79
65      08/08/2012,01 07 09 11 14 15 23 25 30 35 38 39 49 51 52 56 63 66 69 78
66      08/07/2012,01 05 21 24 30 31 33 35 37 44 47 50 51 53 55 57 60 61 65 73
67      08/06/2012,05 06 17 18 21 24 31 32 34 36 39 45 47 57 61 62 66 70 76 77
68      08/05/2012,06 08 16 18 20 21 22 26 33 34 35 38 41 44 62 64 68 69 72 78
69      08/04/2012,03 12 20 21 35 36 39 42 47 48 54 56 57 58 62 66 67 72 74 76
70      08/03/2012,04 05 08 09 13 14 18 20 21 25 31 35 37 39 44 51 52 55 74 76
71      08/02/2012,02 05 15 17 19 21 29 42 45 50 54 60 64 66 68 70 71 72 74 78
72      08/01/2012,04 07 08 19 22 24 25 28 40 41 45 46 47 53 60 61 66 68 73 78
73      07/31/2012,03 14 15 21 24 32 39 40 48 49 56 57 62 64 66 67 68 73 74 75
74      07/30/2012,01 10 12 13 18 33 34 39 41 49 56 59 61 62 66 69 70 71 78 80
75      07/29/2012,05 07 13 18 19 21 27 28 29 33 36 38 39 44 47 55 65 71 78 80
76      07/28/2012,02 05 07 11 17 18 25 29 40 45 51 52 55 57 62 63 69 74 77 80
77      07/27/2012,04 05 09 10 22 26 29 38 41 43 50 52 56 59 62 67 71 75 76 80
78      07/26/2012,02 05 11 14 22 29 31 32 35 38 41 51 52 61 62 65 67 69 72 80
79      07/25/2012,07 10 11 14 16 21 24 26 28 32 35 46 50 51 52 54 57 66 68 69
80      07/24/2012,04 05 11 12 15 22 33 39 42 43 53 55 56 59 63 65 67 70 73 75
81      07/23/2012,03 04 07 08 15 23 40 43 44 46 47 48 56 63 64 71 73 75 78 79
82      07/22/2012,02 06 13 15 16 19 23 39 43 48 50 53 54 57 58 59 64 68 71 78
83      07/21/2012,02 05 06 10 16 17 21 22 29 33 39 45 50 51 53 58 69 71 73 79
84      07/20/2012,02 03 12 13 15 17 26 27 34 35 37 38 40 41 54 58 60 69 74 78
85      07/19/2012,01 09 14 16 24 25 30 32 33 37 40 46 48 54 59 61 62 67 74 78
86      07/18/2012,03 05 08 13 15 21 27 30 31 42 45 46 47 49 50 54 61 66 67 71
87      07/17/2012,04 14 15 16 20 24 26 29 34 40 41 47 49 57 62 64 66 67 70 76
88      07/16/2012,05 08 09 15 17 22 23 35 37 41 42 45 47 55 61 64 66 69 72 77
89      07/15/2012,01 13 17 22 26 28 30 48 54 55 62 65 66 67 69 70 71 75 76 77
90      07/14/2012,01 03 07 08 10 18 29 31 39 42 46 51 54 57 58 59 62 69 73 79
91      07/13/2012,07 14 23 26 29 30 32 33 34 35 39 41 43 44 47 57 61 73 75 80
92      07/12/2012,05 06 10 21 22 28 35 39 41 46 48 49 53 56 59 65 72 73 74 78
93      07/11/2012,04 09 11 18 30 40 41 42 45 54 57 59 60 62 66 68 69 71 72 78
94      07/10/2012,01 04 06 17 22 26 28 29 32 37 38 39 46 51 57 61 63 65 71 80
95      07/09/2012,03 06 14 23 24 27 28 34 35 36 38 40 48 60 62 68 71 72 75 78
96      07/08/2012,09 10 12 13 15 16 22 28 37 41 42 45 50 52 53 58 60 70 74 75
97      07/07/2012,01 05 07 11 15 16 18 30 31 32 38 42 48 52 59 62 66 70 76 80
98      07/06/2012,02 03 05 06 07 09 10 11 21 22 32 35 37 45 49 54 58 60 72 79
99      07/05/2012,01 03 04 15 19 20 22 23 28 31 33 36 45 54 60 70 71 72 76 77
00      07/04/2012,01 02 03 26 34 38 39 41 43 47 49 50 52 55 56 59 62 66 67 77
```

01 07/03/2012,03 06 10 11 16 18 30 31 39 41 42 43 48 58 59 65 68 70 76 80
02 07/02/2012,02 03 07 17 22 23 33 41 43 55 56 59 64 65 67 69 73 76 77 78
03 07/01/2012,01 02 03 06 09 10 13 19 21 23 25 26 27 30 37 39 45 49 58 76
04 06/30/2012,05 07 08 13 19 21 23 25 28 30 33 38 50 60 63 65 70 76 78 80
05 06/29/2012,01 04 05 06 16 17 18 24 25 29 34 42 53 60 64 66 67 70 71 79
06 06/28/2012,03 05 08 09 11 12 17 19 23 24 32 40 41 44 46 48 53 66 72 77
07 06/27/2012,02 03 07 11 13 23 32 34 35 36 42 43 47 54 61 63 64 69 70 79
08 06/26/2012,03 07 12 13 26 27 33 34 44 46 48 52 58 65 68 70 73 74 76 78
09 06/25/2012,01 08 09 11 12 13 23 24 28 34 40 49 50 54 61 62 63 67 70 75
10 06/24/2012,07 12 18 23 25 29 32 35 43 45 47 48 56 67 68 69 71 74 75 76
11 06/23/2012,02 17 18 19 29 30 32 34 37 41 43 47 52 53 55 56 64 74 77 78
12 06/22/2012,04 11 13 14 21 22 28 30 34 39 48 53 65 70 72 74 75 77 78 80
13 06/21/2012,02 03 06 09 14 22 25 33 35 38 39 41 44 50 51 54 57 61 62 64
14 06/20/2012,02 03 04 05 10 11 20 30 32 33 35 36 39 41 44 56 71 74 75 80
15 06/19/2012,06 08 13 14 19 23 26 27 28 33 36 40 41 45 53 63 73 75 77 80
16 06/18/2012,03 18 21 22 27 29 32 38 45 51 55 57 59 61 64 65 69 70 76 79
17 06/17/2012,08 09 13 22 24 36 37 40 42 46 49 52 62 63 70 71 73 74 77 80
18 06/16/2012,10 11 15 16 23 25 30 37 40 44 45 46 48 50 51 53 61 65 66 76
19 06/15/2012,04 07 08 11 13 20 24 26 32 47 49 50 60 69 72 74 75 76 79 80
20 06/14/2012,04 07 21 27 38 39 40 47 53 54 56 58 63 64 70 71 72 74 76 78
21 06/13/2012,02 06 07 08 09 11 17 20 42 44 47 48 50 51 53 58 59 60 65 72
22 06/12/2012,03 05 15 23 25 30 31 34 40 47 54 55 61 62 63 65 66 74 76 78
23 06/11/2012,07 14 23 24 27 29 35 41 42 45 47 48 51 52 54 57 61 63 68 77
24 06/10/2012,02 03 05 17 19 27 31 37 38 42 47 54 55 61 64 67 68 70 77 79
25 06/09/2012,03 05 06 14 16 21 23 28 34 36 37 44 46 51 54 58 62 63 72 80
26 06/08/2012,09 12 18 19 20 22 28 36 38 41 45 47 54 61 63 67 69 75 76 78
27 06/07/2012,13 16 17 20 21 22 25 27 42 44 46 50 55 60 62 63 66 74 77 80
28 06/06/2012,03 11 16 19 26 27 31 33 40 43 48 49 51 55 60 63 65 66 76 80
29 06/05/2012,03 14 15 19 20 24 25 26 27 28 29 32 35 36 38 56 60 70 73 79
30 06/04/2012,01 04 05 12 13 15 16 17 22 28 33 36 42 50 52 61 62 68 73 77
31 06/03/2012,13 20 26 27 28 29 31 33 39 47 49 50 52 53 55 60 63 65 66 73
32 06/02/2012,08 12 13 14 24 26 31 33 35 38 39 43 47 49 54 56 66 74 76 77
33 06/01/2012,04 06 11 14 21 24 29 37 38 43 44 46 47 51 60 66 69 73 76 80
34 05/31/2012,03 07 13 18 20 25 28 36 40 44 47 49 53 56 59 67 68 72 73 80
35 05/30/2012,05 17 22 24 26 32 36 48 50 51 53 58 59 61 62 66 71 72 77 79
36 05/29/2012,09 10 14 17 18 19 20 22 29 32 40 43 48 49 54 55 57 59 73 77
37 05/28/2012,07 13 14 23 28 29 36 38 39 40 42 45 48 54 62 65 72 74 76 77
38 05/27/2012,03 04 10 16 17 20 24 28 30 31 33 37 42 45 50 68 70 74 76 78
39 05/26/2012,03 09 10 11 14 16 19 20 24 29 31 32 36 38 50 57 60 74 75 77
40 05/25/2012,04 10 13 15 16 21 23 24 26 30 40 45 48 54 63 65 66 69 74 75
41 05/24/2012,05 10 12 26 31 34 36 40 44 51 52 54 55 58 65 66 68 71 73 76
42 05/23/2012,02 09 11 14 16 24 25 28 32 33 36 42 45 47 55 61 71 73 75 79
43 05/22/2012,02 06 10 11 13 14 21 22 24 25 30 38 40 41 45 50 56 63 66 80
44 05/21/2012,01 04 06 09 10 14 16 20 27 29 35 40 45 48 50 61 63 72 73 75
45 05/20/2012,04 13 14 19 26 30 31 37 39 45 46 49 55 57 61 67 70 76 79 80
46 05/19/2012,03 05 07 12 14 20 29 31 32 38 39 55 58 60 62 66 71 74 76 79
47 05/18/2012,10 13 14 19 22 23 28 33 34 37 38 41 43 46 53 57 58 59 65 69
48 05/17/2012,02 06 12 19 20 21 24 25 30 33 38 46 49 50 51 58 61 62 64 66
49 05/16/2012,01 02 08 13 14 22 36 39 41 45 46 49 51 54 56 58 63 64 69 80
50 05/15/2012,02 03 07 10 17 22 25 36 38 47 52 57 60 61 65 66 68 69 70 78
51 05/14/2012,10 11 12 13 23 31 41 43 48 49 52 53 57 60 67 69 72 76 79 80
52 05/13/2012,05 14 22 30 32 34 38 41 43 45 49 54 56 62 68 73 75 76 77 78
53 05/12/2012,06 10 17 20 21 23 25 31 35 38 48 49 50 56 58 61 66 74 76 77
54 05/11/2012,04 10 11 12 21 31 44 52 53 54 59 60 63 67 69 70 72 74 75 79

```
55      05/10/2012,07 12 15 18 26 27 31 32 34 38 45 46 50 51 54 58 62 64 69 72
56      05/09/2012,03 14 18 26 27 28 29 32 34 36 38 51 54 56 57 64 67 70 77 80
57      05/08/2012,05 08 09 10 11 14 15 17 22 26 28 39 40 45 47 48 54 56 58 77
58      05/07/2012,08 09 10 19 20 23 24 25 26 27 28 31 40 43 46 53 62 63 75 77
59      05/06/2012,02 05 09 10 11 16 23 26 31 38 41 46 49 55 56 60 63 74 76 77
60      05/05/2012,01 05 13 15 17 22 23 30 35 39 43 52 53 56 59 62 63 65 69 78
61      05/04/2012,02 06 08 17 18 20 22 31 33 38 40 47 52 54 57 58 68 72 76 77
62      05/03/2012,05 13 14 16 26 27 34 36 39 40 42 50 51 59 60 64 69 71 72 73
63      05/02/2012,01 02 04 12 13 14 16 31 34 36 39 44 50 62 63 65 66 70 71 74
64      05/01/2012,01 04 05 06 12 18 23 34 37 39 41 43 50 54 61 67 72 73 78 79
65      04/30/2012,05 06 07 20 25 28 29 34 37 40 41 50 53 54 56 57 61 64 75 76
66      04/29/2012,01 04 07 11 12 14 22 26 27 48 56 57 63 67 74 75 76 77 78 80
67      04/28/2012,05 10 13 14 27 29 32 38 40 48 56 66 67 69 72 73 74 76 77 78
68      04/27/2012,01 06 09 14 16 18 19 20 21 25 26 36 48 53 57 60 66 67 75 77
69      04/26/2012,04 07 08 16 20 24 28 33 45 47 56 57 60 63 64 69 71 73 76 77
70      04/25/2012,04 06 20 37 40 41 43 46 47 48 49 55 60 64 65 66 70 71 78 79
71      04/24/2012,03 09 17 21 24 34 37 44 46 48 50 51 59 61 62 64 65 69 75 77
72      04/23/2012,03 10 12 15 17 20 22 27 29 33 41 46 53 56 57 59 67 73 75 76
73      04/22/2012,08 09 15 19 29 32 33 36 38 42 45 46 52 54 58 60 61 63 71 77
74      04/21/2012,04 07 08 13 15 28 30 33 40 43 44 48 51 53 54 55 58 69 78 80
75      04/20/2012,01 16 20 22 23 25 29 32 33 45 49 50 51 54 55 60 62 69 73 78
76      04/19/2012,05 15 18 22 23 26 31 37 38 39 44 48 51 56 60 62 69 71 77 79
77      04/18/2012,03 11 14 17 23 25 28 30 33 35 36 41 48 56 59 60 66 67 71 79
78      04/17/2012,01 02 03 09 13 14 15 23 28 35 36 41 44 47 52 59 62 63 72 76
79      04/16/2012,02 05 08 15 17 28 32 36 37 38 40 46 47 50 51 53 60 64 73 76
80      04/15/2012,03 04 14 15 17 19 31 34 38 40 47 48 51 56 60 64 66 73 76 79
81      04/14/2012,09 10 12 13 16 17 19 25 26 28 29 39 45 50 51 55 56 62 66 69
82      04/13/2012,03 05 06 07 11 28 30 32 35 37 38 44 47 51 52 54 55 65 67 79
83      04/12/2012,01 06 08 13 15 19 23 24 34 41 42 44 46 51 53 54 72 74 78 79
84      04/11/2012,02 03 05 06 17 18 20 23 29 43 51 53 59 60 62 63 68 74 76 79
85      04/10/2012,07 08 13 18 19 21 22 27 31 33 43 48 53 62 63 64 65 71 75 77
86      04/09/2012,01 03 04 08 18 23 24 28 31 34 36 37 40 42 46 52 55 57 59 74
87      04/08/2012,01 02 04 06 09 10 11 13 17 33 38 49 52 55 63 65 66 68 75 78
88      04/07/2012,02 08 18 21 24 25 27 32 38 39 47 48 49 56 57 61 68 71 76 79
89      04/06/2012,11 16 20 23 27 29 32 36 39 44 56 58 60 63 65 69 70 76 77 80
90      04/05/2012,01 09 10 20 21 22 28 29 31 35 37 43 51 56 58 61 68 70 72 74
91      04/04/2012,08 11 14 17 18 21 25 26 27 33 34 36 43 46 51 54 58 64 73 77
92      04/03/2012,05 07 10 14 15 17 19 23 24 29 40 43 49 53 58 62 63 65 74 76
93      04/02/2012,01 03 04 06 15 16 21 22 26 27 29 36 42 43 46 47 51 57 65 70
94      04/01/2012,14 17 26 29 30 32 41 42 48 54 56 59 60 63 64 65 67 69 70 72
95      03/31/2012,05 07 11 19 20 21 24 30 32 36 41 43 50 51 55 58 64 67 73 75
96      03/30/2012,03 13 14 16 18 23 25 29 34 37 39 43 44 45 54 56 61 69 71 78
97      03/29/2012,04 10 15 18 21 27 36 37 40 44 51 52 54 55 57 58 62 67 68 76
98      03/28/2012,10 14 17 25 32 40 42 43 45 52 54 57 61 62 64 67 68 72 76 79
99      03/27/2012,06 08 15 17 23 24 25 29 33 34 35 41 55 60 64 66 69 72 74 76
00      03/26/2012,01 08 11 18 19 22 26 31 32 35 38 40 44 47 48 62 69 73 77 79
01      03/25/2012,02 03 10 14 16 20 21 27 30 31 32 42 43 50 52 58 65 66 75 76
02      03/24/2012,05 07 09 10 16 22 27 28 29 32 33 35 36 37 43 46 64 73 77 80
03      03/23/2012,02 05 11 12 13 15 19 34 36 41 42 55 57 60 61 63 68 69 73 79
04      03/22/2012,03 12 15 19 20 23 25 26 40 44 46 48 51 53 57 60 62 70 73 78
05      03/21/2012,03 05 08 14 17 24 30 33 35 42 43 49 53 60 63 64 65 71 73 80
06      03/20/2012,06 13 14 15 16 17 28 29 32 34 37 39 48 51 52 54 55 58 72 77
07      03/19/2012,04 12 15 24 28 31 32 33 35 40 42 52 55 59 60 72 73 74 75 79
08      03/18/2012,02 04 12 25 26 29 34 40 45 50 51 53 56 58 61 65 67 68 72 74
```

09 03/17/2012,08 09 14 22 23 29 36 41 42 43 46 51 56 63 66 67 73 75 78 80
10 03/16/2012,01 07 17 21 24 27 32 34 38 41 44 47 48 52 55 56 64 66 75 78
11 03/15/2012,02 09 21 27 28 29 32 33 37 43 47 48 52 57 58 60 67 69 74 79
12 03/14/2012,03 06 11 16 17 23 24 27 31 35 38 41 43 47 51 59 62 65 67 70
13 03/13/2012,01 09 10 19 20 31 32 33 34 39 42 44 47 49 53 55 63 64 66 80
14 03/12/2012,03 07 14 17 19 20 21 24 25 36 41 43 44 49 57 59 63 65 70 71
15 03/11/2012,02 05 08 09 11 14 18 24 27 30 38 49 51 54 62 68 72 75 76 80
16 03/10/2012,07 19 25 27 30 33 34 38 39 46 48 49 52 55 57 65 66 71 74 75
17 03/09/2012,04 05 09 15 17 21 28 40 41 42 55 58 62 67 70 75 76 77 79 80
18 03/08/2012,04 09 11 16 21 23 24 31 33 36 40 48 49 53 54 56 58 68 75 76
19 03/07/2012,01 06 14 19 20 22 33 39 40 41 42 44 47 60 63 68 69 71 72 76
20 03/06/2012,01 02 05 06 09 15 17 20 29 38 43 45 54 55 58 64 65 66 69 75
21 03/05/2012,10 16 21 30 38 39 41 44 45 47 49 54 55 56 60 66 68 72 77 78
22 03/04/2012,03 06 12 19 28 29 33 38 39 42 43 45 57 62 64 66 70 71 72 78
23 03/03/2012,02 08 12 18 26 27 31 32 33 35 39 45 52 53 55 57 65 66 73 79
24 03/02/2012,01 05 06 10 12 13 20 25 26 32 36 39 40 56 59 62 65 76 79 80
25 03/01/2012,06 08 09 14 17 20 29 32 39 42 49 51 52 54 57 63 64 67 68 73
26 02/29/2012,02 05 06 08 14 16 22 23 26 35 40 44 46 47 48 52 60 61 63 65
27 02/28/2012,01 02 08 09 10 17 18 28 30 36 41 49 50 55 59 60 65 72 73 79
28 02/27/2012,03 05 08 11 12 18 21 22 27 28 29 41 42 48 50 55 60 64 68 71
29 02/26/2012,03 12 14 16 21 23 25 28 29 30 34 49 51 55 56 61 63 64 76 77
30 02/25/2012,04 09 10 16 17 18 26 34 36 41 44 48 55 56 59 64 65 73 75 76
31 02/24/2012,04 08 09 10 19 20 21 30 31 33 35 39 45 48 54 61 67 70 74 80
32 02/23/2012,02 07 09 13 22 24 26 28 29 34 35 39 53 56 60 62 67 69 72 73
33 02/22/2012,02 05 06 10 12 13 17 24 25 40 42 46 47 60 62 70 73 76 78 79
34 02/21/2012,05 07 08 18 21 27 28 31 42 43 46 47 51 55 61 62 66 67 73 79
35 02/20/2012,06 07 16 20 23 26 28 30 31 35 42 43 48 52 54 56 59 65 68 73
36 02/19/2012,06 12 14 15 18 24 31 34 41 44 48 49 51 52 54 61 64 65 66 76
37 02/18/2012,03 05 10 11 12 14 20 21 22 23 24 32 36 43 45 57 60 62 64 73
38 02/17/2012,01 02 06 08 14 25 27 30 31 35 39 40 42 43 53 56 57 62 70 75
39 02/16/2012,09 19 21 24 25 28 34 36 43 50 54 57 60 62 67 69 70 77 78 80
40 02/15/2012,04 10 14 20 23 29 31 34 35 37 41 46 49 53 55 66 69 74 75 78
41 02/14/2012,07 09 17 18 23 29 34 36 40 41 45 48 49 51 52 54 56 63 70 80
42 02/13/2012,10 11 12 13 36 37 39 40 49 51 53 54 57 66 70 74 75 76 77 79
43 02/12/2012,01 04 09 11 16 21 25 26 30 32 37 45 46 51 60 64 69 70 76 78
44 02/11/2012,17 19 30 32 34 36 37 40 41 43 45 47 51 58 62 71 72 73 75 77
45 02/10/2012,01 09 10 15 17 18 19 33 36 38 45 46 50 54 66 70 71 73 74 77
46 02/09/2012,03 07 08 13 16 18 19 22 28 31 38 41 47 51 64 69 70 72 74 77
47 02/08/2012,10 12 13 16 19 20 23 29 34 36 38 40 43 46 48 60 62 64 66 80
48 02/07/2012,01 03 10 21 26 31 33 35 36 46 51 53 54 57 59 70 71 74 77 78
49 02/06/2012,03 05 12 14 24 29 30 32 39 40 41 42 46 53 55 57 60 61 62 70
50 02/05/2012,06 08 15 16 22 23 25 27 33 34 36 38 40 41 42 43 51 62 76 79
51 02/04/2012,01 09 10 13 15 23 26 30 33 40 46 49 53 54 62 67 70 71 73 75
52 02/03/2012,03 07 13 19 21 22 30 32 33 35 37 41 42 51 52 57 58 66 70 77
53 02/02/2012,10 11 14 16 17 18 19 21 23 27 36 38 39 49 52 60 67 71 76 77
54 02/01/2012,07 10 13 14 19 20 21 23 30 39 41 43 48 54 61 66 70 71 76 80
55 01/31/2012,08 11 12 14 15 17 19 24 26 30 43 49 53 57 63 65 69 72 77 78
56 01/30/2012,02 04 07 08 14 18 20 27 36 45 47 48 49 52 53 56 57 60 69 72
57 01/29/2012,01 04 05 09 10 12 17 18 20 27 43 48 49 50 55 58 61 66 70 74
58 01/28/2012,01 05 14 17 25 28 35 38 41 46 50 53 54 57 59 65 70 71 75 79
59 01/27/2012,04 06 14 18 23 28 30 35 38 39 43 44 52 53 57 59 61 66 70 71
60 01/26/2012,01 03 07 08 09 10 15 16 20 24 34 39 46 47 52 53 56 62 74 75
61 01/25/2012,01 12 17 19 27 28 40 43 45 47 49 51 52 54 55 58 74 77 79 80
62 01/24/2012,03 04 08 20 29 30 31 41 45 51 53 56 58 60 61 65 76 77 79 80

```
63      01/23/2012,04 09 10 15 17 21 28 29 31 35 39 41 51 54 56 62 66 67 68 76
64      01/22/2012,02 05 06 07 11 19 22 27 29 34 35 38 39 51 54 59 64 66 75 80
65      01/21/2012,04 05 06 09 14 20 23 26 31 40 42 44 45 48 51 61 67 72 74 78
66      01/20/2012,02 04 05 07 08 16 21 22 24 31 40 51 54 62 64 66 67 71 75 77
67      01/19/2012,03 06 09 12 17 24 25 26 30 31 32 33 39 42 45 49 58 65 66 69
68      01/18/2012,01 05 09 11 17 18 22 28 30 32 35 36 39 47 59 71 72 77 78 79
69      01/17/2012,02 09 11 14 15 19 23 24 25 29 35 36 41 42 44 57 60 66 71 76
70      01/16/2012,03 11 14 16 17 22 30 41 43 45 49 52 56 59 60 61 62 66 70 79
71      01/15/2012,01 02 08 14 15 20 21 22 29 40 41 52 56 57 60 64 68 69 71 79
72      01/14/2012,03 09 19 25 27 29 30 31 35 40 46 49 52 55 59 70 74 76 79 80
73      01/13/2012,01 02 12 16 20 22 23 25 36 42 49 51 54 55 66 67 70 71 78 80
74      01/12/2012,03 05 06 08 09 10 14 15 16 18 26 41 54 56 59 63 65 70 73 75
75      01/11/2012,03 04 07 10 17 29 34 35 37 42 44 48 50 51 53 55 67 68 73 78
76      01/10/2012,04 05 11 21 24 25 28 32 38 47 49 52 53 54 56 62 65 67 73 80
77      01/09/2012,03 04 17 21 31 33 34 43 44 45 46 47 48 51 53 54 55 59 61 62
78      01/08/2012,04 13 14 16 22 26 38 39 40 41 42 45 52 53 58 59 67 69 73 75
79      01/07/2012,01 02 03 07 09 10 15 17 21 38 40 41 50 52 55 60 63 66 70 78
80      01/06/2012,02 06 13 16 19 22 28 29 34 41 43 48 54 55 59 60 63 64 65 69
81      01/05/2012,05 06 09 11 25 29 31 34 35 36 38 49 50 52 53 59 61 66 72 80
82      01/04/2012,10 11 15 21 25 32 34 35 38 40 43 55 57 58 59 64 68 69 72 78
83      01/03/2012,06 10 13 18 21 24 31 38 43 49 51 55 63 64 68 74 76 77 78 80
84      01/02/2012,04 07 08 09 10 12 14 17 38 39 41 44 48 56 58 62 69 73 78 79
85      01/01/2012,02 03 08 11 14 18 24 26 27 28 29 30 32 39 45 49 56 69 72 79
86      12/31/2011,04 06 07 11 12 20 24 25 33 35 40 44 54 59 67 68 69 71 73 78
87      12/30/2011,10 14 15 19 24 26 41 44 47 54 57 59 60 63 66 68 69 74 76 80
88      12/29/2011,07 09 17 18 23 31 35 36 38 39 40 42 43 49 53 59 62 64 66 77
89      12/28/2011,04 05 06 11 16 17 23 28 33 40 41 48 50 55 56 60 61 65 68 76
90      12/27/2011,01 05 06 13 15 17 18 20 21 24 25 26 31 36 50 65 67 72 76 79
91      12/26/2011,03 08 09 16 25 39 40 41 43 44 46 49 51 56 61 62 69 74 75 80
92      12/25/2011,03 07 11 15 18 28 33 36 37 41 43 44 45 56 62 68 72 75 77 80
93      12/24/2011,12 13 16 24 26 28 30 31 40 41 49 51 56 63 70 71 72 75 76 77
94      12/23/2011,02 06 12 13 14 17 19 23 27 34 35 49 53 56 59 61 63 73 74 78
95      12/22/2011,03 05 06 12 15 19 20 22 39 40 45 48 51 52 57 64 65 67 68 70
96      12/21/2011,05 14 16 21 23 28 40 42 44 47 49 57 58 59 68 70 73 76 77 79
97      12/20/2011,04 08 09 12 13 17 20 22 38 41 42 43 54 58 68 72 74 75 77 79
98      12/19/2011,14 15 16 22 24 28 31 36 37 41 46 47 52 57 62 69 70 76 77 79
99      12/18/2011,01 04 11 15 28 30 33 36 42 43 53 57 61 63 65 66 70 72 73 78
00      12/17/2011,01 06 19 26 31 32 35 37 38 43 47 50 51 56 57 58 70 72 76 77
01      12/16/2011,01 16 18 20 28 33 38 42 45 49 51 52 54 55 58 65 67 70 72 80
02      12/15/2011,03 14 15 17 18 20 22 27 36 42 44 49 50 52 54 57 59 65 74 77
03      12/14/2011,03 06 08 19 21 23 25 28 29 43 45 47 53 56 60 63 65 69 71 78
04      12/13/2011,10 11 12 19 20 24 25 28 33 37 45 47 48 49 50 53 59 62 67 75
05      12/12/2011,04 09 12 15 18 19 22 27 32 35 44 45 54 55 56 58 62 63 69 73
06      12/11/2011,01 03 09 17 28 30 31 33 37 43 47 56 67 68 71 72 73 76 79 80
07      12/10/2011,01 02 07 13 24 25 27 30 35 38 41 46 53 54 59 60 65 69 74 77
08      12/09/2011,01 04 05 07 09 13 14 15 26 29 32 38 43 52 55 56 59 69 72 80
09      12/08/2011,03 05 07 15 16 24 28 36 46 48 53 54 59 61 66 70 72 73 74 76
10      12/07/2011,03 11 12 16 19 20 39 41 42 43 46 52 53 54 56 60 65 69 71 80
11      12/06/2011,03 06 15 18 25 35 38 43 44 51 52 56 57 60 66 70 71 72 73 75
12      12/05/2011,03 05 08 10 14 23 24 26 27 32 38 39 44 47 50 52 59 64 74 79
13      12/04/2011,03 07 12 15 16 28 31 33 38 47 49 51 54 56 59 66 67 71 73 76
14      12/03/2011,01 02 04 05 06 16 24 26 38 40 41 46 48 59 64 66 67 68 78 79
15      12/02/2011,01 02 03 06 11 13 14 25 33 36 39 43 44 45 52 62 65 70 74 80
16      12/01/2011,01 04 06 15 16 20 22 26 30 31 40 51 59 60 64 68 70 74 75 80
```

```
11/30/2011,04 07 19 25 26 27 29 32 37 41 43 44 51 54 56 64 65 69 71 73
11/29/2011,06 07 08 09 10 12 15 16 19 26 28 34 44 50 51 57 62 71 74 79
11/28/2011,03 08 10 15 16 17 18 23 27 28 31 32 36 41 42 53 59 65 66 68
11/27/2011,01 04 07 11 12 14 17 21 23 28 29 30 40 61 62 63 66 72 76 79
11/26/2011,01 06 08 13 26 27 28 38 39 43 47 48 55 56 59 60 67 71 76 77
11/25/2011,06 10 18 24 27 28 32 35 43 47 57 58 60 61 68 70 72 73 78 80
11/24/2011,04 09 12 18 21 24 25 36 43 48 49 55 58 59 61 64 72 73 74 77
11/23/2011,06 08 11 12 17 24 25 29 43 44 46 56 57 58 59 60 62 65 72 74
11/22/2011,05 16 18 19 22 25 28 30 32 33 34 42 52 61 63 66 68 75 76 77
11/21/2011,06 07 13 15 16 20 23 33 34 35 41 42 50 61 62 63 67 71 72 73
11/20/2011,01 04 05 07 20 22 23 25 34 37 39 42 51 53 58 61 65 67 70 77
11/19/2011,02 13 18 19 26 27 35 38 42 45 52 53 57 59 64 66 67 69 72 79
11/18/2011,02 06 11 18 20 24 32 33 37 47 57 58 66 67 71 72 74 77 79 80
11/17/2011,01 02 16 18 21 24 28 29 34 36 38 39 41 43 59 62 65 67 68 69
11/16/2011,03 08 13 14 23 26 27 32 35 36 43 50 56 62 63 64 68 69 70 73
11/15/2011,08 12 13 16 20 28 30 35 41 42 53 55 57 59 61 62 67 72 73 79
11/14/2011,02 11 12 16 23 25 30 31 34 35 40 48 50 51 52 57 59 61 69 72
11/13/2011,03 04 06 13 20 33 38 46 50 54 56 58 60 64 67 68 70 75 77 79
11/12/2011,11 19 20 21 24 28 34 36 40 44 46 48 51 62 64 66 68 74 75 80
11/11/2011,04 09 11 12 24 31 35 36 37 42 44 47 50 51 55 58 73 75 76 79
11/10/2011,06 07 08 09 15 17 22 30 31 40 48 55 57 58 60 61 65 73 78 80
11/09/2011,01 03 07 09 19 20 26 32 37 42 48 51 52 53 60 64 67 70 77 79
11/08/2011,02 08 09 10 13 15 16 21 22 28 29 31 38 44 48 55 56 59 63 68
11/07/2011,01 05 06 08 14 17 27 36 38 40 42 43 49 54 57 58 62 65 67 71
11/06/2011,01 02 07 08 09 21 24 30 31 40 41 50 51 54 55 56 57 59 62 63
11/05/2011,03 06 09 10 19 25 31 32 33 34 35 36 49 51 58 61 70 71 72 79
11/04/2011,02 05 06 07 09 10 14 18 26 28 29 33 37 41 48 51 53 67 73 75
11/03/2011,20 25 26 27 32 33 35 38 41 43 52 55 61 62 68 70 71 72 75 80
11/02/2011,02 03 09 11 14 17 27 32 34 40 43 46 48 52 54 57 61 66 71 74
11/01/2011,06 10 13 17 18 19 22 24 26 37 38 39 45 48 50 63 70 71 75 78
10/31/2011,02 05 14 15 20 23 24 27 28 30 31 44 48 52 53 55 61 63 71 77
10/30/2011,08 11 12 20 22 23 26 29 33 34 40 41 42 44 54 58 62 72 75 79
10/29/2011,01 03 04 13 18 19 24 25 27 30 40 41 46 51 55 56 60 73 75 78
10/28/2011,01 06 10 19 20 23 28 29 34 35 37 38 41 49 55 59 64 69 75 76
10/27/2011,03 06 07 09 15 22 26 29 40 41 43 45 50 53 54 57 59 60 62 71
10/26/2011,05 12 15 16 21 22 36 38 41 42 45 46 47 55 58 59 72 73 78 80
10/25/2011,07 09 10 12 16 29 32 40 43 45 48 49 52 53 54 57 60 62 70 71
10/24/2011,02 05 08 12 15 17 20 21 28 30 31 33 40 43 57 58 66 73 76 79
10/23/2011,01 04 05 14 15 18 20 21 22 28 32 38 41 48 53 59 61 66 72 74
10/22/2011,08 12 23 24 25 29 37 39 45 51 55 56 57 59 61 66 71 73 76 77
10/21/2011,01 04 09 10 19 20 21 25 26 32 40 41 42 43 49 51 53 74 75 76
10/20/2011,04 07 16 24 27 28 36 38 39 40 41 44 56 63 65 67 70 72 76 77
10/19/2011,01 02 04 05 06 10 11 16 18 22 25 26 27 28 35 43 53 58 60 69
10/18/2011,01 02 05 15 17 21 23 26 29 30 32 43 48 54 56 57 59 69 71 78
10/17/2011,01 04 08 16 21 22 25 27 37 40 41 42 43 47 54 60 66 69 74 77
10/16/2011,01 06 15 21 22 23 24 28 32 46 47 48 56 57 63 65 67 69 72 76
10/15/2011,01 12 17 18 23 31 32 34 41 47 49 53 55 58 64 66 68 73 74 79
10/14/2011,01 07 12 13 18 20 23 25 35 36 37 44 47 48 52 54 67 74 76 79
10/13/2011,02 10 12 15 16 17 25 26 27 37 42 43 46 47 53 74 75 77 79 80
10/12/2011,04 12 15 17 25 27 30 31 33 35 38 39 44 45 53 60 62 68 74 77
10/11/2011,03 05 06 11 17 18 19 23 26 38 39 41 43 53 55 57 64 65 68 69
10/10/2011,06 09 13 16 18 26 28 30 33 37 50 53 55 65 67 72 74 76 79 80
10/09/2011,03 06 12 14 15 21 28 29 33 42 49 64 67 69 70 72 73 76 78 80
10/08/2011,03 05 08 15 19 20 21 23 30 31 41 43 48 55 60 61 62 70 78 79
```

```
71      10/07/2011,02 04 05 13 23 27 35 37 38 39 43 44 45 47 53 56 67 68 70 78
72      10/06/2011,02 04 10 11 15 19 21 24 33 34 50 54 57 58 62 68 70 74 75 78
73      10/05/2011,03 04 11 12 14 17 22 28 31 34 36 38 41 45 49 51 55 59 74 76
74      10/04/2011,01 09 10 15 16 18 30 32 36 39 44 45 47 55 58 60 61 68 75 76
75      10/03/2011,01 06 16 18 21 29 30 33 37 42 43 55 57 58 60 64 65 69 72 73
76      10/02/2011,01 04 07 15 19 24 25 30 31 33 35 41 43 46 48 53 60 67 69 80
77      10/01/2011,01 02 16 20 21 24 28 30 31 32 35 42 43 44 48 56 57 62 72 76
78      09/30/2011,06 09 12 21 25 27 30 34 35 36 41 42 45 57 60 67 72 73 75 79
79      09/29/2011,03 04 06 08 14 18 22 24 34 40 42 44 50 57 58 60 64 66 68 73
80      09/28/2011,05 06 08 09 19 20 28 29 30 32 34 35 42 51 53 55 65 67 69 78
81      09/27/2011,02 06 08 13 14 19 21 23 31 37 43 51 53 54 57 58 69 75 76 79
82      09/26/2011,02 10 14 15 17 19 23 26 29 38 39 41 42 46 49 56 61 62 73 76
83      09/25/2011,01 03 07 08 18 20 21 24 29 33 38 40 42 53 63 64 69 73 79 80
84      09/24/2011,01 04 07 11 15 21 25 26 35 41 52 56 58 64 65 66 67 72 73 74
85      09/23/2011,16 17 18 23 27 29 30 35 38 41 46 53 58 59 67 69 72 74 77 80
86      09/22/2011,02 05 14 16 19 22 27 34 35 38 44 48 51 53 56 58 59 67 71 80
87      09/21/2011,03 12 13 22 25 36 37 41 42 43 44 49 53 59 61 69 72 74 75 77
88      09/20/2011,12 13 18 25 26 27 32 35 36 42 44 50 55 59 61 70 73 74 77 80
89      09/19/2011,05 06 09 15 16 19 22 23 41 53 61 63 69 70 72 73 76 77 79 80
90      09/18/2011,05 06 09 13 16 22 23 29 31 43 45 49 51 60 63 66 69 70 74 75
91      09/17/2011,06 15 18 26 27 32 33 36 37 38 39 47 49 53 54 62 64 67 71 80
92      09/16/2011,04 07 17 18 25 27 30 33 44 45 46 48 51 52 54 58 62 68 70 73
93      09/15/2011,04 07 09 13 20 21 23 26 30 34 36 43 51 52 58 59 63 69 75 77
94      09/14/2011,01 02 03 10 14 18 23 27 28 35 36 52 53 55 60 65 67 71 73 79
95      09/13/2011,10 12 15 19 27 30 33 38 45 51 52 53 55 59 60 68 71 75 78 80
96      09/12/2011,02 16 17 18 19 24 32 35 36 37 41 44 51 52 58 62 66 67 71 79
97      09/11/2011,06 16 18 24 28 32 39 40 42 47 52 53 56 59 61 65 66 67 72 76
98      09/10/2011,06 15 21 27 28 36 37 41 42 44 45 49 53 59 62 64 65 70 76 78
99      09/09/2011,01 08 10 19 20 21 22 33 34 35 36 39 50 58 65 66 67 70 78 80
00      09/08/2011,10 15 22 26 28 32 36 39 45 46 48 49 50 58 59 61 68 75 78 79
01      09/07/2011,04 11 13 17 21 30 31 35 38 45 51 54 56 58 59 60 61 62 71 74
02      09/06/2011,03 06 17 18 20 23 26 27 32 39 40 46 47 48 58 62 66 68 69 74
03      09/05/2011,04 05 09 15 21 28 29 30 33 42 48 49 51 53 54 64 68 72 73 74
04      09/04/2011,02 08 13 15 22 23 26 34 38 42 46 51 53 54 55 58 68 69 71 73
05      09/03/2011,05 09 13 21 25 26 29 33 36 39 49 53 58 59 64 66 71 73 78 80
06      09/02/2011,02 05 13 17 19 21 22 23 25 28 29 41 44 57 58 62 63 72 75 77
07      09/01/2011,01 09 12 13 20 26 36 37 42 43 50 55 56 63 68 74 76 78 79 80
08      08/31/2011,03 19 22 25 30 33 43 46 54 57 58 60 61 63 64 66 69 72 73 76
09      08/30/2011,01 03 05 09 22 26 36 39 40 41 52 58 63 64 65 66 67 74 78 80
10      08/29/2011,01 04 06 08 11 17 18 25 32 45 50 53 54 57 60 63 67 69 70 80
11      08/28/2011,01 08 09 13 21 23 30 31 34 49 63 64 65 69 71 72 73 74 76 80
12      08/27/2011,01 03 05 07 09 13 14 16 21 34 37 40 43 47 49 60 64 66 74 80
13      08/26/2011,02 04 05 08 09 10 15 18 21 24 27 31 40 45 48 58 63 66 73 78
14      08/25/2011,02 07 15 16 17 23 32 39 44 48 50 54 56 58 63 70 72 73 75 79
15      08/24/2011,07 10 12 16 20 23 29 33 34 36 40 49 52 56 60 67 69 71 75 78
16      08/23/2011,07 08 11 13 17 22 25 30 37 39 46 47 50 59 63 68 69 72 74 77
17      08/22/2011,06 12 18 21 25 27 29 36 40 44 47 48 52 53 60 64 68 71 73 76
18      08/21/2011,10 18 22 26 27 32 33 34 35 36 38 42 44 46 50 52 61 70 75 78
19      08/20/2011,05 11 17 18 20 21 23 29 30 31 32 39 60 61 63 69 71 72 74 78
20      08/19/2011,05 07 10 18 21 31 32 33 36 38 42 43 51 52 54 55 58 59 68 80
21      08/18/2011,11 12 16 19 21 23 25 30 32 34 38 45 46 48 57 66 67 68 73 78
22      08/17/2011,04 09 10 14 21 24 28 32 34 36 41 42 43 59 61 64 66 67 68 76
23      08/16/2011,03 06 08 17 20 22 26 27 30 31 33 41 46 49 53 58 59 60 68 79
24      08/15/2011,04 06 07 11 14 18 27 29 39 44 48 51 52 56 57 59 62 69 72 75
```

```
08/14/2011,08 11 13 16 17 19 25 26 30 32 37 50 54 59 60 61 64 71 72 79
08/13/2011,04 12 13 14 20 26 40 45 48 54 55 58 59 62 64 66 73 74 77 78
08/12/2011,09 10 18 27 34 35 38 40 45 48 55 56 58 60 62 63 67 70 73 78
08/11/2011,01 10 11 12 15 23 25 28 29 30 34 35 36 44 46 50 51 55 67 79
08/10/2011,03 08 17 18 23 25 26 30 31 48 49 50 51 53 55 62 63 68 78 80
08/09/2011,01 06 07 12 13 17 33 35 39 41 42 47 51 52 57 63 65 67 68 76
08/08/2011,02 06 07 09 16 22 23 25 29 36 38 46 53 54 60 61 62 65 71 74
08/07/2011,05 07 08 13 16 17 19 28 37 41 42 44 45 46 48 52 55 64 68 70
08/06/2011,04 07 08 09 15 16 28 30 31 33 35 36 44 49 52 53 57 60 62 74
08/05/2011,04 19 21 23 31 37 39 40 43 45 49 52 57 59 61 62 63 65 68 76
08/04/2011,02 03 06 10 11 20 23 25 40 43 64 66 68 69 71 72 73 76 78 80
08/03/2011,06 10 11 16 21 33 35 37 38 40 42 44 57 63 68 70 71 76 77 80
08/02/2011,01 06 17 18 20 21 24 42 43 49 54 56 57 60 61 68 71 72 73 80
08/01/2011,03 06 09 11 14 15 20 28 36 39 44 48 52 54 56 57 59 63 64 66
07/31/2011,01 09 13 17 21 23 24 28 30 38 43 46 48 50 51 58 62 67 75 79
07/30/2011,01 03 13 16 17 31 33 35 43 44 47 49 50 55 56 60 62 65 72 74
07/29/2011,03 04 12 19 20 22 23 24 31 48 53 57 61 63 65 66 68 72 74 75
07/28/2011,01 08 14 15 16 24 25 27 29 32 35 39 45 49 55 61 67 72 77 80
07/27/2011,01 05 07 12 19 23 24 26 32 40 45 47 53 58 62 64 67 68 73 77
07/26/2011,01 02 06 07 09 14 18 19 21 25 29 38 40 41 42 47 63 64 67 79
07/25/2011,03 07 22 23 28 29 33 43 49 50 51 56 57 61 63 69 71 74 76 79
07/24/2011,01 02 07 08 10 11 12 14 18 23 24 43 56 59 63 66 72 74 75 78
07/23/2011,09 16 17 18 22 30 34 38 42 45 46 47 57 61 62 63 64 66 72 77
07/22/2011,03 04 13 14 15 16 23 28 31 33 35 40 41 43 45 53 62 69 71 75
07/21/2011,01 10 15 16 25 28 31 34 40 42 44 45 52 53 54 58 62 63 76 77
07/20/2011,01 05 11 31 33 36 38 39 40 45 46 53 57 58 59 61 63 66 70 79
07/19/2011,02 04 14 15 17 25 27 29 38 40 42 50 57 62 70 72 75 77 78 79
07/18/2011,09 11 14 16 17 19 30 35 36 41 51 53 55 58 60 63 65 71 77 78
07/17/2011,01 03 09 11 19 20 24 27 28 30 33 34 38 44 45 52 56 62 65 73
07/16/2011,05 13 14 17 19 22 24 28 32 34 35 48 51 61 63 64 68 69 72 79
07/15/2011,05 10 16 19 21 24 38 42 44 46 48 51 52 53 60 67 69 70 72 76
07/14/2011,10 11 22 23 24 27 30 31 32 36 42 43 45 58 66 67 71 74 77 80
07/13/2011,02 05 08 10 11 23 29 35 37 45 46 47 48 50 53 60 68 70 73 74
07/12/2011,07 08 14 19 23 28 29 31 39 50 52 57 58 63 68 69 70 77 78 80
07/11/2011,12 13 14 18 34 36 47 48 50 54 57 60 65 68 70 71 72 73 78 80
07/10/2011,01 09 11 13 18 20 27 32 33 37 43 44 50 53 63 64 67 70 74 75
07/09/2011,06 09 10 12 15 21 23 27 28 31 35 36 45 46 53 55 56 57 66 71
07/08/2011,03 05 06 09 13 23 24 25 26 27 28 30 33 34 49 53 56 60 68 79
07/07/2011,02 10 11 16 23 25 26 29 33 34 35 40 44 47 54 57 58 60 62 72
07/06/2011,01 09 11 14 17 20 23 27 31 36 39 49 51 52 53 59 62 63 74 75
07/05/2011,02 09 16 20 23 27 33 38 46 51 52 53 59 61 65 66 67 69 74 80
07/04/2011,05 10 14 20 22 23 24 26 37 42 54 55 56 58 62 66 71 72 73 76
07/03/2011,02 03 08 20 23 27 29 39 40 44 45 47 49 51 54 57 60 63 70 74
07/02/2011,01 11 14 16 17 23 25 26 29 35 41 42 47 55 56 59 62 64 76 79
07/01/2011,02 08 10 12 17 22 33 35 40 42 51 52 54 56 61 62 75 76 77 78
06/30/2011,13 14 18 19 22 31 46 50 51 54 56 58 59 64 66 70 72 73 74 78
06/29/2011,01 02 11 13 15 20 21 22 26 33 45 47 50 51 61 62 63 64 68 69
06/28/2011,02 03 11 12 13 14 25 34 36 43 47 50 54 55 56 62 66 70 71 78
06/27/2011,07 14 16 21 22 36 40 44 51 54 55 56 65 66 67 68 74 76 77 79
06/26/2011,03 04 08 11 12 14 16 17 19 20 25 26 34 37 43 46 49 56 63 78
06/25/2011,01 05 06 10 11 13 14 22 50 51 53 56 59 65 70 71 72 74 75 80
06/24/2011,01 12 18 19 20 21 33 39 53 54 57 64 65 66 68 71 75 76 78 79
06/23/2011,05 07 10 13 18 19 20 21 23 26 32 41 42 44 53 55 71 72 73 76
06/22/2011,05 07 10 15 22 27 32 35 39 43 46 54 55 58 60 61 62 73 74 78
```

```
79      06/21/2011,03 07 09 13 15 17 18 19 21 29 30 31 44 46 52 59 63 71 78 80
80      06/20/2011,03 04 05 11 12 27 28 29 34 35 38 48 51 52 54 60 62 64 71 79
81      06/19/2011,05 06 07 16 21 22 25 28 30 34 35 37 39 41 42 60 67 73 76 80
82      06/18/2011,04 06 10 11 12 14 15 21 23 24 31 44 45 49 51 66 76 77 78 80
83      06/17/2011,01 02 03 07 08 15 20 26 29 35 45 50 52 53 56 63 65 70 72 73
84      06/16/2011,06 09 16 17 18 20 21 23 26 35 36 44 45 54 59 67 68 73 75 78
85      06/15/2011,07 10 15 18 19 20 24 25 33 34 40 49 50 56 58 65 70 74 77 78
86      06/14/2011,01 02 05 09 14 17 18 31 38 39 40 41 42 49 53 54 57 58 65 69
87      06/13/2011,04 07 14 15 16 18 22 24 26 27 35 40 41 45 48 50 52 54 66 79
88      06/12/2011,05 14 20 22 23 27 28 40 43 47 57 59 60 63 68 70 75 76 77 78
89      06/11/2011,01 08 09 13 14 15 19 20 22 26 33 35 54 60 64 66 68 72 73 78
90      06/10/2011,05 07 15 18 22 27 30 39 40 42 44 50 54 56 58 59 69 75 78 80
91      06/09/2011,03 10 13 23 24 26 28 38 39 40 41 42 47 52 59 70 72 76 77 78
92      06/08/2011,02 04 05 06 12 24 25 28 32 36 43 47 50 51 56 61 69 72 77 78
93      06/07/2011,07 10 13 17 19 24 28 29 31 39 44 48 50 51 56 61 66 68 73 78
94      06/06/2011,05 08 17 18 20 30 33 34 35 37 38 40 46 48 56 60 68 75 77 80
95      06/05/2011,01 04 11 20 21 22 29 35 41 45 53 55 59 62 64 65 67 69 70 78
96      06/04/2011,01 05 06 13 19 21 23 24 31 33 43 44 47 51 54 55 61 62 68 69
97      06/03/2011,04 06 07 09 11 12 13 14 15 18 22 24 31 33 48 50 54 56 65 70
98      06/02/2011,03 06 08 09 11 24 28 29 42 43 47 50 56 58 59 70 72 76 77 80
99      06/01/2011,02 08 10 11 14 21 29 36 50 52 54 55 56 66 70 71 72 73 77 80
00      05/31/2011,08 10 14 18 22 26 28 31 32 34 36 37 44 48 67 69 70 75 76 79
01      05/30/2011,05 09 11 12 20 21 24 27 30 34 44 47 53 59 64 66 73 77 78 80
02      05/29/2011,02 03 10 21 28 29 30 35 40 41 44 46 48 55 56 62 67 72 76 80
03      05/28/2011,01 09 11 12 13 16 21 22 24 28 35 44 47 48 52 67 68 75 77 79
04      05/27/2011,04 16 17 26 27 37 38 41 45 47 48 51 52 60 62 64 70 72 78 79
05      05/26/2011,02 05 07 09 14 18 21 23 27 39 46 50 53 55 60 62 63 68 74 80
06      05/25/2011,09 13 14 20 27 30 37 41 48 52 55 56 62 65 66 70 71 77 79 80
07      05/24/2011,06 07 10 13 14 19 20 22 23 24 32 43 44 45 51 61 62 67 70 75
08      05/23/2011,05 08 09 21 25 32 36 39 45 47 49 52 57 58 60 66 67 70 79 80
09      05/22/2011,02 04 18 25 27 29 31 33 39 41 43 44 50 51 52 53 57 64 66 76
10      05/21/2011,01 04 06 09 10 22 23 24 26 27 29 32 35 36 40 52 56 70 77 78
11      05/20/2011,02 07 08 09 10 15 19 21 22 26 31 32 37 44 47 48 49 58 79 80
12      05/19/2011,01 07 08 14 15 17 18 29 39 40 41 49 50 55 58 60 63 66 68 79
13      05/18/2011,04 05 06 16 17 18 23 33 39 49 50 51 53 57 59 61 62 64 65 67
14      05/17/2011,06 07 15 18 19 23 26 35 36 38 39 51 52 56 57 59 62 68 70 76
15      05/16/2011,01 02 05 08 11 12 13 15 19 31 34 46 51 53 54 62 66 73 74 77
16      05/15/2011,01 04 07 14 15 29 30 32 36 38 39 44 49 53 55 56 58 62 73 75
17      05/14/2011,04 06 07 08 10 11 15 19 28 30 35 41 43 44 46 47 51 63 69 72
18      05/13/2011,02 03 10 11 15 17 21 23 25 28 29 41 43 46 54 55 57 63 67 71
19      05/12/2011,01 03 07 14 16 21 27 31 38 39 40 46 50 56 57 60 65 67 69 74
20      05/11/2011,05 10 13 19 20 21 22 30 33 36 41 51 52 55 57 62 67 68 78 80
21      05/10/2011,07 20 21 22 25 31 34 35 37 42 48 55 56 62 64 67 74 76 79 80
22      05/09/2011,06 07 09 11 12 21 28 31 33 34 43 45 52 55 56 64 65 74 78 79
23      05/08/2011,08 11 24 26 27 28 32 36 39 44 47 51 55 57 58 63 71 76 77 78
24      05/07/2011,15 18 19 25 26 27 28 32 37 39 51 53 55 57 58 61 66 69 74 77
25      05/06/2011,08 11 15 18 19 23 26 31 33 37 38 39 42 45 53 60 64 67 68 70
26      05/05/2011,05 08 10 11 13 14 15 20 21 23 27 28 29 32 36 61 62 71 72 76
27      05/04/2011,02 11 12 14 21 26 27 34 36 38 40 42 51 52 54 55 57 59 61 78
28      05/03/2011,02 04 06 12 14 21 23 29 42 43 49 54 55 56 62 63 64 70 75 78
29      05/02/2011,01 03 07 12 14 15 21 24 27 32 41 44 47 51 56 59 60 64 70 71
30      05/01/2011,02 03 05 06 16 20 21 25 29 35 46 48 50 58 59 60 63 64 74 78
31      04/30/2011,01 02 17 19 22 25 30 38 43 44 49 55 56 57 60 63 67 72 74 78
32      04/29/2011,02 13 16 20 22 27 29 32 35 41 42 47 49 51 53 57 63 64 65 66
```

```
04/28/2011,03 06 08 22 26 27 33 34 39 42 45 48 49 51 61 62 64 69 75 77
04/27/2011,01 14 15 16 23 28 40 42 43 47 53 61 63 64 68 69 73 75 76 78
04/26/2011,02 05 07 23 26 28 33 34 35 37 40 41 49 50 52 53 60 71 73 76
04/25/2011,08 12 22 24 25 36 39 41 47 48 49 50 52 53 54 55 60 69 70 73
04/24/2011,01 05 09 14 19 23 24 25 29 31 38 41 52 53 55 65 66 68 72 76
04/23/2011,02 05 06 10 11 23 33 44 45 46 58 65 66 67 68 70 73 75 79 80
04/22/2011,01 03 08 18 19 24 31 32 34 36 38 39 40 41 49 54 57 60 73 78
04/21/2011,03 10 13 20 21 31 34 35 36 37 40 41 42 47 48 52 65 70 73 77
04/20/2011,08 09 11 14 15 16 24 26 30 33 37 40 47 49 50 58 59 66 71 78
04/19/2011,05 07 08 12 15 16 22 30 32 42 48 49 53 58 60 63 67 75 78 80
04/18/2011,01 03 05 06 14 27 28 31 33 42 44 46 49 51 55 57 58 60 71 72
04/17/2011,07 09 17 19 24 26 35 43 48 50 56 57 58 64 65 68 76 78 79 80
04/16/2011,04 06 09 11 14 15 18 19 23 28 30 41 42 44 46 58 64 68 75 78
04/15/2011,08 10 24 28 32 33 36 39 41 46 49 50 51 55 59 60 71 72 73 79
04/14/2011,01 03 06 10 11 20 26 30 35 41 44 45 55 59 60 63 67 76 77 79
04/13/2011,01 11 12 13 17 18 21 22 26 40 41 46 49 52 58 61 63 65 70 78
04/12/2011,08 17 22 27 32 33 34 37 38 39 44 47 50 55 57 58 65 67 68 76
04/11/2011,02 03 04 07 10 14 17 18 36 41 44 46 50 53 57 60 73 75 78 79
04/10/2011,01 11 14 15 21 23 31 39 42 45 48 49 50 51 61 63 66 72 75 80
04/09/2011,01 06 13 20 22 29 33 38 43 45 49 54 58 63 64 71 72 76 79 80
04/08/2011,04 06 15 16 20 23 25 26 29 32 39 40 42 54 59 63 64 68 74 78
04/07/2011,07 16 18 21 23 32 33 41 50 52 57 58 60 66 72 73 74 75 76 78
04/06/2011,03 06 13 14 15 18 21 24 26 27 29 31 35 46 47 48 56 60 72 73
04/05/2011,01 05 06 12 20 22 23 25 26 27 30 31 40 45 47 50 57 66 69 70
04/04/2011,05 11 13 21 22 30 32 37 39 42 46 47 48 49 50 54 64 73 78 79
04/03/2011,01 02 04 10 17 21 29 34 39 42 44 45 49 51 57 58 60 63 73 76
04/02/2011,02 03 04 15 16 20 26 30 38 44 45 51 56 57 61 62 72 74 78 79
04/01/2011,02 07 12 13 18 21 22 28 33 35 38 39 46 48 50 52 53 56 76 80
03/31/2011,03 04 07 15 19 27 28 31 33 35 39 43 50 51 66 69 75 76 78 79
03/30/2011,07 23 24 25 26 29 32 35 40 46 51 56 58 67 68 70 74 77 79 80
03/29/2011,02 13 17 33 37 39 44 45 48 49 50 53 60 62 64 65 70 72 74 77
03/28/2011,02 06 14 20 21 24 32 33 44 47 48 57 61 64 67 70 73 75 77 80
03/27/2011,01 04 05 07 08 10 19 23 26 32 43 45 47 49 52 54 56 57 72 77
03/26/2011,02 03 17 21 27 29 32 33 39 40 42 52 54 59 60 61 70 71 79 80
03/25/2011,03 05 06 12 13 15 19 23 31 33 36 38 41 42 43 45 64 65 70 75
03/24/2011,02 03 06 08 18 19 22 24 26 40 43 45 47 48 51 54 56 67 73 74
03/23/2011,07 15 22 23 25 26 27 28 34 47 48 49 50 52 59 66 73 74 77 80
03/22/2011,06 10 12 20 25 28 29 39 40 45 48 50 51 52 54 60 61 73 76 80
03/21/2011,05 11 17 18 23 28 29 30 35 36 38 48 53 57 61 62 66 67 68 79
03/20/2011,02 05 16 17 19 27 30 33 38 40 54 55 60 64 66 67 68 72 74 75
03/19/2011,06 07 08 23 27 28 30 34 43 48 51 53 57 58 59 60 63 68 72 73
03/18/2011,07 13 14 30 33 34 37 39 42 46 47 49 53 55 60 62 66 71 72 78
03/17/2011,03 06 13 16 20 26 27 28 32 33 34 37 38 52 55 59 66 68 73 77
03/16/2011,09 11 13 14 15 27 30 38 41 52 56 59 63 64 66 69 71 78 79 80
03/15/2011,01 02 04 11 13 15 17 26 31 32 33 35 40 47 66 71 72 74 75 77
03/14/2011,02 06 08 17 21 25 28 29 41 42 46 47 48 51 52 57 60 72 74 76
03/13/2011,01 09 11 12 16 23 25 26 36 41 42 45 47 48 50 68 70 74 76 78
03/12/2011,01 09 23 25 26 29 30 33 37 45 50 51 57 58 59 64 72 74 79 80
03/11/2011,03 05 06 07 09 15 17 19 22 29 36 46 49 57 58 60 63 64 72 73
03/10/2011,03 09 11 13 14 19 20 23 31 34 39 40 41 43 59 61 64 69 72 78
03/09/2011,01 15 19 20 23 25 28 36 37 39 40 43 46 50 53 54 63 71 72 77
03/08/2011,01 05 07 16 24 28 30 33 36 48 49 56 57 60 61 75 76 77 78 80
03/07/2011,05 06 08 09 13 14 22 27 29 30 34 42 48 49 54 57 61 70 71 79
03/06/2011,01 04 05 07 09 11 14 23 24 27 28 29 31 36 50 53 62 66 74 76
```

```
87      03/05/2011,04 05 06 15 16 18 20 22 33 36 41 44 47 61 62 73 75 77 78 79
88      03/04/2011,04 06 07 08 14 21 24 32 43 45 48 50 57 59 61 65 68 69 72 74
89      03/03/2011,03 09 17 19 21 23 33 34 35 36 38 40 45 50 51 53 56 60 69 71
90      03/02/2011,03 10 13 25 26 32 39 41 42 48 50 56 57 58 61 62 67 72 75 79
91      03/01/2011,04 05 14 19 22 34 36 37 39 42 51 52 56 57 58 65 67 69 71 74
92      02/28/2011,01 04 08 11 12 21 26 32 38 41 42 57 59 60 68 69 72 77 78 80
93      02/27/2011,16 19 26 28 31 40 42 43 46 48 50 56 57 61 63 69 73 74 78 79
94      02/26/2011,05 16 17 22 23 28 29 30 34 35 40 44 55 59 61 63 74 75 76 78
95      02/25/2011,07 11 15 16 18 20 27 29 30 35 44 48 58 60 61 63 64 66 67 69
96      02/24/2011,02 09 11 13 25 26 32 33 44 46 48 50 52 54 55 59 67 74 77 78
97      02/23/2011,01 06 07 08 09 14 16 17 20 25 28 30 40 41 47 49 57 61 66 79
98      02/22/2011,03 07 08 09 10 18 22 28 29 30 32 34 43 50 53 65 72 74 76 77
99      02/21/2011,01 04 05 06 15 16 25 32 33 38 40 41 47 50 60 61 62 67 71 79
00      02/20/2011,03 12 18 22 24 28 35 41 44 49 51 55 56 57 61 62 67 72 75 76
01      02/19/2011,01 08 10 13 16 19 22 23 26 27 31 32 39 53 55 57 66 70 79 80
02      02/18/2011,01 03 07 09 19 20 22 27 30 36 39 40 42 50 51 62 67 69 72 75
03      02/17/2011,03 11 15 20 21 25 37 40 41 42 52 56 58 66 70 71 75 76 79 80
04      02/16/2011,03 04 07 08 09 11 21 24 27 33 35 38 40 41 42 46 70 72 76 77
05      02/15/2011,04 07 12 21 22 25 32 35 41 43 45 47 48 59 61 63 64 70 73 78
06      02/14/2011,02 03 04 11 15 18 20 27 31 37 38 42 47 51 54 56 57 71 73 76
07      02/13/2011,03 08 11 23 25 27 29 35 36 39 41 48 55 59 62 64 65 70 72 79
08      02/12/2011,01 02 05 09 17 19 24 27 28 29 30 33 47 51 53 64 69 71 74 77
09      02/11/2011,07 08 12 30 34 37 39 42 43 53 58 59 61 63 64 66 70 72 78 79
10      02/10/2011,08 11 14 16 17 19 20 22 33 34 40 41 44 55 57 59 67 73 78 80
11      02/09/2011,04 08 09 17 21 25 27 30 32 36 39 40 41 44 48 53 66 75 78 79
12      02/08/2011,08 10 12 13 15 18 26 28 30 33 35 38 39 41 58 59 65 67 74 80
13      02/07/2011,09 10 11 12 19 21 24 32 33 34 35 39 45 48 50 65 68 71 72 73
14      02/06/2011,01 02 06 07 08 11 21 24 34 35 37 38 39 45 48 56 59 60 64 68
15      02/05/2011,03 05 12 27 29 30 31 34 42 45 54 57 58 60 63 65 66 71 76 78
16      02/04/2011,01 05 08 09 23 30 32 42 52 55 56 58 59 62 64 65 73 74 77 80
17      02/03/2011,06 12 15 16 17 21 24 35 43 44 48 54 55 57 61 65 66 73 74 80
18      02/02/2011,04 08 13 15 19 25 26 27 36 43 53 57 59 61 63 64 71 78 79 80
19      02/01/2011,02 04 05 07 16 18 21 28 29 34 35 41 44 46 54 55 56 57 59 60
20      01/31/2011,01 02 09 21 22 23 31 32 36 37 39 42 47 55 59 63 64 71 76 78
21      01/30/2011,04 07 14 22 24 25 31 33 35 36 43 46 48 50 51 55 57 61 71 72
22      01/29/2011,01 05 09 22 23 36 38 43 46 51 55 59 62 65 66 69 73 75 77 79
23      01/28/2011,04 05 11 17 20 24 30 31 34 35 36 37 40 45 48 56 59 61 66 77
24      01/27/2011,02 03 29 31 35 39 41 45 46 47 49 50 51 53 61 64 66 68 69 75
25      01/26/2011,09 17 20 21 23 33 38 40 46 49 52 53 57 58 70 71 73 76 79 80
26      01/25/2011,04 05 06 19 20 22 29 30 34 35 36 38 43 45 47 50 65 69 73 76
27      01/24/2011,04 06 11 22 23 25 40 41 44 45 46 52 54 61 65 67 68 72 74 78
28      01/23/2011,01 03 04 07 13 15 16 19 23 31 39 45 57 58 59 68 70 72 73 77
29      01/22/2011,03 09 12 24 26 30 32 38 40 43 44 45 50 52 61 66 68 69 77 80
30      01/21/2011,03 08 13 18 21 22 32 33 40 42 44 47 49 52 62 67 69 77 78 80
31      01/20/2011,02 03 09 15 22 23 31 32 34 35 36 38 40 50 57 61 62 69 74 76
32      01/19/2011,02 05 22 29 33 35 36 45 47 49 50 54 56 58 61 65 71 72 76 78
33      01/18/2011,01 13 22 23 24 30 31 33 42 43 44 47 58 59 60 61 66 72 73 74
34      01/17/2011,04 13 23 25 26 27 28 30 34 36 42 43 45 48 57 62 63 68 74 79
35      01/16/2011,01 04 05 11 12 15 16 28 36 38 49 53 55 56 58 60 64 67 78 79
36      01/15/2011,07 10 12 17 19 21 30 36 42 43 47 56 57 59 60 61 63 64 66 72
37      01/14/2011,08 09 12 13 16 18 19 21 28 35 36 37 38 43 58 68 71 74 75 80
38      01/13/2011,01 06 10 16 24 29 36 41 45 52 57 59 60 62 65 70 71 72 76 79
39      01/12/2011,03 06 20 27 28 29 31 37 38 43 45 46 49 54 62 63 67 70 77 80
40      01/11/2011,02 06 07 15 22 23 28 35 41 47 48 56 58 59 62 69 74 78 79 80
```

```
01/10/2011,08 11 18 23 25 26 34 36 39 45 52 59 61 62 64 65 66 67 68 77
01/09/2011,12 17 20 23 32 36 40 42 43 45 48 52 53 54 62 64 66 67 68 72
01/08/2011,04 10 12 18 24 38 39 41 49 52 54 55 57 59 62 67 70 72 75 80
01/07/2011,01 05 07 12 13 15 18 19 24 25 31 38 39 40 42 43 48 52 64 68
01/06/2011,01 03 05 10 12 18 20 25 33 37 41 44 48 50 62 65 68 72 74 77
01/05/2011,04 14 16 19 20 25 26 30 31 32 34 38 39 43 54 57 71 73 75 76
01/04/2011,01 08 09 11 12 13 19 23 30 39 44 55 59 63 66 67 68 69 77 78
01/03/2011,02 06 10 14 15 16 18 22 26 38 40 53 54 57 59 67 68 69 76 78
01/02/2011,09 13 16 20 21 24 25 48 49 51 54 55 58 60 61 62 63 64 71 78
01/01/2011,02 06 08 14 15 16 17 22 26 30 31 40 46 50 51 52 54 55 69 78
12/31/2010,04 09 10 22 23 26 28 30 33 39 40 43 49 54 61 63 67 70 76 79
12/30/2010,15 16 21 25 38 40 43 45 47 50 54 57 63 72 73 74 75 76 79 80
12/29/2010,02 06 07 14 15 17 18 26 37 42 49 51 52 57 58 65 69 70 74 78
12/28/2010,07 13 19 23 24 30 32 38 45 48 49 51 52 55 58 65 66 70 71 75
12/27/2010,02 09 10 12 20 23 25 28 52 56 61 62 63 69 70 74 76 77 79 80
12/26/2010,04 10 13 25 27 31 32 39 40 41 42 44 55 58 59 62 66 70 72 75
12/24/2010,03 05 11 13 14 17 22 29 34 35 43 50 55 56 57 59 60 64 68 76
12/23/2010,02 03 04 06 09 15 22 32 34 37 41 42 50 52 61 62 67 70 72 75
12/22/2010,02 05 07 10 23 26 27 31 33 34 44 45 46 50 51 54 55 56 57 66
12/21/2010,01 04 07 09 15 22 32 33 36 39 40 44 47 55 59 63 64 66 67 74
12/20/2010,04 11 19 20 25 30 33 38 42 45 49 52 53 56 59 62 64 66 77 78
12/19/2010,02 03 05 08 13 18 20 24 27 31 33 55 62 64 65 67 70 73 76 78
12/18/2010,02 03 12 14 29 30 31 36 40 43 44 46 47 48 52 57 60 65 67 75
12/17/2010,06 15 16 25 28 29 32 40 41 42 50 51 53 55 58 60 61 70 72 78
12/16/2010,03 04 16 18 22 27 32 33 39 41 42 44 47 50 52 53 56 57 64 70
12/15/2010,03 06 09 10 11 19 23 24 31 32 43 53 55 58 59 61 64 65 76 77
12/14/2010,01 03 04 10 11 13 17 20 24 28 39 46 47 52 54 56 63 67 69 77
12/13/2010,01 03 04 08 18 23 31 32 36 41 45 47 54 61 64 68 69 72 79 80
12/12/2010,03 04 05 08 10 14 16 20 30 31 34 36 54 59 62 76 77 78 79 80
12/11/2010,01 06 15 16 29 33 34 37 38 39 40 43 45 46 48 57 74 75 77 78
12/10/2010,03 07 14 21 23 33 42 46 52 55 59 62 66 67 70 72 74 77 79 80
12/09/2010,07 09 10 14 16 20 21 23 25 31 34 39 46 49 54 55 56 63 71 78
12/08/2010,05 06 15 18 25 32 36 41 42 44 48 51 53 61 64 66 68 76 77 80
12/07/2010,02 05 06 07 08 10 17 26 31 37 45 54 59 65 66 68 72 76 77 79
12/06/2010,03 05 08 20 25 27 28 32 35 36 37 39 42 44 50 55 62 69 71 79
12/05/2010,15 18 24 27 31 32 33 42 44 47 50 53 55 67 70 71 72 73 77 78
12/04/2010,01 02 03 05 10 11 15 19 23 33 50 56 61 62 65 70 72 74 75 80
12/03/2010,02 04 05 11 13 14 15 17 24 30 44 52 54 58 59 61 63 68 78 79
12/02/2010,02 03 06 09 15 16 22 24 29 34 46 52 54 57 58 59 65 66 70 80
12/01/2010,02 12 15 17 19 21 27 28 30 34 39 43 44 46 51 69 71 73 76 77
11/30/2010,06 07 09 10 13 25 26 28 31 38 43 45 48 54 56 66 67 69 74 75
11/29/2010,07 08 11 17 20 24 30 37 42 43 44 46 51 55 56 64 66 72 76 79
11/28/2010,01 04 07 10 12 21 26 27 29 41 51 61 62 64 65 68 74 76 77 79
11/27/2010,02 05 16 29 33 34 36 37 40 41 51 52 57 59 61 63 67 70 74 76
11/26/2010,02 04 10 11 12 17 18 19 23 24 27 30 37 38 48 54 56 61 62 68
11/25/2010,09 10 11 18 20 27 29 43 46 48 49 59 62 63 67 68 69 74 76 79
11/24/2010,13 16 21 23 25 26 27 28 35 36 39 43 47 48 56 57 60 65 70 78
11/23/2010,02 05 09 13 19 21 23 25 32 37 42 46 47 51 52 54 58 63 71 78
11/22/2010,04 06 11 14 19 20 27 35 37 40 41 43 51 63 64 68 70 74 75 79
11/21/2010,01 02 07 15 16 20 25 28 33 38 39 43 48 51 53 58 59 67 70 74
11/20/2010,01 05 11 12 13 16 18 19 29 32 35 43 45 47 52 55 59 71 74 76
11/19/2010,01 05 06 07 08 16 25 30 32 33 36 44 47 51 54 59 62 74 77 80
11/18/2010,07 09 10 11 19 20 21 22 30 32 35 38 41 45 51 55 68 72 73 76
11/17/2010,02 06 09 12 13 25 29 32 36 41 43 45 48 54 55 61 67 75 79 80
```

```
95      11/16/2010,10 21 22 26 31 32 33 34 38 41 46 48 58 60 64 67 68 71 74 80
96      11/15/2010,16 17 19 22 29 33 35 37 38 41 44 48 50 58 59 64 65 68 72 76
97      11/14/2010,10 15 16 18 20 22 24 27 31 32 34 35 46 55 63 69 72 73 75 78
98      11/13/2010,01 04 08 12 17 22 23 36 38 45 46 47 50 51 52 55 61 64 71 72
99      11/12/2010,10 12 13 15 20 21 23 24 26 30 39 47 52 57 60 62 68 72 77 80
00      11/11/2010,01 05 06 09 10 13 19 41 43 47 48 50 53 54 62 65 69 74 77 80
01      11/10/2010,05 09 11 12 16 21 34 43 44 45 47 49 52 55 58 60 66 67 71 73
02      11/09/2010,02 06 09 12 13 17 24 25 31 38 39 47 48 50 55 57 64 74 79 80
03      11/08/2010,06 10 12 14 15 18 21 22 27 30 31 35 37 41 48 49 50 53 72 75
04      11/07/2010,03 08 10 13 15 23 28 30 31 34 36 37 40 42 47 49 51 55 73 77
05      11/06/2010,02 09 13 14 23 29 32 39 45 56 58 59 64 66 69 70 72 74 78 80
06      11/05/2010,05 06 08 12 22 24 32 36 50 52 53 59 60 62 63 65 69 73 77 78
07      11/04/2010,08 11 12 28 29 33 38 40 48 54 57 59 62 69 71 72 73 74 78 80
08      11/03/2010,01 04 08 09 10 18 22 26 30 32 37 40 44 54 59 62 68 73 76 77
09      11/02/2010,09 11 18 24 26 29 32 33 38 47 48 50 56 61 63 66 68 70 79 80
10      11/01/2010,02 08 13 14 20 21 30 34 39 41 46 50 51 53 58 61 63 64 75 76
11      10/31/2010,02 03 04 07 09 14 16 18 20 23 25 26 32 49 53 56 61 65 75 79
12      10/30/2010,01 02 07 11 13 17 20 28 29 34 35 56 57 58 64 74 76 77 78 80
13      10/29/2010,06 07 08 10 12 14 16 18 21 30 33 40 41 42 50 53 59 68 72 80
14      10/28/2010,02 09 13 20 21 27 30 35 39 40 42 43 45 49 52 62 69 70 72 74
15      10/27/2010,14 15 24 26 35 39 41 44 46 50 52 56 59 61 62 64 68 76 77 78
16      10/26/2010,01 05 06 14 17 24 31 34 37 38 44 52 55 59 63 66 67 73 76 77
17      10/25/2010,04 08 12 16 20 22 29 30 31 32 36 40 44 47 53 56 59 62 67 75
18      10/24/2010,04 10 12 13 20 23 28 33 34 44 46 53 59 61 62 64 69 73 78 80
19      10/23/2010,02 08 12 24 26 31 34 35 38 43 48 51 56 60 64 65 69 70 71 78
20      10/22/2010,02 10 16 18 19 22 23 37 49 52 53 54 56 58 60 63 72 75 77 78
21      10/21/2010,01 04 07 08 15 23 25 37 42 46 47 48 53 54 59 68 73 75 77 79
22      10/20/2010,03 05 09 15 25 30 31 32 33 41 44 52 58 63 67 68 70 76 78 80
23      10/19/2010,04 06 07 08 09 23 34 35 43 47 49 52 53 54 55 57 63 73 77 78
24      10/18/2010,04 05 08 10 13 24 29 38 40 41 42 46 49 50 54 60 62 63 69 73
25      10/17/2010,07 09 12 18 23 24 25 26 27 32 37 40 41 50 57 65 74 76 78 80
26      10/16/2010,06 07 09 10 13 17 23 25 26 30 35 44 48 52 54 56 60 67 77 80
27      10/15/2010,03 04 06 22 24 25 28 29 44 49 51 52 58 62 63 65 67 73 74 76
28      10/14/2010,01 03 05 06 08 11 12 15 16 22 43 44 46 47 48 52 57 64 71 75
29      10/13/2010,02 03 04 08 09 17 20 22 35 43 45 50 53 57 64 69 71 72 75 79
30      10/12/2010,09 17 19 23 24 25 29 36 37 38 40 42 52 53 54 56 61 62 77 80
31      10/11/2010,04 08 13 15 16 17 19 23 28 29 32 36 52 54 55 57 75 76 79 80
32      10/10/2010,01 07 10 11 12 13 15 16 19 21 24 26 29 38 42 51 61 71 73 76
33      10/09/2010,03 05 06 09 15 21 26 33 38 44 47 50 57 60 61 62 67 70 72 75
34      10/08/2010,06 08 09 14 22 23 24 26 27 30 32 40 48 50 51 52 55 61 68 75
35      10/07/2010,05 06 16 17 18 29 31 34 38 44 45 49 58 59 64 68 70 72 74 75
36      10/06/2010,01 10 11 16 21 23 31 38 46 48 50 51 59 61 62 64 66 75 76 80
37      10/05/2010,02 04 12 14 15 18 22 23 25 33 44 48 52 53 55 56 58 64 71 72
38      10/04/2010,04 13 20 21 22 23 25 31 34 45 46 49 54 59 63 64 67 68 78 80
39      10/03/2010,03 04 06 07 09 13 18 37 39 41 47 48 53 56 62 73 74 75 76 80
40      10/02/2010,01 03 10 18 23 25 35 38 41 46 47 50 55 60 63 65 69 71 74 75
41      10/01/2010,01 02 10 13 18 19 28 29 34 39 44 45 51 55 56 66 74 76 79 80
42      09/30/2010,04 06 12 13 15 20 24 28 36 41 46 49 51 52 66 69 73 74 75 78
43      09/29/2010,01 07 09 16 31 39 40 41 46 48 51 54 57 58 61 62 64 70 75 76
44      09/28/2010,02 03 04 07 10 16 20 21 25 29 31 33 42 44 45 47 54 70 74 78
45      09/27/2010,01 03 05 06 09 19 27 33 41 48 53 54 55 64 66 68 71 75 77 78
46      09/26/2010,01 02 07 15 22 29 30 31 33 34 36 37 39 46 59 63 69 73 76 79
47      09/25/2010,05 06 07 13 16 21 22 25 26 36 39 41 47 49 50 55 57 72 76 78
48      09/24/2010,01 03 04 12 13 15 24 36 38 39 43 45 61 65 66 67 68 73 74 80
```

```
09/23/2010,08 14 16 18 22 24 25 29 51 52 53 54 57 58 62 71 73 74 76 80
09/22/2010,07 10 13 14 16 21 22 33 37 39 41 43 46 51 57 58 59 63 74 75
09/21/2010,09 10 16 18 26 29 30 38 40 45 48 52 54 55 57 60 61 67 71 75
09/20/2010,20 22 23 24 27 29 30 36 40 44 48 51 53 56 57 64 69 70 72 78
09/19/2010,05 06 13 14 20 27 37 40 47 52 54 62 64 66 68 71 73 76 78 80
09/18/2010,04 07 11 13 15 18 22 24 37 45 46 49 50 53 55 66 67 74 75 76
09/17/2010,01 02 09 14 16 25 38 42 43 46 48 49 53 57 59 67 69 73 74 78
09/16/2010,06 07 13 18 19 21 22 26 34 38 46 48 49 50 52 54 57 59 67 78
09/15/2010,06 07 14 17 18 19 26 35 38 42 43 56 57 60 61 62 68 70 72 76
09/14/2010,06 12 14 19 20 21 22 30 39 42 47 49 50 60 61 63 69 70 76 77
09/13/2010,01 04 09 16 19 22 26 34 35 37 41 52 54 55 56 59 61 69 79 80
09/12/2010,02 05 14 18 22 31 34 36 49 52 55 57 61 63 64 65 68 71 76 77
09/11/2010,02 05 06 07 14 15 20 26 28 31 35 38 39 48 54 55 59 75 77 79
09/10/2010,01 08 18 28 30 33 34 37 43 46 49 52 57 60 61 63 66 67 73 77
09/09/2010,01 03 04 13 17 21 22 25 26 45 47 49 51 65 69 70 72 73 74 77
09/08/2010,02 11 20 21 22 27 38 39 42 44 48 50 52 55 62 67 69 74 77 80
09/07/2010,01 03 05 18 23 25 27 34 37 41 50 52 55 59 60 63 66 75 78 79
09/06/2010,01 08 09 10 11 16 18 28 33 38 41 42 44 46 49 52 56 61 76 79
09/05/2010,01 05 06 11 13 20 23 27 31 35 37 38 54 58 61 70 74 77 78 80
09/04/2010,04 06 09 11 12 17 19 28 34 39 44 49 54 56 59 63 66 70 75 76
09/03/2010,04 05 10 14 18 22 26 27 30 31 33 39 46 49 51 58 59 64 68 70
09/02/2010,02 03 05 11 14 18 21 25 34 38 55 56 59 60 64 67 70 71 76 77
09/01/2010,06 11 12 13 14 20 21 23 24 33 39 40 41 49 58 65 66 74 75 78
08/31/2010,01 05 11 14 15 17 22 23 29 32 42 47 53 54 55 59 60 64 65 70
08/30/2010,06 08 10 15 18 19 20 25 29 40 41 50 56 58 61 62 64 66 72 74
08/29/2010,03 04 09 12 14 15 17 21 25 27 29 31 35 41 49 59 66 71 75 79
08/28/2010,01 02 03 05 06 13 14 20 37 38 46 48 51 55 57 58 61 64 65 69
08/27/2010,06 08 09 15 17 19 25 40 41 45 46 47 52 53 57 58 65 70 73 77
08/26/2010,04 06 07 09 13 14 15 16 17 27 30 36 44 48 64 67 69 74 78 80
08/25/2010,10 11 14 15 18 19 24 38 41 45 50 53 56 59 60 63 72 74 77 78
08/24/2010,01 02 12 14 20 21 29 34 40 42 48 49 53 54 56 60 72 75 76 77
08/23/2010,02 05 07 10 14 17 29 35 41 42 50 52 54 55 58 59 60 70 76 80
08/22/2010,09 16 19 20 29 31 35 37 39 42 44 49 57 58 59 62 66 76 78 79
08/21/2010,02 11 12 16 19 28 30 36 48 49 50 58 59 60 70 71 72 73 76 78
08/20/2010,04 19 22 24 27 38 40 44 47 50 53 54 57 59 61 64 70 71 73 75
08/19/2010,01 04 05 08 16 21 28 33 41 46 47 53 59 62 64 65 70 72 76 77
08/18/2010,02 03 09 11 21 23 26 30 34 42 47 49 53 55 57 62 64 65 68 71
08/17/2010,04 08 09 12 30 32 35 39 46 50 51 54 55 60 62 64 67 73 74 76
08/16/2010,01 16 17 19 21 22 30 31 34 35 36 37 53 55 57 61 63 65 66 78
08/15/2010,04 06 07 12 17 19 24 25 34 40 47 50 52 57 59 64 72 75 76 80
08/14/2010,07 12 17 19 21 23 24 26 29 41 45 47 50 53 54 56 63 66 71 74
08/13/2010,03 06 07 11 17 19 24 26 28 34 40 43 54 57 58 60 65 66 72 74
08/12/2010,01 03 04 08 11 13 14 24 25 26 31 34 35 36 38 42 55 59 60 63
08/11/2010,10 11 16 19 20 21 23 25 26 27 35 36 39 42 46 50 52 64 65 79
08/10/2010,03 16 19 22 26 34 36 38 44 46 48 52 56 62 68 71 72 73 78 79
08/09/2010,04 05 06 10 12 13 17 19 22 23 25 38 42 47 57 61 63 67 72 80
08/08/2010,03 05 11 23 27 29 31 36 37 41 43 44 51 54 55 59 65 66 69 79
08/07/2010,08 09 13 23 29 30 37 43 44 51 53 56 58 59 60 61 64 65 76 80
08/06/2010,01 04 08 09 11 15 17 19 23 27 31 36 39 42 47 58 61 68 72 73
08/05/2010,02 05 07 12 16 21 24 33 41 43 45 49 56 57 61 63 68 74 75 77
08/04/2010,01 09 10 12 19 23 24 25 27 29 45 46 52 55 57 69 70 71 73 79
08/03/2010,04 06 09 10 12 15 18 21 23 27 34 36 39 41 44 48 59 64 66 79
08/02/2010,07 13 15 17 37 42 43 44 49 50 55 58 59 66 67 72 75 76 77 78
08/01/2010,10 23 24 26 28 33 34 35 36 38 46 50 57 59 66 67 68 72 73 77
```

03 07/31/2010,04 05 07 13 15 18 19 23 26 27 28 29 31 36 40 41 52 57 63 79
04 07/30/2010,02 07 12 14 15 16 18 20 24 39 41 42 43 46 48 51 65 74 77 78
05 07/29/2010,08 09 10 14 23 24 27 28 30 35 43 48 54 55 67 69 71 74 75 78
06 07/28/2010,02 08 15 17 19 20 22 24 26 36 41 48 56 58 60 64 70 71 76 80
07 07/27/2010,05 09 16 19 32 34 41 42 46 47 48 51 56 59 60 62 65 68 70 78
08 07/26/2010,02 04 08 16 19 29 34 35 36 45 46 49 55 60 61 62 65 66 68 74
09 07/25/2010,01 09 12 17 18 19 20 22 23 24 25 26 42 43 44 49 51 64 73 78
10 07/24/2010,02 03 06 18 29 33 35 39 43 45 50 51 55 56 61 62 70 75 76 79
11 07/23/2010,01 06 08 14 15 18 27 33 36 37 39 40 49 54 59 62 65 68 75 77
12 07/22/2010,05 06 08 11 22 24 34 41 43 51 52 55 56 58 61 62 68 69 73 77
13 07/21/2010,08 12 19 21 35 38 39 42 43 46 48 51 54 55 57 62 63 69 71 76
14 07/20/2010,04 06 10 14 22 27 32 34 37 50 52 53 56 59 63 65 67 72 76 80
15 07/19/2010,01 04 06 11 12 14 17 22 25 28 34 36 40 46 49 50 55 67 68 79
16 07/18/2010,02 06 11 17 21 29 30 38 39 43 46 53 60 62 64 66 67 71 75 80
17 07/17/2010,05 09 16 27 30 34 36 43 44 47 49 50 53 55 63 72 74 75 78 79
18 07/16/2010,05 06 09 10 11 13 21 24 28 32 40 42 46 49 50 59 72 74 76 78
19 07/15/2010,04 06 08 21 22 23 24 25 31 32 33 40 41 43 52 59 60 62 67 69
20 07/14/2010,01 04 06 12 14 22 26 28 36 42 51 56 60 61 62 67 71 75 78 79
21 07/13/2010,04 14 18 19 22 24 28 43 48 50 55 56 60 61 62 65 66 73 75 78
22 07/12/2010,05 08 11 15 18 19 23 26 31 33 35 41 44 47 57 61 65 66 73 77
23 07/11/2010,08 09 11 13 15 18 20 21 27 35 36 40 50 51 53 67 72 73 75 80
24 07/10/2010,03 13 19 20 22 25 28 32 33 35 36 41 42 44 59 62 64 66 69 73
25 07/09/2010,03 15 26 28 35 39 45 46 50 53 55 58 61 63 66 74 76 77 78 79
26 07/08/2010,02 04 10 14 15 17 20 23 24 27 31 32 37 43 47 48 50 62 69 71
27 07/07/2010,01 02 06 07 15 20 21 22 25 41 43 44 51 56 58 64 68 73 74 80
28 07/06/2010,06 08 16 23 24 26 28 29 30 34 37 39 45 49 50 58 60 62 69 79
29 07/05/2010,05 08 11 15 17 18 22 27 39 44 47 51 55 60 65 66 69 74 78 79
30 07/04/2010,01 04 08 14 17 26 30 33 37 39 48 60 61 65 66 68 71 72 77 79
31 07/03/2010,03 07 16 22 23 24 26 36 42 45 47 54 55 57 65 71 74 76 77 80
32 07/02/2010,01 04 11 12 15 19 20 24 28 33 44 47 54 61 65 66 68 72 74 76
33 07/01/2010,02 05 07 13 15 20 34 38 42 44 47 50 53 56 69 71 72 75 76 78
34 06/30/2010,14 15 20 21 22 27 28 31 36 39 41 42 49 52 54 55 59 61 72 77
35 06/29/2010,18 21 24 32 33 35 39 41 43 47 49 52 53 57 67 70 71 75 78 79
36 06/28/2010,01 03 08 13 15 21 26 34 38 42 45 47 51 52 53 58 59 70 71 72
37 06/27/2010,03 07 08 12 18 21 29 32 33 35 38 41 43 63 66 68 69 70 73 78
38 06/26/2010,08 13 14 18 23 26 30 32 37 38 42 47 48 49 51 52 56 58 63 64
39 06/25/2010,12 19 23 24 26 29 30 31 37 40 47 53 58 59 62 63 69 72 76 78
40 06/24/2010,01 02 05 08 11 17 21 24 29 30 32 35 50 51 59 65 67 68 76 80
41 06/23/2010,02 06 13 18 20 24 26 38 44 48 53 56 60 63 65 67 71 73 75 78
42 06/22/2010,02 03 10 13 14 15 18 20 25 26 40 41 45 55 56 59 69 72 76 80
43 06/21/2010,02 03 06 08 09 12 17 18 29 30 39 53 58 60 64 65 66 69 71 78
44 06/20/2010,01 03 05 10 11 15 17 21 28 30 31 37 38 41 52 53 54 55 59 71
45 06/19/2010,12 14 16 17 20 22 24 26 30 31 34 38 49 55 56 58 60 66 76 79
46 06/18/2010,01 04 05 07 14 23 26 33 35 39 43 46 49 51 54 57 63 64 70 72
47 06/17/2010,02 05 08 09 15 32 39 43 44 50 51 53 54 55 58 60 64 72 73 77
48 06/16/2010,08 22 23 25 30 37 39 41 46 50 51 52 53 54 56 60 63 68 71 72
49 06/15/2010,01 09 10 13 16 25 26 27 28 37 42 44 46 54 55 60 61 66 69 76
50 06/14/2010,11 14 16 18 23 37 40 49 55 57 58 61 62 64 65 69 70 71 74 78
51 06/13/2010,07 08 09 12 13 18 25 32 36 41 43 44 54 56 57 59 60 72 74 78
52 06/12/2010,01 14 20 21 22 26 29 33 35 38 41 50 54 60 61 63 70 72 77 80
53 06/11/2010,29 32 33 34 36 38 49 51 52 53 56 57 58 61 64 68 71 73 79 80
54 06/10/2010,04 06 07 13 14 18 24 37 40 45 49 54 57 58 59 61 62 67 71 73
55 06/09/2010,02 03 09 14 18 19 24 25 32 33 35 38 39 44 52 53 57 68 70 80
56 06/08/2010,02 16 17 19 22 24 34 37 41 43 46 52 53 55 67 68 71 74 75 76

```
57      06/07/2010,02 05 07 10 13 20 23 26 29 32 33 34 35 40 42 45 53 60 62 65
58      06/06/2010,06 08 11 15 16 17 23 27 31 37 40 43 47 48 50 54 60 73 77 78
59      06/05/2010,03 04 27 38 39 43 47 49 52 54 58 61 63 65 66 67 68 74 75 79
60      06/04/2010,02 20 28 30 34 35 37 40 41 44 45 47 48 52 65 66 67 69 76 80
61      06/03/2010,02 09 13 15 18 20 22 25 32 37 41 43 52 55 57 60 66 67 68 72
62      06/02/2010,05 07 09 10 28 30 33 35 36 37 46 47 48 49 55 56 61 68 71 79
63      06/01/2010,07 11 12 19 20 21 25 33 36 40 43 48 50 52 61 63 66 67 71 73
64      05/31/2010,02 03 04 15 18 21 25 26 27 28 29 30 35 41 43 48 54 63 78 80
65      05/30/2010,03 04 15 22 26 29 32 36 38 40 42 48 49 52 57 61 66 78 79 80
66      05/29/2010,02 03 11 19 21 27 28 29 30 32 43 44 54 58 63 65 70 77 78 79
67      05/28/2010,03 06 07 10 13 14 18 19 21 22 27 34 37 38 39 49 58 67 72 79
68      05/27/2010,01 02 08 10 23 28 31 33 35 40 44 48 51 54 57 58 59 63 70 79
69      05/26/2010,01 05 11 12 18 25 27 32 40 44 48 53 54 55 56 63 69 72 77 80
70      05/25/2010,07 10 14 16 18 21 32 38 42 43 50 52 55 56 58 63 65 76 79 80
71      05/24/2010,12 13 17 20 21 27 31 33 34 40 49 51 57 58 61 62 67 68 73 76
72      05/23/2010,16 17 21 31 32 33 34 41 45 49 50 58 61 65 66 68 69 73 79 80
73      05/22/2010,06 10 11 16 17 21 26 28 29 30 35 39 42 45 55 66 72 73 75 80
74      05/21/2010,01 11 17 19 24 25 26 32 34 38 42 43 44 47 48 49 53 63 68 71
75      05/20/2010,01 05 15 20 21 23 26 34 37 40 42 47 51 53 57 64 68 69 71 76
76      05/19/2010,01 02 09 10 14 29 31 34 43 44 45 51 52 53 58 61 65 71 73 74
77      05/18/2010,01 09 17 20 22 28 37 44 49 56 57 64 66 68 71 72 73 75 76 77
78      05/17/2010,02 03 07 11 12 21 32 45 46 47 48 49 52 56 60 61 65 66 68 71
79      05/16/2010,01 10 11 22 30 31 33 34 35 36 40 41 43 45 51 54 58 71 73 80
80      05/15/2010,09 11 12 16 17 20 24 30 35 53 55 58 60 61 62 69 71 72 79 80
81      05/14/2010,03 07 13 19 20 21 24 25 34 42 43 46 47 50 53 55 58 64 70 79
82      05/13/2010,05 08 10 11 13 15 28 31 32 33 35 40 42 45 57 60 65 70 73 78
83      05/12/2010,01 02 04 15 20 21 26 30 39 43 44 46 47 48 53 55 62 65 70 78
84      05/11/2010,02 04 06 07 08 11 17 19 22 29 34 41 44 48 53 56 60 64 65 80
85      05/10/2010,05 06 12 15 19 30 39 40 51 55 58 59 61 67 71 74 75 76 77 80
86      05/09/2010,02 05 06 07 11 15 16 17 21 22 24 27 29 30 32 53 58 71 72 75
87      05/08/2010,03 13 14 15 22 37 38 39 47 48 52 54 55 56 61 63 65 68 72 75
88      05/07/2010,03 08 09 11 12 13 14 22 31 33 35 39 50 51 56 60 61 66 67 80
89      05/06/2010,06 10 17 27 30 38 39 40 43 45 46 47 51 54 57 60 64 65 66 76
90      05/05/2010,01 05 06 08 14 19 20 21 23 24 47 58 60 63 66 67 70 72 73 78
91      05/04/2010,04 18 23 25 30 32 34 35 37 40 47 49 55 58 60 61 64 67 70 78
92      05/03/2010,04 11 15 17 27 35 43 48 49 50 65 66 68 71 72 74 75 78 79 80
93      05/02/2010,11 13 14 20 26 28 30 32 33 38 39 42 53 54 55 56 59 61 65 66
94      05/01/2010,01 09 10 11 14 19 22 26 30 31 37 38 54 55 57 64 67 75 77 80
95      04/30/2010,01 04 09 18 20 24 32 35 36 37 42 44 46 52 56 57 63 67 76 77
96      04/29/2010,02 04 09 11 13 19 20 21 23 28 32 42 46 51 52 60 64 65 69 78
97      04/28/2010,02 06 08 16 18 26 34 44 51 52 54 65 66 68 69 73 74 78 79 80
98      04/27/2010,01 03 09 14 26 31 32 35 38 40 43 44 46 51 54 55 60 76 78 80
99      04/26/2010,05 09 11 28 30 31 32 33 43 44 46 47 49 53 54 62 68 69 73 79
00      04/25/2010,03 04 07 13 20 23 25 28 31 34 35 37 39 43 46 53 63 72 73 76
01      04/24/2010,01 02 06 08 11 12 17 18 22 24 25 27 29 41 47 49 52 62 69 70
02      04/23/2010,02 10 13 14 20 21 24 30 32 33 36 45 49 50 53 54 57 64 71 73
03      04/22/2010,13 14 19 27 31 32 39 43 44 51 53 55 60 62 67 68 69 73 76 80
04      04/21/2010,01 07 09 11 12 17 19 26 35 38 43 48 53 62 64 65 66 67 68 78
05      04/20/2010,05 08 11 12 14 16 26 28 33 34 36 40 41 50 57 58 64 65 77 78
06      04/19/2010,10 11 21 22 23 26 30 32 35 36 39 42 50 55 60 63 64 69 76 77
07      04/18/2010,05 06 07 09 13 15 17 21 23 26 34 44 47 50 52 54 56 66 77 79
08      04/17/2010,05 07 13 16 18 19 20 27 34 36 44 46 47 55 58 61 64 67 68 73
09      04/16/2010,03 08 12 16 23 24 30 32 34 35 38 44 47 51 59 62 65 68 72 74
10      04/15/2010,04 06 12 20 23 25 26 35 36 39 45 49 57 59 66 67 68 72 74 78
```

```
04/14/2010,05 09 17 18 19 27 34 38 42 49 59 66 67 68 69 72 73 74 76 80
04/13/2010,03 08 09 17 22 26 32 33 36 40 41 42 43 44 45 56 57 59 66 71
04/12/2010,02 05 08 09 10 11 12 15 24 26 30 32 47 51 58 68 72 75 78 80
04/11/2010,07 12 13 16 19 27 30 36 37 43 44 47 56 59 63 65 66 68 73 80
04/10/2010,02 03 05 06 07 10 13 16 19 20 25 26 29 30 40 41 48 57 63 70
04/09/2010,02 09 10 11 27 31 33 38 39 41 44 48 50 60 61 68 70 76 77 80
04/08/2010,09 13 16 19 22 23 25 27 31 32 43 47 48 51 53 68 69 75 76 78
04/07/2010,01 05 09 11 15 16 18 21 26 29 40 43 52 55 57 59 60 62 71 76
04/06/2010,02 05 08 09 18 27 28 38 40 43 50 52 53 55 56 59 74 75 76 79
04/05/2010,07 10 11 17 27 33 38 44 46 48 49 54 55 57 63 65 69 73 75 78
04/04/2010,01 04 06 08 12 19 26 35 36 41 44 54 59 63 67 75 76 77 78 80
04/03/2010,04 12 17 19 21 23 26 28 39 41 47 51 56 57 59 64 68 69 72 73
04/02/2010,01 05 06 15 21 23 24 40 41 43 48 52 58 60 63 68 70 75 76 78
04/01/2010,03 05 07 15 16 20 27 33 36 44 49 60 68 69 73 75 76 77 79 80
03/31/2010,01 07 09 13 19 23 30 31 32 34 36 42 48 50 54 55 58 60 68 78
03/30/2010,03 04 06 09 11 12 13 14 17 25 35 46 49 50 53 54 75 77 78 80
03/29/2010,11 14 15 16 19 22 28 30 32 33 34 45 48 55 58 62 66 69 70 73
03/28/2010,03 07 09 12 15 18 24 27 29 30 36 41 43 48 53 54 62 67 71 76
03/27/2010,02 05 07 12 17 20 29 35 36 38 45 50 55 60 61 63 68 70 71 75
03/26/2010,05 07 11 14 17 20 22 23 32 33 38 41 55 57 62 65 71 72 76 80
03/25/2010,04 07 18 25 26 28 30 31 32 35 41 45 49 61 65 66 69 71 73 78
03/24/2010,04 07 14 19 22 26 31 38 42 43 44 46 49 52 54 60 64 71 74 76
03/23/2010,03 10 13 27 37 38 42 43 45 46 49 52 54 56 57 64 67 73 74 76
03/22/2010,01 17 26 27 30 32 34 37 42 43 44 45 57 58 61 70 71 75 77 80
03/21/2010,02 06 10 13 21 26 33 35 37 48 53 54 58 63 66 69 73 74 77 80
03/20/2010,01 04 09 10 11 12 23 29 35 40 44 47 51 53 56 60 69 70 73 78
03/19/2010,04 07 08 09 10 11 16 22 26 27 39 40 50 54 57 58 64 70 75 76
03/18/2010,02 03 05 09 18 28 31 37 43 44 45 48 54 57 58 69 70 71 74 79
03/17/2010,07 18 22 23 24 27 29 36 43 44 49 53 61 63 64 66 67 72 73 76
03/16/2010,07 10 13 17 29 31 34 35 42 50 51 56 57 59 63 68 72 73 75 78
03/15/2010,07 10 14 17 34 36 37 38 45 48 53 54 57 62 63 65 67 70 77 78
03/14/2010,01 04 09 12 16 23 25 26 32 34 35 36 52 53 59 62 67 68 72 80
03/13/2010,08 17 21 23 25 26 28 29 40 45 46 48 53 59 60 63 64 70 73 74
03/12/2010,04 09 17 23 30 33 35 37 39 40 41 45 46 49 52 53 61 66 73 74
03/11/2010,02 06 08 10 19 23 26 46 51 52 55 56 62 64 65 67 69 70 73 75
03/10/2010,06 12 14 19 20 21 22 24 25 30 31 39 40 45 46 50 63 65 71 72
03/09/2010,01 07 11 22 24 27 28 32 33 35 38 41 49 55 59 63 64 72 74 75
03/08/2010,01 04 10 11 12 14 19 26 29 37 41 44 50 51 52 54 58 69 70 80
03/07/2010,09 11 12 14 17 18 19 25 28 29 36 39 42 44 49 55 56 60 72 75
03/06/2010,02 05 07 11 16 18 21 23 28 35 45 46 47 49 50 52 62 72 78 79
03/05/2010,06 07 11 21 23 27 28 29 33 41 43 44 47 52 67 68 69 73 74 78
03/04/2010,04 05 07 10 14 18 20 23 26 28 29 34 44 48 50 60 65 68 71 76
03/03/2010,05 08 13 14 24 27 31 34 39 40 43 52 54 59 62 64 67 72 74 76
03/02/2010,04 05 13 17 20 21 25 27 33 37 39 43 49 50 64 68 71 74 75 77
03/01/2010,08 28 30 31 32 36 37 44 48 55 56 60 61 62 68 70 74 77 79 80
02/28/2010,01 02 07 09 12 16 28 30 32 41 44 45 47 48 60 63 65 68 70 71
02/27/2010,01 11 12 13 14 21 23 25 29 35 38 39 41 42 50 58 60 66 76 80
02/26/2010,01 02 10 16 18 19 22 24 28 36 37 39 40 41 45 61 72 73 75 76
02/25/2010,03 05 12 13 15 16 18 23 29 34 35 41 45 48 52 56 61 73 77 79
02/24/2010,05 06 09 13 15 16 19 24 28 35 38 42 50 55 66 68 69 70 78 79
02/23/2010,06 10 12 14 20 26 32 33 35 36 41 46 47 48 52 57 58 66 79 80
02/22/2010,02 03 06 10 11 15 20 25 26 47 52 53 54 55 56 59 63 67 76 79
02/21/2010,03 05 06 10 15 19 21 26 35 36 55 56 58 63 65 68 69 71 78 80
02/20/2010,01 03 06 08 09 18 24 28 29 30 32 36 43 44 58 64 66 72 75 78
```

```
65      02/19/2010,05 09 11 12 13 14 17 19 26 33 34 35 37 39 43 48 51 54 62 65
66      02/18/2010,07 09 14 15 17 18 20 23 24 25 30 39 41 43 45 51 54 56 67 75
67      02/17/2010,05 13 14 30 32 33 34 40 41 43 45 51 53 54 55 56 66 72 73 80
68      02/16/2010,01 10 11 13 19 22 24 25 31 36 40 42 46 48 51 54 62 69 71 76
69      02/15/2010,12 14 15 21 22 26 33 37 46 52 53 55 57 58 60 66 70 73 74 78
70      02/14/2010,02 05 08 09 15 16 17 30 37 40 42 44 48 59 60 61 64 70 72 75
71      02/13/2010,01 03 05 08 09 10 11 14 17 23 26 34 35 37 39 50 57 59 68 75
72      02/12/2010,07 08 12 15 21 26 29 30 35 42 43 49 54 57 58 60 62 70 71 76
73      02/11/2010,02 06 10 12 13 16 18 25 34 35 38 39 42 47 53 55 58 64 75 77
74      02/10/2010,01 05 06 22 24 25 30 31 38 41 45 46 49 51 55 58 60 71 75 78
75      02/09/2010,04 07 09 15 17 19 21 22 23 26 28 30 33 35 37 44 53 57 62 78
76      02/08/2010,01 02 04 05 12 20 25 26 32 33 38 42 46 47 53 58 60 61 66 69
77      02/07/2010,03 04 05 17 18 22 23 27 28 33 34 37 46 48 55 56 62 67 71 76
78      02/06/2010,01 03 06 13 17 26 30 33 34 42 43 44 52 53 55 56 62 76 77 78
79      02/05/2010,05 10 21 28 31 32 34 35 36 37 45 47 49 52 58 59 65 66 68 75
80      02/04/2010,01 08 11 26 27 30 37 38 39 40 47 51 52 53 57 60 70 71 77 80
81      02/03/2010,07 08 09 12 15 17 30 34 37 39 42 43 49 56 57 60 70 74 78 79
82      02/02/2010,01 07 13 16 19 23 27 35 36 41 42 44 45 49 52 55 62 64 69 75
83      02/01/2010,08 09 10 11 12 14 18 22 26 32 34 39 43 54 55 68 76 77 79 80
84      01/31/2010,03 08 09 16 19 21 22 28 31 37 40 47 48 49 53 55 65 71 77 78
85      01/30/2010,04 08 13 20 22 28 30 34 35 36 40 45 46 47 53 54 66 69 72 74
86      01/29/2010,03 09 13 14 16 17 21 22 34 35 40 43 46 58 59 60 63 67 72 79
87      01/28/2010,01 11 16 19 20 22 27 33 37 46 47 54 58 61 62 63 68 69 73 79
88      01/27/2010,01 06 14 16 17 19 20 21 23 30 31 34 37 39 48 49 53 57 59 67
89      01/26/2010,11 16 20 23 24 30 31 33 36 37 48 54 57 60 63 68 70 75 77 78
90      01/25/2010,06 09 17 18 19 28 29 39 40 47 49 50 52 57 62 66 70 73 78 79
91      01/24/2010,08 10 15 20 21 26 27 28 31 32 36 42 45 48 50 52 67 70 73 74
92      01/23/2010,09 12 13 16 18 20 23 26 31 35 40 42 44 46 50 54 61 65 66 72
93      01/22/2010,02 07 11 15 23 28 35 38 39 50 52 54 60 63 66 67 68 69 77 80
94      01/21/2010,06 07 18 19 26 30 32 36 47 52 54 55 62 64 70 71 72 76 78 79
95      01/20/2010,01 05 09 10 23 25 28 32 34 35 49 53 55 56 58 59 65 67 68 77
96      01/19/2010,01 02 05 07 10 14 15 20 26 28 31 37 45 47 50 52 54 59 64 72
97      01/18/2010,03 05 09 17 23 25 28 29 35 39 40 44 50 52 58 59 62 78 79 80
98      01/17/2010,08 10 14 23 25 27 31 36 48 52 53 54 61 64 66 68 74 75 76 77
99      01/16/2010,04 06 12 18 22 23 35 41 42 47 48 52 55 59 60 61 65 70 75 76
00      01/15/2010,01 03 07 15 18 19 20 26 29 34 38 40 42 46 52 59 66 67 68 77
01      01/14/2010,02 04 05 09 10 17 32 34 37 38 41 42 44 48 57 61 69 71 72 77
02      01/13/2010,06 11 12 13 16 22 26 29 32 33 36 40 50 59 60 62 68 74 78 79
03      01/12/2010,01 03 07 16 18 31 32 34 36 38 40 42 47 49 57 59 63 71 74 79
04      01/11/2010,01 07 11 16 20 22 27 28 29 30 31 33 34 35 40 58 63 66 73 78
05      01/10/2010,01 02 04 08 17 22 23 24 27 29 31 53 54 57 66 68 70 73 75 78
06      01/09/2010,03 10 12 14 16 19 26 37 40 45 48 49 50 55 59 69 71 76 77 79
07      01/08/2010,04 07 12 14 15 18 25 29 30 32 33 38 41 43 45 59 63 68 71 76
08      01/07/2010,01 02 04 05 08 17 19 20 21 40 45 46 47 49 55 56 62 63 71 74
09      01/06/2010,01 07 13 17 21 27 29 34 36 43 47 51 53 60 62 65 68 71 78 80
10      01/05/2010,02 07 11 12 13 15 17 19 20 31 32 35 46 47 49 56 59 64 73 80
11      01/04/2010,01 02 12 14 17 18 26 28 31 38 43 46 48 52 56 58 65 67 68 70
12      01/03/2010,02 07 11 18 23 25 26 31 32 35 36 38 46 55 58 66 68 73 79 80
13      01/02/2010,05 06 08 19 24 28 30 35 42 44 48 53 55 59 62 64 66 72 73 76
14      01/01/2010,03 04 08 10 12 16 20 25 28 30 36 50 51 52 60 62 64 69 75 79
15      12/31/2009,01 04 06 18 19 21 22 24 35 44 46 47 51 53 54 64 67 68 72 75
16      12/30/2009,02 08 11 14 16 26 27 31 34 36 37 41 43 44 56 60 62 71 79 80
17      12/29/2009,02 04 10 12 13 20 24 25 28 37 40 42 44 54 56 61 62 63 75 79
18      12/28/2009,01 06 07 14 15 19 32 33 37 39 55 58 59 62 63 66 73 74 78 79
```

```
12/27/2009,05 06 07 16 18 29 33 42 43 47 48 49 51 52 57 59 63 64 65 73
12/26/2009,09 10 12 17 18 20 25 32 41 44 45 47 55 59 60 61 62 65 68 70
12/24/2009,06 14 15 21 23 34 35 37 51 52 56 57 60 61 70 71 73 74 75 76
12/23/2009,03 04 05 08 11 16 18 21 29 34 39 41 42 43 48 57 72 73 79 80
12/22/2009,01 02 05 06 08 12 13 14 20 26 28 36 38 48 49 57 61 75 79 80
12/21/2009,02 07 20 21 27 35 41 44 46 47 53 54 58 64 68 69 70 71 78 79
12/20/2009,03 05 06 13 22 26 33 39 40 47 48 49 55 58 59 62 64 72 74 75
12/19/2009,06 07 10 17 22 23 24 25 32 34 37 41 44 45 48 56 59 60 78 79
12/18/2009,05 07 08 15 23 34 39 41 50 52 54 60 62 66 67 70 71 72 75 80
12/17/2009,11 12 14 26 30 32 37 40 47 48 49 57 58 59 60 64 67 68 72 77
12/16/2009,02 03 09 16 20 35 38 39 46 47 52 55 56 57 60 65 66 70 74 78
12/15/2009,05 13 14 20 25 26 29 30 31 34 45 46 50 52 57 66 67 73 78 79
12/14/2009,04 06 11 17 18 20 23 24 26 33 34 38 49 55 68 71 72 74 76 78
12/13/2009,08 09 10 13 17 18 21 24 25 29 30 39 44 49 53 55 58 62 69 80
12/12/2009,05 11 14 16 23 24 29 32 34 35 36 37 43 44 48 52 60 65 68 76
12/11/2009,01 02 03 09 10 11 15 31 36 37 39 40 47 48 53 55 59 68 69 80
12/10/2009,02 06 07 19 20 21 22 24 29 34 43 47 48 50 58 65 67 69 77 80
12/09/2009,01 05 10 17 19 25 26 31 33 37 38 39 43 44 50 54 57 63 74 80
12/08/2009,01 13 14 16 17 19 20 24 36 37 38 42 44 58 63 64 72 73 77 79
12/07/2009,05 09 12 22 23 24 25 27 34 39 42 46 47 53 54 61 70 71 74 80
12/06/2009,01 05 11 17 20 28 39 40 41 43 44 48 51 55 56 60 61 67 70 74
12/05/2009,03 06 08 16 21 22 27 34 36 43 51 52 54 62 67 68 70 71 75 76
12/04/2009,01 02 05 13 16 18 19 23 26 28 33 46 48 54 55 61 63 71 72 74
12/03/2009,01 04 12 13 16 17 22 24 28 34 38 42 52 54 55 59 60 66 70 76
12/02/2009,03 04 05 07 11 16 19 38 39 43 44 51 52 56 61 63 65 68 70 71
12/01/2009,01 03 15 24 25 26 28 30 33 40 41 43 45 46 56 62 64 65 69 78
11/30/2009,02 04 15 24 25 26 28 32 34 39 45 46 48 50 53 60 63 64 69 72
11/29/2009,01 02 03 10 12 15 23 25 30 31 33 34 36 39 40 41 48 50 66 74
11/28/2009,08 09 16 27 28 30 31 32 33 43 52 56 59 60 61 64 65 68 74 76
11/27/2009,01 02 03 11 16 17 18 27 28 30 31 35 37 38 39 50 52 54 71 73
11/26/2009,10 11 18 21 28 30 34 36 37 43 44 47 52 56 58 68 69 75 79 80
11/25/2009,02 09 10 13 16 25 26 35 36 39 42 45 48 52 53 59 66 74 79 80
11/24/2009,08 15 18 20 21 22 24 25 31 32 33 35 42 44 45 47 54 64 68 72
11/23/2009,01 03 08 15 18 19 20 27 28 31 41 42 47 52 55 57 65 66 77 78
11/22/2009,14 20 25 29 36 42 43 44 51 52 58 61 63 66 69 71 74 77 78 80
11/21/2009,03 09 12 25 27 29 34 35 37 39 45 48 54 56 58 60 62 66 67 80
11/20/2009,04 05 06 12 13 17 21 25 40 42 50 58 61 65 66 68 72 75 77 79
11/19/2009,05 09 13 26 27 29 33 36 45 47 48 49 55 57 58 69 72 76 78 79
11/18/2009,06 07 09 15 17 31 32 34 36 40 50 52 56 61 69 70 73 74 77 79
11/17/2009,11 13 17 20 27 31 32 36 41 44 50 51 54 55 57 65 69 76 79 80
11/16/2009,03 05 11 12 18 21 26 29 31 35 37 39 46 48 49 53 55 58 64 74
11/15/2009,04 14 17 22 29 30 32 36 37 46 55 58 60 62 65 69 70 75 77 80
11/14/2009,09 12 18 19 22 32 38 40 41 43 46 47 53 54 55 58 59 62 73 76
11/13/2009,01 04 06 07 10 15 16 18 24 26 30 34 36 38 39 44 47 51 57 74
11/12/2009,01 07 08 11 15 27 28 30 32 37 39 40 44 46 52 56 57 64 75 78
11/11/2009,01 06 12 13 16 25 29 34 40 41 42 44 48 51 58 62 64 71 73 79
11/10/2009,04 06 11 14 17 28 35 42 45 46 48 58 62 66 67 69 71 75 76 80
11/09/2009,08 13 14 15 17 18 19 24 27 30 34 35 36 43 49 61 62 68 70 80
11/08/2009,02 07 29 30 33 34 36 37 44 45 48 51 54 58 61 65 69 70 73 74
11/07/2009,01 04 05 08 11 14 16 20 21 27 29 30 35 36 43 45 48 54 59 69
11/06/2009,05 06 08 10 21 24 28 32 35 37 42 46 54 55 60 61 64 70 75 79
11/05/2009,05 07 11 13 17 23 32 36 38 44 49 51 53 54 56 58 59 61 66 77
11/04/2009,03 12 17 22 24 28 35 39 43 44 45 48 50 57 66 69 74 75 79 80
11/03/2009,05 13 17 26 28 32 38 40 41 43 45 49 59 61 63 70 71 74 77 78
```

```
11/02/2009,01 03 06 11 14 16 18 20 29 38 39 41 47 50 52 57 60 62 67 72
11/01/2009,02 11 14 17 21 23 28 30 34 39 40 43 49 53 56 57 58 60 63 66
10/31/2009,05 09 13 23 24 28 30 35 36 40 44 47 49 52 59 61 66 74 77 80
10/30/2009,02 08 09 19 22 25 29 33 38 41 46 51 58 66 67 69 70 71 76 80
10/29/2009,07 09 12 19 21 26 37 40 41 43 45 46 49 54 58 63 64 69 73 76
10/28/2009,03 06 08 25 28 32 36 39 41 44 45 46 54 55 58 63 66 72 75 76
10/27/2009,05 08 09 13 16 22 25 26 30 32 37 40 46 47 53 63 65 67 72 80
10/26/2009,04 09 17 19 20 23 29 36 37 39 46 52 54 56 57 61 65 66 69 79
10/25/2009,01 03 07 18 19 28 37 41 43 48 49 52 56 67 69 70 71 76 77 80
10/24/2009,01 07 08 14 18 19 20 32 50 52 55 56 62 65 66 70 71 73 75 76
10/23/2009,03 08 10 12 13 15 18 21 28 29 31 36 43 46 48 57 58 64 66 67
10/22/2009,04 14 19 20 21 23 28 37 38 39 40 41 42 53 56 60 65 69 70 76
10/21/2009,01 03 13 14 18 22 27 35 37 39 41 52 54 60 65 68 72 73 76 77
10/20/2009,01 02 04 08 13 15 18 21 22 24 25 28 38 42 43 52 63 68 71 80
10/19/2009,01 09 11 14 18 21 22 25 27 28 32 36 41 45 57 60 61 67 70 75
10/18/2009,04 19 25 26 28 31 36 38 41 43 46 49 56 59 61 64 65 67 70 76
10/17/2009,02 10 13 16 26 29 38 44 47 50 55 56 62 63 64 66 68 69 71 76
10/16/2009,03 08 10 16 24 26 28 29 35 38 44 45 48 51 55 59 68 72 75 78
10/15/2009,01 04 05 12 13 14 16 22 23 24 27 28 31 33 44 56 62 65 74 75
10/14/2009,08 12 14 15 16 19 20 23 28 34 38 39 47 50 51 60 62 63 74 77
10/13/2009,03 06 07 09 21 22 23 26 29 30 31 35 37 42 46 48 49 50 72 80
10/12/2009,01 04 16 17 19 28 30 33 42 43 44 45 55 58 59 63 66 69 77 78
10/11/2009,05 06 10 11 14 15 16 18 19 24 27 40 46 47 48 49 52 57 62 71
10/10/2009,04 13 14 16 18 20 22 23 28 29 30 31 34 55 56 57 70 72 73 76
10/09/2009,01 02 13 19 22 26 28 31 40 42 49 56 57 58 60 63 68 69 71 73
10/08/2009,02 07 09 11 12 15 20 31 32 37 38 44 46 50 51 60 63 68 74 80
10/07/2009,07 10 18 24 25 33 36 41 45 46 47 49 50 53 61 62 63 65 68 74
10/06/2009,05 08 12 14 17 19 20 23 25 26 29 34 37 40 45 49 60 62 74 80
10/05/2009,02 07 08 09 12 13 18 24 26 28 34 35 36 48 50 51 53 70 75 80
10/04/2009,01 05 13 16 18 22 24 26 27 28 34 40 46 47 51 57 66 69 74 76
10/03/2009,05 06 07 13 15 18 19 24 33 44 45 53 55 56 59 61 63 64 72 78
10/02/2009,01 12 22 23 27 31 36 38 39 43 53 54 59 64 66 71 74 76 79 80
10/01/2009,05 12 14 18 19 20 27 28 30 31 32 36 38 39 47 62 64 73 74 78
09/30/2009,03 11 12 16 17 18 22 24 33 37 40 42 46 53 59 64 65 66 69 78
09/29/2009,02 04 05 06 07 15 18 26 28 29 34 36 39 42 46 54 70 71 73 79
09/28/2009,07 08 11 14 15 16 17 19 20 23 34 38 42 49 51 52 57 58 63 67
09/27/2009,03 07 09 20 25 26 27 35 40 45 46 47 48 52 53 54 55 66 70 72
09/26/2009,04 05 13 17 24 26 28 38 42 46 47 49 55 59 65 67 69 76 77 79
09/25/2009,10 16 19 22 28 32 38 40 46 48 49 53 57 60 62 67 74 75 76 77
09/24/2009,07 09 10 14 19 22 26 31 33 34 39 42 48 52 54 67 72 73 74 80
09/23/2009,11 20 21 24 26 28 30 43 46 50 55 57 59 63 65 71 72 73 75 77
09/22/2009,05 23 25 28 32 33 36 46 48 52 55 57 60 61 64 66 72 73 74 75
09/21/2009,05 07 09 12 14 16 24 25 33 43 49 52 53 64 67 68 70 71 74 76
09/20/2009,06 08 09 13 18 20 22 27 29 35 39 43 47 52 64 68 70 73 78 79
09/19/2009,02 07 22 24 25 27 28 30 36 37 42 43 44 50 53 59 60 61 76 80
09/18/2009,01 04 07 08 10 14 24 36 37 38 39 42 53 54 60 62 63 68 73 75
09/17/2009,05 08 12 14 15 25 31 33 35 39 42 43 48 49 53 61 64 72 73 76
09/16/2009,01 04 07 08 10 11 14 17 18 19 20 21 22 30 31 34 40 48 55 60
09/15/2009,04 05 07 18 24 26 36 43 44 45 47 54 56 58 60 62 63 73 74 78
09/14/2009,02 08 10 11 12 13 14 17 23 24 25 30 31 51 61 66 67 74 77 78
09/13/2009,03 10 14 15 22 26 30 31 37 41 45 47 49 53 55 60 69 70 73 74
09/12/2009,02 07 10 12 17 19 22 23 26 27 29 33 43 46 49 60 63 69 72 77
09/11/2009,04 08 09 11 15 17 24 32 33 36 37 38 39 41 53 56 66 71 73 78
09/10/2009,03 05 17 26 31 33 36 39 40 41 42 43 48 52 53 57 76 78 79 80
```

```
09/09/2009,05 12 19 22 26 27 29 30 32 37 38 39 42 45 46 56 58 67 68 78
09/08/2009,02 03 04 05 16 18 20 21 31 34 37 40 48 59 63 64 66 69 75 80
09/07/2009,02 03 09 10 15 16 22 26 42 43 46 52 59 60 61 64 67 68 69 70
09/06/2009,02 06 17 20 22 24 29 30 33 36 37 39 45 46 48 55 57 67 70 77
09/05/2009,04 08 10 15 26 31 35 37 39 42 44 46 48 55 58 60 63 65 68 72
09/04/2009,01 02 10 15 22 37 42 43 45 46 49 50 53 57 59 63 73 74 78 80
09/03/2009,06 08 09 10 14 15 17 19 29 33 35 43 49 58 68 69 71 76 78 80
09/02/2009,03 15 24 27 28 30 33 37 39 46 47 50 52 65 66 67 68 72 73 77
09/01/2009,02 11 12 14 18 28 29 30 33 38 43 44 45 46 47 57 58 62 75 80
08/31/2009,03 16 21 27 30 33 40 44 46 48 50 54 55 58 59 65 67 70 78 79
08/30/2009,03 04 08 13 14 19 20 25 27 36 37 39 40 43 58 68 71 73 75 77
08/29/2009,14 17 19 20 21 37 44 45 52 53 54 58 59 63 65 67 71 73 74 78
08/28/2009,01 02 04 05 12 16 21 23 24 32 42 43 50 52 55 61 63 64 65 74
08/27/2009,05 06 10 14 15 16 18 21 28 31 34 35 40 48 50 51 52 53 55 66
08/26/2009,01 06 10 11 15 19 21 23 37 40 41 42 48 49 50 62 70 73 74 78
08/25/2009,09 26 28 32 36 40 41 42 46 50 53 56 60 66 67 68 72 74 75 76
08/24/2009,04 06 10 13 23 28 32 34 40 44 53 56 57 60 66 70 72 73 79 80
08/23/2009,02 11 13 17 21 33 38 40 41 46 48 53 54 55 56 65 68 69 71 80
08/22/2009,01 03 09 15 22 25 31 33 38 40 43 46 47 48 49 62 65 67 68 73
08/21/2009,01 12 20 22 30 31 35 40 41 49 51 55 58 59 68 71 72 73 75 77
08/20/2009,12 14 15 19 20 21 30 31 34 35 37 40 46 59 60 63 67 73 78 80
08/19/2009,01 05 06 08 15 19 21 24 25 37 43 44 46 47 49 56 61 63 74 79
08/18/2009,05 08 12 13 25 30 40 41 42 43 50 55 58 59 61 62 65 68 72 80
08/17/2009,01 05 06 08 13 14 16 22 23 26 28 40 45 46 54 59 63 69 73 80
08/16/2009,03 10 12 17 25 28 31 41 42 46 48 49 56 58 59 66 67 76 77 78
08/15/2009,02 03 09 10 11 13 14 19 26 27 36 37 42 54 55 58 62 68 78 80
08/14/2009,04 11 12 24 27 30 32 35 36 37 38 43 47 54 59 65 68 72 74 77
08/13/2009,03 13 15 18 19 21 24 26 32 33 40 41 42 49 58 59 67 70 71 72
08/12/2009,01 13 21 22 23 27 31 33 43 45 49 50 55 56 58 68 73 74 76 80
08/11/2009,01 03 04 05 12 14 17 34 36 40 50 51 52 54 57 58 63 66 70 73
08/10/2009,01 06 09 16 27 28 31 37 39 44 46 48 50 52 54 59 64 73 74 79
08/09/2009,01 02 04 06 13 21 22 25 33 36 39 42 44 47 48 57 58 66 75 78
08/08/2009,03 04 07 08 17 20 24 31 32 34 42 50 51 52 58 59 69 72 74 76
08/07/2009,01 02 14 19 20 25 26 27 34 35 36 39 51 58 61 62 72 73 74 75
08/06/2009,02 06 08 18 22 26 32 37 42 45 49 51 56 60 62 67 70 73 76 80
08/05/2009,01 02 06 08 12 21 22 27 29 30 38 40 44 52 55 56 66 76 79 80
08/04/2009,03 09 13 17 18 20 21 25 29 31 33 37 38 42 62 67 68 69 73 76
08/03/2009,01 08 14 19 21 26 30 33 36 43 44 51 53 55 59 61 66 67 70 77
08/02/2009,05 08 11 12 15 19 22 27 30 32 34 38 48 57 58 63 67 72 73 74
08/01/2009,02 12 13 23 24 28 29 40 41 43 48 49 50 53 55 58 66 68 69 76
07/31/2009,04 07 10 11 16 20 28 35 38 42 46 48 49 55 56 57 58 61 74 79
07/30/2009,01 10 23 24 28 29 31 32 41 42 43 54 58 67 69 72 73 77 78 80
07/29/2009,02 06 08 11 15 19 21 25 27 28 31 34 42 43 49 56 71 72 74 75
07/28/2009,08 11 12 17 19 22 28 29 31 38 39 40 48 58 60 65 67 69 76 80
07/27/2009,02 05 06 10 12 20 21 22 24 32 33 35 44 47 55 63 65 74 75 77
07/26/2009,01 06 09 13 15 16 17 24 25 26 29 37 39 44 52 69 71 77 79 80
07/25/2009,01 06 07 09 11 21 24 31 35 36 38 41 43 52 55 65 66 71 72 74
07/24/2009,06 08 13 19 29 32 34 35 36 42 45 49 62 66 67 68 69 71 75 79
07/23/2009,04 06 15 16 19 20 23 28 29 31 32 33 37 52 57 58 59 61 71 73
07/22/2009,02 03 07 12 13 18 19 27 30 35 38 46 50 53 56 59 60 65 66 69
07/21/2009,04 10 14 15 19 27 29 43 46 47 49 53 60 64 65 66 67 75 77 79
07/20/2009,01 09 12 14 23 27 33 37 38 43 49 52 53 58 59 70 71 72 78 79
07/19/2009,01 11 13 20 22 25 29 31 34 35 37 47 49 54 56 60 62 65 71 75
07/18/2009,08 11 13 18 19 26 27 28 35 43 47 51 53 57 59 60 61 64 73 74
```

```
07/17/2009,01 07 10 23 31 38 39 45 47 55 56 60 65 66 67 68 75 76 78 80
07/16/2009,05 07 10 13 16 19 24 28 30 38 39 40 43 46 48 52 61 71 73 74
07/15/2009,02 09 11 13 15 21 24 26 27 29 40 42 48 56 59 64 65 69 73 75
07/14/2009,03 04 05 20 28 31 35 40 44 45 46 54 56 58 59 65 67 72 77 79
07/13/2009,04 05 16 18 20 23 28 32 36 37 41 45 49 56 58 65 67 69 72 79
07/12/2009,03 05 11 14 15 19 34 40 49 50 54 57 59 60 61 63 66 71 76 79
07/11/2009,02 14 18 19 22 31 32 36 37 41 49 51 55 60 62 70 71 73 74 78
07/10/2009,02 06 08 09 10 22 32 37 38 44 49 50 51 56 60 63 68 75 76 80
07/09/2009,05 07 09 10 11 17 20 21 22 29 30 31 36 38 41 42 47 64 71 77
07/08/2009,01 07 08 10 11 17 18 31 34 37 46 49 53 57 60 66 67 70 72 75
07/07/2009,09 12 21 24 25 27 28 29 32 37 41 44 48 49 51 52 56 59 65 66
07/06/2009,07 10 12 16 26 29 30 32 38 46 51 52 60 62 67 72 74 75 79 80
07/05/2009,07 08 11 12 15 27 28 29 44 49 51 53 55 62 65 67 68 71 75 80
07/04/2009,01 02 16 17 18 23 24 25 29 34 40 44 45 46 47 59 64 69 73 76
07/03/2009,01 03 05 06 15 20 28 32 39 40 41 42 44 49 54 63 65 66 69 70
07/02/2009,01 02 07 13 24 26 32 36 38 42 45 47 49 50 60 65 66 73 77 80
07/01/2009,04 12 14 18 26 28 29 32 36 39 45 52 53 55 60 62 66 74 75 77
06/30/2009,03 04 09 16 19 23 24 30 31 35 40 43 45 47 52 55 56 66 72 80
06/29/2009,01 03 08 10 15 20 21 22 30 32 37 41 50 54 55 58 59 75 79 80
06/28/2009,24 25 28 29 30 31 33 35 36 37 43 45 50 55 57 59 61 73 74 78
06/27/2009,01 07 08 09 13 20 25 34 37 38 42 47 49 55 59 63 66 70 71 73
06/26/2009,04 14 25 26 32 36 39 40 50 52 53 59 61 62 63 67 69 73 76 80
06/25/2009,07 12 13 17 18 19 20 26 28 29 33 51 54 57 59 62 64 72 74 80
06/24/2009,14 15 19 20 21 29 32 40 41 44 51 57 60 62 69 72 73 74 75 78
06/23/2009,03 07 11 13 16 18 19 27 39 43 45 46 48 50 54 56 64 67 72 80
06/22/2009,01 06 09 13 14 17 20 22 28 31 35 40 41 46 54 59 69 72 74 78
06/21/2009,04 05 07 08 09 11 23 28 34 40 50 55 56 57 62 65 68 71 73 76
06/20/2009,04 09 10 18 22 23 29 31 32 34 43 46 53 57 58 60 68 69 71 75
06/19/2009,01 10 14 16 24 25 34 36 39 41 45 52 54 56 57 60 66 69 77 79
06/18/2009,04 07 10 11 13 17 20 28 34 35 43 44 48 56 58 59 62 67 74 75
06/17/2009,01 06 09 12 13 14 15 19 22 23 36 37 38 42 48 52 58 59 72 73
06/16/2009,01 05 06 14 15 16 17 20 22 24 26 31 35 40 47 58 65 71 76 80
06/15/2009,01 12 14 19 20 21 26 31 32 33 42 43 45 47 49 50 51 53 62 63
06/14/2009,09 13 14 15 18 22 28 29 30 31 36 37 46 50 55 57 73 74 76 77
06/13/2009,01 10 11 13 14 18 21 24 34 37 40 51 59 60 61 68 69 71 73 80
06/12/2009,01 02 03 04 12 13 18 19 20 23 28 36 40 41 45 54 61 66 67 75
06/11/2009,04 05 08 13 15 23 27 35 37 38 40 54 56 57 63 69 72 76 78 80
06/10/2009,02 05 13 14 15 18 20 22 26 34 35 36 39 44 46 50 52 54 56 65
06/09/2009,01 03 07 13 15 22 25 31 34 36 37 41 48 54 57 58 67 69 73 79
06/08/2009,09 11 12 23 25 30 36 37 41 51 52 53 57 60 63 71 73 74 77 80
06/07/2009,02 06 08 10 12 19 24 25 26 32 37 38 40 42 45 57 65 66 67 79
06/06/2009,05 10 12 23 24 27 32 34 35 36 42 47 56 58 60 64 68 71 75 79
06/05/2009,03 04 15 18 21 34 35 37 42 50 52 56 58 60 61 66 67 75 77 79
06/04/2009,11 12 15 18 24 25 26 31 32 34 37 38 47 51 55 66 67 71 72 77
06/03/2009,02 03 07 09 11 13 18 19 27 28 29 34 40 46 49 58 59 70 73 74
06/02/2009,03 06 15 17 23 25 26 30 32 35 38 50 56 58 66 67 68 69 73 78
06/01/2009,05 13 16 21 25 29 32 34 35 36 42 47 48 63 65 67 72 73 75 80
05/31/2009,01 05 08 19 21 28 29 32 33 34 38 42 47 50 53 59 63 65 77 79
05/30/2009,03 05 08 11 14 20 21 22 29 31 35 42 53 54 59 62 65 73 74 77
05/29/2009,01 10 11 14 23 27 28 29 32 33 34 35 38 42 48 59 69 70 71 76
05/28/2009,10 11 13 15 16 22 30 31 33 35 36 37 39 43 55 58 68 70 71 74
05/27/2009,02 04 09 17 20 23 26 30 31 37 43 44 48 55 64 65 70 72 78 80
05/26/2009,06 10 12 25 27 29 30 32 34 37 39 48 50 52 54 59 62 67 70 71
05/25/2009,05 10 16 28 30 36 37 41 42 44 50 51 56 62 64 66 73 76 77 80
```

```
05/24/2009,04 05 21 25 26 28 32 37 38 40 41 45 46 49 58 59 64 68 71 72
05/23/2009,05 06 10 22 29 35 38 42 43 46 49 51 60 64 67 69 75 76 78 80
05/22/2009,01 11 13 14 15 32 35 37 38 44 49 54 64 65 68 71 73 74 78 79
05/21/2009,05 06 07 15 18 24 25 28 29 35 37 39 47 50 53 55 58 60 62 70
05/20/2009,03 10 16 17 18 24 27 30 33 42 44 49 51 56 58 64 70 75 79 80
05/19/2009,01 04 17 18 20 30 35 36 40 46 51 55 56 59 65 67 69 71 76 80
05/18/2009,04 05 11 19 20 21 22 23 28 31 33 36 40 41 47 51 55 66 75 80
05/17/2009,07 08 14 23 30 32 33 36 39 41 47 48 49 50 54 63 65 66 74 75
05/16/2009,01 05 19 26 29 31 32 37 41 42 43 51 52 59 63 65 69 72 73 78
05/15/2009,01 04 06 08 09 14 21 31 32 37 42 44 46 49 50 53 62 65 74 79
05/14/2009,02 03 17 20 21 22 26 27 29 32 38 39 46 48 49 57 59 60 71 80
05/13/2009,01 05 09 12 13 15 18 28 29 30 41 47 52 53 59 61 70 77 78 79
05/12/2009,02 03 04 08 12 14 22 24 26 37 38 39 41 52 56 67 69 74 75 78
05/11/2009,05 06 11 14 17 23 29 31 32 41 44 51 54 55 58 60 67 74 78 79
05/10/2009,09 12 18 24 25 32 35 43 53 55 56 58 62 64 69 73 74 75 77 80
05/09/2009,02 15 16 22 25 27 31 48 53 55 60 62 65 67 68 70 71 72 73 77
05/08/2009,14 15 22 25 27 28 31 32 35 42 50 52 55 58 59 69 73 74 77 80
05/07/2009,06 07 10 13 14 21 22 41 46 47 50 53 54 66 72 73 74 75 79 80
05/06/2009,06 15 17 18 19 20 21 25 27 29 33 38 40 42 50 62 66 67 79 80
05/05/2009,01 02 13 16 23 24 29 30 36 38 41 46 48 51 55 56 60 69 70 71
05/04/2009,06 18 19 24 25 28 30 32 33 34 37 38 44 51 56 60 62 66 70 71
05/03/2009,13 17 18 20 23 25 26 29 33 37 40 46 57 66 69 70 72 73 74 80
05/02/2009,01 03 06 11 13 15 17 18 20 26 30 43 54 61 68 71 74 75 76 77
05/01/2009,02 03 07 14 15 18 23 26 29 30 36 38 40 45 49 50 64 68 69 75
04/30/2009,01 02 17 18 19 24 27 30 31 40 42 43 44 45 49 52 58 62 63 65
04/29/2009,05 09 10 12 15 16 19 22 35 36 39 41 42 48 52 58 61 62 78 80
04/28/2009,09 15 28 29 33 34 35 42 44 45 46 49 58 61 69 72 73 74 75 76
04/27/2009,06 08 10 14 23 26 29 34 35 42 46 49 50 51 52 53 56 57 78 80
04/26/2009,04 05 07 09 15 16 20 24 25 29 34 37 41 45 54 57 69 75 76 78
04/25/2009,05 12 14 17 18 19 20 21 25 29 33 36 38 40 42 45 57 64 77 79
04/24/2009,03 08 12 21 30 33 34 36 37 45 46 47 57 58 63 67 68 73 78 80
04/23/2009,06 17 19 21 30 36 37 43 44 46 57 60 64 65 66 67 68 72 73 80
04/22/2009,04 05 08 09 10 26 30 35 42 44 45 58 59 60 63 65 66 69 78 79
04/21/2009,02 03 06 13 14 15 16 19 22 25 26 27 37 41 44 51 53 61 67 79
04/20/2009,03 04 11 13 21 28 37 39 40 43 44 48 50 56 59 61 62 69 76 79
04/19/2009,05 07 14 17 18 27 28 31 37 39 47 48 56 64 68 72 76 78 79 80
04/18/2009,07 08 09 11 14 15 22 23 24 27 32 37 41 51 52 54 57 64 69 79
04/17/2009,04 05 06 07 10 11 13 14 15 21 32 44 52 54 55 57 59 63 75 79
04/16/2009,03 06 07 10 25 28 31 34 39 42 44 45 46 47 49 54 57 76 77 80
04/15/2009,01 02 05 08 17 19 28 29 30 38 41 44 48 59 61 62 66 67 68 71
04/14/2009,03 13 19 29 32 37 39 46 47 48 53 54 55 57 61 64 69 71 73 79
04/13/2009,01 12 13 17 18 20 25 29 30 39 40 41 45 50 57 69 72 73 74 79
04/12/2009,04 06 15 21 27 34 36 40 45 50 51 53 56 61 65 67 70 72 73 77
04/11/2009,03 06 09 10 14 19 22 23 29 32 38 43 47 50 51 54 68 72 78 80
04/10/2009,02 03 16 17 19 20 27 28 32 39 41 49 55 56 58 60 63 67 74 79
04/09/2009,03 12 13 16 25 29 32 33 35 37 38 42 48 54 55 60 62 64 72 78
04/08/2009,05 11 13 20 30 31 37 38 39 41 44 45 46 50 52 60 63 64 66 71
04/07/2009,01 11 15 16 26 28 29 30 33 34 42 44 47 49 51 57 61 62 66 76
04/06/2009,05 06 15 21 27 33 35 42 44 46 50 55 57 60 61 64 67 68 71 78
04/05/2009,04 08 10 12 22 25 30 31 34 44 50 52 55 56 58 61 65 68 78 80
04/04/2009,04 05 10 14 24 28 30 31 35 37 46 48 55 62 63 64 66 74 78 80
04/03/2009,03 05 10 12 14 18 19 20 21 24 25 39 43 52 69 72 73 76 78 80
04/02/2009,08 10 13 14 25 27 31 33 35 38 42 48 52 56 57 59 68 73 77 78
04/01/2009,05 10 14 16 18 29 33 40 42 45 53 54 56 60 66 68 70 72 76 78
```

```
89      03/31/2009,08 13 16 17 28 32 40 41 42 44 45 53 56 57 58 59 60 61 66 73
90      03/30/2009,02 03 06 12 13 14 17 20 21 24 25 33 39 40 45 47 56 71 74 80
91      03/29/2009,03 11 14 17 29 39 40 49 56 60 65 66 67 69 70 71 72 73 74 76
92      03/28/2009,04 05 16 18 24 25 33 34 36 45 51 52 54 55 67 71 74 75 77 80
93      03/27/2009,02 09 13 15 18 20 24 26 32 34 39 40 46 52 55 56 63 70 76 78
94      03/26/2009,10 11 17 20 21 27 31 32 44 45 49 50 55 56 57 61 62 64 70 71
95      03/25/2009,04 06 07 08 11 13 15 17 19 21 37 42 51 55 56 59 63 65 69 75
96      03/24/2009,10 11 12 13 14 21 26 27 30 45 47 48 49 60 62 65 67 71 76 78
97      03/23/2009,04 05 07 17 23 24 40 41 48 50 51 53 55 57 59 63 75 77 79 80
98      03/22/2009,01 04 11 12 18 19 26 27 31 33 35 37 40 43 54 57 70 76 77 80
99      03/21/2009,05 08 11 19 25 28 31 33 35 41 42 44 50 52 58 62 68 71 74 76
00      03/20/2009,04 16 20 22 24 25 28 30 31 37 40 45 48 49 53 64 66 71 74 79
01      03/19/2009,04 14 19 23 25 26 28 35 36 37 38 43 48 49 55 56 61 62 68 74
02      03/18/2009,03 07 16 17 20 21 25 27 32 33 36 44 56 65 66 71 72 73 74 79
03      03/17/2009,07 08 20 25 29 31 32 42 43 51 52 54 55 59 62 63 69 73 77 79
04      03/16/2009,09 10 13 20 21 22 37 43 44 49 55 56 60 62 64 65 71 73 75 78
05      03/15/2009,03 05 07 09 13 16 17 22 27 35 38 39 43 45 48 51 59 68 69 75
06      03/14/2009,02 12 13 15 18 29 32 39 42 51 52 57 58 61 62 63 65 66 69 74
07      03/13/2009,01 03 04 07 09 14 16 21 33 37 38 43 45 51 52 57 58 59 70 73
08      03/12/2009,02 04 09 10 15 19 23 32 47 49 51 54 58 62 63 66 71 72 73 76
09      03/11/2009,01 02 05 28 35 37 38 42 43 44 46 50 58 59 64 66 70 71 76 77
10      03/10/2009,03 06 13 17 18 23 25 27 41 44 45 50 52 53 58 60 67 73 75 77
11      03/09/2009,03 17 19 22 28 33 36 37 42 46 48 50 54 58 60 61 64 65 66 67
12      03/08/2009,03 07 09 11 15 17 19 20 22 23 33 36 43 44 45 47 51 64 68 78
13      03/07/2009,01 02 04 08 09 11 12 22 24 36 40 42 43 46 49 56 59 62 66 73
14      03/06/2009,03 09 12 13 24 25 29 30 37 38 39 42 47 49 60 66 72 74 77 80
15      03/05/2009,02 05 06 07 08 09 11 13 16 20 29 39 42 55 57 60 65 68 74 77
16      03/04/2009,02 07 09 17 18 29 33 36 39 42 43 48 53 57 59 60 62 64 72 73
17      03/03/2009,05 07 10 13 18 24 25 39 41 44 45 57 58 59 61 63 71 72 73 80
18      03/02/2009,03 16 23 27 31 37 42 43 46 48 52 54 58 61 63 64 66 67 76 79
19      03/01/2009,01 04 10 12 16 17 25 27 30 43 45 46 47 48 53 57 60 63 67 79
20      02/28/2009,03 07 11 12 14 21 25 35 37 39 41 42 50 52 54 68 70 73 74 80
21      02/27/2009,04 06 11 13 16 17 24 25 27 28 29 41 47 57 59 63 67 73 74 79
22      02/26/2009,01 03 05 06 11 12 17 22 26 32 41 49 50 57 58 59 64 69 70 75
23      02/25/2009,02 08 10 15 18 20 25 33 35 38 42 44 46 51 57 59 61 66 68 79
24      02/24/2009,06 10 11 14 16 19 22 34 38 41 43 49 50 59 65 68 69 70 76 78
25      02/23/2009,01 02 05 08 13 21 22 23 26 32 35 37 43 45 62 63 70 71 77 79
26      02/22/2009,04 15 17 18 29 31 34 35 41 43 45 48 49 52 57 59 60 64 71 75
27      02/21/2009,02 06 10 15 24 27 29 32 36 37 39 49 50 54 55 66 73 74 75 80
28      02/20/2009,03 09 16 17 23 25 27 29 32 38 40 51 52 54 56 58 59 63 78 79
29      02/19/2009,02 05 18 22 25 30 31 33 38 46 52 54 55 60 62 65 69 70 74 79
30      02/18/2009,03 16 19 22 24 35 36 37 40 41 49 52 55 57 58 62 64 67 75 77
31      02/17/2009,06 07 08 09 16 17 21 26 36 37 38 45 51 55 57 64 69 70 71 79
32      02/16/2009,02 05 14 15 19 21 23 24 44 48 49 53 54 55 59 62 71 72 79 80
33      02/15/2009,01 03 11 14 15 18 19 21 26 31 32 34 37 43 60 61 65 68 70 80
34      02/14/2009,01 02 03 18 23 30 32 33 36 37 40 46 48 55 62 69 70 71 74 76
35      02/13/2009,05 08 10 11 12 14 26 30 31 46 50 54 59 60 62 63 64 67 77 78
36      02/12/2009,14 15 16 24 26 35 37 40 41 47 54 56 58 62 64 68 69 73 75 79
37      02/11/2009,03 07 12 14 16 17 18 22 23 30 31 33 44 52 57 58 60 65 67 71
38      02/10/2009,01 10 12 14 15 18 25 26 27 38 39 46 52 54 59 63 64 65 66 77
39      02/09/2009,01 13 14 17 24 26 29 32 35 38 43 46 48 50 54 63 67 73 79 80
40      02/08/2009,02 06 12 14 23 24 34 38 42 47 53 54 58 67 69 70 71 73 76 80
41      02/07/2009,02 09 14 17 18 23 28 31 32 36 38 42 49 50 55 58 61 62 65 77
42      02/06/2009,01 03 04 05 09 11 23 24 26 27 31 34 35 41 49 64 65 70 73 74
```

```
02/05/2009,03 10 11 20 27 32 33 36 38 40 41 42 44 45 51 57 59 61 66 78
02/04/2009,01 02 03 05 14 21 33 39 45 47 49 53 54 65 66 69 74 75 76 80
02/03/2009,02 05 16 18 25 26 29 31 36 37 39 42 45 50 52 54 66 67 78 80
02/02/2009,03 10 15 21 23 29 36 37 38 41 45 48 55 57 59 62 67 68 72 77
02/01/2009,01 08 15 17 22 24 27 30 38 39 48 50 53 57 61 68 71 76 78 80
01/31/2009,04 06 08 19 25 27 30 31 32 34 38 44 45 49 51 58 59 65 73 78
01/30/2009,01 02 03 10 15 19 20 25 31 47 52 56 59 60 63 68 71 72 74 79
01/29/2009,06 07 23 28 29 34 42 45 47 51 53 55 57 60 61 66 67 72 73 77
01/28/2009,04 08 17 19 24 25 33 35 38 43 44 46 47 49 51 54 66 74 76 79
01/27/2009,06 10 13 14 20 23 28 34 40 41 42 48 54 57 67 73 77 78 79 80
01/26/2009,02 04 05 06 13 19 21 28 36 46 49 56 60 61 65 66 69 70 72 78
01/25/2009,02 04 19 28 32 36 37 41 43 48 50 52 57 62 63 67 72 73 76 80
01/24/2009,04 14 16 24 28 31 33 39 40 44 46 47 49 51 58 60 66 67 71 78
01/23/2009,05 12 15 17 24 27 28 34 39 42 44 45 55 56 58 61 62 64 69 72
01/22/2009,03 10 17 18 20 22 24 33 40 42 51 52 54 61 64 67 69 73 78 80
01/21/2009,05 07 16 19 21 25 27 30 31 39 41 48 52 54 55 63 65 73 74 80
01/20/2009,06 09 14 16 19 21 22 29 30 32 43 51 53 56 61 62 68 75 77 80
01/19/2009,02 03 05 14 18 22 24 27 29 30 31 34 35 38 47 57 60 68 71 73
01/18/2009,03 13 15 22 26 27 31 34 38 39 43 51 52 54 56 57 63 64 68 78
01/17/2009,07 08 14 15 17 27 29 34 45 48 52 62 66 68 69 70 72 75 76 79
01/16/2009,03 04 15 18 19 21 22 26 37 38 42 48 54 55 56 58 69 71 72 74
01/15/2009,09 11 13 14 15 18 25 35 36 39 43 47 49 51 53 54 56 61 76 79
01/14/2009,01 03 06 10 15 17 27 33 35 40 41 48 55 57 58 62 64 66 68 72
01/13/2009,04 07 12 17 19 21 32 34 46 47 48 54 64 65 67 70 75 77 79 80
01/12/2009,05 13 17 22 26 27 31 33 34 35 38 40 41 46 53 54 63 68 71 73
01/11/2009,01 03 13 15 19 21 26 34 40 41 51 54 58 59 61 63 69 70 74 79
01/10/2009,02 07 08 10 15 17 19 24 33 40 41 45 47 52 56 57 62 67 68 69
01/09/2009,01 09 11 24 33 42 43 44 49 50 51 57 58 59 62 64 72 73 79 80
01/08/2009,07 08 10 11 12 13 16 17 23 24 25 26 28 33 37 49 72 76 77 78
01/07/2009,04 07 11 19 24 27 28 30 31 37 38 48 50 55 57 59 63 69 76 78
01/06/2009,03 04 10 19 20 24 28 30 42 44 46 49 52 59 64 65 66 67 68 74
01/05/2009,04 05 08 09 10 15 16 23 24 26 27 29 47 48 54 57 60 68 70 80
01/04/2009,03 08 11 17 19 23 27 35 44 45 48 56 57 60 64 72 73 74 75 78
01/03/2009,04 11 12 21 22 24 25 26 28 32 34 52 54 55 56 57 62 69 71 74
01/02/2009,02 14 21 24 25 27 28 29 32 37 38 39 50 54 57 58 62 75 77 80
01/01/2009,08 10 12 13 19 22 34 35 37 50 52 59 61 63 66 75 76 77 79 80
12/31/2008,01 07 12 13 14 20 26 32 34 42 44 45 50 59 64 65 70 76 77 79
12/30/2008,05 06 10 14 20 22 25 27 28 29 32 40 48 53 54 59 60 64 68 69
12/29/2008,01 02 06 12 13 23 28 29 33 34 35 37 40 42 55 56 61 64 68 70
12/28/2008,03 05 06 15 19 21 25 26 32 35 48 51 52 56 58 59 63 64 70 71
12/27/2008,04 05 07 11 12 17 19 23 25 31 37 38 39 41 44 60 64 73 74 78
12/26/2008,06 08 11 17 18 23 25 33 34 35 36 45 47 48 49 52 54 73 74 76
12/24/2008,02 06 07 11 17 20 26 28 30 31 38 45 49 57 68 70 72 75 76 78
12/23/2008,03 07 08 16 18 23 28 29 30 32 39 44 48 49 55 67 69 73 77 80
12/22/2008,01 02 05 13 17 25 26 31 33 42 46 49 56 58 63 65 68 74 76 78
12/21/2008,04 05 06 11 18 20 21 22 23 31 32 33 37 38 39 41 52 60 64 72
12/20/2008,03 10 16 17 21 22 25 29 36 38 40 45 47 48 49 51 52 54 67 72
12/19/2008,04 10 11 12 20 22 24 25 32 35 40 41 58 59 60 65 67 70 78 79
12/18/2008,11 16 21 26 28 34 37 42 43 47 52 54 55 62 64 65 66 68 73 74
12/17/2008,05 13 16 17 18 21 22 27 28 33 34 36 47 48 57 63 64 72 76 79
12/16/2008,01 05 08 09 11 13 14 17 19 21 22 27 30 39 43 47 52 65 77 79
12/15/2008,07 16 19 21 23 34 35 50 52 55 57 59 61 62 63 64 65 72 76 80
12/14/2008,02 05 08 10 14 24 28 35 36 43 50 51 54 58 60 67 69 72 74 78
12/13/2008,02 10 12 14 26 29 33 35 37 46 47 49 52 55 56 63 66 71 72 76
```

```
97      12/12/2008,02 04 07 12 16 17 32 37 40 45 48 54 55 56 57 66 75 76 79 80
98      12/11/2008,03 14 16 17 21 22 23 31 34 40 49 54 55 59 62 67 73 74 75 77
99      12/10/2008,08 09 10 13 16 18 21 25 28 41 48 56 59 62 65 66 70 72 78 80
00      12/09/2008,08 09 10 15 21 23 27 28 30 34 38 42 43 53 62 63 65 70 72 77
01      12/08/2008,03 07 09 12 24 26 27 36 37 41 42 43 48 50 52 57 58 61 67 71
02      12/07/2008,03 06 07 20 28 34 39 42 43 45 47 48 51 52 57 58 59 61 62 77
03      12/06/2008,04 15 18 20 30 40 42 43 44 45 47 51 53 55 56 57 65 72 73 75
04      12/05/2008,02 04 06 07 08 10 11 22 24 26 34 35 36 40 52 53 59 67 68 78
05      12/04/2008,01 02 03 07 13 14 17 25 26 30 35 37 40 49 55 68 71 72 75 77
06      12/03/2008,01 10 31 34 35 37 38 39 42 43 44 45 47 49 54 56 66 69 72 73
07      12/02/2008,05 06 10 19 21 23 30 34 45 46 47 49 51 54 56 59 66 73 74 77
08      12/01/2008,02 06 07 15 20 21 23 25 26 33 54 60 61 62 63 64 70 71 72 75
09      11/30/2008,03 04 07 12 15 16 20 24 26 28 35 39 44 47 51 54 58 61 63 79
10      11/29/2008,07 13 24 29 33 34 36 37 46 48 49 53 57 59 62 63 69 73 77 80
11      11/28/2008,02 03 04 06 14 15 21 25 26 39 41 44 51 53 59 68 70 71 73 79
12      11/27/2008,05 06 08 12 18 20 22 24 26 28 31 33 39 40 41 53 60 64 76 78
13      11/26/2008,01 02 05 07 11 14 18 19 22 26 34 35 38 41 51 65 72 73 79 80
14      11/25/2008,02 04 11 13 15 16 20 23 24 29 31 42 46 54 58 66 67 72 74 79
15      11/24/2008,03 08 13 19 22 23 26 29 30 33 36 40 44 46 55 64 70 73 77 80
16      11/23/2008,01 02 04 06 13 17 21 24 29 30 33 36 42 43 45 47 55 56 58 61
17      11/22/2008,04 14 22 26 29 31 32 33 35 37 40 47 49 50 59 63 64 69 75 77
18      11/21/2008,12 13 17 21 23 29 32 45 47 48 57 63 65 67 69 70 71 74 76 77
19      11/20/2008,02 05 08 23 24 26 28 37 41 45 47 49 51 52 54 55 66 70 77 80
20      11/19/2008,05 06 16 17 19 20 21 31 34 36 43 45 48 49 54 56 59 70 76 80
21      11/18/2008,03 07 08 10 11 12 14 15 20 29 34 43 44 45 57 60 62 73 76 78
22      11/17/2008,02 04 06 08 13 14 15 21 33 38 39 45 48 50 51 61 65 67 68 75
23      11/16/2008,04 09 16 17 23 33 44 47 49 54 58 59 60 62 65 66 69 71 77 80
24      11/15/2008,05 08 09 20 23 25 26 27 28 33 46 50 53 55 56 57 60 61 65 66
25      11/14/2008,07 08 11 13 18 20 26 29 34 35 45 53 54 55 57 62 65 67 70 79
26      11/13/2008,02 03 06 07 09 11 21 22 23 26 30 32 37 40 43 48 49 70 71 78
27      11/12/2008,01 04 05 08 20 23 30 31 35 40 44 46 51 52 55 56 64 68 73 78
28      11/11/2008,03 09 10 14 23 24 26 27 31 34 45 50 51 52 62 63 64 66 75 76
29      11/10/2008,01 03 05 06 11 17 22 23 26 32 50 51 53 54 55 56 63 69 72 80
30      11/09/2008,02 06 09 10 13 15 19 20 27 34 41 43 48 52 55 59 60 65 69 79
31      11/08/2008,05 12 14 17 18 23 26 30 36 40 47 49 55 57 61 63 70 72 74 79
32      11/07/2008,03 10 12 14 20 24 26 27 30 32 37 38 39 41 42 44 55 72 79 80
33      11/06/2008,02 04 06 14 15 16 20 34 37 39 46 52 55 56 57 60 67 69 71 76
34      11/05/2008,02 04 07 09 12 24 28 32 36 42 47 48 55 57 58 59 66 69 74 79
35      11/04/2008,06 12 16 17 22 29 32 33 38 45 55 56 63 67 70 71 73 74 76 78
36      11/03/2008,05 06 09 12 14 20 22 23 26 28 37 51 57 59 60 66 73 77 78 80
37      11/02/2008,01 03 09 11 12 19 20 24 26 29 41 42 50 52 55 58 59 64 69 80
38      11/01/2008,11 17 25 33 36 38 39 46 53 55 59 61 64 65 66 69 71 74 75 76
39      10/31/2008,01 03 04 05 21 22 25 32 33 38 43 50 52 57 60 61 69 73 74 80
40      10/30/2008,04 07 11 13 16 22 30 36 37 43 50 51 52 54 58 59 66 68 72 80
41      10/29/2008,07 12 14 18 19 21 23 28 34 36 49 50 53 55 58 65 66 69 77 80
42      10/28/2008,01 03 06 08 18 19 25 26 27 28 35 36 41 52 54 58 59 69 70 71
43      10/27/2008,01 04 06 10 22 30 32 33 34 39 42 44 45 47 49 66 75 77 79 80
44      10/26/2008,03 09 20 24 25 27 28 31 32 35 39 52 57 58 63 66 70 73 75 78
45      10/25/2008,11 15 21 24 28 29 31 34 38 39 42 43 55 58 61 64 74 75 79 80
46      10/24/2008,01 02 07 12 13 15 19 21 30 32 40 47 52 55 56 59 62 63 73 79
47      10/23/2008,07 09 14 15 21 23 40 42 52 53 54 57 58 60 62 67 70 71 75 76
48      10/22/2008,02 07 09 11 13 14 27 29 42 44 45 53 55 59 60 65 73 76 78 80
49      10/21/2008,04 05 07 14 21 22 29 33 37 38 39 40 43 51 53 54 56 59 70 75
50      10/20/2008,10 11 13 17 18 19 22 25 34 40 41 48 52 53 59 66 69 75 76 77
```

```
51      10/19/2008,01 06 10 14 22 24 26 32 33 42 43 45 47 57 58 63 73 74 75 78
52      10/18/2008,02 03 04 06 11 20 31 33 38 46 50 52 53 61 63 66 68 69 71 76
53      10/17/2008,01 07 16 18 20 22 25 30 33 34 38 40 43 49 52 59 68 69 74 75
54      10/16/2008,02 06 09 14 15 16 24 27 28 29 31 37 40 47 48 50 53 62 68 73
55      10/15/2008,03 10 14 27 29 30 34 37 43 45 52 59 61 67 68 69 71 72 73 78
56      10/14/2008,05 08 14 19 23 25 27 28 29 32 34 44 45 51 56 60 61 67 73 79
57      10/13/2008,03 04 07 09 10 22 29 30 42 46 49 54 56 59 62 67 69 76 79 80
58      10/12/2008,03 12 13 16 23 25 26 31 33 36 39 54 58 60 61 63 68 73 75 76
59      10/11/2008,03 05 06 07 08 09 11 13 30 37 40 41 47 55 58 67 71 75 77 80
60      10/10/2008,02 05 19 21 22 29 39 40 44 50 57 60 64 67 69 70 75 77 79 80
61      10/09/2008,01 06 11 27 29 31 34 36 40 42 43 46 51 57 60 63 64 65 70 75
62      10/08/2008,01 03 05 10 19 25 28 30 33 36 46 48 50 53 55 56 61 64 74 78
63      10/07/2008,02 04 06 15 24 33 34 36 37 39 40 45 46 48 58 59 62 64 78 79
64      10/06/2008,06 07 09 10 15 16 18 33 34 36 39 42 45 49 51 62 63 69 70 79
65      10/05/2008,03 04 09 12 14 15 16 17 36 39 40 43 45 47 54 55 57 60 66 77
66      10/04/2008,02 07 08 17 18 20 23 24 25 26 39 47 49 52 54 55 57 63 65 72
67      10/03/2008,06 12 13 14 15 17 19 22 29 31 38 39 40 47 51 55 59 63 67 75
68      10/02/2008,05 08 10 12 16 18 30 34 35 38 40 44 46 55 56 57 58 63 69 75
69      10/01/2008,02 06 08 11 13 18 20 24 28 41 42 43 46 47 60 63 68 74 76 78
70      09/30/2008,07 10 17 29 32 36 44 45 47 54 58 59 60 63 69 71 72 75 78 79
71      09/29/2008,03 10 14 15 16 19 21 26 35 37 51 53 54 55 59 62 64 65 75 80
72      09/28/2008,05 06 13 16 20 25 33 35 42 45 46 47 51 55 58 60 69 73 75 78
73      09/27/2008,03 04 10 11 15 21 31 32 34 36 42 43 45 46 52 55 57 58 59 64
74      09/26/2008,01 03 07 12 17 23 33 35 37 39 44 45 53 55 64 67 71 76 79 80
75      09/25/2008,01 07 08 09 10 13 17 20 25 39 46 47 55 59 62 67 69 70 75 80
76      09/24/2008,01 05 09 10 11 18 24 25 29 36 38 40 45 48 50 61 67 68 76 80
77      09/23/2008,08 14 23 38 44 47 48 51 53 55 57 59 61 62 63 65 66 67 69 80
78      09/22/2008,07 08 09 15 19 26 29 33 48 49 50 51 52 57 58 59 69 70 73 75
79      09/21/2008,09 18 30 33 36 39 41 43 45 47 48 50 56 63 65 66 70 75 76 78
80      09/20/2008,02 13 15 19 21 22 25 26 29 33 41 53 54 56 63 64 66 71 78 80
81      09/19/2008,05 13 17 18 19 24 26 34 35 39 45 46 47 56 59 60 66 71 73 76
82      09/18/2008,07 10 12 14 19 21 22 23 28 44 46 52 59 62 64 65 66 70 71 77
83      09/17/2008,02 05 07 17 18 23 29 33 34 36 39 40 42 52 53 56 63 67 71 72
84      09/16/2008,01 03 04 10 12 16 20 29 32 34 40 44 54 56 58 60 64 68 71 79
85      09/15/2008,14 20 24 26 27 30 35 37 40 45 49 50 53 57 60 62 64 69 75 78
86      09/14/2008,05 09 20 22 23 29 33 37 40 43 46 48 49 54 56 57 62 63 64 66
87      09/13/2008,02 03 05 12 14 22 24 25 30 32 35 38 43 45 49 55 59 61 68 70
88      09/12/2008,02 08 09 10 12 19 22 23 25 33 34 35 37 44 50 51 56 60 67 69
89      09/11/2008,01 05 08 13 14 18 25 26 35 36 38 53 54 59 65 69 75 77 79 80
90      09/10/2008,03 06 15 16 24 33 42 43 46 57 58 59 66 67 68 70 73 78 79 80
91      09/09/2008,04 12 20 21 22 26 30 31 32 33 35 38 43 46 49 53 64 66 73 80
92      09/08/2008,06 07 09 16 17 19 25 28 30 31 34 35 37 40 43 54 58 61 63 80
93      09/07/2008,03 08 13 16 18 19 24 28 29 31 34 40 41 46 53 65 69 71 73 80
94      09/06/2008,03 09 11 14 15 21 23 24 26 30 38 41 43 47 52 62 63 67 72 76
95      09/05/2008,05 13 24 26 30 34 41 46 48 50 52 59 67 69 74 75 76 77 78 80
96      09/04/2008,07 12 17 19 23 26 37 40 41 44 46 52 53 56 57 58 64 67 70 73
97      09/03/2008,03 05 10 11 14 17 23 24 28 29 42 46 53 56 61 64 66 69 72 73
98      09/02/2008,05 08 16 22 25 26 28 32 37 46 47 49 50 51 53 56 59 61 69 74
99      09/01/2008,06 07 12 13 19 20 30 32 44 49 51 52 54 60 61 69 76 77 78 80
00      08/31/2008,04 20 21 23 24 30 31 40 42 43 44 45 48 53 69 70 74 76 78 79
01      08/30/2008,11 12 13 14 16 17 18 19 23 32 34 38 48 49 52 58 63 69 75 78
02      08/29/2008,05 06 08 09 10 14 16 32 33 34 38 47 50 62 66 67 70 72 73 80
03      08/28/2008,03 05 18 19 21 29 31 33 34 38 45 48 51 52 57 59 61 62 71 72
04      08/27/2008,01 02 05 10 12 15 18 34 38 45 47 49 51 56 60 61 69 72 73 80
```

05 08/26/2008,01 06 07 10 19 28 33 34 38 45 47 52 54 56 64 66 67 68 72 78
06 08/25/2008,03 04 05 11 17 18 23 30 35 36 41 43 46 51 54 68 74 77 79 80
07 08/24/2008,05 06 09 12 13 15 23 24 30 35 36 37 44 46 55 56 58 60 66 72
08 08/23/2008,02 05 08 09 22 25 28 30 38 39 41 45 48 58 59 64 68 73 75 77
09 08/22/2008,12 13 15 17 18 19 27 31 32 34 45 47 53 60 62 66 69 72 75 80
10 08/21/2008,08 13 15 17 27 30 36 39 46 48 49 51 54 60 61 65 69 70 73 76
11 08/20/2008,03 06 31 39 41 49 52 54 55 56 57 59 66 67 71 72 74 75 77 80
12 08/19/2008,02 06 09 13 15 16 18 21 32 37 44 50 51 52 57 58 65 67 74 80
13 08/18/2008,02 10 18 26 27 29 31 33 38 40 42 45 48 49 52 59 65 69 70 74
14 08/17/2008,04 06 09 10 18 20 21 22 27 28 35 38 40 45 46 57 58 60 61 65
15 08/16/2008,17 20 21 22 26 27 28 33 37 48 50 51 55 59 61 67 70 71 74 78
16 08/15/2008,08 14 15 19 21 26 27 28 31 36 43 46 48 49 51 53 65 69 76 77
17 08/14/2008,03 04 06 07 10 13 14 15 19 33 34 37 38 40 42 59 67 69 79 80
18 08/13/2008,01 03 05 07 09 18 22 23 25 30 34 40 50 56 57 58 65 68 74 76
19 08/12/2008,01 14 19 20 22 25 28 29 30 35 36 42 48 51 53 54 61 72 79 80
20 08/11/2008,01 09 11 18 19 20 21 24 27 28 33 44 49 51 58 59 65 66 69 73
21 08/10/2008,03 04 05 06 11 17 18 20 26 27 34 36 38 45 48 49 59 67 69 76
22 08/09/2008,05 08 09 12 13 15 17 20 21 22 23 38 44 51 57 60 66 67 72 79
23 08/08/2008,01 03 05 11 15 18 26 29 38 41 50 54 57 60 63 67 69 71 78 79
24 08/07/2008,05 12 18 21 25 30 32 34 38 41 45 50 55 57 60 63 71 76 79 80
25 08/06/2008,04 05 11 12 17 18 22 24 28 30 35 36 37 40 42 44 53 56 74 79
26 08/05/2008,01 06 09 15 16 38 42 44 45 47 49 50 54 61 71 73 74 75 76 80
27 08/04/2008,02 03 11 24 25 26 27 33 37 38 40 44 45 47 59 62 64 66 78 79
28 08/03/2008,07 13 14 19 20 24 27 37 39 46 49 56 59 62 70 71 75 76 77 79
29 08/02/2008,03 05 08 16 25 28 31 40 41 45 48 57 58 62 68 70 72 73 74 80
30 08/01/2008,01 04 19 22 25 30 31 34 39 44 49 51 59 66 67 68 70 76 77 78
31 07/31/2008,03 05 10 21 22 26 30 36 39 46 47 51 54 59 63 64 66 67 73 76
32 07/30/2008,05 09 10 15 18 23 26 33 42 44 46 51 54 57 64 67 73 78 79 80
33 07/29/2008,01 07 15 19 26 27 32 33 36 37 45 46 52 55 58 67 72 75 77 80
34 07/28/2008,01 02 05 06 09 10 23 24 29 36 38 42 53 56 57 58 60 61 65 70
35 07/27/2008,03 11 16 17 19 30 31 33 35 36 37 40 41 42 47 59 61 64 77 78
36 07/26/2008,06 12 14 16 19 23 27 31 32 42 43 49 50 52 54 58 62 63 72 76
37 07/25/2008,02 07 09 24 25 26 28 30 31 34 40 44 52 56 57 59 64 74 75 78
38 07/24/2008,01 11 13 19 20 26 28 35 38 45 49 50 54 55 63 67 68 69 72 75
39 07/23/2008,06 07 25 28 32 36 37 38 40 42 46 55 56 57 59 62 64 71 72 79
40 07/22/2008,03 04 05 06 07 08 10 14 17 21 27 30 31 35 39 49 51 57 69 74
41 07/21/2008,05 08 19 21 22 29 32 39 40 43 46 47 53 59 62 69 74 76 78 79
42 07/20/2008,04 08 31 35 36 38 39 40 41 42 43 44 45 46 49 50 54 60 72 80
43 07/19/2008,05 06 08 09 10 19 20 26 28 31 35 42 45 49 50 56 58 66 71 77
44 07/18/2008,02 08 12 20 22 27 29 35 38 39 49 51 54 58 62 66 72 76 78 79
45 07/17/2008,01 04 07 10 13 15 21 30 31 34 54 57 60 62 63 64 66 68 70 79
46 07/16/2008,01 05 12 14 15 25 26 31 36 39 40 54 60 62 66 68 70 71 72 80
47 07/15/2008,03 04 07 08 10 19 24 28 29 30 37 40 44 51 55 56 57 59 65 74
48 07/14/2008,04 08 09 15 17 19 21 30 38 42 48 52 56 57 61 63 68 71 77 80
49 07/13/2008,01 06 10 12 15 16 17 24 26 31 42 53 54 63 65 66 69 74 76 80
50 07/12/2008,09 11 13 14 18 19 21 23 33 34 36 39 40 46 48 58 59 68 79 80
51 07/11/2008,02 03 10 12 24 25 33 35 38 41 47 50 54 57 58 70 71 72 78 79
52 07/10/2008,03 07 10 12 13 14 27 30 37 38 39 40 45 48 51 52 62 68 72 76
53 07/09/2008,02 05 10 12 16 22 30 31 35 38 49 51 52 54 60 65 66 76 77 80
54 07/08/2008,01 05 07 09 15 20 23 25 26 27 29 32 34 39 41 50 51 52 54 71
55 07/07/2008,04 10 20 21 23 24 29 36 38 39 47 49 50 52 55 62 74 76 77 78
56 07/06/2008,09 12 25 26 32 34 38 39 45 47 51 57 62 63 66 67 71 72 76 79
57 07/05/2008,07 15 17 19 23 24 30 46 48 53 54 55 56 59 60 67 69 72 75 77
58 07/04/2008,01 02 17 21 22 30 32 36 37 38 44 48 57 62 63 64 70 75 76 79

```
07/03/2008,05 11 14 24 25 31 34 39 43 47 50 52 54 64 66 72 73 75 77 79
07/02/2008,05 06 07 09 10 11 16 18 22 27 30 34 45 52 56 58 64 72 76 77
07/01/2008,02 05 08 11 14 25 27 30 35 37 38 40 47 48 57 58 63 64 69 73
06/30/2008,08 09 21 24 25 26 29 33 34 44 47 49 59 63 64 65 68 73 74 75
06/29/2008,03 07 08 09 11 12 14 16 24 29 33 35 40 43 50 53 56 60 69 77
06/28/2008,08 17 18 19 20 23 25 27 28 31 32 35 39 40 48 57 61 64 65 72
06/27/2008,03 07 10 14 17 18 21 33 34 41 46 47 53 60 63 64 69 76 79 80
06/26/2008,03 05 08 15 17 21 22 30 38 43 46 47 49 51 54 57 59 69 72 74
06/25/2008,01 03 11 12 16 17 26 32 38 41 45 47 52 53 57 59 63 64 74 76
06/24/2008,05 07 13 14 17 20 25 28 36 39 42 51 65 67 68 71 72 73 78 79
06/23/2008,06 10 14 15 18 24 26 28 30 33 35 36 42 46 49 58 60 63 64 76
06/22/2008,02 04 06 10 14 15 16 18 19 20 21 23 30 37 45 51 58 59 79 80
06/21/2008,01 06 08 10 17 21 22 24 27 30 40 42 49 53 62 63 67 68 71 77
06/20/2008,02 03 04 05 06 07 08 15 20 21 27 42 49 51 52 56 68 70 72 79
06/19/2008,07 14 23 27 32 33 35 39 40 41 42 48 51 53 54 55 64 70 76 79
06/18/2008,04 09 21 23 30 37 39 44 45 46 47 51 52 55 57 68 69 75 79 80
06/17/2008,05 07 09 10 11 15 17 18 19 21 25 27 33 45 47 53 55 64 73 75
06/16/2008,01 03 05 07 18 21 25 37 41 42 44 50 58 59 60 62 63 64 68 79
06/15/2008,03 06 10 18 21 22 29 30 42 43 44 46 51 54 58 70 73 74 76 79
06/14/2008,01 02 14 16 21 22 23 42 45 46 52 55 59 68 69 70 75 76 77 78
06/13/2008,02 05 07 13 18 24 33 34 39 44 56 58 59 62 64 65 69 75 76 79
06/12/2008,04 05 09 10 15 17 23 28 29 37 38 49 53 57 61 67 69 74 75 79
06/11/2008,01 12 14 17 19 24 25 30 36 38 39 51 56 60 62 69 75 76 78 80
06/10/2008,01 02 03 12 15 21 26 28 34 38 40 41 51 52 61 71 72 73 74 79
06/09/2008,06 07 11 14 21 23 24 25 26 35 37 39 40 42 58 62 64 66 74 78
06/08/2008,06 10 24 28 29 30 37 46 47 52 54 55 56 58 60 65 66 71 76 80
06/07/2008,01 02 05 07 17 23 26 27 29 30 37 39 43 44 53 54 59 60 61 65
06/06/2008,03 07 10 12 13 15 16 29 34 39 42 45 50 53 55 59 62 64 70 79
06/05/2008,02 03 11 15 17 30 34 35 37 50 54 57 58 63 66 67 68 74 75 78
06/04/2008,07 16 19 25 42 44 47 48 49 50 55 56 59 66 69 70 72 76 78 80
06/03/2008,03 16 17 20 21 22 24 26 30 37 41 46 48 55 57 63 65 71 73 78
06/02/2008,04 05 09 10 16 21 26 29 31 34 40 41 42 43 44 52 54 65 74 80
06/01/2008,06 09 13 23 27 31 32 34 35 38 40 49 50 59 60 64 68 74 76 77
05/31/2008,01 04 08 09 13 20 32 33 36 38 40 41 48 50 51 56 58 61 72 73
05/30/2008,02 06 16 25 32 34 35 37 38 40 50 51 56 63 65 66 71 75 76 78
05/29/2008,03 15 21 22 30 31 35 43 44 46 51 53 54 56 68 70 74 75 76 77
05/28/2008,01 05 08 11 12 13 15 20 25 26 27 31 44 46 48 61 62 68 69 79
05/27/2008,01 03 06 07 10 16 17 20 21 22 30 31 39 42 44 48 50 68 69 70
05/26/2008,01 08 12 18 19 25 27 31 41 46 50 52 53 56 57 62 73 74 76 77
05/25/2008,01 02 03 10 12 15 20 28 30 38 39 44 45 47 56 60 71 74 77 80
05/24/2008,02 07 19 22 25 26 30 35 45 50 56 57 58 61 64 66 67 74 75 78
05/23/2008,03 06 08 09 13 14 17 20 22 32 38 42 45 50 57 58 61 67 68 71
05/22/2008,02 03 08 27 33 38 40 43 49 50 52 54 57 64 65 67 72 77 78 80
05/21/2008,03 11 18 21 22 27 28 29 34 41 43 50 56 59 60 61 63 65 78 79
05/20/2008,01 10 11 15 16 17 18 29 33 38 41 45 52 56 60 62 65 70 72 74
05/19/2008,04 05 06 10 14 16 18 22 23 24 25 29 38 39 41 45 47 52 64 67
05/18/2008,05 06 12 17 21 28 32 40 43 44 48 52 55 56 70 72 73 75 78 80
05/17/2008,02 08 14 16 17 24 27 30 32 40 52 58 62 63 64 65 66 68 70 80
05/16/2008,03 04 14 29 31 33 35 38 41 48 50 51 56 61 65 68 69 72 78 79
05/15/2008,05 11 14 17 20 21 24 25 26 36 40 41 48 60 64 67 70 72 75 79
05/14/2008,05 13 18 24 28 33 35 38 40 44 47 50 56 63 67 73 74 76 77 78
05/13/2008,03 20 23 24 25 31 35 38 39 44 55 57 58 60 67 68 72 78 79 80
05/12/2008,01 03 10 14 32 33 35 37 38 40 41 42 47 50 57 58 59 66 68 72
05/11/2008,01 05 08 09 15 20 37 40 42 43 44 46 50 51 56 60 68 72 76 78
```

```
13      05/10/2008,02 03 07 12 13 14 21 27 32 40 44 45 47 55 56 59 62 64 67 79
14      05/09/2008,05 08 11 14 17 20 24 31 35 36 38 48 54 55 57 59 67 72 74 79
15      05/08/2008,03 09 11 20 25 29 30 33 34 43 45 46 49 52 61 65 68 77 79 80
16      05/07/2008,01 03 08 09 20 23 25 27 28 31 40 42 46 48 49 50 51 67 70 73
17      05/06/2008,02 09 12 18 20 29 32 39 41 42 43 44 46 47 49 52 59 60 65 78
18      05/05/2008,06 15 19 21 22 29 30 35 36 41 46 48 53 56 58 61 65 69 71 74
19      05/04/2008,01 08 12 14 15 17 19 26 29 30 37 39 43 51 52 55 62 72 73 80
20      05/03/2008,07 10 11 14 15 16 18 25 32 34 35 40 41 42 52 56 60 67 68 69
21      05/02/2008,04 05 09 13 18 19 22 23 24 28 29 38 41 45 50 57 67 68 73 74
22      05/01/2008,01 06 08 14 16 17 21 28 37 38 39 44 46 48 54 55 63 64 68 75
23      04/30/2008,09 11 12 15 18 22 23 25 26 30 35 36 44 47 49 56 59 67 70 74
24      04/29/2008,03 04 05 17 23 31 32 34 35 36 48 49 52 53 55 57 58 64 67 74
25      04/28/2008,01 06 11 14 15 18 21 22 25 31 32 33 36 44 47 60 62 63 67 72
26      04/27/2008,03 05 06 07 18 22 25 36 41 51 57 59 62 64 67 70 71 72 73 80
27      04/26/2008,04 09 20 26 30 33 40 44 46 48 50 58 59 62 64 68 70 73 75 79
28      04/25/2008,01 05 09 14 17 21 23 25 26 31 34 36 39 41 46 57 61 66 77 79
29      04/24/2008,02 05 07 08 20 21 22 26 31 33 38 39 42 45 49 54 57 62 66 77
30      04/23/2008,02 03 04 08 18 20 22 23 28 32 41 46 49 56 57 60 73 74 76 80
31      04/22/2008,02 03 14 15 16 25 27 28 31 32 40 42 46 48 53 54 61 69 70 71
32      04/21/2008,03 04 11 12 13 17 19 22 31 33 34 43 44 49 51 52 53 54 62 76
33      04/20/2008,02 04 07 15 16 18 21 22 23 26 28 36 42 49 50 55 65 76 78 80
34      04/19/2008,02 12 15 16 27 41 46 49 57 60 61 63 66 67 68 69 71 72 77 78
35      04/18/2008,09 10 13 14 23 28 31 33 36 43 49 51 52 55 58 59 61 63 66 80
36      04/17/2008,09 10 11 12 15 19 23 26 34 35 36 39 41 42 56 62 67 68 76 80
37      04/16/2008,03 04 10 14 19 23 27 31 33 37 38 40 45 46 47 48 57 58 63 73
38      04/15/2008,04 05 08 09 13 14 17 25 26 34 37 39 42 44 47 51 52 64 75 80
39      04/14/2008,05 06 11 21 30 37 39 41 46 48 51 59 63 66 67 73 74 75 78 79
40      04/13/2008,01 06 10 11 12 18 19 27 28 29 31 33 43 49 61 62 69 70 74 79
41      04/12/2008,08 15 17 18 22 24 28 30 32 36 46 51 52 55 64 70 72 74 76 79
42      04/11/2008,14 17 18 23 25 27 29 30 37 40 45 49 57 58 64 67 69 72 77 78
43      04/10/2008,02 04 08 11 21 22 35 43 45 48 49 53 60 66 69 70 71 75 78 79
44      04/09/2008,05 10 13 14 18 19 21 29 34 43 46 50 53 54 58 61 63 67 71 74
45      04/08/2008,01 02 09 10 14 16 18 20 25 26 29 31 32 43 51 60 61 64 70 79
46      04/07/2008,01 02 08 11 14 16 18 19 39 44 45 49 51 53 57 59 71 75 76 78
47      04/06/2008,06 10 17 18 23 29 31 40 42 44 45 49 50 59 63 65 67 68 70 77
48      04/05/2008,08 09 10 12 23 24 25 31 32 48 49 54 60 61 69 70 71 72 78 79
49      04/04/2008,06 10 14 18 24 27 31 35 38 40 43 46 49 53 55 58 70 71 74 79
50      04/03/2008,02 06 10 13 15 21 22 23 24 30 33 44 48 51 56 58 63 72 77 78
51      04/02/2008,03 07 20 21 23 25 31 41 42 43 48 53 56 59 60 65 70 73 77 79
52      04/01/2008,01 03 05 06 10 17 26 30 37 41 45 47 52 57 59 60 65 70 71 80
53      03/31/2008,01 02 07 09 17 18 23 33 38 40 41 43 44 52 58 60 61 66 67 78
54      03/30/2008,05 10 11 15 20 22 24 25 28 36 39 43 46 49 50 53 65 66 67 77
55      03/29/2008,01 04 15 16 29 33 37 42 44 50 53 54 58 59 63 64 66 67 70 78
56      03/28/2008,03 05 09 14 16 21 23 27 29 31 32 38 53 56 61 62 69 70 71 78
57      03/27/2008,01 06 07 22 23 24 26 29 30 33 34 37 47 50 51 54 60 62 76 77
58      03/26/2008,04 09 13 14 15 17 22 24 27 29 31 34 37 42 46 51 56 65 72 75
59      03/25/2008,04 07 08 10 14 16 19 20 21 31 37 41 49 58 59 61 65 66 72 78
60      03/24/2008,03 08 13 16 17 20 29 30 31 33 48 58 60 61 65 66 68 71 74 80
61      03/23/2008,05 07 12 14 20 24 28 31 36 39 50 51 52 56 59 61 69 74 75 78
62      03/22/2008,01 04 07 17 21 26 32 33 34 39 41 44 45 46 50 55 59 63 70 80
63      03/21/2008,09 15 17 19 20 27 33 34 39 41 46 51 52 53 62 63 65 66 75 77
64      03/20/2008,04 05 07 10 13 16 17 41 42 43 49 51 55 57 58 60 65 75 78 79
65      03/19/2008,02 07 08 10 20 21 25 26 32 34 35 36 37 40 41 51 53 54 60 76
66      03/18/2008,02 03 07 12 13 14 15 17 18 24 34 38 42 45 48 51 54 61 62 69
```

```
03/17/2008,01 02 04 12 16 30 32 36 37 42 45 52 59 61 65 71 73 76 77 80
03/16/2008,09 10 13 18 19 23 24 30 32 34 37 41 44 56 61 67 70 72 76 77
03/15/2008,05 08 09 15 27 30 31 32 46 47 49 53 56 61 63 66 70 75 77 80
03/14/2008,01 05 16 18 20 21 23 29 30 33 36 39 40 41 49 50 59 74 76 79
03/13/2008,05 07 08 09 11 17 18 21 23 44 47 51 55 60 61 69 71 72 74 79
03/12/2008,02 03 05 06 08 10 11 15 21 24 27 31 38 52 58 59 61 63 68 74
03/11/2008,11 12 18 20 30 32 33 34 35 38 41 42 48 52 53 54 58 61 62 71
03/10/2008,02 05 09 10 13 17 20 27 30 31 34 40 49 54 58 63 68 71 73 79
03/09/2008,02 09 16 20 26 27 28 29 32 37 41 47 55 59 61 62 66 69 71 79
03/08/2008,05 07 09 19 20 21 23 25 26 30 34 40 42 53 54 57 58 64 70 75
03/07/2008,06 09 13 15 17 25 29 30 31 40 41 44 46 48 53 54 57 60 74 77
03/06/2008,03 06 23 28 32 36 37 38 39 44 46 53 54 55 58 62 68 72 78 80
03/05/2008,01 02 07 13 18 21 25 26 29 31 32 35 45 52 55 58 60 65 68 71
03/04/2008,04 05 06 09 15 17 22 29 30 31 42 44 46 49 53 57 59 64 70 75
03/03/2008,01 07 09 10 19 20 25 26 29 31 41 44 53 57 58 60 63 68 70 76
03/02/2008,15 22 24 29 30 33 34 36 40 41 56 57 58 61 66 68 74 76 78 79
03/01/2008,05 07 11 17 23 25 34 36 44 49 50 54 60 61 62 63 72 73 76 79
02/29/2008,08 10 14 17 22 25 28 33 38 42 47 53 54 57 61 67 69 72 78 79
02/28/2008,05 10 14 15 16 19 20 26 34 38 39 43 44 53 56 58 61 65 69 71
02/27/2008,01 08 10 18 25 26 31 44 49 51 56 57 58 59 61 63 64 68 77 78
02/26/2008,13 24 27 28 30 40 42 43 44 50 51 52 56 63 64 66 67 72 76 78
02/25/2008,13 14 19 21 23 26 30 37 44 45 49 50 51 57 58 61 64 74 75 80
02/24/2008,03 07 09 10 12 13 16 22 39 40 52 59 60 61 62 66 69 72 74 75
02/23/2008,02 15 16 17 19 20 28 30 39 44 46 49 52 53 59 61 64 66 73 80
02/22/2008,02 05 11 15 16 18 19 25 29 31 37 39 44 48 49 54 58 59 68 70
02/21/2008,10 15 17 18 24 28 29 37 39 40 47 48 49 51 60 63 64 65 70 74
02/20/2008,02 10 26 28 30 31 33 34 40 45 49 51 57 61 63 70 72 73 75 79
02/19/2008,01 04 07 08 10 11 19 39 48 49 51 54 56 60 61 63 65 66 75 78
02/18/2008,02 05 14 22 27 28 29 31 32 43 44 45 47 48 53 55 57 70 72 76
02/17/2008,04 05 06 07 08 15 16 18 20 30 36 46 48 56 59 64 71 72 75 76
02/16/2008,03 05 07 10 21 22 25 29 30 33 36 37 40 44 45 54 59 66 70 74
02/15/2008,07 12 22 25 27 29 30 33 35 37 38 41 43 45 48 55 60 73 79 80
02/14/2008,01 05 06 08 10 12 18 22 25 35 36 39 41 43 47 69 73 77 78 80
02/13/2008,02 04 05 07 08 09 18 19 21 25 35 39 41 45 51 58 64 70 71 78
02/12/2008,16 21 22 27 29 31 34 35 47 53 54 58 62 65 68 69 71 72 75 77
02/11/2008,01 02 04 17 18 19 24 33 37 39 44 51 53 63 69 70 71 77 78 79
02/10/2008,01 02 06 08 12 16 22 27 28 32 33 38 42 45 46 58 59 71 72 74
02/09/2008,08 12 14 27 28 29 31 34 39 42 43 49 50 52 56 65 67 71 75 77
02/08/2008,06 07 09 17 18 22 25 28 29 31 45 49 51 54 57 62 63 67 71 78
02/07/2008,06 07 15 17 18 20 23 25 26 28 45 46 49 50 54 57 58 69 74 75
02/06/2008,04 05 09 13 16 18 21 23 30 36 39 40 42 45 51 53 59 62 74 79
02/05/2008,01 07 10 11 14 17 27 31 32 33 35 36 46 50 55 57 63 66 72 75
02/04/2008,01 04 05 09 23 25 26 30 33 40 48 49 50 56 60 66 67 70 75 76
02/03/2008,05 08 11 12 13 17 19 22 31 42 47 48 59 63 65 66 69 71 72 80
02/02/2008,02 10 16 18 20 22 23 27 31 35 36 43 56 62 66 70 73 77 78 80
02/01/2008,08 09 13 23 32 34 35 39 46 47 48 50 52 59 60 63 66 70 77 80
01/31/2008,03 05 15 17 18 21 25 30 31 39 42 46 48 51 52 64 66 67 75 78
01/30/2008,04 12 13 15 20 24 33 35 37 45 53 55 58 61 64 65 74 75 78 79
01/29/2008,02 03 04 05 11 20 27 30 39 40 41 42 48 51 54 57 65 74 77 78
01/28/2008,02 11 13 15 16 23 30 33 40 43 47 48 49 50 52 61 71 73 76 78
01/27/2008,08 18 34 36 37 38 41 43 47 52 53 56 59 65 66 69 72 75 76 77
01/26/2008,03 04 06 16 25 31 34 36 41 47 48 51 54 60 66 67 68 70 71 74
01/25/2008,02 06 14 18 21 30 35 39 40 41 44 48 53 56 60 67 71 72 74 78
01/24/2008,04 07 09 10 17 19 22 30 36 46 47 48 56 64 66 70 72 74 77 79
```

```
01/23/2008,02 04 13 21 24 26 27 29 30 32 42 46 51 53 56 57 64 65 66 69
01/22/2008,01 02 04 06 09 10 17 23 32 33 40 42 54 56 58 62 66 72 74 75
01/21/2008,10 28 31 32 33 35 36 40 41 43 44 46 47 55 59 64 74 75 78 80
01/20/2008,03 04 06 16 19 21 44 46 52 53 60 61 62 65 66 70 73 75 76 78
01/19/2008,04 05 06 09 14 17 18 22 32 33 36 41 42 43 56 60 70 74 76 77
01/18/2008,01 06 07 14 17 21 27 30 33 45 50 53 57 60 65 69 71 74 76 80
01/17/2008,04 06 11 12 15 28 34 35 37 38 42 44 46 49 52 55 62 67 73 79
01/16/2008,06 07 09 10 11 14 16 17 20 36 43 46 47 65 67 69 72 78 79 80
01/15/2008,02 07 08 12 18 20 21 22 27 34 35 39 47 51 52 57 64 67 72 74
01/14/2008,02 13 14 19 28 30 32 36 39 41 42 56 58 61 64 68 71 77 79 80
01/13/2008,02 03 13 22 29 32 33 38 42 48 50 51 54 57 59 60 62 68 72 78
01/12/2008,03 19 24 26 30 32 33 35 36 43 47 49 53 56 59 64 70 74 75 78
01/11/2008,01 04 10 11 14 16 18 19 26 27 43 47 55 57 59 63 74 75 78 79
01/10/2008,11 16 22 23 27 28 33 34 36 46 50 54 55 59 60 62 66 69 72 73
01/09/2008,04 07 13 25 31 32 39 41 44 46 48 49 64 66 67 69 72 73 76 80
01/08/2008,02 04 11 13 14 27 31 32 34 37 46 55 56 58 59 63 73 75 76 79
01/07/2008,02 03 04 18 19 23 40 42 44 48 52 56 61 66 71 74 76 78 79 80
01/06/2008,06 07 08 21 28 31 32 38 41 53 57 61 63 64 66 68 69 70 71 72
01/05/2008,10 12 17 21 23 30 32 35 37 39 41 56 58 61 65 72 73 74 77 80
01/04/2008,01 05 07 21 23 25 26 38 39 46 49 51 52 54 57 64 65 69 72 73
01/03/2008,06 08 12 13 14 21 22 30 33 35 39 49 52 55 63 64 66 70 78 80
01/02/2008,09 13 17 20 21 22 25 33 38 42 48 49 50 61 68 70 72 73 74 77
01/01/2008,02 07 08 10 11 21 22 32 39 52 53 55 56 63 64 66 68 69 71 80
12/31/2007,02 04 12 13 16 19 21 27 28 31 40 42 54 55 56 62 70 71 72 77
12/30/2007,07 14 18 19 29 30 32 33 35 41 43 45 46 53 55 67 72 75 76 77
12/29/2007,07 08 20 21 23 25 31 43 45 54 56 57 58 61 63 65 68 69 78 79
12/28/2007,06 09 13 14 15 21 23 27 30 31 33 41 42 46 48 51 60 64 73 74
12/27/2007,06 08 11 19 25 26 28 31 36 43 48 50 52 56 61 63 70 74 75 80
12/26/2007,03 06 10 12 16 19 20 23 27 38 40 54 61 63 65 68 71 77 78 80
12/24/2007,02 03 05 08 20 26 27 39 40 41 52 54 57 59 64 67 69 73 76 79
12/23/2007,10 22 25 26 31 32 33 35 36 43 48 51 54 57 59 62 65 68 71 78
12/22/2007,02 04 05 09 14 21 25 27 30 35 38 41 47 54 57 63 65 67 70 72
12/21/2007,05 07 16 21 25 27 39 41 43 45 46 47 53 55 64 72 74 75 76 78
12/20/2007,05 06 07 11 12 14 20 21 27 50 53 56 60 61 65 68 70 76 77 80
12/19/2007,10 18 20 26 34 38 40 46 48 50 51 55 59 65 66 67 69 72 73 78
12/18/2007,02 03 04 05 08 11 13 14 15 20 25 42 47 51 52 58 60 66 67 74
12/17/2007,01 04 06 08 11 12 16 19 21 23 24 29 34 38 42 44 45 49 64 74
12/16/2007,02 13 15 18 24 25 27 33 40 43 46 47 48 53 54 61 62 65 72 76
12/15/2007,03 07 11 15 23 28 32 33 36 41 50 52 56 58 59 62 67 68 70 75
12/14/2007,02 10 15 21 23 24 29 34 35 37 42 50 56 59 60 61 65 68 73 79
12/13/2007,03 06 09 10 16 21 26 39 40 45 46 47 52 53 56 63 67 75 76 79
12/12/2007,02 04 07 11 12 17 27 28 31 39 40 46 49 52 54 63 65 68 74 80
12/11/2007,02 03 09 14 18 34 38 40 43 44 49 52 53 55 56 64 70 72 75 77
12/10/2007,04 06 08 09 12 17 18 21 24 26 31 47 48 59 61 64 68 69 74 78
12/09/2007,01 02 03 07 11 13 17 22 24 27 31 37 47 49 53 55 58 64 71 79
12/08/2007,06 09 11 12 13 15 21 24 26 27 34 39 46 48 51 54 58 61 70 74
12/07/2007,06 07 16 19 20 21 22 27 48 49 55 57 59 62 63 65 70 75 76 77
12/06/2007,01 10 12 15 17 18 19 24 27 31 40 45 46 47 49 57 58 62 66 71
12/05/2007,01 04 13 19 23 32 36 42 48 49 53 61 64 68 70 71 73 75 78 79
12/04/2007,02 06 07 09 12 16 19 24 26 29 31 32 43 44 46 50 51 58 66 75
12/03/2007,02 14 22 23 32 33 41 47 48 52 56 59 65 70 72 74 75 77 78 79
12/02/2007,02 06 09 10 11 14 30 34 45 46 47 50 51 55 56 60 63 65 67 77
12/01/2007,01 14 16 18 21 25 27 29 30 31 33 37 38 39 40 55 64 68 75 79
11/30/2007,01 12 19 27 29 30 33 37 39 41 42 43 46 49 50 53 57 62 71 79
```

```
75      11/29/2007,01 02 05 07 09 13 18 22 23 28 39 40 42 49 58 61 65 68 75 78
76      11/28/2007,02 07 08 20 25 27 32 36 38 39 40 42 43 52 63 64 66 68 74 76
77      11/27/2007,07 13 15 18 24 32 34 35 40 42 47 48 49 50 51 62 70 76 79 80
78      11/26/2007,01 03 04 11 12 14 16 18 24 27 36 37 45 46 48 49 58 67 71 72
79      11/25/2007,02 03 10 15 21 22 24 27 30 31 41 44 45 47 54 55 59 63 66 75
80      11/24/2007,11 12 17 20 28 31 33 34 35 36 37 43 44 58 61 65 67 70 72 73
81      11/23/2007,03 12 14 17 19 25 26 29 32 34 38 40 45 50 53 63 67 68 75 77
82      11/22/2007,02 03 05 09 10 11 15 16 19 20 22 28 33 35 45 48 52 58 74 78
83      11/21/2007,07 14 16 24 25 26 27 32 35 40 46 47 52 53 54 56 60 62 73 80
84      11/20/2007,11 13 14 28 29 33 35 42 47 48 55 58 60 62 64 65 67 69 71 78
85      11/19/2007,06 07 11 14 19 32 33 41 42 46 52 57 58 62 64 66 70 72 74 80
86      11/18/2007,04 06 21 23 26 35 36 38 42 48 50 54 61 63 68 70 71 74 77 79
87      11/17/2007,03 07 11 12 13 15 17 18 23 33 37 42 45 56 60 67 69 76 79 80
88      11/16/2007,01 14 15 20 22 29 31 36 44 46 49 50 51 52 53 60 62 65 79 80
89      11/15/2007,04 18 23 32 34 39 42 44 45 49 51 55 60 61 66 67 73 75 77 80
90      11/14/2007,02 09 13 14 16 17 18 19 24 31 34 49 51 52 58 63 68 70 73 74
91      11/13/2007,06 09 14 20 28 29 32 35 37 40 46 47 49 65 68 69 70 72 76 78
92      11/12/2007,01 02 09 13 15 18 21 26 28 29 36 40 49 53 56 64 65 67 69 72
93      11/11/2007,06 14 16 17 22 27 29 40 42 43 46 49 52 55 57 58 63 66 70 71
94      11/10/2007,01 02 04 07 08 16 22 23 25 28 29 33 42 53 57 64 70 74 77 78
95      11/09/2007,03 13 16 21 22 26 31 37 38 41 42 43 53 55 57 67 68 73 74 77
96      11/08/2007,04 07 09 11 16 19 21 29 30 32 46 48 50 56 59 60 67 68 74 75
97      11/07/2007,02 04 12 13 14 19 20 27 32 43 44 48 51 54 57 58 60 71 74 79
98      11/06/2007,08 11 12 13 15 21 23 25 29 36 47 51 53 55 56 57 63 64 69 74
99      11/05/2007,02 06 14 17 23 24 35 39 41 44 47 57 58 63 66 67 70 71 77 80
00      11/04/2007,02 04 07 23 25 29 36 48 51 56 57 67 68 69 73 75 76 77 78 79
01      11/03/2007,04 09 14 24 25 27 28 29 30 37 38 42 59 62 63 64 68 75 76 77
02      11/02/2007,02 13 18 31 37 38 40 41 43 44 45 47 48 51 52 57 61 65 70 72
03      11/01/2007,08 10 11 14 25 31 36 46 47 49 52 54 58 59 60 62 65 70 71 76
04      10/31/2007,02 03 04 14 15 16 20 22 23 24 26 29 35 44 45 52 55 59 72 79
05      10/30/2007,04 07 09 12 13 16 19 33 35 37 45 55 57 60 61 65 66 67 70 77
06      10/29/2007,05 07 15 24 27 28 34 38 41 42 44 49 52 54 59 64 66 69 74 76
07      10/28/2007,04 05 08 09 15 18 20 25 27 28 30 31 36 47 49 55 63 64 68 77
08      10/27/2007,03 10 13 14 19 22 23 25 29 43 45 48 52 57 60 62 63 70 75 79
09      10/26/2007,01 03 05 06 09 24 25 26 33 36 41 44 48 55 57 67 69 72 73 79
10      10/25/2007,02 07 08 13 23 28 32 35 36 42 44 45 49 55 57 63 66 71 76 80
11      10/24/2007,02 03 04 07 09 14 18 22 36 41 46 55 56 58 60 71 72 73 74 77
12      10/23/2007,03 04 10 16 18 19 20 22 26 32 41 43 49 51 57 58 69 75 76 79
13      10/22/2007,04 09 12 20 21 23 35 38 39 48 49 54 55 56 59 60 62 73 76 79
14      10/21/2007,03 14 15 20 22 24 25 30 31 34 37 38 46 57 62 67 68 72 75 80
15      10/20/2007,04 05 06 17 18 22 25 29 30 42 45 46 52 55 58 61 62 66 69 73
16      10/19/2007,01 05 06 19 30 34 36 39 42 54 55 57 60 62 63 67 69 74 77 79
17      10/18/2007,02 05 06 10 24 33 35 39 42 46 55 56 58 68 69 72 75 76 78 80
18      10/17/2007,01 04 05 13 14 26 27 29 30 42 45 47 49 50 51 52 61 65 70 75
19      10/16/2007,03 05 07 15 16 18 19 27 31 32 36 37 43 50 55 60 70 71 73 79
20      10/15/2007,06 10 22 24 25 32 36 37 39 40 48 51 56 57 61 68 69 70 78 79
21      10/14/2007,06 13 14 21 22 24 29 37 41 43 50 51 54 55 56 57 67 72 77 79
22      10/13/2007,03 04 15 18 20 21 27 28 30 32 40 46 49 55 59 62 66 69 74 76
23      10/12/2007,06 13 14 21 22 27 31 41 43 54 55 59 60 63 67 73 74 76 77 78
24      10/11/2007,02 07 15 18 25 27 30 31 33 38 45 56 62 63 64 70 74 76 78 79
25      10/10/2007,04 11 13 14 15 19 22 23 25 35 37 45 47 51 57 60 70 71 75 79
26      10/09/2007,05 09 12 13 14 21 25 27 28 30 31 50 51 53 56 57 69 71 72 75
27      10/08/2007,01 10 11 12 18 20 27 38 41 45 46 47 48 51 52 58 61 63 68 76
28      10/07/2007,03 05 16 20 25 38 42 44 45 59 60 61 62 64 65 71 73 76 77 78
```

```
10/06/2007,05 11 15 19 20 27 33 39 43 44 45 52 53 55 60 61 64 70 74 75
10/05/2007,03 04 05 10 15 25 26 27 37 39 40 45 49 51 52 54 55 65 68 77
10/04/2007,01 09 15 18 21 22 26 27 28 35 42 45 50 51 60 63 72 77 78 79
10/03/2007,01 05 08 18 21 27 28 30 32 35 45 55 60 64 66 67 71 73 77 80
10/02/2007,07 08 19 21 24 25 29 30 32 33 36 41 45 47 48 54 56 60 63 70
10/01/2007,07 10 17 18 22 30 31 36 37 42 43 48 52 58 62 63 65 70 73 78
09/30/2007,04 12 13 14 19 20 23 38 43 48 51 53 60 66 69 70 71 74 78 79
09/29/2007,03 09 13 14 15 18 23 27 33 34 38 39 41 42 50 53 65 73 74 75
09/28/2007,01 04 05 10 11 13 21 22 26 28 33 53 54 56 57 59 69 72 76 77
09/27/2007,06 14 16 19 20 26 35 39 43 44 45 49 50 55 57 58 64 67 70 71
09/26/2007,10 11 13 17 19 21 24 30 35 44 47 54 57 61 62 65 66 69 70 76
09/25/2007,04 05 10 12 22 24 25 35 36 38 46 47 50 51 53 64 66 69 70 80
09/24/2007,06 07 20 25 28 32 33 34 35 36 37 40 45 46 47 49 56 61 73 74
09/23/2007,03 09 14 17 18 22 24 25 29 31 36 38 49 52 60 67 69 73 78 80
09/22/2007,07 09 10 11 19 22 24 34 39 45 48 54 61 62 66 67 72 75 77 78
09/21/2007,03 06 08 16 18 21 22 24 25 38 40 42 48 51 53 54 63 65 69 80
09/20/2007,03 07 11 12 13 14 15 16 19 20 34 35 38 46 63 69 71 73 79 80
09/19/2007,04 05 07 11 13 23 24 30 37 39 43 45 51 54 65 69 75 78 79 80
09/18/2007,04 12 13 14 17 18 22 30 32 36 42 44 51 52 53 59 67 69 72 75
09/17/2007,03 06 11 12 15 22 28 36 38 40 43 53 56 62 67 71 73 74 75 80
09/16/2007,01 12 13 14 22 24 28 31 36 37 40 46 47 48 56 62 71 73 78 80
09/15/2007,01 02 04 06 20 23 28 29 35 36 37 39 40 50 62 63 67 73 76 77
09/14/2007,07 09 15 16 23 30 33 35 38 40 44 50 56 60 61 64 65 73 74 79
09/13/2007,02 05 06 10 14 23 30 33 37 38 50 51 52 55 59 67 69 74 76 78
09/12/2007,03 06 07 09 10 11 14 18 25 35 37 41 48 60 61 62 66 69 78 80
09/11/2007,04 05 09 13 18 24 26 27 30 36 40 45 50 51 53 54 55 59 63 65
09/10/2007,01 07 17 18 20 22 26 27 29 30 34 45 53 54 60 63 64 67 69 71
09/09/2007,01 03 08 11 12 17 22 23 27 28 32 36 43 50 56 58 64 69 73 76
09/08/2007,04 05 06 09 10 24 25 26 28 29 31 34 36 39 47 56 67 69 73 79
09/07/2007,03 05 12 23 28 33 36 46 47 53 55 56 61 63 66 67 68 72 78 79
09/06/2007,01 02 05 11 15 16 18 25 27 28 29 31 37 40 41 55 72 74 76 78
09/05/2007,04 05 07 16 17 21 26 32 33 39 41 42 47 57 58 60 61 64 73 76
09/04/2007,04 13 15 17 19 26 28 31 34 36 37 38 39 41 47 59 64 67 71 76
09/03/2007,02 10 11 15 23 25 30 31 34 44 45 47 48 52 58 60 69 78 79 80
09/02/2007,03 04 05 06 12 15 18 20 28 30 35 47 55 57 61 62 64 65 76 78
09/01/2007,01 06 09 10 11 12 14 15 18 19 34 35 36 38 41 45 46 65 75 76
08/31/2007,01 09 16 17 20 22 29 30 31 32 33 37 40 41 55 57 65 67 79 80
08/30/2007,02 13 18 25 29 30 31 32 36 42 55 56 57 62 63 66 70 71 72 74
08/29/2007,01 02 05 10 11 12 14 23 25 26 28 31 33 41 45 50 57 59 70 72
08/28/2007,04 05 06 08 09 18 20 22 23 30 33 38 51 52 53 54 62 64 67 78
08/27/2007,03 06 07 11 13 15 19 23 32 37 42 44 54 61 69 73 74 76 77 80
08/26/2007,04 11 13 24 33 37 39 42 45 46 48 53 56 58 59 63 64 70 71 78
08/25/2007,02 03 05 06 08 17 18 20 23 24 25 28 30 33 34 36 45 68 70 77
08/24/2007,02 08 09 17 18 21 24 25 33 35 37 38 53 58 60 66 67 68 77 79
08/23/2007,01 05 10 13 16 21 25 26 27 28 30 36 48 49 56 60 64 65 71 76
08/22/2007,05 06 09 11 13 15 23 35 36 37 39 45 47 48 50 51 54 66 67 68
08/21/2007,04 10 12 13 21 27 28 29 32 34 36 37 39 45 52 56 59 62 70 78
08/20/2007,01 02 08 13 17 30 34 36 42 44 46 49 51 52 56 62 63 71 75 79
08/19/2007,04 07 11 12 14 16 21 24 26 41 42 54 59 60 67 69 70 73 75 77
08/18/2007,10 11 12 15 19 21 32 35 37 40 46 50 51 52 56 62 67 71 74 80
08/17/2007,01 20 21 22 23 25 26 31 33 34 38 47 48 51 57 59 63 73 74 78
08/16/2007,06 08 09 11 18 19 20 25 27 29 31 32 51 59 63 67 68 70 78 79
08/15/2007,02 03 06 19 20 22 24 27 30 33 34 38 44 50 55 57 59 67 77 79
08/14/2007,02 05 07 10 11 12 13 15 18 19 30 36 46 61 70 72 73 74 78 80
```

```
83      08/13/2007,08 12 16 20 21 28 41 56 58 60 62 66 67 68 71 72 74 76 78 80
84      08/12/2007,06 17 21 22 23 30 32 34 36 41 44 50 53 56 58 59 62 64 66 76
85      08/11/2007,03 07 08 11 14 20 21 26 31 38 42 43 46 48 57 60 66 67 74 76
86      08/10/2007,01 07 13 16 18 23 26 27 28 29 35 37 38 42 48 49 50 67 73 79
87      08/09/2007,08 14 15 20 23 24 25 27 28 29 31 36 43 45 49 55 59 60 71 74
88      08/08/2007,01 03 08 16 20 30 31 32 34 44 49 52 55 56 61 64 66 68 69 76
89      08/07/2007,09 18 19 28 29 30 31 32 36 46 50 56 57 59 63 64 67 68 70 80
90      08/06/2007,02 03 15 16 19 20 21 32 38 43 51 52 54 62 63 65 68 76 77 80
91      08/05/2007,03 07 08 14 17 20 22 26 30 40 45 46 50 51 52 54 55 60 67 79
92      08/04/2007,01 04 06 07 12 14 15 17 18 19 23 24 32 35 36 37 60 61 73 78
93      08/03/2007,03 07 08 14 16 19 27 29 33 34 40 53 57 59 63 69 70 71 78 79
94      08/02/2007,01 05 07 14 18 19 23 28 29 33 34 38 39 51 52 57 60 65 66 68
95      08/01/2007,05 10 16 18 19 20 23 24 26 35 47 48 55 61 63 65 66 69 74 75
96      07/31/2007,02 03 04 19 27 31 32 38 40 42 43 46 48 50 53 56 60 64 68 69
97      07/30/2007,06 07 08 10 14 21 28 29 31 36 41 44 45 52 61 66 68 71 73 77
98      07/29/2007,13 17 20 23 24 29 36 40 41 44 46 51 56 59 62 70 72 78 79 80
99      07/28/2007,10 12 13 14 18 23 30 31 34 42 50 53 59 62 63 65 68 70 73 74
00      07/27/2007,04 12 13 16 17 22 25 26 29 31 33 41 43 54 55 56 59 60 73 75
01      07/26/2007,05 07 12 18 19 22 27 29 31 33 41 43 44 45 46 56 57 62 63 75
02      07/25/2007,03 12 15 16 21 24 26 39 41 45 48 57 58 59 60 65 69 70 72 76
03      07/24/2007,06 10 16 25 27 30 33 37 39 45 53 54 55 56 59 60 65 67 70 75
04      07/23/2007,01 05 09 11 13 15 17 18 20 24 34 37 46 47 50 64 68 74 77 78
05      07/22/2007,01 02 04 07 10 20 22 25 28 35 40 42 45 47 48 54 57 61 74 79
06      07/21/2007,01 04 05 06 12 15 16 22 24 27 28 34 42 45 51 58 61 69 75 77
07      07/20/2007,03 04 11 12 15 23 32 34 35 37 42 48 51 52 54 60 64 69 76 78
08      07/19/2007,09 15 20 28 30 35 42 51 54 55 57 59 60 65 67 73 74 75 76 77
09      07/18/2007,02 05 08 12 13 15 16 17 40 42 43 48 50 51 52 53 73 75 76 79
10      07/17/2007,07 08 13 14 15 26 29 33 34 38 39 40 42 43 46 51 63 68 69 73
11      07/16/2007,01 02 03 04 08 16 19 24 32 40 41 44 48 55 66 70 71 77 78 79
12      07/15/2007,01 04 14 20 25 27 31 44 45 47 51 59 61 66 69 70 72 74 78 79
13      07/14/2007,02 08 13 15 16 17 28 29 36 37 42 43 49 56 62 64 65 67 68 78
14      07/13/2007,03 05 07 17 20 27 28 32 35 37 40 44 45 49 53 55 56 64 66 72
15      07/12/2007,01 04 06 08 10 17 18 37 43 44 48 52 53 58 63 69 70 72 73 76
16      07/11/2007,03 05 07 10 11 18 24 28 35 37 39 42 43 45 46 53 57 59 67 72
17      07/10/2007,04 07 13 15 17 20 24 25 26 30 37 50 53 54 61 63 64 65 77 80
18      07/09/2007,01 03 07 14 17 18 26 27 36 38 43 46 48 49 56 57 63 66 69 72
19      07/08/2007,12 13 15 23 24 27 29 31 36 40 42 46 47 50 51 61 69 70 76 79
20      07/07/2007,06 10 17 18 22 24 31 34 40 42 44 50 54 55 56 58 62 64 73 75
21      07/06/2007,07 09 11 13 20 21 22 24 26 28 33 58 60 62 66 71 73 75 76 79
22      07/05/2007,02 04 20 33 34 36 39 40 41 48 50 51 55 56 58 66 69 73 75 78
23      07/04/2007,05 09 11 13 16 23 26 28 37 41 49 54 56 59 61 63 64 72 73 74
24      07/03/2007,01 03 05 07 08 12 21 27 30 31 38 46 47 57 60 63 64 75 77 80
25      07/02/2007,01 04 11 13 14 20 25 26 31 34 35 48 51 52 54 60 61 63 66 72
26      07/01/2007,04 05 11 13 18 20 21 24 28 29 30 33 45 49 53 54 67 73 75 79
27      06/30/2007,01 02 07 09 10 11 13 17 24 30 37 39 45 46 56 61 63 66 71 76
28      06/29/2007,02 04 06 11 12 14 15 19 20 24 31 38 41 42 46 54 63 64 65 78
29      06/28/2007,06 10 14 20 26 29 44 47 50 51 52 53 54 55 61 62 63 68 76 78
30      06/27/2007,02 03 06 09 12 13 15 18 19 20 24 30 35 41 44 47 51 74 79 80
31      06/26/2007,02 17 23 25 38 39 41 44 46 47 48 54 56 57 64 69 71 74 76 79
32      06/25/2007,04 06 11 12 13 19 21 26 27 28 33 35 43 46 54 57 60 64 65 69
33      06/24/2007,04 09 14 15 17 25 27 31 33 36 44 48 57 58 64 68 71 72 73 76
34      06/23/2007,06 07 10 12 17 20 23 33 35 37 40 41 56 58 62 64 66 71 76 80
35      06/22/2007,05 16 17 18 20 21 25 30 32 38 39 52 55 60 67 70 73 74 75 79
36      06/21/2007,05 06 07 12 36 39 47 48 53 55 56 57 58 60 65 69 73 76 78 79
```

```
06/20/2007,13 15 21 27 29 33 35 36 40 46 48 51 52 54 55 57 60 67 69 76
06/19/2007,01 02 03 05 08 09 10 14 18 22 28 30 41 50 52 55 59 61 63 66
06/18/2007,01 05 18 21 25 26 31 34 36 43 44 46 49 54 56 63 65 69 75 80
06/17/2007,02 08 09 13 15 21 31 32 39 40 45 49 51 54 59 63 69 72 79 80
06/16/2007,03 07 09 10 11 17 24 30 38 40 42 44 46 54 57 63 66 70 78 79
06/15/2007,08 11 12 22 33 34 35 42 46 47 49 53 59 61 64 68 72 73 74 79
06/14/2007,01 07 10 21 26 28 36 39 41 42 46 49 52 53 59 64 65 76 78 79
06/13/2007,01 02 05 06 08 11 13 19 24 34 45 49 53 54 64 66 68 71 72 76
06/12/2007,04 07 08 18 22 23 27 29 30 36 38 49 51 53 56 60 69 71 72 79
06/11/2007,01 09 11 13 17 22 33 39 42 44 45 47 50 53 55 56 65 68 69 77
06/10/2007,05 12 14 18 22 27 28 40 45 49 50 52 54 55 64 73 75 77 78 80
06/09/2007,06 09 14 18 20 23 24 25 27 29 37 45 48 49 59 70 73 74 75 80
06/08/2007,01 02 16 22 26 27 30 46 52 53 56 59 61 62 64 67 69 74 78 79
06/07/2007,07 11 14 21 28 30 33 35 36 44 46 50 58 59 61 63 68 70 71 79
06/06/2007,01 03 05 13 16 23 32 42 43 47 49 52 57 58 60 70 72 76 77 80
06/05/2007,01 06 11 12 13 15 19 21 22 25 33 45 51 52 58 61 64 69 70 77
06/04/2007,01 02 04 08 14 20 24 27 31 33 36 39 40 43 45 47 63 69 71 74
06/03/2007,02 05 06 13 16 19 29 30 32 34 35 38 44 46 57 58 60 65 66 74
06/02/2007,01 02 06 07 08 10 17 18 22 27 31 39 40 41 50 58 59 64 71 80
06/01/2007,03 20 22 23 25 30 31 39 44 46 51 58 61 63 66 69 70 73 77 78
05/31/2007,03 07 10 16 24 26 27 32 33 39 45 48 49 51 53 57 73 74 79 80
05/30/2007,01 08 19 21 23 27 33 34 41 43 49 53 54 56 60 62 69 71 72 74
05/29/2007,05 06 11 12 15 24 27 34 35 36 41 43 52 55 58 66 70 71 75 80
05/28/2007,01 09 10 15 17 19 21 23 24 35 47 48 49 50 52 55 64 74 77 79
05/27/2007,03 05 08 13 19 23 26 31 34 37 38 42 44 47 50 56 66 67 69 80
05/26/2007,04 06 07 10 21 26 30 37 46 55 58 59 61 62 65 72 73 74 75 76
05/25/2007,06 08 12 16 17 21 24 26 27 33 34 41 42 43 47 50 61 63 64 72
05/24/2007,05 10 12 17 19 28 29 31 34 37 39 41 43 47 50 51 59 62 64 68
05/23/2007,10 12 14 17 22 26 31 33 37 45 48 55 59 60 66 67 69 75 76 80
05/22/2007,08 11 14 15 19 21 22 24 32 35 39 41 48 53 58 70 72 73 79 80
05/21/2007,02 08 10 12 18 20 25 26 31 34 35 36 41 54 55 64 66 67 68 78
05/20/2007,01 02 09 19 23 24 27 31 37 38 39 42 52 54 59 66 68 75 77 78
05/19/2007,02 13 14 19 23 34 35 37 42 45 47 49 50 51 52 54 60 72 74 80
05/18/2007,02 03 06 07 10 21 23 25 26 31 34 40 41 42 51 55 72 73 77 79
05/17/2007,02 03 04 16 17 19 26 27 28 33 37 46 49 50 51 57 59 67 68 79
05/16/2007,11 13 18 20 22 26 27 29 33 38 42 44 54 56 59 62 70 72 75 79
05/15/2007,01 08 14 17 19 20 22 24 32 35 36 43 45 46 54 57 60 69 70 75
05/14/2007,01 03 06 07 11 15 24 31 38 39 46 47 49 52 53 55 63 64 71 78
05/13/2007,01 05 09 11 19 20 22 35 45 48 55 61 64 65 68 70 72 78 79 80
05/12/2007,01 07 11 13 14 17 31 38 43 44 45 48 51 58 64 69 70 75 76 77
05/11/2007,05 08 09 13 18 26 34 44 46 47 55 64 67 69 71 73 74 76 79 80
05/10/2007,02 06 14 15 24 26 32 35 37 44 49 51 52 59 61 62 66 69 72 75
05/09/2007,02 03 04 09 11 13 26 28 32 35 38 41 48 51 57 61 62 64 70 75
05/08/2007,11 16 20 24 25 36 40 42 44 46 48 50 52 57 61 71 72 73 79 80
05/07/2007,01 10 14 17 24 27 29 35 37 44 47 53 54 57 58 60 63 69 73 78
05/06/2007,02 05 06 20 22 29 31 37 45 46 52 57 62 65 67 69 70 72 76 79
05/05/2007,03 07 09 13 15 18 21 22 23 28 31 34 37 48 57 60 71 73 77 79
05/04/2007,02 08 10 14 21 31 37 38 40 52 61 62 64 68 72 73 74 75 77 79
05/03/2007,02 03 05 10 17 23 34 36 38 39 42 46 59 60 61 69 70 71 73 80
05/02/2007,05 08 09 13 21 31 35 49 50 51 53 55 65 66 67 68 69 72 77 80
05/01/2007,04 09 13 15 21 22 27 37 40 41 46 47 50 56 58 60 68 74 77 79
04/30/2007,03 05 06 13 21 26 32 35 37 43 45 47 49 50 51 54 56 57 69 77
04/29/2007,08 11 20 24 25 27 32 48 51 52 53 60 62 64 73 74 75 76 77 78
04/28/2007,02 04 09 11 12 13 20 26 32 34 45 46 47 48 49 59 62 67 71 72
```

```
91      04/27/2007,02 08 15 16 19 20 24 28 31 34 35 42 44 45 49 50 51 58 62 69
92      04/26/2007,09 14 16 18 22 23 24 26 29 33 35 39 50 52 53 69 70 71 73 75
93      04/25/2007,11 14 15 17 21 23 26 28 32 37 41 48 49 50 56 66 68 69 71 76
94      04/24/2007,04 10 14 17 21 26 27 33 36 38 41 45 56 57 61 62 65 70 71 79
95      04/23/2007,05 07 09 11 12 16 20 25 31 34 35 40 49 54 59 60 67 68 75 76
96      04/22/2007,02 06 07 09 15 17 20 32 42 43 45 48 51 64 66 68 69 71 76 78
97      04/21/2007,02 03 09 19 21 31 42 46 48 51 58 59 60 61 62 66 69 75 76 77
98      04/20/2007,01 03 04 06 10 12 17 25 26 28 29 30 33 35 36 38 57 59 68 79
99      04/19/2007,01 04 06 08 09 10 13 16 18 31 33 38 39 42 46 51 60 67 68 71
00      04/18/2007,03 04 05 07 10 13 14 21 30 35 36 38 51 61 66 67 69 70 76 80
01      04/17/2007,09 10 11 16 21 23 28 30 34 37 40 42 43 44 47 54 63 65 69 78
02      04/16/2007,06 09 11 12 13 20 27 35 36 40 41 42 51 54 55 57 66 71 72 76
03      04/15/2007,02 04 06 07 13 22 30 32 38 39 44 51 53 54 62 65 68 71 72 75
04      04/14/2007,16 17 19 22 26 27 31 32 37 41 42 45 50 52 53 55 58 68 69 78
05      04/13/2007,04 05 10 11 16 17 22 24 26 29 34 40 41 42 46 48 53 54 64 70
06      04/12/2007,05 11 13 14 15 20 23 27 30 34 35 41 43 51 61 69 72 74 77 80
07      04/11/2007,01 06 08 21 22 23 24 25 26 34 38 39 43 44 51 68 75 77 78 79
08      04/10/2007,09 10 13 14 20 21 22 32 33 36 43 47 52 59 60 70 72 74 75 77
09      04/09/2007,02 03 07 11 14 19 20 22 26 27 34 35 39 41 42 45 57 61 68 76
10      04/08/2007,03 12 14 16 23 27 32 33 37 38 40 43 44 46 48 55 58 61 69 73
11      04/07/2007,11 16 20 21 22 26 29 30 36 39 42 51 54 55 65 66 68 77 79 80
12      04/06/2007,06 07 10 13 15 26 27 28 34 38 40 48 49 52 56 62 63 71 75 80
13      04/05/2007,08 09 19 24 26 30 34 35 38 40 42 48 50 56 58 61 70 71 72 75
14      04/04/2007,02 03 08 12 16 19 20 26 29 33 35 39 53 54 56 59 62 64 65 78
15      04/03/2007,10 13 15 16 21 23 24 27 28 32 34 35 36 39 48 63 68 73 74 77
16      04/02/2007,06 08 13 16 20 25 29 31 32 35 39 40 44 54 55 56 57 62 65 80
17      04/01/2007,01 05 07 15 20 25 27 38 39 42 43 53 59 63 67 68 71 72 75 76
18      03/31/2007,04 10 11 20 22 24 25 45 46 49 51 52 57 61 63 67 68 69 71 76
19      03/30/2007,02 04 11 20 22 25 31 38 43 45 46 48 50 61 62 64 69 70 72 73
20      03/29/2007,05 10 12 14 24 27 39 44 46 47 51 54 55 61 62 63 69 70 79 80
21      03/28/2007,01 04 05 06 12 23 25 30 32 40 42 44 48 50 52 57 61 70 71 72
22      03/27/2007,03 07 08 10 11 13 22 24 36 38 42 44 52 53 54 57 62 67 70 71
23      03/26/2007,07 11 18 20 22 23 34 36 47 48 49 50 52 58 60 65 66 73 76 78
24      03/25/2007,03 05 06 07 10 11 12 18 21 25 26 35 38 40 42 45 49 70 75 79
25      03/24/2007,03 13 20 22 23 33 39 40 42 43 46 49 50 53 65 68 70 71 72 79
26      03/23/2007,01 02 04 08 09 16 17 22 31 33 35 41 43 46 51 52 54 55 65 74
27      03/22/2007,01 02 03 10 12 18 21 27 38 42 45 53 56 66 69 72 73 75 76 80
28      03/21/2007,01 07 09 13 15 22 25 27 32 33 39 41 43 44 49 50 52 55 57 80
29      03/20/2007,05 08 11 12 16 19 23 25 31 35 36 40 52 53 57 63 67 71 79 80
30      03/19/2007,03 05 06 10 12 15 19 32 40 49 53 57 58 60 61 70 71 72 73 78
31      03/18/2007,01 04 05 18 19 22 23 27 30 35 40 49 50 51 66 68 73 75 76 77
32      03/17/2007,03 04 07 11 18 19 20 22 37 38 39 47 58 61 65 66 70 71 72 80
33      03/16/2007,05 08 09 18 21 29 30 32 36 38 44 45 47 51 58 61 65 68 77 80
34      03/15/2007,01 04 05 11 15 17 20 21 24 26 33 35 39 48 49 52 55 56 65 73
35      03/14/2007,03 04 09 13 15 16 20 21 23 34 39 48 49 53 54 56 63 68 78 80
36      03/13/2007,02 03 06 18 19 22 27 30 36 53 55 56 57 61 63 66 69 70 73 77
37      03/12/2007,08 10 12 16 19 23 24 25 26 29 35 46 52 54 58 59 70 72 75 76
38      03/11/2007,02 03 04 12 13 18 30 33 37 38 39 42 43 46 49 54 56 60 62 78
39      03/10/2007,04 06 11 13 14 20 22 28 30 33 42 45 46 57 63 64 66 67 75 79
40      03/09/2007,01 05 10 17 19 21 22 34 36 37 38 42 43 49 56 62 67 68 71 76
41      03/08/2007,08 15 18 23 26 30 32 33 35 40 43 50 54 55 56 60 65 70 73 79
42      03/07/2007,01 07 14 15 22 25 28 29 34 39 41 45 47 48 59 66 70 73 78 79
43      03/06/2007,02 08 15 21 22 28 40 41 45 48 50 57 61 62 66 68 71 72 74 80
44      03/05/2007,01 07 08 12 15 17 18 21 32 33 37 38 40 49 51 52 53 59 60 69
```

```
03/04/2007,05 06 07 16 29 30 31 35 38 43 47 49 54 59 60 63 69 70 72 77
03/03/2007,11 19 20 21 22 26 29 32 36 40 42 46 51 57 60 64 65 71 79 80
03/02/2007,01 05 10 12 13 19 20 24 26 27 30 32 38 42 47 54 66 74 75 78
03/01/2007,02 03 05 09 14 17 20 22 33 37 39 44 45 48 51 54 55 58 70 76
02/28/2007,04 07 17 21 24 27 31 34 35 38 40 41 44 50 55 69 70 73 75 77
02/27/2007,03 06 09 20 21 34 42 45 46 49 50 52 53 56 59 68 72 77 79 80
02/26/2007,01 02 05 06 08 10 26 27 28 29 34 42 44 48 55 57 58 61 68 78
02/25/2007,03 05 09 12 14 15 16 17 36 39 40 43 44 46 52 63 64 65 78 79
02/24/2007,02 03 14 16 19 38 53 54 55 56 58 61 62 64 66 70 73 74 75 76
02/23/2007,03 04 07 09 10 11 13 16 17 30 32 33 36 40 42 59 66 67 71 74
02/22/2007,03 08 09 10 11 14 23 24 39 47 53 54 58 60 64 66 68 75 76 79
02/21/2007,02 06 08 12 13 19 23 27 32 34 36 40 44 50 55 59 61 66 70 72
02/20/2007,02 04 13 22 24 25 27 28 29 30 34 40 48 51 52 53 54 63 67 75
02/19/2007,02 07 13 16 17 22 23 26 29 35 36 43 52 55 58 72 73 75 78 80
02/18/2007,11 13 14 18 20 23 26 38 46 48 53 54 55 57 65 66 71 74 77 80
02/17/2007,04 13 14 16 20 32 35 36 40 43 46 47 48 49 53 55 61 65 76 80
02/16/2007,01 03 06 08 09 10 16 20 29 34 41 46 57 62 63 64 65 77 79 80
02/15/2007,02 05 07 08 10 11 15 17 23 26 30 32 38 40 49 51 55 58 61 65
02/14/2007,02 04 05 07 12 13 16 17 24 35 38 42 46 49 56 59 61 67 69 73
02/13/2007,01 05 08 10 11 18 20 21 23 28 29 41 47 48 51 64 72 73 76 80
02/12/2007,06 09 18 27 28 31 33 38 40 41 50 53 56 57 61 63 68 74 77 80
02/11/2007,03 06 07 15 27 31 35 41 44 51 53 55 56 58 60 62 63 69 70 71
02/10/2007,02 04 05 12 14 16 20 23 27 34 36 37 44 51 52 56 71 72 75 80
02/09/2007,08 20 25 27 29 33 39 42 43 44 45 47 54 66 68 69 70 74 77 78
02/08/2007,02 06 10 13 15 27 28 32 34 39 41 46 49 52 54 69 75 77 78 79
02/07/2007,03 08 12 25 26 40 44 50 55 59 60 61 63 64 68 71 72 76 78 79
02/06/2007,03 04 08 16 19 21 22 24 25 28 31 39 40 56 57 58 61 69 70 77
02/05/2007,04 10 15 16 19 34 43 46 47 49 50 52 53 54 57 59 62 63 78 80
02/04/2007,02 08 12 13 17 18 19 30 44 45 46 54 62 63 65 68 70 71 79 80
02/03/2007,10 14 18 22 23 24 30 32 35 37 54 55 59 61 70 71 73 74 76 80
02/02/2007,02 04 06 10 26 28 36 39 40 41 44 45 58 65 67 69 70 74 77 80
02/01/2007,04 05 09 11 12 22 24 35 38 39 40 44 45 46 47 48 54 69 75 79
01/31/2007,03 11 14 16 18 20 25 28 32 38 40 53 57 65 67 71 73 74 75 77
01/30/2007,06 08 10 11 13 14 18 22 27 28 39 42 54 56 60 63 64 68 73 74
01/29/2007,01 09 12 16 21 27 29 33 34 36 39 48 49 51 53 56 60 63 69 80
01/28/2007,02 06 16 21 31 39 40 49 50 51 52 53 54 55 58 59 63 64 74 78
01/27/2007,07 10 12 16 20 22 26 27 32 36 39 43 44 45 52 58 61 63 73 74
01/26/2007,05 09 10 11 14 18 19 20 21 34 35 41 44 62 64 66 67 68 72 76
01/25/2007,15 22 28 31 40 42 44 45 46 58 59 62 63 65 67 69 70 71 75 79
01/24/2007,01 06 09 13 14 17 19 21 24 28 30 35 36 38 42 44 54 65 71 80
01/23/2007,01 05 07 13 15 19 26 31 32 33 37 41 43 47 51 65 67 72 76 77
01/22/2007,14 16 20 22 29 30 31 40 42 43 52 53 55 56 58 62 64 68 76 78
01/21/2007,01 10 16 19 22 23 24 27 29 30 35 36 41 42 43 53 62 66 67 76
01/20/2007,08 10 12 24 25 27 30 32 37 44 47 48 49 51 58 60 61 64 66 80
01/19/2007,01 11 12 17 19 35 40 41 47 53 56 58 59 60 65 67 71 74 76 78
01/18/2007,04 08 14 30 31 33 37 48 53 55 56 61 62 64 65 67 69 74 78 80
01/17/2007,02 04 05 08 16 22 23 26 34 43 52 59 65 66 68 71 76 77 79 80
01/16/2007,08 09 14 16 18 21 22 23 35 37 38 43 47 49 60 61 65 66 68 79
01/15/2007,01 07 12 21 24 26 38 41 45 46 51 54 60 61 64 67 70 74 76 78
01/14/2007,06 10 20 25 27 33 34 36 37 38 40 41 43 49 51 53 55 71 74 76
01/13/2007,05 06 13 20 26 28 30 32 36 39 40 48 49 55 59 60 61 71 73 80
01/12/2007,01 03 11 13 16 20 21 22 24 35 36 41 46 58 59 62 64 67 75 80
01/11/2007,04 19 22 24 29 30 33 36 39 43 45 47 48 54 55 56 58 59 70 74
01/10/2007,06 09 20 24 25 28 29 41 42 45 46 49 51 53 55 67 69 70 77 78
```

```
99      01/09/2007,06 07 08 15 18 20 23 25 33 39 50 56 59 63 64 65 67 68 78 80
00      01/08/2007,03 08 09 14 17 20 21 27 41 42 44 50 51 53 54 55 65 75 77 78
01      01/07/2007,01 02 03 08 09 11 13 16 17 21 26 30 33 34 41 44 51 60 63 72
02      01/06/2007,02 07 08 10 11 12 17 22 23 24 27 33 39 46 49 53 61 67 70 73
03      01/05/2007,07 13 14 22 25 26 28 31 32 33 38 43 54 58 69 72 74 75 77 79
04      01/04/2007,04 05 07 08 10 11 15 19 22 24 29 46 48 49 57 63 68 73 74 75
05      01/03/2007,02 10 11 12 13 15 22 23 24 26 27 34 37 48 49 62 65 67 74 80
06      01/02/2007,03 10 12 15 16 19 23 24 28 31 50 52 56 57 58 62 67 70 75 76
07      01/01/2007,02 05 07 15 16 20 21 26 32 37 42 52 53 58 59 60 71 74 77 78
08      12/31/2006,01 02 03 05 09 20 22 24 29 32 40 47 48 52 55 60 62 67 74 80
09      12/30/2006,03 04 08 09 10 11 13 18 25 28 30 35 37 42 46 53 54 60 67 74
10      12/29/2006,02 08 10 12 17 18 21 31 37 38 46 48 56 58 65 66 72 77 78 79
11      12/28/2006,01 02 07 13 15 16 22 23 30 32 36 53 57 58 63 68 69 70 74 80
12      12/27/2006,14 17 19 26 27 29 30 32 33 36 37 41 44 45 53 54 56 62 67 69
13      12/26/2006,07 09 13 18 23 27 38 42 54 55 60 61 62 67 68 69 70 73 77 79
14      12/24/2006,05 08 10 11 18 22 23 25 28 29 31 33 37 45 47 51 59 67 68 74
15      12/23/2006,03 07 09 11 12 13 17 22 23 28 30 38 44 45 46 53 54 59 73 80
16      12/22/2006,05 10 13 15 17 20 25 27 31 36 40 48 50 56 57 60 62 68 71 80
17      12/21/2006,03 06 07 08 27 28 29 30 32 38 45 54 56 62 64 71 72 73 74 79
18      12/20/2006,06 09 10 11 16 17 18 21 29 32 35 37 40 41 51 53 54 56 57 66
19      12/19/2006,01 03 11 13 15 20 25 28 31 38 45 47 49 50 55 56 58 68 71 79
20      12/18/2006,03 09 14 15 21 22 23 25 27 32 40 43 45 46 51 52 67 69 71 74
21      12/17/2006,01 02 04 06 08 16 17 22 29 30 39 40 44 48 53 56 59 60 71 80
22      12/16/2006,02 03 05 06 15 18 27 32 34 37 38 39 41 43 48 50 58 65 67 70
23      12/15/2006,02 07 16 18 19 22 26 29 31 34 37 41 43 49 52 62 69 70 77 80
24      12/14/2006,01 04 10 11 28 29 35 38 40 49 51 54 56 59 64 65 69 76 77 78
25      12/13/2006,04 06 10 12 14 16 25 35 38 46 47 48 50 51 55 60 68 71 72 73
26      12/12/2006,02 05 09 18 23 24 33 35 37 44 45 49 50 58 66 68 69 71 74 80
27      12/11/2006,03 04 13 15 21 31 39 40 43 44 45 48 50 56 59 61 63 66 67 80
28      12/10/2006,01 02 04 05 06 19 21 22 31 37 38 39 43 45 51 55 62 68 74 80
29      12/09/2006,02 03 05 06 11 16 18 25 28 31 36 39 46 52 65 66 68 69 72 80
30      12/08/2006,11 12 18 19 21 23 25 28 30 31 41 44 47 52 54 55 58 60 61 71
31      12/07/2006,01 03 11 12 20 21 22 33 34 36 47 48 49 52 58 70 71 72 73 75
32      12/06/2006,01 11 17 19 22 26 27 29 30 32 36 43 55 57 61 62 65 72 73 78
33      12/05/2006,03 04 08 09 19 23 24 29 31 32 33 45 46 54 57 64 66 72 75 80
34      12/04/2006,03 10 15 19 20 24 26 27 28 40 43 46 50 57 61 66 68 70 72 75
35      12/03/2006,05 09 12 17 20 25 28 31 37 43 44 48 54 56 61 65 67 68 70 73
36      12/02/2006,04 15 23 24 25 31 37 40 44 49 50 51 56 57 60 61 64 71 74 78
37      12/01/2006,02 05 07 14 15 22 23 29 30 38 42 44 53 58 61 65 68 69 70 77
38      11/30/2006,11 12 14 19 21 24 25 28 38 41 42 44 47 50 52 57 63 67 71 72
39      11/29/2006,05 06 08 09 10 11 12 25 26 29 31 40 41 46 47 56 63 68 72 76
40      11/28/2006,05 06 10 12 13 26 29 31 33 40 55 56 57 58 62 68 71 76 78 79
41      11/27/2006,08 09 12 14 19 20 23 24 36 38 40 47 48 49 53 55 58 65 71 74
42      11/26/2006,01 12 17 19 20 25 32 39 40 45 52 59 62 63 68 69 72 73 79 80
43      11/25/2006,05 07 09 16 20 22 29 35 45 48 49 54 58 64 67 69 71 73 74 75
44      11/24/2006,02 12 13 14 19 20 32 33 36 40 41 44 46 47 48 49 50 56 63 69
45      11/23/2006,05 07 09 12 13 18 20 21 25 32 35 36 41 42 55 60 63 73 77 80
46      11/22/2006,02 03 05 06 21 32 39 48 49 51 52 54 59 62 63 64 66 70 76 77
47      11/21/2006,06 10 15 17 21 30 36 40 41 46 50 55 60 62 63 64 67 68 69 76
48      11/20/2006,10 11 14 15 16 24 28 30 35 36 41 45 53 55 56 57 63 71 73 75
49      11/19/2006,06 12 14 16 19 22 28 30 37 41 42 50 52 57 61 67 73 74 75 79
50      11/18/2006,04 05 09 13 20 24 25 29 38 41 49 52 62 63 65 67 71 74 76 80
51      11/17/2006,04 05 09 12 13 19 21 24 27 31 33 40 43 54 62 68 71 74 77 78
52      11/16/2006,01 02 04 05 10 18 20 21 28 31 34 44 47 50 52 56 60 67 78 79
```

```
53      11/15/2006,02 06 09 20 32 34 45 47 55 56 58 63 64 65 67 70 77 78 79 80
54      11/14/2006,05 10 16 17 21 23 25 30 33 40 43 47 50 52 58 62 65 67 70 76
55      11/13/2006,01 02 06 17 22 27 28 31 34 35 40 42 52 55 56 58 62 65 66 79
56      11/12/2006,02 03 05 07 23 31 34 39 44 45 48 52 55 57 58 61 68 73 74 77
57      11/11/2006,01 08 13 17 26 33 34 35 41 42 43 44 46 54 59 61 67 68 73 77
58      11/10/2006,02 03 17 18 21 28 29 32 33 34 41 60 62 68 72 73 74 76 77 78
59      11/09/2006,01 02 04 07 16 32 35 36 38 39 42 47 49 50 51 54 62 75 76 80
60      11/08/2006,01 04 07 08 09 10 11 14 20 33 52 53 56 57 61 63 65 77 78 80
61      11/07/2006,03 09 23 26 33 36 37 44 47 50 53 56 57 61 62 69 73 74 75 78
62      11/06/2006,01 06 07 19 29 31 36 38 40 42 46 50 60 61 63 66 69 71 77 80
63      11/05/2006,01 02 05 07 09 10 13 25 34 35 36 37 40 48 50 52 61 65 74 79
64      11/04/2006,06 10 12 13 14 15 23 24 27 28 29 37 43 44 46 52 64 68 74 77
65      11/03/2006,05 09 15 20 26 31 32 34 41 46 47 50 51 52 53 56 62 72 79 80
66      11/02/2006,04 10 13 16 18 19 23 31 32 39 51 55 56 60 61 62 64 73 78 80
67      11/01/2006,01 05 14 16 17 19 23 25 28 33 38 44 48 50 58 60 64 66 77 78
68      10/31/2006,04 11 16 18 22 23 24 25 31 35 37 40 43 45 46 49 51 61 65 68
69      10/30/2006,01 02 04 13 17 18 21 24 29 38 39 40 46 47 48 51 58 62 68 79
70      10/29/2006,02 04 05 06 08 09 12 27 32 38 39 40 41 46 55 56 64 65 67 76
71      10/28/2006,06 09 12 13 15 20 23 26 42 43 47 48 54 58 61 63 68 70 73 75
72      10/27/2006,04 05 07 08 09 12 20 23 27 30 31 41 50 60 63 70 74 76 77 78
73      10/26/2006,07 10 13 16 17 22 24 26 35 37 49 51 61 63 65 66 67 74 78 79
74      10/25/2006,10 13 15 17 20 23 27 28 30 35 39 40 41 47 49 57 62 65 69 78
75      10/24/2006,01 03 10 21 24 26 27 31 40 44 47 48 49 55 56 57 64 66 67 77
76      10/23/2006,07 13 17 18 22 24 26 33 35 36 38 42 48 49 50 63 67 70 71 76
77      10/22/2006,01 04 05 06 11 17 20 24 27 29 34 36 41 56 58 60 66 70 73 77
78      10/21/2006,01 03 08 14 22 23 28 32 38 41 42 43 48 63 65 67 68 71 76 78
79      10/20/2006,02 11 14 18 21 29 30 38 46 49 51 52 53 55 60 62 64 67 70 71
80      10/19/2006,06 07 12 16 18 21 25 30 34 36 48 55 58 60 61 63 66 69 78 80
81      10/18/2006,01 05 09 12 16 24 26 28 32 40 48 53 56 59 61 63 68 70 73 76
82      10/17/2006,08 14 16 17 18 31 35 36 38 44 46 47 48 49 50 57 61 63 66 73
83      10/16/2006,03 05 06 18 21 29 35 37 44 48 51 52 57 64 67 68 71 72 73 79
84      10/15/2006,03 04 05 17 20 27 31 32 34 36 43 44 46 47 48 51 53 57 64 80
85      10/14/2006,09 11 12 16 23 24 26 30 33 37 38 39 46 48 50 55 63 66 69 76
86      10/13/2006,01 06 11 12 17 23 24 27 32 33 34 35 40 49 63 64 65 66 75 77
87      10/12/2006,03 04 09 11 17 20 24 25 26 34 41 43 47 51 60 65 70 71 72 76
88      10/11/2006,03 04 06 09 13 16 20 24 25 34 40 50 51 52 56 63 69 71 72 73
89      10/10/2006,05 06 07 12 18 24 26 32 35 36 47 48 52 58 60 66 69 70 73 76
90      10/09/2006,02 20 21 22 23 28 38 46 48 49 50 52 54 55 58 60 64 78 79 80
91      10/08/2006,03 07 21 22 29 30 31 35 38 44 48 56 57 58 60 67 68 70 72 78
92      10/07/2006,03 06 07 08 09 12 19 22 27 36 48 51 52 53 56 57 76 77 78 80
93      10/06/2006,02 03 07 17 18 23 29 30 31 32 36 40 49 53 55 56 57 59 60 66
94      10/05/2006,01 03 06 08 16 17 19 22 30 36 37 39 44 53 62 63 67 68 77 78
95      10/04/2006,04 13 18 23 24 30 37 40 43 45 51 52 58 60 62 63 65 67 68 76
96      10/03/2006,04 09 16 18 20 21 23 27 30 37 39 41 43 47 51 61 72 73 75 77
97      10/02/2006,02 04 07 08 09 10 14 19 30 36 37 39 50 51 55 61 68 73 74 80
98      10/01/2006,01 03 09 14 18 28 31 33 36 37 41 48 51 54 56 58 63 67 75 78
99      09/30/2006,09 16 21 23 25 27 28 29 33 34 36 37 39 56 59 63 70 72 73 76
00      09/29/2006,03 07 10 12 16 32 34 45 51 52 55 56 58 64 66 70 74 77 78 79
01      09/28/2006,01 11 18 24 29 31 34 38 39 41 42 48 49 52 54 56 62 64 68 70
02      09/27/2006,10 11 12 15 18 23 25 27 31 38 41 44 45 53 59 63 65 70 73 79
03      09/26/2006,03 11 15 16 19 23 25 26 33 37 40 44 52 55 57 59 65 67 71 77
04      09/25/2006,01 03 04 06 08 11 16 23 37 39 42 44 48 64 65 67 68 70 72 80
05      09/24/2006,04 07 10 13 17 21 23 25 30 32 38 40 44 51 55 70 73 77 78 80
06      09/23/2006,01 04 06 07 10 12 15 20 22 27 28 30 41 46 49 57 60 61 63 66
```

07 09/22/2006,08 12 13 15 18 29 32 33 35 37 43 44 45 46 53 66 67 71 72 76
08 09/21/2006,06 13 14 17 19 28 30 40 41 47 49 54 55 61 65 68 76 77 79 80
09 09/20/2006,02 03 04 09 17 22 28 30 31 33 42 46 52 56 63 67 73 76 79 80
10 09/19/2006,02 08 11 12 17 19 22 23 26 39 42 45 46 49 50 55 66 68 70 78
11 09/18/2006,02 03 08 09 16 23 25 27 31 32 33 35 36 47 65 67 68 73 74 78
12 09/17/2006,04 08 10 14 22 33 35 44 48 51 52 53 60 63 65 67 68 69 73 74
13 09/16/2006,04 08 10 13 15 25 28 35 37 41 43 47 53 54 57 58 60 61 66 74
14 09/15/2006,08 10 12 15 22 25 27 28 32 33 34 46 47 58 64 65 68 75 76 78
15 09/14/2006,06 11 16 18 21 32 36 37 38 41 45 57 59 63 71 72 73 74 75 80
16 09/13/2006,01 11 13 20 23 24 28 31 32 36 44 45 52 61 63 65 67 74 77 79
17 09/12/2006,07 09 10 13 14 18 19 20 30 32 35 38 46 48 59 62 64 65 71 77
18 09/11/2006,01 03 08 21 22 24 25 32 33 34 35 40 41 44 46 51 60 61 65 71
19 09/10/2006,08 13 15 17 18 21 29 33 38 39 40 45 47 64 68 70 72 75 78 80
20 09/09/2006,03 04 09 18 20 22 26 29 31 33 37 38 39 42 44 49 63 68 76 78
21 09/08/2006,01 02 03 04 09 12 13 17 21 26 28 29 37 47 48 59 65 67 68 73
22 09/07/2006,05 12 13 14 15 20 30 31 34 35 41 51 55 57 66 69 72 73 75 79
23 09/06/2006,03 04 05 08 13 19 30 32 33 43 45 51 54 59 60 68 69 71 77 79
24 09/05/2006,09 12 13 15 23 27 30 31 32 35 36 37 47 49 52 56 58 64 65 68
25 09/04/2006,02 04 06 08 10 14 18 20 30 32 35 42 45 49 54 63 67 70 72 79
26 09/03/2006,06 08 11 14 18 25 26 29 32 33 34 35 44 48 49 54 57 68 70 77
27 09/02/2006,02 05 06 10 11 12 14 18 37 42 48 49 55 56 57 64 67 72 77 80
28 09/01/2006,05 07 10 12 13 15 27 29 32 35 37 38 48 52 56 57 60 65 68 76
29 08/31/2006,03 07 09 17 19 24 33 39 43 47 50 54 58 59 60 64 70 71 72 74
30 08/30/2006,02 03 05 11 13 19 20 21 34 36 42 47 50 52 55 56 57 63 64 75
31 08/29/2006,01 02 04 09 10 12 18 22 25 26 28 30 31 33 52 57 59 68 76 78
32 08/28/2006,03 04 09 13 14 17 22 25 28 31 37 39 52 57 62 67 68 71 72 78
33 08/27/2006,10 14 17 20 22 26 27 28 30 31 33 48 50 51 55 58 59 60 72 76
34 08/26/2006,03 06 12 14 17 22 26 31 32 38 40 41 44 45 52 59 60 63 64 72
35 08/25/2006,03 04 11 22 23 30 33 35 40 41 44 45 49 50 62 63 65 67 69 74
36 08/24/2006,02 10 13 14 18 19 20 24 25 33 34 35 36 41 44 49 50 59 74 75
37 08/23/2006,08 09 10 18 19 22 25 32 38 40 43 47 48 52 54 58 60 66 67 75
38 08/22/2006,02 03 07 10 12 15 18 23 25 27 28 29 47 49 50 51 56 65 70 73
39 08/21/2006,04 12 14 15 16 31 33 36 37 39 40 43 44 46 53 58 63 66 75 77
40 08/20/2006,08 19 20 21 22 25 29 34 36 37 41 43 44 63 69 70 73 74 76 78
41 08/19/2006,03 05 08 21 23 26 27 31 34 35 36 37 41 49 52 55 60 62 65 79
42 08/18/2006,04 07 09 11 13 14 15 22 23 31 33 35 40 50 61 65 67 69 71 77
43 08/17/2006,04 06 15 19 21 23 29 39 44 46 51 56 62 65 68 70 72 73 75 79
44 08/16/2006,03 04 08 15 20 23 27 35 42 45 51 52 59 64 66 68 73 74 76 79
45 08/15/2006,01 03 04 06 13 14 22 25 26 27 35 37 46 49 54 60 68 71 72 73
46 08/14/2006,05 13 16 23 24 25 36 37 39 41 42 43 45 49 57 61 68 69 71 79
47 08/13/2006,02 07 09 14 16 17 21 22 23 28 29 35 36 38 45 53 58 65 73 78
48 08/12/2006,01 04 14 23 30 35 37 40 48 51 54 61 62 64 65 70 73 75 79 80
49 08/11/2006,04 16 17 21 28 29 34 36 38 42 43 44 47 48 51 54 61 66 71 80
50 08/10/2006,07 19 22 23 36 38 42 44 45 48 52 54 55 59 60 61 62 68 72 76
51 08/09/2006,01 05 08 10 13 19 27 28 36 37 38 39 42 43 51 58 65 72 78 80
52 08/08/2006,05 06 09 17 21 22 23 25 26 31 34 35 39 46 50 56 64 65 69 71
53 08/07/2006,02 08 12 17 19 32 35 38 41 46 49 53 61 63 66 68 69 70 75 77
54 08/06/2006,01 10 12 13 18 22 23 26 28 29 37 38 41 47 49 55 56 58 61 73
55 08/05/2006,08 09 15 17 19 25 27 29 30 33 36 48 49 53 58 68 73 77 78 79
56 08/04/2006,13 16 20 23 24 27 30 34 37 47 53 55 62 63 67 69 72 78 79 80
57 08/03/2006,03 05 07 08 15 17 27 40 45 46 47 50 53 54 56 58 61 63 67 74
58 08/02/2006,02 04 06 08 14 17 22 26 27 31 40 41 42 44 48 49 66 76 77 80
59 08/01/2006,04 16 19 22 26 28 30 36 37 40 44 46 52 54 57 59 62 64 71 74
60 07/31/2006,04 07 11 12 13 14 18 21 24 25 33 35 36 38 44 50 56 62 71 77

```
61      07/30/2006,07 09 10 12 14 15 16 34 35 36 38 40 41 44 51 55 57 58 66 78
62      07/29/2006,01 10 14 16 28 40 44 48 51 53 54 63 64 65 68 71 74 77 78 79
63      07/28/2006,03 08 11 13 17 18 21 24 25 30 36 56 57 59 63 69 72 75 77 79
64      07/27/2006,05 11 19 21 26 27 32 36 37 42 44 48 52 60 66 67 68 72 73 75
65      07/26/2006,03 11 14 18 22 24 27 28 30 36 46 47 48 55 57 59 65 68 69 74
66      07/25/2006,01 07 10 24 28 29 45 48 52 54 55 56 60 62 63 65 66 71 72 80
67      07/24/2006,11 13 14 16 21 22 25 31 38 39 48 49 51 53 56 59 64 70 71 77
68      07/23/2006,05 07 08 14 17 21 24 28 29 30 33 39 42 43 52 53 54 62 66 78
69      07/22/2006,04 06 07 11 19 22 25 27 32 35 40 45 49 52 53 61 63 71 73 77
70      07/21/2006,04 11 17 22 25 26 28 34 35 36 47 53 55 56 57 58 60 68 77 79
71      07/20/2006,02 05 12 27 28 35 43 44 46 47 48 49 53 54 59 64 65 71 73 77
72      07/19/2006,03 04 06 08 14 15 22 28 39 41 42 47 48 51 54 59 63 68 73 76
73      07/18/2006,01 05 10 19 22 23 30 32 33 34 37 38 45 47 56 62 72 74 75 78
74      07/17/2006,05 07 10 11 14 19 30 33 35 38 46 49 54 56 60 62 66 69 75 79
75      07/16/2006,01 03 06 10 15 16 20 21 22 28 32 34 53 60 66 68 69 73 78 80
76      07/15/2006,10 11 14 18 21 23 24 28 30 34 38 39 42 44 46 48 59 70 71 74
77      07/14/2006,08 12 18 21 23 37 38 42 46 47 48 51 53 57 58 61 65 70 71 77
78      07/13/2006,01 05 10 16 19 29 31 36 40 44 49 50 51 57 61 64 68 70 73 75
79      07/12/2006,03 04 06 07 08 18 24 33 34 41 48 49 52 56 59 62 65 66 67 78
80      07/11/2006,03 10 14 15 22 28 33 39 42 43 44 48 50 52 53 55 64 74 75 76
81      07/10/2006,04 09 10 22 23 26 29 31 38 41 43 50 52 53 58 61 65 73 75 80
82      07/09/2006,01 03 06 11 13 33 41 42 43 44 51 52 58 61 62 66 68 70 76 78
83      07/08/2006,02 09 12 16 23 26 27 30 33 38 40 46 49 52 60 67 71 73 78 80
84      07/07/2006,02 06 12 14 23 27 28 31 33 41 53 54 55 63 64 65 67 70 75 77
85      07/06/2006,01 11 13 14 17 20 24 26 27 31 32 37 41 42 44 48 52 54 60 66
86      07/05/2006,07 09 10 15 17 21 22 26 32 48 53 59 63 64 66 68 69 72 73 79
87      07/04/2006,03 05 11 12 15 24 28 31 38 39 41 45 55 60 63 68 69 75 78 79
88      07/03/2006,02 03 05 08 13 15 17 18 27 32 33 43 47 61 68 69 71 74 76 79
89      07/02/2006,10 11 14 16 20 21 22 33 35 36 39 55 57 58 59 66 67 70 71 75
90      07/01/2006,05 08 22 26 29 32 38 40 47 51 52 57 61 62 67 70 71 77 78 79
91      06/30/2006,02 08 09 10 16 18 22 24 25 45 52 53 55 65 67 70 71 72 76 78
92      06/29/2006,05 15 16 21 22 31 33 35 36 40 55 60 62 64 67 69 73 74 77 78
93      06/28/2006,04 09 13 14 15 16 21 26 31 33 38 39 48 55 57 59 67 69 73 76
94      06/27/2006,01 06 07 20 22 26 38 39 40 41 42 52 58 61 64 69 72 74 77 80
95      06/26/2006,02 03 05 07 09 10 24 30 33 37 40 41 42 44 48 49 57 61 64 68
96      06/25/2006,01 02 05 09 22 26 28 29 31 41 43 44 52 53 56 59 60 66 74 78
97      06/24/2006,06 10 15 17 18 22 33 34 37 38 47 50 66 67 68 69 72 73 74 80
98      06/23/2006,07 13 17 22 26 27 29 34 35 36 39 41 45 46 47 53 55 59 71 80
99      06/22/2006,03 16 17 21 22 27 32 37 41 43 50 58 60 61 64 72 73 77 78 80
00      06/21/2006,09 11 12 16 20 26 33 39 43 47 49 50 54 57 58 61 66 68 69 71
01      06/20/2006,10 11 17 21 22 25 34 44 46 48 53 55 56 57 61 62 67 68 77 80
02      06/19/2006,08 10 11 13 14 20 26 27 33 39 41 43 47 53 54 58 62 66 70 71
03      06/18/2006,02 04 05 09 10 12 15 17 23 27 28 29 36 40 45 64 68 72 73 77
04      06/17/2006,05 06 12 19 23 25 32 33 41 56 58 61 62 66 68 70 73 74 76 77
05      06/16/2006,16 19 20 21 22 27 28 29 33 35 41 50 52 57 59 60 64 67 71 72
06      06/15/2006,01 03 05 17 19 23 27 29 32 42 44 45 53 56 58 59 63 70 76 78
07      06/14/2006,03 09 15 16 18 19 20 37 38 39 43 47 51 60 61 63 66 67 71 79
08      06/13/2006,01 09 10 16 17 18 19 23 26 30 42 54 55 69 70 72 74 75 77 80
09      06/12/2006,02 20 23 29 31 35 37 40 43 51 52 54 58 59 61 65 67 70 73 74
10      06/11/2006,02 04 05 09 13 14 16 24 29 33 37 39 48 49 55 63 66 69 71 78
11      06/10/2006,04 06 07 08 12 14 15 17 19 23 24 27 28 35 38 41 48 53 73 80
12      06/09/2006,05 11 15 16 23 24 26 27 28 31 33 38 47 59 64 65 71 74 77 80
13      06/08/2006,01 04 05 09 10 14 18 27 31 38 40 41 42 45 47 52 54 65 68 73
14      06/07/2006,01 03 07 09 19 20 30 34 35 36 39 42 44 45 46 61 72 75 76 77
```

```
15      06/06/2006,06 10 12 14 18 21 22 31 32 33 36 47 54 60 65 66 68 70 71 80
16      06/05/2006,02 05 21 25 31 35 36 38 43 47 49 53 55 57 58 63 70 72 79 80
17      06/04/2006,01 17 20 26 29 31 37 41 43 49 51 55 56 58 60 66 69 74 77 80
18      06/03/2006,13 14 16 17 24 25 26 28 30 34 35 39 49 50 56 62 65 66 68 71
19      06/02/2006,02 08 12 14 15 16 17 22 26 29 31 33 43 50 54 61 65 72 76 78
20      06/01/2006,03 10 13 17 19 21 23 28 29 30 36 40 47 51 52 55 59 61 67 69
21      05/31/2006,04 05 09 11 14 25 26 27 30 31 32 33 38 41 42 58 61 62 65 79
22      05/30/2006,03 06 09 13 14 20 27 34 35 37 38 42 43 46 52 57 64 67 70 74
23      05/29/2006,05 11 13 17 23 27 28 31 34 38 43 44 63 66 67 70 71 74 77 78
24      05/28/2006,01 10 13 16 23 25 30 32 33 35 38 43 46 49 51 60 68 74 75 78
25      05/27/2006,05 10 11 21 26 29 34 35 38 40 44 46 50 51 53 55 57 60 74 77
26      05/26/2006,01 03 11 16 17 19 22 24 25 28 29 37 46 47 53 54 65 75 77 78
27      05/25/2006,02 03 07 14 15 23 29 32 34 35 38 40 42 44 63 65 69 72 74 75
28      05/24/2006,02 04 06 12 16 23 32 34 36 37 45 50 54 58 66 69 70 72 77 78
29      05/23/2006,06 10 11 14 18 19 26 29 31 32 34 35 38 54 57 58 63 64 69 80
30      05/22/2006,04 06 08 12 14 19 21 23 31 32 35 37 38 43 47 57 58 64 74 77
31      05/21/2006,03 08 11 12 13 15 18 21 23 25 26 33 49 50 53 57 59 73 75 76
32      05/20/2006,03 05 07 09 10 11 13 15 17 20 29 40 48 49 50 60 63 67 71 78
33      05/19/2006,03 07 09 13 14 19 27 35 40 42 46 47 57 59 62 71 73 74 75 80
34      05/18/2006,01 11 14 15 18 22 23 28 29 30 31 32 33 37 38 45 52 66 67 79
35      05/17/2006,09 10 11 15 23 28 29 30 34 47 49 50 55 59 63 66 68 69 74 75
36      05/16/2006,01 13 16 23 31 43 50 51 52 53 61 65 66 67 72 73 75 76 77 78
37      05/15/2006,01 04 09 10 13 15 18 21 24 30 33 41 42 44 55 57 64 65 77 79
38      05/14/2006,09 11 14 22 23 24 29 31 38 39 41 42 43 44 46 50 51 53 64 69
39      05/13/2006,02 04 07 10 22 32 34 36 40 42 49 50 56 57 59 61 66 67 77 79
40      05/12/2006,10 13 17 21 25 26 27 29 32 34 36 37 51 54 62 66 69 70 73 74
41      05/11/2006,02 10 14 15 16 19 26 30 32 36 38 41 43 47 54 64 65 72 73 78
42      05/10/2006,03 07 08 19 20 22 26 30 41 42 45 46 53 56 59 60 63 65 67 73
43      05/09/2006,02 05 08 10 16 18 19 30 31 33 38 40 50 51 57 58 65 69 73 74
44      05/08/2006,01 03 05 07 18 23 25 30 40 41 43 44 48 49 51 59 64 65 66 72
45      05/07/2006,13 15 17 27 28 33 35 38 41 46 47 51 57 59 64 65 70 73 76 79
46      05/06/2006,06 08 10 22 23 25 28 36 38 39 42 46 61 63 65 68 72 75 76 79
47      05/05/2006,05 11 13 15 16 22 25 30 35 39 42 54 57 59 60 72 74 78 79 80
48      05/04/2006,02 06 09 10 12 25 27 30 31 33 34 36 41 50 58 61 63 67 71 79
49      05/03/2006,05 06 08 09 11 12 16 17 18 24 27 28 29 42 43 59 60 63 71 79
50      05/02/2006,06 11 18 29 34 39 43 46 48 49 52 56 57 61 68 70 71 75 76 77
51      05/01/2006,05 06 08 11 14 17 18 20 28 30 32 42 50 51 55 57 58 60 69 73
52      04/30/2006,02 07 09 16 17 22 26 27 29 31 34 36 37 41 43 45 52 55 61 66
53      04/29/2006,03 04 12 17 22 27 28 29 32 37 43 49 56 60 62 63 69 70 77 78
54      04/28/2006,05 07 08 16 17 18 28 35 42 44 46 48 49 50 52 53 56 69 71 74
55      04/27/2006,02 03 11 17 22 27 33 35 39 41 42 45 52 57 60 63 66 73 74 78
56      04/26/2006,02 03 06 07 13 14 17 19 29 42 44 46 49 52 60 65 66 67 69 80
57      04/25/2006,01 03 04 07 08 09 16 25 28 32 37 39 44 47 55 57 64 78 79 80
58      04/24/2006,01 03 08 09 10 14 15 19 21 25 32 44 50 52 61 62 72 74 75 76
59      04/23/2006,08 13 15 16 21 26 27 39 42 45 46 48 50 54 57 58 59 67 76 78
60      04/22/2006,12 19 22 29 33 35 39 42 49 50 53 57 61 63 67 68 72 73 78 79
61      04/21/2006,02 03 08 11 19 20 21 35 37 41 42 44 53 57 61 67 72 74 75 78
62      04/20/2006,01 05 17 19 24 26 28 29 31 35 37 40 47 50 52 57 61 62 65 76
63      04/19/2006,04 05 13 20 22 25 28 29 30 37 41 43 44 46 51 53 66 67 68 79
64      04/18/2006,10 19 20 23 24 28 35 41 47 52 55 57 59 61 63 69 74 76 77 79
65      04/17/2006,01 04 13 14 15 17 18 22 26 31 37 38 39 40 51 66 67 73 76 80
66      04/16/2006,05 07 08 11 14 32 34 39 48 50 52 53 54 55 61 63 64 66 68 71
67      04/15/2006,04 05 06 07 09 11 14 26 27 32 37 47 53 56 63 65 67 74 78 79
68      04/14/2006,08 10 12 14 19 24 33 40 41 43 44 48 55 63 64 65 70 71 73 79
```

```
69      04/13/2006,01 02 08 11 12 15 16 17 22 23 32 42 47 49 51 55 58 64 74 78
70      04/12/2006,01 09 13 14 18 20 23 26 30 38 41 44 45 48 51 56 58 63 69 77
71      04/11/2006,04 05 07 09 13 17 30 31 32 36 44 46 50 51 54 56 60 63 67 79
72      04/10/2006,02 05 13 23 24 25 28 31 32 33 38 39 46 47 49 52 66 72 74 78
73      04/09/2006,07 10 14 20 24 33 34 35 40 43 44 48 51 52 57 60 62 66 67 69
74      04/08/2006,04 13 17 19 21 36 37 38 41 42 43 47 51 58 67 68 70 72 77 80
75      04/07/2006,02 04 13 16 19 27 33 40 41 46 47 48 50 55 57 63 65 66 67 72
76      04/06/2006,06 07 08 12 17 18 31 37 39 41 53 62 63 65 66 67 70 73 74 76
77      04/05/2006,11 12 18 22 27 29 30 35 37 39 40 41 45 48 51 56 57 59 72 80
78      04/04/2006,05 07 12 13 15 19 25 29 31 32 33 35 36 49 54 56 59 63 76 77
79      04/03/2006,06 09 15 16 17 21 33 35 38 39 40 44 47 49 51 56 57 58 72 74
80      04/02/2006,03 07 09 12 16 17 18 26 27 30 35 39 47 49 61 64 66 67 72 76
81      04/01/2006,02 05 10 12 13 14 21 31 40 41 45 50 51 60 62 65 66 68 74 76
82      03/31/2006,01 06 11 16 19 22 26 32 36 40 41 42 44 48 51 57 67 71 72 79
83      03/30/2006,04 06 09 13 16 19 22 37 42 46 47 48 49 50 55 58 66 72 77 80
84      03/29/2006,05 09 12 15 23 28 29 32 35 37 41 42 50 55 57 66 68 71 73 78
85      03/28/2006,01 04 13 18 19 29 31 32 34 35 36 38 42 48 52 56 60 61 66 76
86      03/27/2006,05 06 08 14 16 22 27 32 36 44 45 50 51 62 69 70 74 76 77 79
87      03/26/2006,04 12 16 17 21 32 33 37 38 43 46 47 56 58 59 60 67 74 76 79
88      03/25/2006,07 12 16 19 23 25 26 29 31 32 39 43 46 50 53 61 64 71 73 75
89      03/24/2006,06 07 08 13 17 22 25 26 27 36 46 57 58 59 63 67 72 74 79 80
90      03/23/2006,01 06 11 12 18 25 28 32 34 35 37 38 41 44 45 48 51 57 73 78
91      03/22/2006,08 11 13 15 16 22 28 30 31 32 38 41 44 47 50 51 59 61 63 65
92      03/21/2006,01 11 15 31 36 37 40 41 42 51 53 55 58 62 65 66 68 69 72 76
93      03/20/2006,07 11 15 16 19 28 30 31 37 39 46 50 64 65 67 70 72 75 76 77
94      03/19/2006,05 07 08 09 17 18 25 29 30 42 46 47 49 55 58 71 77 78 79 80
95      03/18/2006,01 04 05 07 11 15 17 20 28 31 32 37 47 54 55 56 60 61 75 80
96      03/17/2006,03 07 08 12 17 18 19 20 24 26 27 35 40 47 52 53 54 65 69 72
97      03/16/2006,01 04 05 11 16 31 35 42 43 51 52 53 60 61 63 64 67 71 73 78
98      03/15/2006,09 10 14 15 16 19 29 34 35 39 43 50 52 54 58 63 65 68 71 79
99      03/14/2006,05 06 08 10 22 23 39 42 44 50 52 54 61 64 65 66 71 73 75 80
00      03/13/2006,07 09 10 11 15 19 26 28 30 33 36 41 46 55 57 59 65 71 78 80
01      03/12/2006,07 12 17 26 28 29 30 31 34 36 40 44 54 55 58 61 63 67 68 74
02      03/11/2006,07 09 12 17 33 35 37 38 39 40 51 53 54 56 58 59 61 69 74 77
03      03/10/2006,05 13 21 24 27 33 36 39 42 44 49 53 58 59 61 65 67 69 79 80
04      03/09/2006,08 09 11 14 17 28 29 30 34 36 38 41 44 46 48 58 60 63 69 75
05      03/08/2006,02 04 11 16 27 28 29 32 35 50 57 60 65 66 69 70 75 77 78 79
06      03/07/2006,09 10 20 21 25 26 30 32 34 38 43 47 51 58 61 64 65 67 75 76
07      03/06/2006,03 06 21 23 30 33 34 35 36 39 40 44 46 62 64 65 69 70 72 79
08      03/05/2006,01 04 08 11 19 23 26 32 34 39 43 52 54 59 60 63 66 67 73 77
09      03/04/2006,02 07 13 14 19 26 28 40 43 50 53 58 60 61 64 65 67 71 79 80
10      03/03/2006,05 07 10 11 15 18 20 27 28 30 31 45 49 53 54 58 62 76 79 80
11      03/02/2006,06 07 11 12 13 14 27 28 33 36 47 51 56 57 67 69 72 78 79 80
12      03/01/2006,01 06 07 08 12 13 18 28 37 39 40 41 43 47 52 56 57 63 76 80
13      02/28/2006,01 08 11 17 26 28 34 38 42 43 45 46 48 52 53 54 58 59 60 79
14      02/27/2006,03 09 21 22 23 24 34 35 37 39 43 45 49 53 54 57 72 74 75 76
15      02/26/2006,02 05 07 10 19 24 26 27 28 32 34 38 40 45 47 48 49 57 63 79
16      02/25/2006,05 11 12 24 30 36 38 43 46 47 49 62 64 65 66 68 70 71 76 79
17      02/24/2006,06 08 10 15 25 26 30 33 35 36 38 39 57 59 60 62 71 72 74 78
18      02/23/2006,01 02 03 15 17 19 20 41 46 48 50 58 59 60 61 65 69 70 72 73
19      02/22/2006,06 08 12 17 19 25 29 36 39 41 43 46 50 51 58 59 71 74 77 78
20      02/21/2006,08 11 13 20 30 31 32 38 40 41 44 48 49 54 59 60 63 67 70 78
21      02/20/2006,01 15 16 17 18 27 29 30 39 41 42 48 50 52 63 66 68 70 74 80
22      02/19/2006,04 13 15 20 23 25 28 33 51 52 60 64 66 67 70 71 72 74 76 77
```

```
02/18/2006,01 15 16 19 27 29 38 40 42 43 44 50 51 56 63 65 70 71 72 79
02/17/2006,04 06 17 20 29 32 33 38 47 49 52 54 55 56 63 69 72 73 77 80
02/16/2006,07 14 22 28 30 32 37 50 52 56 58 61 63 68 71 72 74 77 78 80
02/15/2006,01 03 04 08 13 14 24 33 34 44 47 49 56 57 59 66 72 73 75 76
02/14/2006,02 04 10 12 14 17 22 24 26 30 37 45 50 52 55 58 66 70 71 79
02/13/2006,04 06 09 10 14 18 26 31 34 38 41 53 58 61 62 63 64 69 71 79
02/12/2006,02 03 05 06 22 30 31 32 35 41 50 51 52 56 59 61 69 74 76 79
02/11/2006,01 03 11 12 17 27 31 34 39 46 47 49 54 57 58 62 73 74 75 76
02/10/2006,04 06 10 13 34 35 40 41 44 46 48 50 52 54 63 64 69 76 77 80
02/09/2006,07 08 13 15 16 17 21 23 28 33 36 41 48 53 57 58 61 68 69 79
02/08/2006,01 06 12 14 17 26 28 35 37 40 43 44 46 50 56 58 61 62 63 64
02/07/2006,01 04 05 06 14 16 26 31 32 40 49 50 53 60 65 66 68 75 79 80
02/06/2006,01 02 06 10 17 20 21 27 30 36 38 39 42 52 53 55 56 66 72 75
02/05/2006,02 17 21 22 23 28 30 39 41 42 46 51 53 58 60 68 72 74 75 80
02/04/2006,01 03 10 14 19 20 23 25 26 28 32 34 36 40 47 55 57 62 68 77
02/03/2006,01 02 03 08 23 24 28 30 35 40 43 48 49 51 57 59 60 65 73 79
02/02/2006,12 15 23 24 27 28 32 33 44 45 51 52 53 60 61 62 69 74 78 80
02/01/2006,03 06 08 17 22 24 30 33 35 40 41 43 48 53 55 61 62 64 73 76
01/31/2006,02 03 10 15 20 25 28 32 36 41 43 51 60 61 63 70 71 72 78 79
01/30/2006,01 11 12 18 20 21 24 32 34 35 37 38 40 42 44 62 70 74 78 80
01/29/2006,11 13 20 21 22 23 25 28 29 36 41 43 49 55 59 65 74 75 76 77
01/28/2006,03 06 07 09 10 15 16 21 24 31 32 35 36 37 46 54 64 65 78 79
01/27/2006,02 05 09 10 11 28 30 31 32 35 41 44 51 52 55 57 60 67 70 74
01/26/2006,11 13 14 18 21 23 33 38 45 49 50 56 57 60 61 63 64 71 74 79
01/25/2006,05 08 09 34 35 39 42 45 46 47 51 52 54 55 62 67 69 71 74 80
01/24/2006,04 05 09 12 21 23 25 29 42 43 44 47 51 53 54 56 58 59 67 74
01/23/2006,05 06 08 13 17 18 20 27 30 31 35 39 42 51 58 61 62 68 76 79
01/22/2006,01 03 10 16 17 18 24 25 27 39 40 54 57 60 61 62 64 71 75 79
01/21/2006,01 03 04 05 19 21 28 29 40 41 47 53 54 56 57 63 64 71 74 78
01/20/2006,01 02 07 11 12 13 16 17 23 27 30 32 34 38 41 42 67 73 75 76
01/19/2006,01 02 05 11 15 18 20 21 25 32 33 39 43 46 49 50 52 54 63 69
01/18/2006,02 03 11 21 27 31 32 35 39 42 43 45 47 55 67 73 74 75 76 77
01/17/2006,03 04 08 09 10 11 18 22 33 41 50 51 54 56 59 61 65 67 71 77
01/16/2006,04 06 22 25 28 30 34 37 38 47 48 49 51 52 58 70 72 73 76 80
01/15/2006,02 06 07 09 11 12 15 20 23 26 40 41 42 50 55 56 64 65 69 77
01/14/2006,03 08 10 12 14 19 25 26 28 29 35 36 48 52 53 61 66 69 75 78
01/13/2006,01 05 10 26 32 34 35 38 40 41 47 48 54 56 60 61 64 69 70 77
01/12/2006,14 15 18 19 22 23 24 27 30 38 44 53 55 60 64 70 74 77 78 79
01/11/2006,01 03 06 07 14 17 18 19 26 31 32 46 52 56 57 58 67 68 69 75
01/10/2006,02 05 08 11 13 29 30 37 39 40 46 52 57 62 67 68 71 73 75 76
01/09/2006,05 07 13 19 25 29 31 34 38 39 40 44 48 49 55 64 69 70 75 79
01/08/2006,04 09 10 13 21 25 26 32 38 39 44 47 49 54 57 63 72 77 78 79
01/07/2006,07 08 10 21 22 24 27 29 30 31 43 44 46 54 57 59 62 74 75 76
01/06/2006,05 09 14 18 25 35 38 40 43 45 49 53 55 57 68 71 73 74 75 79
01/05/2006,07 09 13 16 24 31 32 38 39 43 44 45 47 50 52 60 64 73 75 77
01/04/2006,03 05 11 16 27 29 31 32 40 43 44 49 50 51 53 57 59 60 63 77
01/03/2006,02 04 07 09 13 25 26 27 30 32 39 40 44 46 60 65 68 69 72 74
01/02/2006,03 04 16 17 18 24 28 29 33 36 37 38 40 44 49 62 68 74 76 78
01/01/2006,01 07 09 10 13 22 23 30 40 44 49 50 52 53 54 55 58 63 65 68
12/31/2005,02 03 07 15 19 22 42 44 48 49 51 55 57 58 60 61 65 66 71 73
12/30/2005,08 13 14 19 21 22 24 25 28 38 44 45 55 57 60 63 67 72 74 77
12/29/2005,02 06 10 13 21 24 25 26 27 28 29 38 48 51 58 60 64 67 73 80
12/28/2005,01 02 06 07 12 24 28 30 36 39 42 47 56 59 67 69 70 71 72 79
12/27/2005,03 08 12 15 18 20 21 23 24 32 37 39 41 43 44 47 72 74 75 76
```

```
77      12/26/2005,11 12 19 21 22 23 25 26 33 36 39 40 42 51 55 56 60 61 62 71
78      12/24/2005,08 11 17 19 20 24 25 26 30 33 39 48 49 58 62 65 69 75 79 80
79      12/23/2005,11 14 21 26 32 33 34 35 36 39 49 50 51 53 55 59 66 72 73 80
80      12/22/2005,04 05 18 25 27 31 33 35 36 43 51 53 58 59 62 72 74 77 78 80
81      12/21/2005,09 18 26 28 29 31 34 36 37 39 44 49 52 53 62 63 64 65 77 80
82      12/20/2005,05 08 10 11 19 22 28 30 32 40 45 47 50 51 53 59 61 64 72 80
83      12/19/2005,02 03 15 18 20 21 28 29 39 42 43 50 51 61 63 68 74 75 78 80
84      12/18/2005,04 05 09 20 22 29 32 33 34 41 43 45 49 57 61 62 72 74 76 78
85      12/17/2005,02 04 06 12 14 16 17 19 21 25 26 27 37 38 52 53 54 56 73 75
86      12/16/2005,03 07 11 12 14 17 18 21 23 24 29 31 38 42 45 56 57 63 70 73
87      12/15/2005,01 09 10 15 20 24 26 27 30 39 43 47 55 57 60 62 64 65 67 69
88      12/14/2005,03 19 20 23 31 32 34 38 41 43 46 47 56 60 62 64 71 72 74 77
89      12/13/2005,04 11 14 22 24 30 40 41 42 45 46 48 60 61 62 65 70 71 73 78
90      12/12/2005,04 07 14 15 17 21 23 30 33 37 46 53 54 57 60 62 66 68 74 80
91      12/11/2005,02 05 07 10 17 28 37 39 42 44 46 57 58 61 62 64 68 69 77 80
92      12/10/2005,06 08 20 22 28 30 37 40 44 45 46 47 52 54 56 57 58 67 71 75
93      12/09/2005,02 05 06 08 13 15 23 25 29 33 50 52 54 58 60 61 66 68 70 73
94      12/08/2005,01 04 05 11 17 18 34 37 38 39 45 48 57 58 62 65 69 75 76 77
95      12/07/2005,04 06 17 20 21 24 28 31 32 34 38 40 41 42 51 55 57 73 74 80
96      12/06/2005,03 19 20 22 28 29 30 45 48 50 51 52 56 59 61 68 72 74 77 79
97      12/05/2005,04 05 10 12 13 14 15 17 23 27 31 35 41 53 57 59 62 63 69 80
98      12/04/2005,08 10 25 32 39 42 57 58 60 65 66 67 68 69 73 74 75 77 78 79
99      12/03/2005,06 07 08 10 14 18 23 24 30 39 45 49 50 51 58 62 66 69 76 79
00      12/02/2005,03 09 17 18 22 24 31 34 36 42 44 49 53 56 58 66 68 69 71 78
01      12/01/2005,04 09 14 18 21 24 28 32 33 44 46 48 49 51 58 60 61 67 73 78
02      11/30/2005,02 15 23 24 25 28 30 34 37 43 46 49 53 55 60 63 67 72 73 79
03      11/29/2005,02 04 08 16 21 22 24 26 31 34 35 36 37 44 46 47 53 61 77 78
04      11/28/2005,05 09 14 17 19 29 32 33 37 39 51 54 56 58 64 65 70 71 72 80
05      11/27/2005,06 08 11 16 24 26 28 30 32 43 44 45 47 49 56 57 60 68 74 78
06      11/26/2005,01 05 06 08 13 22 26 31 35 43 44 47 50 62 63 68 69 75 77 80
07      11/25/2005,07 15 18 21 26 27 28 33 43 46 47 51 52 53 59 62 66 69 73 77
08      11/24/2005,04 06 08 20 27 30 32 33 40 41 42 44 56 58 60 70 72 73 75 78
09      11/23/2005,05 09 23 24 25 29 32 35 40 42 46 49 53 56 58 60 66 76 78 79
10      11/22/2005,04 11 18 21 24 29 33 35 38 41 44 50 54 55 56 60 61 70 71 76
11      11/21/2005,05 14 16 19 21 23 24 28 32 34 42 47 48 50 62 64 66 67 72 78
12      11/20/2005,01 02 06 12 20 23 33 35 36 39 40 42 45 49 51 57 64 69 73 78
13      11/19/2005,01 08 10 17 18 19 21 22 25 26 27 29 48 51 60 64 66 70 76 77
14      11/18/2005,01 07 08 18 20 21 23 26 29 30 42 43 52 54 55 56 59 68 71 73
15      11/17/2005,08 12 16 18 21 32 33 35 36 37 41 43 46 56 59 64 65 68 73 74
16      11/16/2005,02 05 06 07 15 17 19 22 27 28 29 33 35 38 39 51 57 65 66 74
17      11/15/2005,06 11 14 20 22 25 28 32 34 35 38 45 46 55 56 61 65 67 73 74
18      11/14/2005,01 02 05 06 07 08 13 40 51 52 59 60 61 62 63 65 72 73 76 80
19      11/13/2005,09 10 18 22 32 33 37 38 40 44 46 47 49 50 52 58 59 68 76 77
20      11/12/2005,10 19 35 36 41 44 45 46 50 51 54 55 57 59 60 61 65 67 72 80
21      11/11/2005,01 05 09 14 15 21 26 28 30 31 41 46 48 51 56 59 64 68 69 78
22      11/10/2005,09 16 25 26 29 36 37 41 43 44 46 48 49 51 63 65 68 69 71 77
23      11/09/2005,02 04 07 11 15 16 24 27 34 35 41 49 52 57 58 60 66 68 70 71
24      11/08/2005,08 10 13 15 19 20 21 23 27 33 37 42 50 55 56 71 73 76 77 78
25      11/07/2005,03 08 12 13 16 17 21 27 28 31 43 44 47 55 57 59 63 64 67 78
26      11/06/2005,07 09 10 20 24 41 43 45 47 49 53 54 57 64 67 69 70 77 79 80
27      11/05/2005,02 03 05 12 16 24 29 36 40 43 46 47 50 57 61 62 70 76 79 80
28      11/04/2005,01 02 10 17 20 23 26 27 33 39 40 42 58 59 62 66 68 74 77 80
29      11/03/2005,01 03 04 22 24 28 33 37 42 45 48 49 51 52 57 65 67 73 77 78
30      11/02/2005,04 05 14 20 25 26 27 30 32 34 35 39 42 43 56 60 61 67 70 72
```

```
11/01/2005,09 17 18 27 31 32 35 37 38 43 46 49 52 58 59 62 66 71 74 78
10/31/2005,02 08 16 18 26 31 32 42 49 51 54 57 61 63 69 70 73 74 76 80
10/30/2005,16 18 19 21 22 24 30 37 43 51 54 57 63 65 68 70 71 74 77 80
10/29/2005,07 12 13 14 16 18 21 24 35 36 40 49 52 58 59 70 71 73 75 78
10/28/2005,04 12 15 26 34 41 43 45 47 53 55 62 65 68 70 71 74 75 78 79
10/27/2005,04 08 09 10 12 14 19 31 33 36 40 45 47 59 68 69 70 77 78 80
10/26/2005,04 11 13 21 22 23 24 25 26 29 38 45 48 53 55 61 69 75 77 80
10/25/2005,01 07 11 16 22 26 31 35 37 46 48 50 56 60 62 64 65 72 77 79
10/24/2005,03 04 11 12 17 26 28 29 35 40 42 46 48 53 60 62 63 68 73 76
10/23/2005,04 05 10 11 13 16 17 18 20 26 43 46 49 52 59 64 65 72 77 78
10/22/2005,01 06 07 12 19 28 31 32 33 36 39 41 46 49 52 60 67 76 77 78
10/21/2005,01 02 05 07 09 11 12 19 22 25 27 33 37 39 40 49 51 52 65 66
10/20/2005,01 08 11 13 15 18 19 31 35 44 49 55 56 58 67 69 70 71 77 79
10/19/2005,02 03 06 07 14 16 23 24 25 26 33 42 50 51 53 57 63 68 79 80
10/18/2005,04 05 07 11 14 18 20 25 34 38 39 44 49 51 62 65 66 69 70 77
10/17/2005,06 12 14 18 20 23 26 37 41 42 46 48 57 64 66 68 70 72 73 78
10/16/2005,04 07 08 12 14 17 20 24 28 29 30 31 33 37 38 50 56 58 73 80
10/15/2005,08 11 19 26 29 30 31 34 36 45 48 60 66 69 73 74 75 76 77 79
10/14/2005,01 02 07 12 20 26 32 33 36 39 41 45 52 55 56 63 67 73 76 78
10/13/2005,01 08 12 19 21 23 25 26 28 36 37 39 56 57 60 61 62 73 74 76
10/12/2005,02 18 21 24 27 29 34 37 38 39 52 54 62 65 66 72 76 77 79 80
10/11/2005,05 06 08 11 22 24 28 30 31 34 40 41 53 61 62 66 68 73 75 80
10/10/2005,05 18 20 21 33 36 37 45 48 50 54 60 63 64 66 67 69 74 76 79
10/09/2005,02 12 14 18 26 32 34 42 43 45 48 51 53 61 62 67 70 74 75 77
10/08/2005,03 05 06 07 13 14 19 20 22 34 39 41 54 58 62 64 68 69 74 79
10/07/2005,01 11 20 23 24 27 29 30 34 43 50 51 52 54 55 58 59 61 62 77
10/06/2005,03 05 06 08 13 20 25 35 43 48 58 61 65 66 68 70 73 75 79 80
10/05/2005,01 02 04 06 11 12 16 17 30 38 46 49 50 54 59 64 67 70 75 80
10/04/2005,05 06 11 13 19 20 21 33 37 38 39 49 50 51 54 57 59 61 70 78
10/03/2005,08 09 10 12 13 14 17 18 32 34 37 47 49 55 56 60 61 66 72 77
10/02/2005,01 05 10 14 18 19 25 27 30 39 43 49 52 58 59 62 63 67 69 76
10/01/2005,02 09 10 12 20 24 27 32 36 37 41 44 49 53 56 67 68 70 76 80
09/30/2005,06 10 12 14 20 23 28 30 33 34 39 40 48 49 54 60 63 64 77 80
09/29/2005,09 11 14 22 26 30 34 37 42 45 51 56 59 61 62 63 66 75 79 80
09/28/2005,05 12 15 17 27 29 30 34 43 44 46 52 54 57 58 62 63 68 74 76
09/27/2005,07 11 17 20 23 26 27 28 31 37 38 43 49 50 51 53 57 63 67 79
09/26/2005,02 05 08 09 13 16 20 30 33 35 43 44 47 49 51 58 64 71 73 75
09/25/2005,02 07 09 10 26 27 33 35 37 38 40 43 45 48 53 54 62 70 73 79
09/24/2005,03 08 10 11 12 19 26 30 36 37 44 49 52 53 58 61 65 69 70 77
09/23/2005,04 08 14 17 18 25 26 27 32 39 40 41 50 51 52 56 58 59 64 68
09/22/2005,04 19 30 33 36 40 41 43 46 47 50 51 53 58 63 64 65 71 75 80
09/21/2005,04 05 08 10 11 16 21 24 35 41 44 46 50 55 64 67 68 73 74 78
09/20/2005,02 04 07 09 13 27 29 36 37 39 40 44 48 55 58 61 65 69 71 74
09/19/2005,02 10 12 18 19 21 22 25 34 35 40 41 42 44 48 52 54 56 62 63
09/18/2005,02 19 24 26 27 29 30 33 38 45 47 52 62 64 67 72 74 75 79 80
09/17/2005,04 05 06 07 10 12 14 19 30 34 35 39 41 47 48 51 60 66 67 78
09/16/2005,04 05 13 16 24 32 33 34 35 41 43 53 54 61 62 63 64 66 71 79
09/15/2005,01 03 14 19 26 28 29 34 37 40 44 50 57 58 64 68 73 75 76 79
09/14/2005,01 06 11 14 18 22 26 29 32 36 37 42 44 46 56 58 61 63 67 79
09/13/2005,10 13 16 20 22 29 30 33 34 36 38 39 43 44 52 58 59 62 64 65
09/12/2005,02 04 09 10 15 20 22 26 29 33 41 42 45 48 55 58 62 63 68 79
09/11/2005,09 11 13 19 20 23 28 34 42 43 45 49 50 51 52 55 58 69 76 77
09/10/2005,04 05 06 08 10 15 20 22 23 26 36 40 41 43 44 45 49 53 64 76
09/09/2005,09 16 21 23 26 37 38 39 41 46 47 57 58 60 62 67 68 69 75 76
```

```
09/08/2005,07 15 17 18 21 25 30 39 47 50 51 52 53 54 55 60 62 68 70 72
09/07/2005,03 04 12 16 17 21 24 30 46 52 54 55 57 60 65 68 70 75 78 80
09/06/2005,02 04 12 16 18 21 22 26 33 34 36 42 53 57 59 62 63 64 71 72
09/05/2005,03 11 13 24 29 30 33 35 40 56 57 62 63 66 72 74 75 77 78 79
09/04/2005,05 09 12 13 16 17 22 25 30 31 32 33 36 37 38 41 47 72 77 79
09/03/2005,02 03 04 05 09 10 12 21 22 24 28 36 40 43 47 50 58 59 62 70
09/02/2005,07 10 17 18 20 22 25 28 30 31 33 38 46 47 49 51 67 72 73 75
09/01/2005,04 06 07 09 12 13 14 19 26 28 33 44 46 48 50 52 53 55 62 66
08/31/2005,09 15 20 21 27 28 31 32 38 41 49 52 56 60 61 63 67 69 75 79
08/30/2005,07 08 12 23 30 37 38 45 47 48 51 52 54 62 64 67 68 69 74 80
08/29/2005,02 03 12 13 20 23 25 30 32 45 46 48 55 56 62 64 65 72 73 76
08/28/2005,03 12 14 22 23 25 27 28 30 32 35 43 45 58 59 68 69 70 76 79
08/27/2005,06 13 22 23 24 25 26 28 29 41 42 46 51 53 57 58 65 66 70 79
08/26/2005,08 10 16 17 21 25 31 34 37 38 41 43 45 47 49 51 55 61 76 80
08/25/2005,03 07 10 19 20 33 35 37 38 46 55 57 62 63 64 65 67 70 71 77
08/24/2005,01 10 14 18 22 32 34 37 38 41 42 43 47 54 58 66 68 72 74 77
08/23/2005,04 05 06 10 12 16 35 37 45 46 49 51 52 55 57 59 61 74 76 80
08/22/2005,01 04 12 19 22 23 24 27 29 30 32 33 38 39 40 41 51 59 64 76
08/21/2005,07 10 19 23 24 28 36 37 47 59 60 62 63 64 65 68 70 71 74 80
08/20/2005,02 04 07 08 14 15 16 17 27 30 34 47 54 55 58 62 68 77 78 80
08/19/2005,15 21 24 27 28 29 30 31 33 37 40 41 42 44 47 50 57 68 77 78
08/18/2005,03 04 16 19 21 22 24 30 34 38 43 45 48 53 64 68 69 72 75 77
08/17/2005,09 20 23 28 30 31 34 37 39 41 42 45 47 50 51 54 69 71 72 79
08/16/2005,11 13 18 21 26 28 29 32 39 42 48 49 50 55 65 66 67 76 78 80
08/15/2005,03 04 07 11 17 24 25 30 32 38 42 44 51 58 61 68 72 77 78 79
08/14/2005,04 08 09 14 15 18 23 26 29 33 38 44 45 50 52 53 56 63 70 71
08/13/2005,01 04 09 14 18 24 30 31 32 41 46 53 54 56 57 60 62 63 68 71
08/12/2005,03 04 05 10 15 20 22 26 29 33 40 44 47 49 52 53 61 69 71 77
08/11/2005,02 04 05 11 13 14 16 17 18 26 32 34 41 43 57 61 66 73 74 76
08/10/2005,01 02 05 19 22 32 33 35 39 44 50 58 59 60 64 67 68 70 75 80
08/09/2005,05 08 11 13 14 15 21 23 27 33 36 42 43 44 50 56 58 69 73 79
08/08/2005,08 15 16 18 19 20 30 35 38 45 47 52 55 56 58 65 69 73 77 80
08/07/2005,01 02 10 14 19 26 27 34 39 42 43 50 51 56 60 67 74 75 78 80
08/06/2005,03 08 09 12 14 15 19 22 25 28 34 35 56 57 63 70 71 74 75 77
08/05/2005,05 09 18 19 20 24 28 29 31 33 40 52 57 60 66 68 72 77 78 79
08/04/2005,02 04 08 14 15 17 18 25 27 28 29 30 36 46 51 54 59 64 68 73
08/03/2005,09 11 14 15 19 22 26 36 40 49 51 54 57 60 63 65 69 73 77 80
08/02/2005,05 07 12 20 24 28 35 39 41 42 43 45 47 56 65 70 71 72 75 76
08/01/2005,02 16 18 21 22 23 31 32 36 37 38 48 51 53 57 58 67 70 77 78
07/31/2005,01 02 06 08 10 16 18 22 26 34 35 38 39 40 56 62 66 74 78 80
07/30/2005,01 06 08 10 13 17 19 31 33 35 38 44 45 48 52 59 61 68 69 80
07/29/2005,02 04 09 11 13 16 20 25 33 34 43 45 48 50 51 52 54 58 73 79
07/28/2005,01 02 04 08 09 36 39 40 43 48 49 52 53 56 58 63 68 69 75 78
07/27/2005,02 06 08 10 12 20 21 27 28 37 39 40 48 50 58 64 66 67 75 78
07/26/2005,04 06 09 20 22 27 29 31 32 34 38 54 59 61 63 65 69 72 76 80
07/25/2005,01 04 05 08 12 16 20 28 37 47 49 50 55 59 61 64 68 74 78 80
07/24/2005,01 03 06 07 17 18 23 28 33 34 37 41 50 54 55 56 73 76 78 80
07/23/2005,01 05 06 14 15 19 21 24 27 30 35 44 45 46 52 59 74 76 78 79
07/22/2005,05 07 10 11 13 19 21 29 35 51 62 65 66 67 68 70 71 76 77 78
07/21/2005,01 02 05 08 16 17 20 31 33 34 42 46 55 59 60 68 71 74 75 79
07/20/2005,08 09 12 14 20 23 25 28 30 31 34 48 53 56 57 59 63 66 74 77
07/19/2005,05 07 10 12 16 19 25 30 40 42 55 58 59 60 62 63 64 73 78 80
07/18/2005,03 05 06 09 10 11 17 18 27 29 41 50 53 56 57 66 69 70 72 79
07/17/2005,01 02 04 09 14 15 27 30 43 46 48 56 58 62 64 71 72 74 75 79
```

```
07/16/2005,01 02 10 23 30 35 38 41 46 48 53 59 60 62 66 67 70 76 78 79
07/15/2005,02 04 05 10 13 15 16 20 22 27 29 32 34 38 39 55 61 63 73 76
07/14/2005,13 15 18 23 27 36 38 43 46 47 53 55 60 61 65 68 69 74 75 80
07/13/2005,04 06 08 13 25 26 27 29 34 35 41 46 54 55 63 70 73 76 78 79
07/12/2005,02 06 07 16 24 27 29 32 37 38 41 44 46 53 63 65 67 68 75 77
07/11/2005,11 15 19 25 29 30 31 32 35 40 41 43 44 46 51 55 60 74 76 80
07/10/2005,03 11 12 19 23 24 30 31 33 37 42 43 44 55 56 60 61 67 71 76
07/09/2005,08 09 18 21 32 33 34 39 44 45 48 52 63 65 67 71 72 73 74 75
07/08/2005,14 15 17 20 25 27 30 31 32 33 39 47 53 57 67 70 72 73 76 80
07/07/2005,01 05 10 12 16 17 18 20 21 24 32 35 39 44 49 59 60 64 68 70
07/06/2005,04 06 24 26 30 32 33 35 39 43 49 51 58 62 67 72 74 77 78 79
07/05/2005,05 06 07 10 27 29 32 34 37 39 40 44 56 57 58 60 63 64 70 73
07/04/2005,02 10 12 13 23 24 26 27 30 38 44 53 55 56 58 60 65 69 70 76
07/03/2005,05 09 10 13 16 18 32 34 35 37 41 52 58 64 66 69 70 71 74 76
07/02/2005,04 06 07 09 23 30 35 40 45 47 49 51 52 59 62 67 70 74 77 78
07/01/2005,02 04 07 09 17 18 27 29 32 33 34 42 52 63 65 72 76 78 79 80
06/30/2005,13 17 21 24 32 38 40 43 44 52 53 57 59 60 64 69 70 71 72 75
06/29/2005,04 12 14 18 19 20 23 24 27 30 37 38 48 52 55 65 73 74 75 79
06/28/2005,01 02 06 07 13 19 23 25 26 31 34 37 40 43 45 47 63 69 71 77
06/27/2005,05 07 13 29 30 31 33 34 39 46 51 54 55 57 59 60 66 68 69 72
06/26/2005,05 09 13 17 19 20 22 27 33 37 43 46 51 61 63 64 65 68 74 78
06/25/2005,01 02 19 32 33 34 35 39 43 49 50 53 54 58 59 68 71 74 75 80
06/24/2005,01 09 10 12 18 19 20 21 25 26 38 43 45 52 53 56 61 63 75 80
06/23/2005,02 10 11 12 19 26 29 34 36 38 46 52 63 65 69 70 72 74 75 80
06/22/2005,01 03 08 16 17 21 22 32 34 35 41 45 52 55 60 64 72 74 75 79
06/21/2005,05 10 12 17 20 21 23 31 33 42 43 48 52 57 58 63 67 69 77 78
06/20/2005,02 04 06 08 14 25 35 39 50 54 58 60 61 63 67 70 71 73 77 80
06/19/2005,04 07 08 10 12 13 16 17 20 24 27 37 38 40 50 58 61 69 76 79
06/18/2005,02 03 06 07 16 23 26 27 34 36 40 42 49 50 51 59 60 68 78 79
06/17/2005,08 09 12 15 24 25 26 29 30 32 37 39 43 56 57 61 65 67 72 80
06/16/2005,03 05 07 08 09 16 18 21 26 27 28 36 39 40 49 60 63 72 77 78
06/15/2005,04 12 14 18 21 23 30 32 34 36 38 43 44 45 47 48 49 68 73 74
06/14/2005,07 08 12 15 16 19 27 32 34 46 50 52 54 58 59 61 62 63 70 75
06/13/2005,04 07 08 14 15 17 21 23 24 27 34 36 39 42 55 56 57 66 75 78
06/12/2005,08 19 21 27 28 30 31 34 39 42 49 52 61 62 65 72 73 74 75 76
06/11/2005,02 04 12 13 14 15 16 17 18 22 28 32 35 51 58 63 69 70 74 77
06/10/2005,01 06 09 12 13 16 17 19 29 32 34 49 55 58 60 64 69 70 73 80
06/09/2005,03 06 11 13 30 32 34 43 45 48 51 52 56 66 67 70 75 76 77 80
06/08/2005,09 11 20 21 24 26 29 33 39 41 57 58 59 61 65 72 73 74 76 77
06/07/2005,02 05 12 16 19 21 32 36 39 42 44 46 48 52 54 57 59 67 68 78
06/06/2005,01 12 13 15 16 25 29 35 38 41 42 44 53 57 63 66 69 71 73 79
06/05/2005,02 05 09 17 21 22 35 36 37 38 41 49 53 55 59 70 73 75 77 79
06/04/2005,04 16 17 19 20 21 26 34 40 42 43 44 45 47 59 60 64 70 71 75
06/03/2005,04 09 10 11 15 16 21 32 35 36 42 46 48 58 61 63 65 74 75 79
06/02/2005,03 08 09 13 16 17 20 28 35 40 42 43 48 52 59 63 64 68 70 74
06/01/2005,02 11 15 20 29 33 35 45 46 48 49 50 58 62 64 65 66 69 70 74
05/31/2005,01 02 08 12 17 30 37 41 42 45 46 51 58 61 64 65 67 74 75 78
05/30/2005,14 15 17 20 21 22 33 40 43 44 45 48 56 57 64 67 69 73 76 80
05/29/2005,02 04 05 07 14 15 20 29 30 33 40 41 43 54 59 69 72 74 78 80
05/28/2005,07 09 10 13 18 19 21 26 32 42 47 48 54 60 62 64 70 72 75 80
05/27/2005,03 04 05 09 10 11 14 15 20 21 29 35 36 42 53 56 69 77 78 79
05/26/2005,06 10 13 22 23 26 34 39 42 44 48 54 57 60 64 66 73 75 78 80
05/25/2005,01 02 03 06 08 25 27 29 30 36 37 54 63 65 68 69 71 72 77 79
05/24/2005,01 02 06 15 18 23 27 28 31 37 38 43 50 57 58 64 66 68 74 79
```

93 05/23/2005,01 02 03 07 10 17 25 26 31 32 33 38 39 43 47 54 58 72 75 80
94 05/22/2005,04 08 11 13 14 24 26 31 34 35 37 42 50 51 53 63 69 70 73 79
95 05/21/2005,03 20 22 23 25 28 29 34 48 50 55 58 61 65 66 67 69 70 71 73
96 05/20/2005,05 06 07 12 13 15 22 23 34 38 40 45 47 49 50 54 55 62 64 69
97 05/19/2005,06 08 11 18 23 31 34 39 40 50 51 52 54 56 58 59 70 75 76 79
98 05/18/2005,04 05 06 09 11 16 23 25 34 35 39 42 52 55 64 67 75 77 78 79
99 05/17/2005,13 15 18 23 26 28 29 30 31 36 37 39 49 51 61 62 65 71 72 74
00 05/16/2005,04 07 08 22 25 28 30 31 39 50 51 53 59 61 64 67 68 70 73 80
01 05/15/2005,08 09 11 17 19 21 25 28 44 45 46 51 52 58 62 65 72 73 77 80
02 05/14/2005,02 04 11 12 17 21 27 31 32 33 36 39 50 52 53 55 61 65 68 69
03 05/13/2005,01 04 06 09 12 14 23 37 43 45 49 56 57 68 69 70 71 72 73 78
04 05/12/2005,06 08 09 10 15 19 28 29 33 40 43 44 50 58 63 68 71 72 75 79
05 05/11/2005,01 02 14 17 24 26 27 28 29 31 32 36 37 38 43 56 58 59 61 70
06 05/10/2005,03 04 05 09 10 14 15 28 31 35 39 46 52 55 58 66 76 78 79 80
07 05/09/2005,04 06 09 11 12 15 19 23 24 25 27 29 37 50 53 59 65 71 72 79
08 05/08/2005,09 15 18 19 23 25 30 37 43 46 50 57 59 64 65 70 71 75 76 78
09 05/07/2005,01 02 09 20 22 23 28 30 34 35 38 41 47 48 49 50 63 68 69 78
10 05/06/2005,05 06 20 28 29 32 33 37 38 48 54 55 58 60 61 62 65 69 78 80
11 05/05/2005,01 02 16 24 28 31 36 37 39 43 47 51 62 64 69 70 72 73 76 78
12 05/04/2005,02 05 08 18 23 26 28 30 34 36 38 39 42 44 51 53 60 64 68 77
13 05/03/2005,12 13 14 18 19 20 22 23 29 41 45 49 55 57 59 64 66 74 77 79
14 05/02/2005,05 07 08 14 16 18 20 24 28 31 32 38 39 48 54 55 57 75 76 80
15 05/01/2005,09 20 22 26 31 32 35 47 48 51 52 56 58 60 61 63 68 69 72 73
16 04/30/2005,01 06 07 09 10 12 15 18 19 20 23 24 31 39 43 49 53 65 66 67
17 04/29/2005,06 12 14 18 24 27 28 30 37 40 42 48 49 58 66 70 71 75 76 78
18 04/28/2005,03 05 08 14 18 27 34 35 43 54 57 61 62 63 67 70 72 73 75 78
19 04/27/2005,01 08 13 14 15 16 22 26 29 32 38 42 51 52 54 55 72 73 76 79
20 04/26/2005,01 02 05 10 11 12 18 24 25 30 32 39 46 52 54 55 60 62 70 76
21 04/25/2005,07 13 18 29 41 43 49 50 51 60 63 64 65 66 67 71 74 76 77 78
22 04/24/2005,02 03 09 11 12 14 16 29 31 36 40 42 57 58 59 62 64 70 73 76
23 04/23/2005,08 14 15 20 29 31 35 36 39 40 49 54 55 57 59 61 62 69 71 75
24 04/22/2005,05 06 08 09 17 19 20 22 29 35 39 47 50 51 52 57 66 71 73 74
25 04/21/2005,01 04 13 15 19 29 33 34 37 38 39 44 48 63 65 66 71 75 77 80
26 04/20/2005,08 09 11 15 20 23 24 27 30 34 36 44 51 54 60 65 68 69 78 80
27 04/19/2005,02 03 11 15 19 22 24 31 34 38 39 40 45 47 61 64 65 67 69 74
28 04/18/2005,02 11 20 25 29 33 34 37 38 49 52 60 64 66 67 70 73 76 77 78
29 04/17/2005,06 12 13 17 19 20 22 34 40 45 47 49 50 52 54 59 67 75 78 79
30 04/16/2005,02 09 10 15 20 26 28 29 34 36 40 47 51 58 63 66 69 71 72 74
31 04/15/2005,02 16 20 22 25 30 31 32 33 40 43 47 51 56 63 66 75 76 78 79
32 04/14/2005,02 03 04 07 08 23 28 36 37 48 51 53 56 62 63 64 68 69 72 78
33 04/13/2005,06 07 11 13 15 16 21 24 28 34 35 36 44 49 53 56 58 59 62 69
34 04/12/2005,05 09 12 14 23 26 29 30 34 36 39 40 46 47 49 64 65 70 77 78
35 04/11/2005,01 02 04 09 13 15 23 25 30 44 46 47 48 50 60 67 68 69 71 80
36 04/10/2005,01 16 21 25 26 34 39 40 42 49 50 53 55 56 58 60 63 69 78 79
37 04/09/2005,06 09 12 13 14 15 16 18 26 27 33 39 43 45 48 61 70 71 79 80
38 04/08/2005,03 04 06 08 11 20 21 26 32 34 36 41 42 46 49 54 64 70 77 80
39 04/07/2005,02 03 05 06 20 29 33 36 47 51 57 58 60 64 68 73 74 76 77 80
40 04/06/2005,02 07 09 10 13 21 24 28 29 30 36 44 48 54 56 62 63 72 75 78
41 04/05/2005,03 09 15 20 21 24 31 32 47 49 51 54 58 60 63 64 69 72 75 77
42 04/04/2005,02 03 12 13 14 18 24 35 37 39 40 45 56 57 65 68 69 73 75 79
43 04/03/2005,05 07 09 10 11 12 15 25 28 33 35 44 45 47 52 54 58 61 73 77
44 04/02/2005,02 08 09 14 15 17 19 23 24 27 35 39 43 52 55 64 66 74 76 80
45 04/01/2005,06 09 11 12 13 14 23 25 31 41 45 53 57 58 63 67 69 74 78 79
46 03/31/2005,04 06 09 10 13 25 28 35 40 48 51 52 54 56 60 65 68 69 71 78

```
47      03/30/2005,03 05 06 11 15 18 27 32 45 50 53 55 57 59 60 62 68 69 74 79
48      03/29/2005,05 07 12 18 28 31 33 38 42 43 48 51 53 54 57 66 68 72 75 76
49      03/28/2005,01 02 11 12 17 18 21 27 35 37 39 44 57 61 62 64 65 66 68 79
50      03/27/2005,01 07 09 11 13 14 26 27 33 38 39 40 46 47 52 62 65 68 72 79
51      03/26/2005,05 12 14 15 16 26 27 30 31 32 34 35 36 39 48 54 58 64 73 77
52      03/25/2005,01 02 24 25 31 32 33 36 41 47 48 49 53 58 60 61 63 69 75 77
53      03/24/2005,01 02 04 05 08 12 20 23 25 33 35 37 40 52 55 59 68 70 73 79
54      03/23/2005,09 11 14 21 23 27 32 35 39 42 50 51 52 58 60 63 64 65 71 79
55      03/22/2005,05 13 23 28 30 32 41 42 49 51 52 54 56 58 61 64 68 73 76 78
56      03/21/2005,03 05 07 09 10 14 21 22 31 32 39 43 45 48 55 62 75 76 77 80
57      03/20/2005,02 04 09 11 18 22 25 28 32 35 38 40 44 45 47 59 60 62 70 79
58      03/19/2005,07 09 10 11 12 14 25 27 37 38 41 44 45 54 55 57 58 70 78 80
59      03/18/2005,09 11 13 15 16 17 20 27 32 35 37 43 51 52 57 59 63 75 77 78
60      03/17/2005,03 05 08 12 17 18 22 23 25 26 27 35 41 44 46 56 65 72 76 77
61      03/16/2005,03 05 06 16 17 23 33 37 43 45 52 56 61 64 65 68 69 70 75 79
62      03/15/2005,03 11 13 18 23 27 30 35 36 40 44 50 54 56 60 62 66 67 70 78
63      03/14/2005,04 06 18 19 30 32 33 39 41 45 50 51 52 53 56 61 62 65 76 77
64      03/13/2005,04 07 10 13 16 17 19 22 23 28 41 43 49 51 57 60 62 65 68 79
65      03/12/2005,01 07 12 21 25 31 34 37 41 42 58 59 61 62 65 67 71 72 74 76
66      03/11/2005,01 02 03 07 12 14 18 26 30 36 42 45 51 53 55 68 69 71 78 80
67      03/10/2005,01 02 12 13 18 25 30 39 41 42 47 51 55 56 57 59 63 66 69 72
68      03/09/2005,02 03 05 13 20 23 28 29 30 33 39 40 44 46 47 59 65 69 70 79
69      03/08/2005,02 07 11 13 14 16 18 25 26 27 42 46 47 50 52 55 57 59 64 80
70      03/07/2005,04 05 08 10 16 17 21 23 25 29 40 41 43 44 46 51 55 65 75 76
71      03/06/2005,04 09 12 14 22 24 26 29 32 34 38 39 41 42 46 56 67 68 75 80
72      03/05/2005,09 11 14 18 21 26 37 38 40 41 42 45 48 55 56 64 66 73 74 76
73      03/04/2005,03 04 08 23 26 28 33 36 37 39 44 49 51 53 58 61 65 71 78 80
74      03/03/2005,01 02 03 06 10 16 19 25 29 43 47 55 56 58 68 69 70 71 75 80
75      03/02/2005,02 11 12 14 18 19 28 38 45 46 50 51 54 55 56 58 63 67 68 75
76      03/01/2005,02 04 05 06 13 18 22 23 26 29 33 38 44 45 48 54 60 72 75 78
77      02/28/2005,01 03 09 14 19 25 30 34 43 44 45 47 50 51 52 57 59 66 68 74
78      02/27/2005,02 09 30 31 32 33 35 36 37 38 39 41 42 44 45 46 49 51 53 72
79      02/26/2005,03 04 10 12 22 23 24 27 35 40 41 43 51 58 60 64 72 75 76 79
80      02/25/2005,04 07 12 17 19 21 26 27 28 29 34 39 46 48 53 54 55 60 72 76
81      02/24/2005,08 15 21 22 25 26 31 32 39 41 42 44 46 49 51 59 61 62 64 66
82      02/23/2005,02 03 05 06 07 14 22 28 29 30 33 34 39 46 60 61 68 71 73 77
83      02/22/2005,01 06 09 11 19 22 27 28 34 39 47 50 53 56 65 68 69 70 74 78
84      02/21/2005,02 04 08 17 22 27 29 32 33 34 41 43 46 52 58 59 62 68 72 74
85      02/20/2005,07 08 14 17 23 24 30 37 39 41 44 48 52 57 63 65 70 71 76 77
86      02/19/2005,01 03 04 05 08 23 26 27 28 29 35 36 38 43 53 56 60 62 74 76
87      02/18/2005,01 03 16 17 18 19 24 26 28 30 32 34 44 46 48 50 65 74 75 76
88      02/17/2005,01 03 06 07 12 21 22 24 25 27 29 30 32 33 34 41 56 62 79 80
89      02/16/2005,02 08 15 24 25 32 34 35 38 43 46 52 55 56 60 61 68 70 72 79
90      02/15/2005,01 11 14 15 23 26 30 31 37 40 44 47 52 53 55 56 60 62 65 67
91      02/14/2005,01 02 10 11 12 16 23 24 27 41 47 48 51 53 60 63 64 70 71 75
92      02/13/2005,02 04 06 08 16 19 21 24 25 28 43 45 51 52 61 62 65 76 77 78
93      02/12/2005,05 08 10 15 20 21 24 27 28 35 42 43 51 53 54 57 58 60 62 70
94      02/11/2005,11 13 22 23 26 33 36 39 40 41 43 48 54 55 60 69 73 78 79 80
95      02/10/2005,01 08 12 14 16 17 21 28 37 52 56 59 61 63 66 67 68 72 73 78
96      02/09/2005,07 09 12 13 17 22 27 33 36 44 49 50 52 53 54 63 70 77 78 79
97      02/08/2005,02 06 09 16 25 26 31 35 36 37 39 42 46 48 54 63 66 74 75 79
98      02/07/2005,02 07 10 11 12 15 30 31 41 42 56 59 63 65 68 69 76 78 79 80
99      02/06/2005,04 13 19 24 30 32 33 39 41 51 53 55 58 59 60 63 70 73 75 76
00      02/05/2005,01 05 07 12 19 22 33 36 39 41 42 47 53 57 59 66 71 73 77 79
```

01 02/04/2005,04 08 09 13 14 18 25 31 32 33 35 41 46 47 58 59 64 65 74 77
02 02/03/2005,04 06 07 15 18 20 23 28 31 33 34 35 46 50 60 70 72 75 77 78
03 02/02/2005,11 16 17 30 31 32 43 49 51 57 59 60 64 65 67 69 72 75 76 80
04 02/01/2005,05 08 13 14 20 30 33 34 35 37 40 55 59 60 62 66 68 69 74 80
05 01/31/2005,04 08 15 19 22 25 30 31 37 40 41 42 48 51 61 63 65 72 74 79
06 01/30/2005,02 03 04 07 08 16 21 26 31 41 42 46 50 54 57 61 65 66 69 72
07 01/29/2005,01 02 03 04 14 15 21 33 35 39 44 50 52 54 56 61 62 63 79 80
08 01/28/2005,05 11 16 17 18 25 26 28 30 32 36 38 41 46 56 58 61 63 77 78
09 01/27/2005,04 07 16 20 24 27 30 31 42 47 49 52 56 57 58 60 65 66 71 73
10 01/26/2005,05 12 14 16 19 21 29 31 33 37 40 45 50 60 63 73 74 77 79 80
11 01/25/2005,04 09 10 11 13 26 38 43 44 46 48 54 56 57 62 65 68 69 77 80
12 01/24/2005,04 12 13 24 25 27 28 30 36 41 42 50 57 59 63 68 76 77 78 80
13 01/23/2005,01 06 10 19 24 33 35 42 46 47 50 51 53 56 58 59 68 69 72 78
14 01/22/2005,03 04 09 10 20 24 25 27 35 39 42 51 52 54 57 58 62 64 76 80
15 01/21/2005,02 15 17 31 32 33 39 41 44 45 46 54 57 63 64 69 70 75 76 80
16 01/20/2005,06 14 16 18 22 29 31 32 34 40 41 46 50 57 64 65 66 70 71 74
17 01/19/2005,03 04 05 08 10 12 15 19 24 35 37 43 44 52 64 65 70 75 77 78
18 01/18/2005,06 21 22 24 25 36 38 39 40 46 47 48 51 58 60 64 67 70 75 76
19 01/17/2005,02 05 16 17 24 27 33 34 35 37 44 45 52 54 63 67 70 73 74 80
20 01/16/2005,04 05 06 09 11 13 16 18 19 23 26 34 45 50 56 64 68 69 75 78
21 01/15/2005,06 08 13 21 22 25 27 30 35 44 47 48 51 53 54 56 61 69 70 72
22 01/14/2005,06 11 19 20 27 29 35 41 43 47 49 53 59 61 62 63 64 66 77 78
23 01/13/2005,06 16 18 21 25 26 28 32 33 35 36 45 51 52 60 62 66 72 75 77
24 01/12/2005,01 05 09 18 24 27 37 38 40 42 46 48 49 53 54 56 61 62 63 75
25 01/11/2005,01 03 07 14 16 23 29 30 36 42 43 49 51 52 58 59 61 63 72 78
26 01/10/2005,03 08 13 20 22 36 48 50 54 57 59 60 61 62 65 72 74 76 77 79
27 01/09/2005,02 04 05 06 14 21 23 26 39 53 54 57 58 64 65 67 68 72 76 79
28 01/08/2005,01 06 08 15 20 26 28 32 35 42 43 45 46 47 57 58 61 62 77 79
29 01/07/2005,01 03 04 10 19 25 30 35 37 38 42 51 52 53 59 62 67 69 75 80
30 01/06/2005,06 08 12 19 21 23 28 31 37 51 53 56 58 60 63 65 70 71 73 79
31 01/05/2005,04 09 11 16 21 25 28 30 35 39 44 45 54 60 61 63 72 73 78 80
32 01/04/2005,04 06 07 11 14 15 18 22 26 29 37 38 52 55 60 62 64 66 68 80
33 01/03/2005,07 10 11 12 21 22 23 24 42 44 49 51 52 56 57 61 63 64 65 74
34 01/02/2005,02 16 22 25 30 40 43 44 45 46 50 54 56 59 72 73 74 76 79 80
35 01/01/2005,11 13 17 20 21 23 24 25 26 28 30 33 40 46 47 50 52 56 58 72
36 12/31/2004,01 08 15 16 19 20 21 24 32 37 46 53 55 59 60 63 69 74 76 79
37 12/30/2004,04 07 11 16 18 19 21 24 29 37 38 40 41 49 50 51 66 69 72 78
38 12/29/2004,03 08 12 16 19 22 25 27 32 41 42 44 45 56 60 61 66 67 73 76
39 12/28/2004,01 03 05 09 13 19 27 28 35 42 43 47 49 53 58 63 64 66 67 73
40 12/27/2004,04 07 11 16 23 24 29 30 39 41 42 49 57 61 62 64 66 74 78 79
41 12/26/2004,02 05 06 07 16 17 25 36 37 43 45 46 47 54 55 67 68 69 72 75
42 12/24/2004,10 13 14 15 18 21 23 24 32 34 39 40 45 46 53 57 60 63 70 80
43 12/23/2004,07 10 12 14 15 16 23 26 34 43 45 47 52 55 57 58 70 71 74 78
44 12/22/2004,02 04 05 11 16 17 18 19 22 23 26 28 29 35 47 48 57 58 72 76
45 12/21/2004,07 09 13 15 16 18 20 25 30 34 35 38 42 44 51 57 61 67 74 76
46 12/20/2004,01 02 08 13 14 15 24 29 32 33 39 43 44 45 57 60 61 64 69 78
47 12/19/2004,02 03 06 10 16 18 20 24 25 31 36 43 45 49 50 53 58 71 72 76
48 12/18/2004,01 02 07 09 18 20 26 29 36 40 44 51 52 55 58 61 63 65 68 77
49 12/17/2004,03 06 12 19 20 22 33 36 43 44 45 47 49 50 55 57 58 65 68 78
50 12/16/2004,04 09 12 13 14 18 20 21 24 29 36 40 43 45 47 50 52 60 62 65
51 12/15/2004,04 06 09 12 14 22 24 26 32 40 47 52 54 55 57 62 63 67 68 70
52 12/14/2004,02 03 06 07 09 17 24 30 31 34 36 37 42 51 52 53 68 73 76 79
53 12/13/2004,01 03 14 15 16 18 21 35 42 43 49 50 58 60 61 65 68 74 75 78
54 12/12/2004,02 03 12 17 24 26 28 31 34 38 43 44 46 48 51 55 70 73 74 78

12/11/2004,05 12 14 16 21 23 26 41 43 48 49 50 55 60 61 65 67 68 73 78
12/10/2004,01 02 04 08 17 21 27 28 29 31 35 36 39 42 48 66 74 76 77 80
12/09/2004,01 02 03 04 14 15 18 26 29 33 37 38 39 41 47 48 54 67 74 75
12/08/2004,05 12 16 17 18 22 33 35 36 37 40 42 45 46 54 57 59 60 66 76
12/07/2004,02 04 07 22 25 30 31 33 34 35 36 42 48 57 62 64 65 70 73 80
12/06/2004,01 02 07 09 11 13 21 23 24 31 34 37 43 45 46 61 63 64 71 76
12/05/2004,08 09 12 16 20 21 30 32 34 37 41 42 44 49 51 52 62 69 70 77
12/04/2004,03 07 21 26 30 32 37 38 44 47 53 55 57 62 63 67 70 73 74 79
12/03/2004,03 04 10 13 18 24 30 34 35 42 46 51 52 54 63 65 66 69 72 80
12/02/2004,02 08 11 17 18 40 41 42 45 47 48 49 57 60 61 65 66 73 75 79
12/01/2004,03 10 14 16 21 23 26 28 31 36 42 47 50 55 56 57 61 70 71 77
11/30/2004,01 03 06 09 12 18 21 28 30 32 37 40 47 48 50 55 58 63 73 79
11/29/2004,04 08 17 18 20 21 25 29 31 44 46 49 50 51 58 62 70 73 74 79
11/28/2004,01 03 09 14 20 23 25 26 31 32 41 47 52 53 54 66 68 71 72 73
11/27/2004,10 16 17 18 20 27 32 34 40 42 43 47 48 51 53 54 60 61 65 67
11/26/2004,01 06 10 24 28 29 32 33 36 39 42 47 53 54 56 60 64 65 68 72
11/25/2004,02 03 06 10 15 20 21 29 48 50 54 55 56 57 59 61 62 65 71 74
11/24/2004,01 02 11 14 15 18 21 23 33 35 37 38 42 61 63 65 66 69 74 76
11/23/2004,12 13 18 21 25 40 41 43 46 47 50 51 52 53 54 59 62 69 70 79
11/22/2004,06 11 13 16 23 25 28 34 38 39 42 43 46 50 57 64 69 71 72 78
11/21/2004,03 09 11 12 16 17 23 27 30 37 43 47 49 50 62 66 67 68 70 73
11/20/2004,01 05 08 09 11 19 29 30 32 34 38 47 48 49 50 51 52 75 77 79
11/19/2004,01 06 07 12 22 34 38 41 42 44 45 46 48 55 56 62 64 68 72 76
11/18/2004,06 11 12 13 15 17 21 29 32 35 36 50 56 57 62 63 65 69 74 79
11/17/2004,04 14 15 17 20 24 27 30 31 33 34 37 50 51 54 56 66 67 72 79
11/16/2004,01 05 08 11 19 28 31 33 36 45 46 53 54 57 61 64 71 74 76 78
11/15/2004,03 04 05 13 18 22 23 27 29 37 41 43 46 61 66 68 74 77 79 80
11/14/2004,06 09 13 14 18 20 38 42 44 45 50 53 59 62 64 66 67 68 73 78
11/13/2004,02 04 07 10 12 14 15 44 45 48 50 51 53 56 63 68 70 71 74 78
11/12/2004,06 07 10 13 20 25 29 30 31 34 35 40 41 45 49 53 59 72 74 80
11/11/2004,02 07 10 11 14 18 26 45 49 50 53 54 56 57 61 64 70 71 74 77
11/10/2004,07 10 16 17 20 30 38 42 46 48 49 52 53 54 55 62 64 71 75 78
11/09/2004,02 06 12 16 18 19 30 37 38 41 42 43 44 58 65 67 72 73 75 78
11/08/2004,02 05 06 07 13 20 36 42 44 45 49 51 55 56 57 65 67 71 73 75
11/07/2004,02 03 04 06 08 24 26 29 30 37 39 41 46 48 59 63 67 70 71 76
11/06/2004,02 09 10 12 15 20 24 25 41 42 47 51 58 59 60 63 68 70 73 74
11/05/2004,12 15 19 20 21 22 24 32 34 37 43 50 51 54 55 59 67 72 74 78
11/04/2004,04 06 07 10 18 19 20 21 24 25 28 31 33 39 47 48 54 61 67 72
11/03/2004,01 02 08 10 12 13 16 19 21 38 40 42 46 47 48 61 64 66 70 72
11/02/2004,07 12 14 15 17 22 23 25 33 34 39 44 48 50 58 63 65 75 78 80
11/01/2004,02 05 11 16 18 27 28 31 33 40 45 46 49 52 56 64 66 71 75 79
10/31/2004,08 13 15 17 21 23 24 32 36 40 51 54 58 59 64 67 68 69 77 78
10/30/2004,04 05 16 17 24 25 26 34 36 39 44 46 50 63 65 68 69 74 75 78
10/29/2004,07 09 18 27 28 30 33 36 52 57 58 61 63 67 68 69 71 75 76 77
10/28/2004,03 05 06 09 10 11 12 37 39 43 49 50 59 60 62 65 73 78 79 80
10/27/2004,02 04 13 18 20 21 25 28 34 35 36 47 50 54 56 61 65 70 77 80
10/26/2004,03 09 10 11 13 17 25 27 34 44 52 60 62 65 71 73 74 76 77 78
10/25/2004,04 05 09 13 22 23 28 33 34 37 38 45 49 51 53 60 64 70 78 80
10/24/2004,07 09 11 15 18 28 32 37 39 40 50 51 53 54 56 58 68 70 75 76
10/23/2004,06 16 17 18 22 25 26 30 34 39 40 42 50 53 60 62 71 73 79 80
10/22/2004,05 07 08 10 16 17 22 33 37 39 47 59 60 61 64 65 66 67 70 79
10/21/2004,16 19 20 33 37 40 43 44 46 47 49 52 53 64 67 69 72 75 78 80
10/20/2004,07 10 15 18 21 23 24 27 29 31 33 43 44 53 56 57 69 72 77 80
10/19/2004,05 08 11 28 33 35 38 42 43 45 48 52 58 60 61 62 66 70 71 72

09 10/18/2004,03 30 31 32 38 39 44 46 47 49 52 53 57 60 62 63 66 73 76 77
10 10/17/2004,03 10 14 20 21 22 23 36 37 40 42 44 45 55 56 57 62 65 66 72
11 10/16/2004,01 07 09 13 17 22 34 35 43 44 45 48 49 59 61 68 71 73 74 76
12 10/15/2004,05 09 13 15 16 22 24 25 27 34 35 36 40 41 46 62 66 71 73 74
13 10/14/2004,01 03 05 07 16 17 18 21 31 32 35 39 42 52 57 58 60 65 70 75
14 10/13/2004,03 08 14 20 21 25 33 34 35 40 43 46 51 52 55 57 63 64 67 77
15 10/12/2004,03 07 13 22 23 37 39 40 46 47 50 54 57 61 66 67 69 74 78 79
16 10/11/2004,02 06 08 15 18 20 25 26 27 29 34 38 48 54 56 57 66 67 73 79
17 10/10/2004,02 04 05 06 14 19 27 32 38 40 42 43 52 53 58 63 64 73 74 78
18 10/09/2004,05 09 11 12 16 18 21 25 29 35 44 45 49 53 54 61 63 67 69 76
19 10/08/2004,01 03 04 12 15 20 27 29 33 34 39 40 44 49 63 64 67 68 70 78
20 10/07/2004,05 10 15 16 17 21 28 29 30 31 34 37 40 42 45 46 63 64 75 80
21 10/06/2004,01 03 08 10 16 18 24 25 32 35 36 44 55 60 65 66 73 75 78 79
22 10/05/2004,10 14 16 19 24 27 34 35 38 39 44 46 47 49 50 65 68 74 77 80
23 10/04/2004,14 15 16 17 18 21 30 33 35 39 45 54 65 69 72 73 75 77 78 79
24 10/03/2004,01 09 13 14 20 21 28 29 32 36 37 44 45 50 60 62 63 70 74 76
25 10/02/2004,04 05 11 21 22 25 29 30 31 32 42 44 48 49 52 57 60 68 69 72
26 10/01/2004,01 08 10 11 18 23 29 35 37 38 39 43 46 57 59 60 70 72 76 79
27 09/30/2004,03 04 08 12 14 20 22 25 26 28 29 34 36 38 47 50 57 66 67 74
28 09/29/2004,09 12 15 18 19 20 23 26 35 37 38 42 43 51 55 57 61 65 72 78
29 09/28/2004,02 03 09 13 19 23 27 32 36 37 42 45 47 49 51 53 56 59 65 71
30 09/27/2004,05 08 11 13 17 22 25 27 33 36 37 44 46 49 50 54 63 69 70 73
31 09/26/2004,01 02 17 18 22 27 32 35 41 45 47 48 49 53 56 61 64 66 72 76
32 09/25/2004,01 04 09 12 21 24 25 28 30 31 32 33 38 39 46 52 61 63 64 73
33 09/24/2004,03 15 16 21 24 28 29 30 32 37 39 44 45 46 48 50 58 60 68 77
34 09/23/2004,03 04 05 06 10 15 17 25 29 31 32 40 41 48 52 60 61 66 67 79
35 09/22/2004,01 07 15 16 19 20 23 27 28 32 33 41 47 48 52 54 59 63 65 69
36 09/21/2004,01 06 10 11 14 17 20 29 34 38 39 44 51 54 59 62 68 74 78 80
37 09/20/2004,04 05 17 18 19 20 25 32 35 39 58 60 63 65 70 74 75 78 79 80
38 09/19/2004,09 15 24 28 31 38 41 46 47 50 54 55 57 58 64 65 71 72 74 80
39 09/18/2004,03 08 10 14 16 17 21 23 25 36 41 42 49 52 57 59 62 63 64 72
40 09/17/2004,06 07 11 15 17 30 37 40 41 42 46 48 54 55 65 68 70 72 73 76
41 09/16/2004,04 06 14 20 22 24 26 34 39 40 42 43 44 52 53 55 71 73 74 78
42 09/15/2004,04 25 30 31 36 37 39 42 45 48 49 54 55 58 62 63 64 69 79 80
43 09/14/2004,01 05 06 07 08 21 28 29 33 36 37 39 41 52 53 64 70 73 75 77
44 09/13/2004,04 07 08 12 14 15 16 20 49 51 55 56 61 62 64 66 68 70 71 72
45 09/12/2004,05 09 12 19 23 27 28 29 30 45 49 51 52 55 61 72 75 76 77 79
46 09/11/2004,08 09 10 15 24 33 36 40 48 58 60 61 62 63 68 70 74 77 78 80
47 09/10/2004,13 18 20 22 28 29 42 46 51 53 54 56 61 62 64 67 69 70 75 76
48 09/09/2004,02 09 11 15 16 18 22 31 33 42 46 50 51 54 55 57 63 70 75 78
49 09/08/2004,03 06 12 21 24 26 31 33 36 39 46 49 50 60 61 68 74 75 76 77
50 09/07/2004,02 11 12 15 17 19 20 25 28 31 36 40 50 52 58 61 66 68 71 73
51 09/06/2004,02 04 06 07 08 12 17 25 31 36 40 56 57 59 61 63 65 66 74 80
52 09/05/2004,02 04 07 10 15 18 19 27 34 51 58 60 61 63 64 68 70 71 76 80
53 09/04/2004,11 16 17 20 25 27 36 41 42 44 50 54 55 57 61 62 64 66 69 77
54 09/03/2004,02 05 06 08 17 22 25 31 36 42 43 44 48 52 60 64 69 76 79 80
55 09/02/2004,11 18 20 25 29 30 34 36 44 46 48 49 55 62 68 73 74 75 77 79
56 09/01/2004,02 08 12 15 20 24 25 28 30 31 32 40 41 43 46 64 72 75 77 78
57 08/31/2004,01 03 04 11 12 20 21 24 26 32 36 39 40 45 56 63 65 71 79 80
58 08/30/2004,01 12 17 23 24 25 26 28 29 33 41 47 51 52 57 61 62 73 74 78
59 08/29/2004,01 04 08 09 11 17 21 23 24 37 42 49 53 56 57 59 63 66 74 76
60 08/28/2004,01 05 09 11 12 16 18 22 23 25 30 37 39 49 54 56 68 77 78 80
61 08/27/2004,02 07 09 11 12 19 22 25 26 27 37 45 48 49 53 57 62 65 72 74
62 08/26/2004,06 08 12 13 14 21 27 31 33 43 44 49 65 66 67 71 74 75 76 79

```
63      08/25/2004,01 08 11 15 22 23 27 28 33 37 38 45 47 51 55 67 68 71 74 75
64      08/24/2004,02 05 07 09 13 14 16 17 23 24 26 31 33 40 43 52 61 64 66 80
65      08/23/2004,04 06 12 13 19 20 21 25 28 29 31 39 40 45 47 53 60 68 75 78
66      08/22/2004,03 08 11 23 26 27 37 41 44 45 48 50 52 55 62 63 64 66 69 71
67      08/21/2004,01 08 11 13 14 27 29 30 40 46 47 50 53 56 58 61 69 70 74 76
68      08/20/2004,01 02 13 18 28 30 33 34 36 42 50 52 59 62 64 65 66 68 69 77
69      08/19/2004,02 12 15 16 21 30 33 35 38 40 45 50 53 55 59 61 64 67 69 76
70      08/18/2004,03 04 13 17 23 28 38 45 47 53 57 58 61 62 65 68 69 70 75 80
71      08/17/2004,03 08 10 15 17 19 23 27 34 37 43 46 50 54 56 64 72 78 79 80
72      08/16/2004,02 05 09 17 21 27 31 32 34 44 50 52 56 58 67 70 73 76 77 80
73      08/15/2004,02 04 05 06 07 13 15 17 19 20 28 29 35 36 47 49 54 68 70 77
74      08/14/2004,08 09 12 16 24 30 31 34 48 49 50 54 55 59 62 65 66 70 72 77
75      08/13/2004,03 05 09 23 24 25 28 29 30 32 34 40 43 48 51 61 68 69 77 79
76      08/12/2004,02 06 07 08 12 13 24 26 27 28 29 45 46 50 52 60 61 63 66 77
77      08/11/2004,02 07 13 15 21 23 30 32 35 37 40 45 52 59 64 65 66 72 73 78
78      08/10/2004,04 05 06 07 27 30 32 39 40 41 43 44 46 51 54 62 64 65 67 73
79      08/09/2004,06 09 12 17 22 23 27 29 36 39 41 43 46 47 50 56 62 67 69 80
80      08/08/2004,10 11 12 15 16 17 21 22 25 33 34 38 44 50 57 58 64 72 76 80
81      08/07/2004,01 03 11 24 26 29 32 33 34 37 39 44 47 48 51 60 64 69 70 72
82      08/06/2004,02 07 14 18 22 26 27 28 29 30 31 33 35 36 45 51 57 63 65 76
83      08/05/2004,02 10 15 16 17 21 24 25 29 40 44 47 48 49 51 53 58 68 70 78
84      08/04/2004,03 11 12 16 19 24 26 27 33 35 39 42 47 49 52 54 58 59 65 66
85      08/03/2004,02 05 09 10 13 16 17 22 31 36 37 42 45 50 52 53 56 62 63 74
86      08/02/2004,04 10 13 18 25 29 38 42 45 46 56 57 58 59 71 72 73 78 79 80
87      08/01/2004,05 10 11 21 23 24 25 27 28 30 31 37 45 61 63 72 74 76 78 79
88      07/31/2004,03 06 08 16 19 23 28 29 36 42 43 50 53 56 66 69 71 73 74 78
89      07/30/2004,03 05 06 16 20 28 39 45 47 48 50 52 53 56 61 66 69 72 74 79
90      07/29/2004,02 04 09 15 18 21 27 28 39 44 50 58 61 67 69 70 71 74 76 77
91      07/28/2004,02 05 08 18 24 27 33 34 35 39 43 46 48 54 55 58 60 63 66 73
92      07/27/2004,02 12 13 20 22 27 29 30 33 35 44 51 53 56 58 59 60 62 67 79
93      07/26/2004,02 04 05 08 16 18 31 38 42 48 49 50 54 56 63 68 69 70 71 76
94      07/25/2004,02 12 16 19 21 23 30 44 47 48 52 53 55 56 57 63 67 72 74 76
95      07/24/2004,03 06 08 10 16 18 19 22 32 48 50 52 56 57 59 61 63 65 71 79
96      07/23/2004,04 10 11 12 18 19 29 35 36 40 41 43 44 59 62 65 67 69 71 78
97      07/22/2004,02 03 05 09 10 11 18 20 25 30 32 35 36 39 43 53 59 65 69 79
98      07/21/2004,02 03 08 10 13 15 18 21 23 31 38 44 47 48 58 59 62 73 75 77
99      07/20/2004,01 08 17 19 20 24 30 32 38 41 42 43 44 49 54 56 64 65 66 69
00      07/19/2004,13 23 28 32 33 34 36 44 46 48 51 56 57 64 65 66 69 73 75 78
01      07/18/2004,01 02 10 11 13 16 17 20 21 24 25 36 48 49 59 65 66 72 75 76
02      07/17/2004,05 07 09 11 21 25 28 29 31 39 43 56 58 65 69 70 72 75 77 80
03      07/16/2004,01 06 10 13 15 16 20 22 24 34 41 42 50 54 58 60 67 71 78 79
04      07/15/2004,03 04 09 10 12 16 27 33 37 43 49 51 52 53 55 59 68 69 71 78
05      07/14/2004,01 09 10 12 14 21 33 34 35 37 45 49 51 53 59 61 63 68 70 71
06      07/13/2004,01 04 06 10 11 16 26 27 30 41 43 45 46 50 52 54 57 60 62 68
07      07/12/2004,11 12 17 18 29 35 37 38 39 46 50 52 55 64 65 67 75 77 78 80
08      07/11/2004,03 04 13 28 29 30 31 32 34 40 45 47 48 49 51 53 65 66 74 79
09      07/10/2004,03 07 08 09 25 28 30 31 33 35 36 40 46 51 56 64 66 70 76 80
10      07/09/2004,09 10 13 16 22 23 25 26 27 30 40 41 47 50 51 52 57 67 68 74
11      07/08/2004,08 12 15 17 20 22 27 30 31 33 42 43 49 64 66 71 74 76 79 80
12      07/07/2004,04 09 12 19 29 37 39 51 52 60 62 64 65 69 71 72 73 77 78 79
13      07/06/2004,02 08 10 11 17 25 27 30 36 38 45 48 56 60 62 64 67 70 76 80
14      07/05/2004,08 10 11 14 15 23 24 34 36 41 49 52 54 57 58 61 63 69 72 78
15      07/04/2004,05 08 10 13 16 19 26 27 30 31 33 34 41 50 57 66 71 73 74 79
16      07/03/2004,04 05 07 09 18 21 22 26 29 30 31 35 36 38 43 46 54 56 61 74
```

```
07/02/2004,01 04 05 09 10 12 17 18 19 26 29 34 43 45 51 54 55 56 69 73
07/01/2004,02 03 06 07 12 14 17 24 25 33 41 45 54 55 58 63 66 70 76 80
06/30/2004,01 05 10 15 19 25 28 40 47 51 52 54 58 59 60 63 66 71 72 76
06/29/2004,06 10 14 16 22 24 30 33 38 40 45 47 48 62 65 66 70 72 78 80
06/28/2004,05 06 09 15 22 24 27 30 39 43 46 48 54 55 63 68 73 76 79 80
06/27/2004,01 04 06 07 15 17 22 23 26 30 41 45 52 54 63 66 67 72 75 77
06/26/2004,02 06 07 11 18 22 27 29 31 33 34 36 38 44 47 53 55 57 65 71
06/25/2004,01 07 09 13 17 18 19 34 38 40 45 50 53 54 63 67 72 73 77 79
06/24/2004,01 07 10 20 23 28 35 37 40 43 48 50 56 60 70 71 73 74 76 78
06/23/2004,08 12 20 22 28 31 32 33 40 43 48 54 55 56 62 64 67 68 74 75
06/22/2004,03 11 13 14 15 18 28 31 38 43 44 47 53 55 60 63 66 67 68 77
06/21/2004,03 05 08 09 10 16 20 22 26 34 35 36 38 43 46 48 61 73 74 75
06/20/2004,01 03 10 14 18 27 29 31 37 38 52 55 56 62 64 65 66 72 78 80
06/19/2004,03 06 09 11 15 16 19 22 23 39 40 42 46 52 53 57 59 61 67 68
06/18/2004,03 04 17 18 20 21 23 25 31 36 46 49 52 58 64 65 66 74 78 79
06/17/2004,03 05 07 21 30 34 37 38 44 45 49 50 51 53 59 63 64 69 75 80
06/16/2004,04 06 07 14 18 21 28 34 38 43 45 47 53 57 60 61 69 70 74 78
06/15/2004,01 07 08 11 14 24 26 34 40 41 49 51 52 53 54 56 64 66 71 80
06/14/2004,01 03 04 10 15 20 22 23 25 34 35 36 42 44 46 48 56 61 65 71
06/13/2004,03 06 11 15 17 24 25 28 32 33 34 36 37 50 57 59 61 68 70 72
06/12/2004,02 03 08 09 17 27 29 33 36 37 43 46 49 50 56 60 63 70 72 78
06/11/2004,01 02 05 07 12 22 25 32 39 40 44 45 50 61 62 63 67 73 75 78
06/10/2004,04 11 17 18 22 23 24 25 28 30 35 38 40 43 46 58 60 64 66 71
06/09/2004,07 08 09 16 21 27 28 30 36 44 48 49 51 59 61 66 69 71 72 73
06/08/2004,02 10 12 13 15 17 22 23 27 29 36 37 40 42 45 59 63 68 77 79
06/07/2004,01 09 25 26 27 28 32 40 51 52 53 59 66 68 69 76 77 78 79 80
06/06/2004,06 07 10 14 18 21 22 36 37 42 45 49 55 56 61 65 70 74 76 77
06/05/2004,03 05 09 12 16 18 26 28 37 41 44 47 52 59 61 68 69 72 76 79
06/04/2004,01 03 04 13 20 21 27 31 33 36 40 41 45 55 58 68 69 71 74 75
06/03/2004,01 03 05 07 08 11 13 15 20 29 31 40 53 57 60 63 65 71 73 75
06/02/2004,02 19 21 31 35 39 43 46 56 58 63 64 65 67 68 73 74 75 79 80
06/01/2004,04 05 06 10 12 17 24 25 31 32 34 36 39 43 49 58 64 65 69 76
05/31/2004,01 04 10 12 16 28 31 33 36 39 40 58 60 62 66 67 72 74 76 79
05/30/2004,06 09 11 16 17 19 20 23 27 31 35 36 44 52 53 54 63 73 75 78
05/29/2004,06 11 12 17 20 22 25 33 44 50 51 57 59 63 67 68 72 74 75 76
05/28/2004,01 04 05 06 12 13 15 16 35 43 45 50 51 53 55 56 57 62 68 76
05/27/2004,01 02 09 12 15 22 23 24 25 32 33 39 43 49 53 56 67 69 72 76
05/26/2004,02 08 12 16 28 30 34 37 38 40 45 47 59 67 69 71 72 78 79 80
05/25/2004,03 04 05 10 13 22 26 30 33 34 35 40 41 45 48 52 53 54 56 63
05/24/2004,17 23 25 28 30 31 32 33 34 35 37 39 40 44 47 50 63 72 73 74
05/23/2004,03 11 13 19 22 23 27 32 33 34 36 51 54 58 59 66 70 72 76 79
05/22/2004,02 03 10 11 14 16 17 22 24 36 37 43 44 48 49 60 61 64 70 76
05/21/2004,02 05 08 10 11 18 25 28 30 33 39 46 51 55 58 62 64 68 73 79
05/20/2004,03 04 06 07 12 22 25 26 30 32 45 46 49 53 55 59 62 63 64 79
05/19/2004,02 07 11 12 14 18 20 25 29 31 32 37 44 46 51 60 61 64 76 79
05/18/2004,05 06 18 22 30 31 35 37 39 40 49 50 56 57 67 68 69 71 75 79
05/17/2004,05 09 12 13 19 28 32 37 38 40 43 44 48 51 53 54 66 67 74 78
05/16/2004,01 03 04 06 15 18 20 30 31 33 35 38 40 42 46 52 54 55 59 63
05/15/2004,01 03 05 06 10 11 12 17 25 27 34 38 45 46 61 63 64 68 70 77
05/14/2004,03 09 17 22 29 31 34 35 38 39 42 46 51 55 60 61 63 72 76 80
05/13/2004,07 16 19 23 31 34 41 45 56 60 62 65 66 67 69 70 71 73 75 78
05/12/2004,01 04 05 06 08 11 21 23 26 35 37 38 39 42 44 46 60 62 72 79
05/11/2004,03 07 12 13 17 20 25 31 34 38 39 48 51 53 61 69 71 73 75 80
05/10/2004,07 09 10 12 16 17 22 23 33 44 49 54 55 57 58 64 65 73 78 79
```

```
71      05/09/2004,09 15 16 17 19 24 30 43 48 51 53 54 59 66 72 73 74 75 79 80
72      05/08/2004,01 03 04 07 10 11 12 20 22 25 30 34 37 40 57 62 63 65 66 75
73      05/07/2004,05 09 10 11 13 27 28 36 41 42 45 47 53 60 61 62 65 68 69 76
74      05/06/2004,12 13 26 28 31 35 41 43 47 50 52 53 55 58 60 61 62 67 71 80
75      05/05/2004,06 10 11 13 20 23 27 29 31 35 38 44 52 54 64 67 69 76 77 78
76      05/04/2004,03 05 12 14 18 22 38 45 48 54 59 65 66 67 68 70 71 73 75 77
77      05/03/2004,03 17 18 19 21 26 29 35 37 43 48 49 54 58 59 60 65 67 70 79
78      05/02/2004,01 02 09 11 13 14 16 21 27 28 46 48 52 56 57 60 61 71 73 74
79      05/01/2004,05 07 14 16 19 21 24 27 35 43 46 51 52 57 60 61 68 76 78 79
80      04/30/2004,02 06 08 16 24 25 27 31 41 47 54 55 58 59 64 65 70 72 73 79
81      04/29/2004,06 12 16 17 20 24 25 33 38 44 45 46 47 48 51 54 58 59 72 74
82      04/28/2004,07 09 10 11 17 18 21 24 28 30 35 36 41 43 49 54 56 59 71 79
83      04/27/2004,02 20 25 27 30 32 35 37 39 51 52 55 58 66 68 69 71 72 73 74
84      04/26/2004,01 08 13 14 19 20 25 29 33 48 51 57 58 65 67 68 69 72 75 79
85      04/25/2004,01 04 13 17 21 24 31 32 34 39 44 46 47 49 55 62 64 72 74 77
86      04/24/2004,01 09 10 15 18 19 21 24 30 36 42 51 58 61 63 67 70 71 78 79
87      04/23/2004,05 06 07 08 11 25 30 34 35 38 39 40 45 50 59 64 70 71 74 76
88      04/22/2004,02 05 06 10 13 23 25 29 32 39 55 57 58 60 62 63 72 76 78 79
89      04/21/2004,02 08 21 22 27 29 31 32 36 37 39 40 41 46 48 56 57 62 65 71
90      04/20/2004,01 10 13 15 20 22 34 36 42 45 59 61 62 64 67 70 75 76 77 78
91      04/19/2004,01 05 09 10 13 15 17 20 24 32 42 45 48 50 52 53 61 72 74 77
92      04/18/2004,06 07 08 10 11 15 23 26 27 28 29 32 33 49 50 56 62 67 76 79
93      04/17/2004,10 20 24 28 30 35 36 38 54 57 59 61 62 63 65 68 71 72 75 77
94      04/16/2004,02 03 06 09 11 15 19 23 27 41 42 49 50 52 53 55 57 68 69 77
95      04/15/2004,01 13 20 25 27 28 32 42 48 59 60 65 66 67 68 70 72 76 77 80
96      04/14/2004,07 10 11 12 13 14 19 22 23 24 29 30 32 36 49 50 52 56 71 75
97      04/13/2004,01 03 07 08 11 19 23 26 34 43 46 47 48 49 60 62 63 64 65 72
98      04/12/2004,13 15 16 18 25 27 30 35 36 38 46 47 54 55 58 60 69 73 75 78
99      04/11/2004,04 06 10 11 12 20 40 42 52 54 56 60 61 62 65 66 69 71 77 79
00      04/10/2004,09 16 21 31 34 35 38 39 41 42 45 50 53 62 64 68 70 75 78 79
01      04/09/2004,09 10 15 17 28 29 32 36 37 39 46 51 60 63 64 67 71 75 76 80
02      04/08/2004,01 03 09 13 23 25 32 36 37 45 46 49 50 52 53 56 64 70 71 72
03      04/07/2004,01 02 10 17 23 29 30 42 43 45 46 50 56 57 60 65 67 69 72 76
04      04/06/2004,03 08 09 12 20 22 29 30 31 33 34 39 42 44 49 53 56 65 69 77
05      04/05/2004,03 08 10 11 16 18 21 33 34 39 41 46 49 68 69 70 72 74 76 80
06      04/04/2004,05 17 18 19 20 23 27 35 36 43 45 52 55 56 64 68 71 73 77 79
07      04/03/2004,06 12 13 17 27 29 39 40 41 42 45 49 50 54 56 57 73 75 76 79
08      04/02/2004,06 11 15 17 21 26 27 29 30 31 32 36 44 57 62 68 69 70 77 80
09      04/01/2004,01 02 03 05 08 09 12 21 25 26 29 35 36 40 45 56 63 65 66 78
10      03/31/2004,16 18 21 22 24 27 29 42 44 45 47 48 49 50 53 55 59 62 65 76
11      03/30/2004,08 09 10 17 22 24 29 37 40 43 44 45 46 52 62 67 69 70 73 78
12      03/29/2004,04 16 19 23 25 27 31 33 34 35 40 46 47 48 53 55 58 59 71 76
13      03/28/2004,02 04 09 10 12 27 28 34 39 44 46 48 56 60 64 65 67 69 73 76
14      03/27/2004,01 02 03 04 20 24 30 32 35 40 42 43 47 58 63 66 67 71 73 74
15      03/26/2004,03 05 08 10 13 15 16 17 25 39 43 45 55 56 59 65 66 67 74 77
16      03/25/2004,05 10 11 12 15 16 19 20 23 28 38 40 41 46 57 58 63 65 70 77
17      03/24/2004,01 02 07 08 10 16 17 21 25 26 31 35 38 39 43 45 53 56 73 80
18      03/23/2004,03 08 10 15 18 20 27 32 36 40 46 48 51 55 58 63 66 70 71 76
19      03/22/2004,07 08 10 12 16 24 26 31 40 45 47 50 52 53 64 67 68 71 74 77
20      03/21/2004,01 05 06 07 10 12 19 23 27 32 33 37 41 47 62 63 67 68 77 79
21      03/20/2004,02 06 10 11 17 23 24 25 30 33 37 45 50 51 53 58 60 65 69 79
22      03/19/2004,01 03 08 10 17 18 22 25 27 28 29 31 32 34 36 41 46 71 76 78
23      03/18/2004,01 02 03 06 07 09 17 19 20 22 28 41 43 44 46 47 50 55 61 67
24      03/17/2004,01 04 09 14 15 16 21 38 39 41 49 55 56 59 65 68 72 73 74 75
```

```
25      03/16/2004,03 05 08 12 17 19 21 23 25 29 31 32 35 38 49 59 62 65 67 70
26      03/15/2004,03 07 09 12 13 15 21 23 24 25 27 29 42 49 51 52 63 64 66 77
27      03/14/2004,15 17 18 19 23 30 31 32 33 34 45 46 49 56 58 65 71 73 74 75
28      03/13/2004,02 10 11 17 19 20 23 24 29 31 49 53 57 58 66 67 69 73 75 78
29      03/12/2004,03 05 08 13 14 15 30 33 37 38 42 43 45 50 52 64 67 73 75 78
30      03/11/2004,04 06 08 09 11 15 18 23 25 27 33 34 41 48 55 61 67 70 73 76
31      03/10/2004,01 02 10 15 16 22 23 25 27 29 32 34 37 53 61 64 68 69 71 79
32      03/09/2004,07 14 15 24 25 30 36 41 42 46 47 51 53 56 58 59 64 67 76 79
33      03/08/2004,02 06 07 09 21 22 27 29 45 46 48 51 55 58 60 61 64 67 69 74
34      03/07/2004,01 04 05 07 10 12 17 20 21 38 39 44 50 52 53 64 65 70 78 79
35      03/06/2004,06 10 15 16 19 20 22 25 26 28 32 45 55 59 63 65 66 72 74 80
36      03/05/2004,01 04 07 11 16 28 29 30 33 34 35 44 45 47 48 51 57 59 70 76
37      03/04/2004,02 06 07 19 24 25 27 29 30 41 52 54 55 64 65 67 69 70 74 78
38      03/03/2004,09 10 11 15 18 20 24 28 31 33 37 46 50 55 58 60 65 67 75 77
39      03/02/2004,03 06 07 19 23 26 29 31 33 40 44 47 51 52 62 67 71 73 74 79
40      03/01/2004,04 06 11 21 22 24 25 31 34 35 36 38 48 59 62 68 72 76 78 79
41      02/29/2004,02 14 21 24 25 26 27 31 35 36 38 39 52 56 59 60 68 69 76 79
42      02/28/2004,02 09 15 23 24 28 31 33 38 39 48 52 54 58 59 61 62 67 68 71
43      02/27/2004,01 03 04 05 10 15 20 22 25 27 30 32 37 40 42 49 55 67 78 80
44      02/26/2004,02 04 15 19 20 26 28 29 31 33 34 41 46 53 55 70 71 77 78 80
45      02/25/2004,03 04 10 11 12 15 19 20 26 28 29 31 35 39 52 59 62 72 77 79
46      02/24/2004,04 05 08 09 13 25 27 28 33 35 37 38 41 42 52 54 63 65 69 77
47      02/23/2004,04 06 07 08 16 17 24 34 40 48 50 54 57 58 59 63 72 73 74 77
48      02/22/2004,02 03 04 06 13 27 30 34 39 40 42 52 59 63 64 67 68 71 73 79
49      02/21/2004,02 04 06 08 09 18 25 26 34 37 38 40 43 44 50 52 55 57 66 78
50      02/20/2004,05 06 07 12 13 14 24 27 28 34 37 39 45 50 58 62 65 66 77 79
51      02/19/2004,03 07 09 12 17 24 26 28 30 34 38 40 52 60 61 62 63 73 74 75
52      02/18/2004,04 07 15 19 22 27 38 50 52 54 60 61 62 63 67 70 77 78 79 80
53      02/17/2004,01 02 04 11 13 16 17 28 30 39 42 44 45 48 55 58 61 62 65 69
54      02/16/2004,04 16 18 22 25 28 31 32 35 38 40 45 54 55 56 60 67 74 76 77
55      02/15/2004,04 05 06 12 15 16 27 34 35 36 40 41 44 48 49 50 60 63 75 76
56      02/14/2004,01 02 07 10 17 26 27 31 32 36 38 46 48 50 60 68 71 74 76 79
57      02/13/2004,05 11 12 14 30 36 37 40 42 44 49 50 62 66 68 69 70 71 73 80
58      02/12/2004,02 03 07 08 09 10 16 17 31 39 40 49 52 60 61 62 63 64 65 66
59      02/11/2004,05 09 12 15 22 23 24 31 36 40 41 43 45 53 57 62 63 67 68 77
60      02/10/2004,02 13 17 18 21 24 25 27 28 29 32 34 38 39 42 44 55 56 58 65
61      02/09/2004,11 12 17 25 32 42 44 47 48 51 61 62 66 67 69 71 75 78 79 80
62      02/08/2004,02 08 09 12 13 23 25 26 27 28 29 34 40 45 49 55 57 66 67 77
63      02/07/2004,01 05 17 18 22 25 26 29 32 33 41 45 53 55 56 57 60 63 78 79
64      02/06/2004,04 05 09 16 19 24 25 29 34 37 38 43 44 46 47 48 60 65 68 74
65      02/05/2004,02 04 11 14 15 23 27 30 36 37 38 44 49 50 58 60 62 64 73 74
66      02/04/2004,12 19 23 27 40 43 44 47 48 50 52 58 59 61 62 64 65 69 72 75
67      02/03/2004,06 07 10 11 24 26 34 39 41 42 43 48 56 65 68 70 71 77 79 80
68      02/02/2004,04 05 11 15 17 22 26 28 31 34 39 43 55 57 59 65 66 71 77 79
69      02/01/2004,02 05 14 19 22 30 35 37 39 43 46 48 50 51 56 59 62 73 74 78
70      01/31/2004,04 21 22 23 27 30 31 37 42 46 50 52 55 57 58 62 68 73 75 76
71      01/30/2004,07 09 10 16 18 23 25 26 37 38 41 43 52 54 57 61 63 64 69 77
72      01/29/2004,01 05 06 07 11 13 14 20 23 27 28 38 44 46 47 59 60 68 74 77
73      01/28/2004,08 16 17 21 24 29 33 44 46 47 50 51 53 54 59 64 66 69 71 72
74      01/27/2004,02 10 20 22 25 26 28 32 37 43 44 54 55 56 59 61 64 65 69 70
75      01/26/2004,08 11 13 22 26 30 31 32 37 38 40 47 48 53 57 59 64 67 74 78
76      01/25/2004,04 13 14 16 17 18 23 25 34 40 43 45 46 50 56 66 67 69 72 76
77      01/24/2004,01 04 12 15 17 25 27 34 37 39 48 51 56 59 68 69 70 73 76 78
78      01/23/2004,03 08 09 16 19 22 24 33 35 37 38 40 47 53 55 57 61 62 79 80
```

```
79      01/22/2004,06 11 15 19 25 31 38 43 49 50 52 53 55 58 60 64 68 69 73 76
80      01/21/2004,03 11 12 13 15 17 21 25 29 35 36 45 51 53 63 65 69 73 76 80
81      01/20/2004,01 03 20 22 26 27 31 32 34 35 39 40 46 47 48 58 62 64 72 80
82      01/19/2004,05 06 09 15 21 23 27 28 30 35 39 41 44 46 49 55 57 64 73 76
83      01/18/2004,02 03 04 07 10 11 14 21 28 29 31 34 51 53 57 58 64 71 75 76
84      01/17/2004,05 07 14 16 17 21 25 32 35 41 45 53 55 60 65 70 74 77 79 80
85      01/16/2004,02 03 11 19 22 25 27 31 33 43 47 56 59 60 63 64 66 69 79 80
86      01/15/2004,01 07 10 12 17 21 23 25 27 28 37 39 43 45 47 55 57 59 62 79
87      01/14/2004,01 15 16 18 28 35 36 40 42 44 45 46 52 54 57 59 60 69 72 75
88      01/13/2004,04 05 06 12 13 17 25 28 37 38 44 56 57 63 68 69 70 72 73 77
89      01/12/2004,01 02 10 12 14 16 18 19 22 27 28 30 31 36 39 48 54 58 64 74
90      01/11/2004,09 10 11 12 15 20 21 30 31 33 39 41 52 53 56 58 66 73 75 78
91      01/10/2004,06 10 12 18 19 24 36 41 44 45 54 56 57 58 60 61 68 70 75 78
92      01/09/2004,01 02 04 05 09 32 35 37 42 43 46 48 49 52 54 57 58 61 72 79
93      01/08/2004,06 07 08 13 15 16 22 29 31 32 39 42 44 47 48 51 58 71 73 80
94      01/07/2004,01 03 09 13 21 27 29 30 32 33 37 42 50 62 63 64 72 73 79 80
95      01/06/2004,03 08 12 14 16 19 22 23 29 40 46 48 52 53 58 59 61 66 68 76
96      01/05/2004,03 06 12 14 15 23 26 27 33 36 42 44 48 49 54 55 56 68 70 75
97      01/04/2004,03 04 10 12 19 21 25 28 31 36 42 50 56 57 58 62 65 67 74 77
98      01/03/2004,07 13 14 19 25 30 37 44 45 48 50 54 59 64 66 72 75 76 78 80
99      01/02/2004,07 29 31 32 38 40 45 46 49 50 51 53 54 55 56 58 64 68 73 78
00      01/01/2004,06 12 15 18 20 27 28 31 33 34 36 45 48 54 57 58 60 70 72 74
01      12/31/2003,01 02 12 22 25 26 27 31 35 44 45 47 48 52 59 62 65 66 69 80
02      12/30/2003,03 07 14 17 25 26 27 32 33 35 36 38 40 46 48 49 53 58 70 78
03      12/29/2003,01 07 08 21 22 23 25 26 29 33 40 45 49 51 52 55 62 67 70 77
04      12/28/2003,03 04 05 07 15 16 20 23 25 26 36 41 49 51 54 62 66 67 72 76
05      12/27/2003,01 07 10 15 16 17 22 24 26 34 46 47 48 51 58 59 63 68 75 80
06      12/26/2003,06 08 09 10 15 16 18 19 21 22 30 33 38 39 41 44 45 65 73 80
07      12/24/2003,04 05 14 17 19 20 26 28 30 35 37 38 40 46 48 58 62 64 68 69
08      12/23/2003,01 07 10 16 17 23 29 31 37 40 41 45 47 50 59 61 69 72 73 78
09      12/22/2003,01 02 03 06 09 10 11 13 26 30 37 60 61 62 63 72 75 76 78 80
10      12/21/2003,01 04 11 13 21 22 32 34 37 42 47 48 59 61 64 66 68 74 77 80
11      12/20/2003,08 10 17 20 22 23 24 26 27 41 46 51 59 60 63 66 68 69 77 79
12      12/19/2003,01 02 03 05 17 27 28 34 38 51 55 60 62 67 70 72 73 74 77 79
13      12/18/2003,06 08 10 13 16 21 23 31 32 38 39 41 46 54 57 58 63 64 68 73
14      12/17/2003,05 06 18 20 23 26 27 34 38 40 49 53 55 58 59 61 66 67 75 77
15      12/16/2003,07 08 10 19 22 24 26 29 33 34 36 41 42 48 52 53 56 60 74 76
16      12/15/2003,02 03 05 07 12 13 17 19 24 25 27 28 29 41 47 61 69 75 77 79
17      12/14/2003,04 05 06 10 14 18 22 26 27 33 41 42 43 58 60 62 63 74 78 80
18      12/13/2003,06 08 13 14 15 16 18 32 35 36 43 46 47 49 54 58 68 71 74 77
19      12/12/2003,03 06 11 15 20 23 40 42 43 44 46 48 54 58 59 65 69 70 75 80
20      12/11/2003,02 06 09 13 16 18 24 25 26 34 36 44 51 52 53 55 57 59 75 79
21      12/10/2003,01 03 05 15 17 22 23 24 26 31 33 39 40 41 43 44 48 54 67 71
22      12/09/2003,04 07 09 11 13 22 24 28 34 36 37 40 41 54 57 58 59 68 71 75
23      12/08/2003,08 13 15 24 26 27 42 48 51 52 57 60 62 63 64 69 71 72 74 77
24      12/07/2003,08 09 10 15 18 20 21 24 28 32 33 34 44 47 50 59 60 61 67 71
25      12/06/2003,04 08 09 14 16 18 22 26 27 33 42 44 50 61 62 64 65 70 71 72
26      12/05/2003,02 03 05 13 17 18 24 36 39 42 44 46 48 57 66 69 70 71 73 77
27      12/04/2003,04 05 10 11 14 15 19 20 21 22 35 38 46 51 54 63 66 74 75 79
28      12/03/2003,07 09 11 15 20 29 33 34 36 38 40 42 46 47 49 57 58 64 69 72
29      12/02/2003,09 12 14 16 18 21 23 26 31 35 37 54 58 63 67 69 70 71 78 79
30      12/01/2003,01 06 08 11 13 14 17 31 33 40 50 54 55 60 64 65 71 74 76 80
31      11/30/2003,01 09 13 22 23 24 33 35 36 38 46 48 51 59 68 69 70 75 76 78
32      11/29/2003,04 05 10 11 16 26 29 35 37 38 40 43 44 46 51 57 58 62 67 79
```

```
11/28/2003,01 02 04 10 11 18 22 27 35 36 39 44 48 50 55 61 64 66 70 78
11/27/2003,16 22 24 31 35 36 41 48 50 54 55 60 61 64 66 70 72 75 76 79
11/26/2003,02 06 12 13 18 24 27 28 43 44 51 58 59 60 63 64 66 67 73 80
11/25/2003,03 05 08 10 15 19 22 24 27 28 37 38 43 50 53 55 60 74 79 80
11/24/2003,05 06 07 08 13 31 36 38 39 43 44 45 48 55 58 59 60 63 66 78
11/23/2003,02 03 04 06 20 28 29 37 40 41 43 47 52 55 58 59 61 63 64 71
11/22/2003,02 03 09 20 21 22 24 33 40 42 43 54 55 56 61 62 64 67 71 80
11/21/2003,07 09 13 16 17 22 23 26 30 31 34 40 44 53 60 67 70 71 73 80
11/20/2003,04 08 12 13 15 17 24 25 31 38 40 49 51 52 55 57 58 62 63 73
11/19/2003,07 12 14 18 36 37 38 52 53 59 60 68 70 71 73 74 75 76 78 79
11/18/2003,01 04 08 10 11 17 22 24 31 32 33 38 52 54 55 64 71 75 76 78
11/17/2003,03 07 08 09 15 17 23 28 37 49 50 52 56 57 58 59 70 71 72 80
11/16/2003,04 06 07 12 13 14 15 16 27 31 32 34 45 46 54 57 61 68 71 73
11/15/2003,05 09 19 24 25 30 37 42 43 44 46 48 49 52 61 63 65 66 73 78
11/14/2003,02 15 18 21 25 27 34 37 39 41 44 50 54 60 61 71 74 76 78 80
11/13/2003,03 04 07 16 18 21 25 26 37 42 43 45 52 54 56 57 58 66 69 75
11/12/2003,02 03 04 08 09 10 11 14 19 20 28 33 35 47 52 55 62 68 69 78
11/11/2003,02 04 14 16 32 35 38 44 47 48 55 58 62 63 67 68 73 76 77 80
11/10/2003,08 10 14 15 17 26 32 36 39 43 47 49 54 57 63 64 68 78 79 80
11/09/2003,01 08 11 15 18 20 21 30 38 46 47 48 50 53 66 67 70 71 73 75
11/08/2003,05 16 18 20 21 29 31 35 36 40 43 44 48 49 55 56 59 63 68 75
11/07/2003,09 11 15 23 25 31 37 39 44 45 46 47 53 57 60 63 65 68 75 78
11/06/2003,01 09 10 11 23 24 28 30 31 32 36 37 39 41 44 54 56 57 62 80
11/05/2003,05 08 15 16 17 20 22 35 36 38 41 42 45 48 63 66 74 76 77 80
11/04/2003,08 13 18 22 23 24 28 30 33 35 39 41 49 50 56 63 68 76 77 80
11/03/2003,03 10 11 19 24 25 28 33 35 38 42 43 47 52 53 55 58 62 69 73
11/02/2003,01 02 03 08 10 17 30 32 40 47 53 54 56 59 61 67 71 74 78 80
11/01/2003,12 15 21 24 25 31 32 35 37 39 41 43 45 58 60 62 68 72 77 79
10/31/2003,05 11 13 14 17 19 24 31 36 41 43 48 51 53 54 58 71 73 75 79
10/30/2003,07 14 17 24 26 29 31 33 35 40 43 45 48 49 50 58 61 64 68 78
10/29/2003,02 03 08 14 23 31 33 39 47 48 51 52 58 59 61 63 64 67 68 69
10/28/2003,02 18 21 25 27 28 30 36 38 43 44 48 49 52 55 65 71 72 73 79
10/27/2003,01 02 03 05 10 13 24 27 28 30 31 35 42 46 47 49 58 60 66 72
10/26/2003,10 14 19 26 28 33 36 39 45 54 57 62 63 70 74 75 76 78 79 80
10/25/2003,02 03 04 05 09 11 17 18 24 25 27 34 40 48 49 51 53 60 70 72
10/24/2003,03 06 11 17 19 22 26 28 32 33 36 40 41 46 62 64 66 71 72 74
10/23/2003,03 10 11 15 16 18 21 28 30 31 39 48 51 55 61 70 72 77 79 80
10/22/2003,04 07 11 15 16 25 34 37 38 39 41 43 49 59 61 64 69 70 71 78
10/21/2003,10 11 14 15 17 25 29 32 37 38 40 47 52 55 58 59 62 64 65 72
10/20/2003,03 12 13 22 26 28 33 41 47 49 51 52 54 60 63 68 71 72 77 78
10/19/2003,08 12 13 16 17 26 27 38 40 41 51 52 53 58 64 65 70 72 77 79
10/18/2003,08 11 13 15 16 27 33 34 38 42 46 51 52 54 58 60 65 70 73 74
10/17/2003,02 05 08 12 16 19 26 29 31 35 40 45 59 63 65 68 69 74 77 78
10/16/2003,12 17 18 19 20 23 26 34 39 41 42 44 45 51 53 57 59 62 70 80
10/15/2003,02 04 07 08 11 15 22 25 28 34 37 45 46 58 60 65 76 77 78 79
10/14/2003,08 09 19 21 24 25 31 32 34 47 50 51 53 54 56 57 60 69 75 78
10/13/2003,05 06 07 09 13 19 22 30 33 35 36 37 43 48 50 61 62 65 67 80
10/12/2003,04 06 09 11 13 17 22 23 29 30 31 41 48 52 63 64 68 73 75 79
10/11/2003,08 15 19 21 24 32 34 35 36 41 43 46 47 51 53 58 59 62 64 75
10/10/2003,03 07 08 09 17 18 25 31 39 46 47 51 55 58 59 66 67 72 73 80
10/09/2003,05 06 07 13 14 15 16 22 24 25 35 38 40 51 56 60 64 67 71 72
10/08/2003,01 03 04 05 07 16 20 25 28 41 44 45 49 51 52 56 58 63 75 77
10/07/2003,01 05 10 14 21 30 33 38 41 43 48 49 50 63 64 66 69 72 74 78
10/06/2003,01 13 21 23 27 31 38 39 41 44 46 47 50 51 56 60 63 65 76 80
```

```
87      10/05/2003,06 10 13 15 16 19 25 31 37 43 44 47 53 55 65 67 69 71 74 78
88      10/04/2003,03 06 14 16 20 25 29 30 34 40 46 47 50 62 67 69 70 71 76 79
89      10/03/2003,04 13 14 15 16 17 20 24 25 31 35 48 49 57 59 62 68 69 71 77
90      10/02/2003,02 10 13 16 22 25 31 32 35 36 45 50 56 57 60 66 68 71 72 80
91      10/01/2003,01 05 11 12 16 22 24 28 29 34 40 41 51 53 62 64 65 67 70 77
92      09/30/2003,02 03 08 09 13 20 22 24 30 32 35 39 41 49 55 62 66 69 75 76
93      09/29/2003,02 15 20 25 30 34 36 37 42 50 53 57 58 59 64 65 73 77 78 79
94      09/28/2003,01 03 04 05 07 10 12 13 23 25 30 36 40 49 55 56 61 64 65 67
95      09/27/2003,02 12 13 15 16 17 32 33 38 40 43 47 51 57 60 61 62 68 71 80
96      09/26/2003,07 10 15 22 25 27 33 36 38 39 40 47 53 57 59 66 69 70 78 79
97      09/25/2003,03 06 10 13 16 17 21 25 30 45 46 49 54 59 60 68 69 75 78 79
98      09/24/2003,04 07 09 16 18 23 31 34 35 37 41 46 47 49 52 55 56 62 78 79
99      09/23/2003,04 07 18 19 20 21 24 27 35 40 41 46 53 57 58 66 73 74 75 77
00      09/22/2003,10 11 17 22 26 29 30 31 33 36 45 47 51 53 58 60 61 64 67 71
01      09/21/2003,01 07 08 09 11 18 19 22 24 25 26 32 43 47 49 55 57 62 68 69
02      09/20/2003,01 02 07 08 10 14 17 28 30 32 36 39 43 49 65 67 69 71 76 77
03      09/19/2003,02 05 06 12 16 18 29 36 37 42 44 45 46 51 53 55 57 61 68 75
04      09/18/2003,01 06 07 09 10 18 24 30 32 34 40 42 44 51 68 71 73 74 77 79
05      09/17/2003,10 12 13 22 26 29 34 36 39 44 46 48 49 50 55 59 60 61 65 66
06      09/16/2003,09 20 23 26 30 31 34 39 40 42 44 47 54 55 56 59 61 72 76 79
07      09/15/2003,04 05 13 16 23 27 28 29 30 44 50 54 58 64 65 68 70 74 78 80
08      09/14/2003,04 06 13 17 20 28 32 40 43 46 48 54 55 57 61 65 68 70 72 73
09      09/13/2003,04 07 09 10 15 19 24 29 30 31 35 36 48 49 50 51 52 60 68 70
10      09/12/2003,02 04 06 13 15 23 24 29 30 31 35 36 40 47 50 52 65 73 77 78
11      09/11/2003,01 04 06 07 09 14 19 23 25 26 27 33 34 36 44 47 53 57 62 65
12      09/10/2003,04 05 07 12 14 17 18 23 30 34 37 43 45 56 59 62 68 69 74 78
13      09/09/2003,01 02 08 13 18 21 22 23 25 27 32 39 43 44 46 47 56 59 60 74
14      09/08/2003,04 08 14 16 18 20 25 27 31 33 37 38 43 49 53 55 58 62 63 78
15      09/07/2003,03 04 08 09 10 13 21 34 38 40 43 48 52 53 60 62 64 68 75 80
16      09/06/2003,05 07 08 22 24 30 32 36 37 41 46 48 50 51 53 54 56 59 77 80
17      09/05/2003,01 02 03 15 18 19 32 35 37 38 40 45 46 47 52 63 65 67 75 80
18      09/04/2003,04 06 13 15 19 24 28 33 35 37 38 39 44 45 53 54 67 69 74 79
19      09/03/2003,01 09 10 18 25 26 27 30 34 36 40 43 46 47 54 60 72 73 75 77
20      09/02/2003,01 02 03 07 09 10 13 18 20 28 31 39 42 43 49 55 58 61 75 80
21      09/01/2003,01 06 07 12 13 14 16 17 20 31 35 40 41 47 49 50 70 75 76 78
22      08/31/2003,03 08 11 13 16 17 21 26 32 35 38 42 48 50 54 58 62 71 75 77
23      08/30/2003,04 09 12 18 27 30 31 32 41 43 48 50 52 56 58 66 68 71 73 79
24      08/29/2003,07 10 16 25 26 28 32 35 45 46 50 52 54 56 58 66 67 71 72 78
25      08/28/2003,04 08 09 11 13 18 24 27 30 31 36 41 42 51 57 61 63 64 72 77
26      08/27/2003,01 05 24 27 31 35 38 40 43 49 54 57 60 61 62 67 71 72 75 77
27      08/26/2003,01 03 08 15 20 25 29 31 37 38 42 43 54 55 56 57 62 69 71 80
28      08/25/2003,03 05 09 18 23 29 33 34 40 41 44 51 53 57 58 59 60 62 65 77
29      08/24/2003,10 12 13 16 18 24 28 30 32 33 37 46 49 51 55 58 63 64 72 75
30      08/23/2003,06 10 11 15 16 18 19 37 42 44 48 52 53 55 65 66 70 72 76 78
31      08/22/2003,02 04 06 07 15 16 17 19 28 30 33 36 39 44 48 49 52 55 57 68
32      08/21/2003,02 07 13 19 22 24 26 29 31 36 45 53 58 59 61 65 71 72 73 74
33      08/20/2003,05 09 10 16 28 30 31 33 34 39 40 41 42 45 47 56 58 60 67 68
34      08/19/2003,01 22 24 25 31 33 37 38 44 53 57 60 62 65 66 70 71 72 73 74
35      08/18/2003,14 22 25 32 34 40 43 44 45 46 50 51 58 62 66 69 70 71 73 75
36      08/17/2003,04 10 14 15 17 23 26 30 36 37 41 43 50 51 52 56 60 70 71 76
37      08/16/2003,01 04 05 06 09 11 12 15 16 25 26 49 51 53 60 62 63 65 69 75
38      08/15/2003,08 10 21 26 27 32 35 39 42 45 47 48 56 57 61 63 66 69 77 80
39      08/14/2003,01 05 13 14 16 22 23 27 29 30 34 37 44 47 50 63 70 72 78 80
40      08/13/2003,05 06 08 12 14 17 19 24 30 33 38 42 44 45 46 55 60 68 77 80
```

```
41      08/12/2003,05 06 07 10 18 20 21 24 26 28 29 42 43 44 50 51 60 61 69 80
42      08/11/2003,13 14 23 24 25 27 36 46 50 53 54 57 58 62 63 67 73 75 77 78
43      08/10/2003,06 12 15 18 20 26 28 29 32 33 43 45 48 52 57 58 62 64 76 77
44      08/09/2003,01 04 05 08 10 13 23 24 27 36 37 38 42 45 57 58 60 64 70 78
45      08/08/2003,05 09 12 13 15 19 21 22 23 34 59 62 63 65 68 69 70 72 75 76
46      08/07/2003,01 03 04 08 09 10 16 18 21 34 41 44 53 56 59 60 68 70 72 73
47      08/06/2003,04 05 07 11 12 19 22 23 29 31 36 42 47 49 51 56 58 59 60 65
48      08/05/2003,01 04 13 16 20 23 29 40 46 48 49 52 54 56 62 65 70 75 76 77
49      08/04/2003,08 09 13 18 20 21 26 29 38 43 45 50 54 55 56 57 58 60 61 80
50      08/03/2003,02 05 07 08 15 23 27 39 47 48 49 53 54 56 58 64 68 74 77 78
51      08/02/2003,03 05 09 16 21 24 25 31 38 43 54 56 58 60 62 63 70 73 75 80
52      08/01/2003,03 15 16 19 20 22 27 28 31 37 46 48 51 59 62 65 68 69 77 80
53      07/31/2003,01 03 16 19 21 24 25 27 31 34 45 46 48 50 52 59 67 76 78 80
54      07/30/2003,01 03 05 07 13 15 17 21 27 40 43 50 54 60 64 65 68 69 74 79
55      07/29/2003,06 10 14 17 25 28 32 33 38 40 45 47 50 51 59 62 65 67 68 70
56      07/28/2003,06 08 10 16 17 21 23 24 25 26 38 39 43 52 62 65 66 71 72 75
57      07/27/2003,18 20 21 22 23 24 26 29 31 40 41 46 48 54 55 60 64 72 76 78
58      07/26/2003,02 05 10 17 19 22 23 25 26 28 31 37 39 46 55 56 58 66 69 78
59      07/25/2003,02 03 23 25 26 31 33 42 43 44 50 51 53 54 61 64 65 66 69 74
60      07/24/2003,08 09 16 18 30 35 37 38 43 44 47 53 55 56 59 61 72 73 78 80
61      07/23/2003,10 12 13 14 17 21 22 26 29 30 31 32 35 39 48 50 63 66 73 75
62      07/22/2003,01 09 10 11 19 20 21 23 24 31 38 48 55 57 61 67 69 70 73 74
63      07/21/2003,04 13 14 15 18 20 24 29 31 37 39 46 49 50 52 63 69 70 72 73
64      07/20/2003,01 03 11 13 20 24 26 32 33 35 40 42 46 48 53 56 62 74 76 78
65      07/19/2003,02 08 09 12 14 15 18 21 22 23 26 33 36 37 52 55 63 67 70 77
66      07/18/2003,01 03 06 09 24 28 29 32 35 42 48 49 53 56 64 66 70 71 77 80
67      07/17/2003,01 02 03 11 13 15 26 33 37 40 42 45 46 58 61 62 64 73 78 80
68      07/16/2003,03 06 10 12 16 17 19 21 24 25 27 29 32 35 36 42 43 56 67 68
69      07/15/2003,01 02 03 08 21 23 24 32 41 45 47 49 52 57 60 63 71 72 74 80
70      07/14/2003,06 10 13 15 18 22 27 29 34 36 45 50 53 54 60 62 69 72 75 78
71      07/13/2003,02 07 11 19 23 24 30 45 46 48 50 52 53 56 64 66 68 73 76 77
72      07/12/2003,02 04 05 09 10 29 33 34 35 38 41 42 47 49 50 51 54 58 66 80
73      07/11/2003,02 03 05 06 13 15 17 25 33 34 35 45 52 54 64 65 66 72 74 76
74      07/10/2003,02 04 06 08 09 14 15 18 26 30 35 39 53 55 57 62 63 77 79 80
75      07/09/2003,02 04 07 08 12 14 19 27 31 32 43 46 52 61 65 73 74 75 78 79
76      07/08/2003,10 12 22 23 26 34 35 36 39 40 41 46 49 53 64 66 67 77 79 80
77      07/07/2003,01 09 27 33 38 39 42 46 49 50 51 52 53 56 58 62 63 65 68 71
78      07/06/2003,06 07 09 10 13 19 22 23 27 28 32 34 36 39 45 55 57 59 65 70
79      07/05/2003,06 07 10 17 18 28 30 31 34 36 37 44 45 46 59 63 67 68 70 76
80      07/04/2003,01 02 04 05 10 20 21 24 26 34 35 44 48 49 59 63 64 65 68 72
81      07/03/2003,07 11 14 17 20 30 38 39 46 47 51 52 53 54 58 63 68 72 73 78
82      07/02/2003,01 11 17 27 30 33 35 42 45 47 51 52 53 54 55 56 61 70 72 73
83      07/01/2003,02 05 07 12 14 17 32 44 46 49 51 52 56 61 64 66 67 72 75 76
84      06/30/2003,01 06 10 21 24 34 39 40 49 50 52 54 60 61 62 66 71 75 76 79
85      06/29/2003,01 05 10 13 14 32 45 46 49 51 52 54 55 62 63 71 72 73 78 79
86      06/28/2003,17 19 20 27 33 34 38 39 40 46 47 52 53 56 59 69 71 73 76 80
87      06/27/2003,04 09 11 13 14 24 28 29 30 37 39 47 48 53 56 57 60 64 68 69
88      06/26/2003,12 15 16 22 23 25 29 32 35 37 39 40 42 47 48 61 66 68 69 72
89      06/25/2003,05 10 11 17 18 22 24 25 34 37 48 51 53 60 63 64 67 70 76 77
90      06/24/2003,01 02 06 12 20 22 26 27 43 47 53 58 59 61 64 66 68 70 76 78
91      06/23/2003,02 04 16 20 25 29 30 33 34 37 40 42 54 58 67 68 70 71 74 75
92      06/22/2003,03 11 12 15 21 23 27 29 36 37 39 43 52 55 56 58 67 76 77 78
93      06/21/2003,04 10 16 22 24 25 29 30 31 35 43 50 53 56 58 60 65 67 74 76
94      06/20/2003,02 06 11 18 19 24 39 42 48 50 52 53 55 57 59 62 66 67 77 80
```

95 06/19/2003,02 10 14 21 22 26 32 33 37 39 40 46 53 57 58 59 70 72 73 80
96 06/18/2003,01 05 12 19 20 30 39 46 47 48 53 56 62 63 65 66 67 70 73 79
97 06/17/2003,07 10 16 18 21 33 34 39 41 42 46 48 50 61 65 66 67 68 72 78
98 06/16/2003,04 08 17 28 31 32 35 40 41 43 45 46 47 49 55 56 59 64 73 76
99 06/15/2003,07 13 14 16 20 21 23 24 26 29 34 42 47 50 54 55 57 58 59 60
00 06/14/2003,01 09 17 18 21 25 36 37 38 40 44 45 51 52 61 63 64 70 72 79
01 06/13/2003,01 06 09 11 13 17 22 24 43 48 50 52 56 59 60 61 64 72 78 80
02 06/12/2003,01 02 03 04 11 12 20 23 28 38 40 42 45 49 60 64 65 67 69 76
03 06/11/2003,04 07 10 11 13 16 18 21 24 32 42 50 52 57 58 59 60 62 71 75
04 06/10/2003,01 05 07 12 15 17 18 20 21 24 27 35 39 44 66 69 70 73 74 79
05 06/09/2003,11 13 19 21 27 28 31 33 36 44 45 47 51 55 62 64 66 69 71 79
06 06/08/2003,03 08 15 19 21 26 33 37 38 47 50 52 59 62 63 69 71 72 76 77
07 06/07/2003,01 03 08 09 20 29 35 39 43 46 49 50 52 53 54 62 66 70 71 72
08 06/06/2003,07 08 20 25 29 30 34 35 40 42 47 49 52 53 59 60 61 65 71 79
09 06/05/2003,05 06 07 21 24 28 29 30 31 32 33 38 41 42 45 53 60 67 68 74
10 06/04/2003,01 05 09 12 19 20 28 29 32 33 37 39 41 45 48 51 56 73 78 79
11 06/03/2003,01 04 10 17 21 29 32 34 37 38 40 45 47 48 53 58 59 60 76 80
12 06/02/2003,03 11 12 13 16 19 22 24 27 30 35 39 45 48 51 57 62 67 68 71
13 06/01/2003,05 07 11 16 20 22 26 29 33 42 52 54 55 62 67 69 70 74 76 77
14 05/31/2003,03 07 13 15 16 18 19 23 26 27 28 38 39 42 43 45 53 67 74 77
15 05/30/2003,11 15 18 22 23 27 40 45 46 49 52 53 56 58 59 60 66 67 70 77
16 05/29/2003,04 09 10 15 21 25 29 30 31 34 40 42 48 50 58 64 65 68 70 75
17 05/28/2003,02 04 14 19 20 21 23 27 31 34 47 52 54 61 62 69 70 72 74 77
18 05/27/2003,01 07 09 12 16 19 27 35 41 42 43 49 50 60 62 64 74 76 77 79
19 05/26/2003,03 05 11 15 18 20 23 31 36 41 42 43 44 46 47 49 53 65 68 72
20 05/25/2003,05 09 15 16 21 26 36 37 41 43 45 46 48 49 50 53 54 58 67 80
21 05/24/2003,02 06 09 11 12 16 21 23 27 44 45 46 47 51 58 65 67 71 72 74
22 05/23/2003,01 09 13 14 16 18 20 22 23 27 31 36 52 56 63 67 69 70 74 77
23 05/22/2003,01 04 08 19 29 32 34 40 46 50 52 54 55 56 58 59 67 73 74 76
24 05/21/2003,02 05 06 07 08 10 13 17 20 27 30 32 45 47 58 69 71 74 78 79
25 05/20/2003,10 14 24 26 31 32 33 34 35 37 39 40 42 43 46 54 60 61 66 68
26 05/19/2003,03 14 16 17 21 22 24 29 32 33 39 40 41 48 52 54 58 59 60 68
27 05/18/2003,06 08 09 13 15 20 34 38 39 40 46 51 53 54 56 58 60 69 75 80
28 05/17/2003,02 14 15 19 20 24 26 36 39 43 46 48 50 54 58 62 64 68 70 78
29 05/16/2003,02 03 04 10 12 18 19 22 26 29 30 36 40 50 59 61 71 75 77 80
30 05/15/2003,07 09 16 22 23 25 26 27 30 40 49 53 58 60 65 66 69 71 72 76
31 05/14/2003,02 03 09 15 18 27 28 37 38 39 46 48 49 55 64 68 71 77 78 79
32 05/13/2003,02 03 05 11 14 20 24 25 36 37 38 44 46 47 51 57 61 62 70 75
33 05/12/2003,04 13 21 27 29 30 31 34 36 38 42 45 46 48 60 63 66 71 77 80
34 05/11/2003,06 08 12 17 19 20 24 27 33 36 38 39 42 46 61 62 67 71 79 80
35 05/10/2003,01 04 05 09 13 16 17 19 21 28 36 43 54 58 60 64 69 70 77 78
36 05/09/2003,03 06 07 09 11 12 18 19 23 27 33 42 46 55 67 71 72 73 77 80
37 05/08/2003,06 10 21 22 24 28 32 36 43 48 54 55 60 61 63 68 69 72 75 79
38 05/07/2003,04 07 09 12 16 23 27 31 32 40 43 45 50 51 53 55 62 67 75 80
39 05/06/2003,01 04 05 07 15 16 17 18 27 34 38 42 46 56 63 64 66 67 76 80
40 05/05/2003,10 16 17 18 21 22 29 33 36 37 39 42 46 55 58 62 65 69 72 77
41 05/04/2003,01 08 10 17 18 19 21 29 35 38 47 54 58 59 60 69 72 73 74 79
42 05/03/2003,04 07 17 21 24 25 27 29 31 32 35 36 38 39 47 53 54 55 61 68
43 05/02/2003,01 03 05 06 13 14 15 20 33 35 39 43 44 52 53 55 56 57 58 64
44 05/01/2003,01 12 20 27 29 31 33 34 41 42 43 46 50 55 59 65 66 67 69 79
45 04/30/2003,01 04 08 13 18 21 23 24 37 51 55 56 58 59 62 64 65 69 74 79
46 04/29/2003,04 07 15 19 20 23 24 28 31 37 38 46 47 50 54 66 68 73 79 80
47 04/28/2003,03 05 08 13 14 18 24 27 31 34 40 43 44 45 46 47 52 63 67 71
48 04/27/2003,07 10 11 13 15 16 20 21 29 31 37 40 42 46 50 52 65 71 72 76

```
49      04/26/2003,02 13 14 15 23 25 26 27 29 32 39 43 48 54 59 60 67 70 73 79
50      04/25/2003,03 06 09 10 12 17 18 19 25 27 32 38 40 49 53 61 62 63 67 74
51      04/24/2003,01 03 08 12 23 24 27 30 33 34 35 41 46 49 50 55 62 67 70 77
52      04/23/2003,01 08 11 15 23 27 30 31 32 38 40 43 52 58 63 68 70 71 77 79
53      04/22/2003,12 19 21 23 25 27 33 37 39 40 43 50 51 53 61 62 63 70 71 73
54      04/21/2003,01 06 11 16 19 21 22 23 25 37 38 39 43 46 51 53 56 67 71 80
55      04/20/2003,09 10 11 14 17 19 23 27 28 30 45 49 52 54 56 58 70 73 78 79
56      04/19/2003,01 06 09 11 18 33 40 44 49 51 52 59 61 63 65 70 71 75 78 80
57      04/18/2003,01 04 05 07 10 15 20 22 26 31 37 38 40 44 55 60 63 67 76 79
58      04/17/2003,06 07 08 10 12 15 16 20 24 33 38 40 43 45 55 62 66 67 71 77
59      04/16/2003,01 02 05 06 15 16 19 20 22 24 30 34 36 44 55 69 71 72 76 78
60      04/15/2003,07 08 09 10 13 23 27 29 38 41 46 51 52 55 56 65 68 69 74 77
61      04/14/2003,08 12 13 19 20 21 27 30 31 41 49 60 63 65 68 70 72 76 77 78
62      04/13/2003,01 03 05 07 08 10 13 17 27 33 34 43 46 48 50 64 72 73 76 78
63      04/12/2003,02 03 04 05 16 18 22 23 25 29 30 32 54 55 56 57 65 72 76 79
64      04/11/2003,04 07 17 29 32 33 35 38 40 43 47 53 56 58 61 64 69 73 78 80
65      04/10/2003,07 12 15 17 21 22 26 27 34 35 37 40 42 48 58 60 68 73 75 79
66      04/09/2003,01 10 15 18 23 25 30 35 38 39 42 45 47 55 56 57 61 71 74 78
67      04/08/2003,01 03 04 12 17 22 27 29 30 44 48 54 64 66 69 70 73 74 76 78
68      04/07/2003,03 07 15 18 29 38 42 44 45 50 57 59 62 68 69 71 73 78 79 80
69      04/06/2003,02 04 08 14 23 26 29 30 38 43 45 47 54 60 63 65 68 72 74 78
70      04/05/2003,05 06 14 20 26 32 33 35 44 47 48 58 59 62 65 69 71 74 75 80
71      04/04/2003,02 07 11 17 18 26 27 29 30 38 44 46 52 60 61 62 63 64 71 76
72      04/03/2003,01 03 08 11 17 20 22 23 24 35 42 51 53 55 59 63 65 75 78 79
73      04/02/2003,01 03 14 16 18 21 26 29 30 35 36 39 40 42 50 63 65 72 73 76
74      04/01/2003,10 13 14 20 23 26 29 31 34 37 38 39 40 44 48 49 56 60 62 70
75      03/31/2003,05 06 11 16 17 24 26 32 36 37 41 47 50 56 59 63 69 72 73 79
76      03/30/2003,04 08 09 10 13 15 22 32 33 36 37 41 42 46 53 59 64 76 77 79
77      03/29/2003,01 03 04 08 12 14 15 16 20 24 25 28 30 34 39 49 51 59 60 65
78      03/28/2003,06 08 11 16 22 35 36 41 50 53 59 60 64 70 71 72 74 75 76 78
79      03/27/2003,02 03 06 07 13 15 22 23 26 31 32 36 38 40 44 47 57 59 65 68
80      03/26/2003,12 16 18 20 21 25 27 28 30 37 41 49 55 56 58 59 68 73 74 76
81      03/25/2003,10 13 25 27 30 32 36 38 42 43 46 50 53 55 59 65 66 67 68 80
82      03/24/2003,18 19 24 25 26 28 29 36 40 45 48 49 52 56 64 65 67 70 73 77
83      03/23/2003,02 05 11 15 21 27 30 32 36 40 41 46 47 49 53 60 63 68 72 80
84      03/22/2003,03 04 05 10 15 16 20 26 28 32 33 42 46 48 50 51 52 57 65 66
85      03/21/2003,04 05 08 09 11 15 20 29 30 32 34 35 36 43 44 63 64 68 73 74
86      03/20/2003,05 07 10 12 13 16 18 19 24 26 29 31 48 49 57 60 61 63 64 74
87      03/19/2003,01 07 09 10 14 19 22 28 30 37 49 50 54 61 69 72 74 75 78 79
88      03/18/2003,03 05 09 12 15 20 23 28 32 34 35 38 45 47 53 57 60 63 67 69
89      03/17/2003,01 05 13 14 22 26 32 39 44 45 49 55 58 63 65 69 70 72 73 77
90      03/16/2003,09 10 11 12 15 19 21 27 29 30 34 40 42 44 52 57 61 62 70 79
91      03/15/2003,04 06 07 08 11 19 21 28 31 36 45 58 61 64 66 68 69 73 78 79
92      03/14/2003,06 07 10 14 21 25 28 29 33 38 47 50 55 56 61 65 67 71 75 77
93      03/13/2003,05 10 12 16 19 21 23 24 32 33 42 44 48 52 60 62 65 67 71 75
94      03/12/2003,01 04 05 09 16 23 29 30 39 44 49 59 60 62 65 68 72 74 75 79
95      03/11/2003,03 04 05 15 16 17 32 38 43 44 45 51 52 54 61 63 68 70 74 78
96      03/10/2003,12 14 18 21 32 38 41 44 45 48 49 56 58 60 62 63 65 68 69 76
97      03/09/2003,09 12 18 21 24 26 27 28 36 39 47 52 54 56 58 64 67 69 70 78
98      03/08/2003,03 11 20 24 25 26 27 33 34 38 42 45 47 53 58 62 65 67 68 75
99      03/07/2003,01 03 09 16 18 27 29 33 34 48 50 57 62 64 67 68 72 73 74 79
00      03/06/2003,01 04 06 14 16 17 34 37 38 41 48 52 53 54 56 57 62 66 68 73
01      03/05/2003,07 15 32 33 35 37 45 47 49 50 52 55 59 63 69 73 74 75 78 80
02      03/04/2003,01 02 06 07 11 18 29 30 34 38 39 44 47 57 61 63 67 71 75 76
```

03 03/03/2003,08 10 21 34 36 39 42 44 45 46 47 57 58 63 67 69 72 73 74 80
04 03/02/2003,03 10 11 20 21 27 28 29 36 39 45 46 58 61 62 63 65 68 69 78
05 03/01/2003,03 11 21 22 28 29 30 31 32 36 39 44 46 50 58 64 67 68 71 76
06 02/28/2003,02 07 10 17 22 27 31 33 34 42 49 53 54 63 64 66 67 72 75 76
07 02/27/2003,02 09 17 21 27 32 34 39 41 43 46 50 54 55 57 60 62 64 68 74
08 02/26/2003,01 05 11 15 16 17 25 26 30 40 51 53 54 55 59 63 70 72 76 78
09 02/25/2003,02 06 14 16 17 20 21 23 26 29 31 36 37 45 53 56 68 70 71 79
10 02/24/2003,09 15 16 17 26 31 40 43 46 49 52 61 62 63 66 69 71 75 76 80
11 02/23/2003,09 10 16 18 24 26 30 36 47 48 51 52 54 62 63 64 71 75 79 80
12 02/22/2003,03 08 09 11 12 20 23 29 30 36 44 58 59 60 62 63 70 74 75 76
13 02/21/2003,01 06 13 15 18 19 20 21 25 38 39 40 47 49 50 53 56 60 69 73
14 02/20/2003,03 08 14 15 21 26 27 31 35 38 39 42 53 56 57 58 59 67 77 79
15 02/19/2003,02 03 07 08 09 17 26 27 32 33 49 53 54 57 63 64 66 69 72 80
16 02/18/2003,02 03 07 10 13 21 22 26 29 37 43 53 57 58 61 62 65 73 74 75
17 02/17/2003,01 07 11 14 17 20 27 35 39 43 49 51 54 56 60 61 66 67 70 71
18 02/16/2003,06 08 09 13 24 27 29 30 32 38 40 50 52 57 61 65 68 71 72 78
19 02/15/2003,01 06 13 20 24 28 29 39 45 46 50 52 54 55 59 60 62 63 64 73
20 02/14/2003,02 05 10 18 19 21 24 26 38 42 43 49 55 58 63 64 66 75 79 80
21 02/13/2003,03 08 09 13 16 18 21 28 29 38 39 44 55 57 58 59 61 66 70 78
22 02/12/2003,01 02 04 07 19 24 27 29 37 38 46 47 52 61 63 69 71 75 76 77
23 02/11/2003,01 03 05 09 10 12 19 34 36 42 43 47 48 63 64 69 74 76 79 80
24 02/10/2003,06 10 12 22 26 29 34 39 41 47 49 50 52 63 68 72 75 76 77 78
25 02/09/2003,01 03 07 09 11 14 17 22 24 28 34 37 38 40 45 46 47 50 58 66
26 02/08/2003,04 05 10 13 15 26 27 28 41 42 43 47 52 53 61 64 66 67 70 78
27 02/07/2003,02 05 06 10 11 13 17 23 24 34 40 43 44 46 48 54 61 65 73 76
28 02/06/2003,11 12 13 14 15 17 28 32 35 37 43 47 50 55 58 65 66 68 75 80
29 02/05/2003,06 09 12 15 16 29 36 44 45 48 50 53 56 58 69 71 75 76 79 80
30 02/04/2003,01 07 08 10 15 20 21 25 33 34 36 38 41 45 53 57 60 62 68 70
31 02/03/2003,02 05 08 11 17 20 23 26 29 30 48 54 56 59 61 68 72 73 78 80
32 02/02/2003,08 13 15 16 26 28 29 30 35 40 42 43 44 49 50 64 65 71 72 75
33 02/01/2003,04 12 19 26 27 29 30 31 32 38 44 49 51 55 59 62 65 66 77 78
34 01/31/2003,05 06 12 15 16 19 23 31 36 37 46 47 48 49 54 62 66 68 72 76
35 01/30/2003,02 04 13 14 15 17 23 25 33 39 41 43 48 53 59 61 67 75 77 79
36 01/29/2003,06 10 16 19 26 28 29 30 31 45 46 53 60 63 66 68 70 75 77 80
37 01/28/2003,03 09 15 19 21 22 29 32 34 41 48 52 56 58 60 61 72 74 76 78
38 01/27/2003,01 04 09 10 11 14 15 29 31 34 35 36 47 48 49 52 54 69 73 74
39 01/26/2003,03 09 12 18 27 29 30 33 34 43 46 47 49 51 63 69 71 72 76 80
40 01/25/2003,16 20 26 27 28 29 30 36 39 42 46 48 50 52 57 58 60 67 72 76
41 01/24/2003,02 06 11 14 19 23 24 25 26 28 30 33 35 38 48 53 54 69 70 80
42 01/23/2003,10 13 14 15 20 24 26 30 31 34 42 48 51 52 53 56 57 60 61 70
43 01/22/2003,04 12 17 21 22 23 26 34 36 43 53 60 61 63 66 67 74 76 77 80
44 01/21/2003,03 10 11 13 15 16 19 22 24 29 35 42 51 56 57 64 65 70 74 76
45 01/20/2003,01 02 10 14 16 18 20 22 27 30 32 48 50 59 60 63 66 67 75 78
46 01/19/2003,01 02 03 09 15 18 20 35 36 42 43 44 48 52 56 59 60 65 71 72
47 01/18/2003,02 03 06 17 19 23 29 36 37 42 49 51 63 65 69 70 71 72 76 77
48 01/17/2003,01 05 09 13 14 17 18 19 20 22 29 30 31 32 43 57 58 63 69 76
49 01/16/2003,08 15 16 18 21 30 32 38 39 43 44 45 48 55 56 58 61 71 72 78
50 01/15/2003,06 09 13 14 15 21 31 34 40 43 45 48 49 52 53 58 62 68 73 79
51 01/14/2003,01 03 04 05 08 09 11 16 21 23 26 31 36 50 54 57 69 71 75 78
52 01/13/2003,10 11 12 13 17 21 22 23 26 28 38 42 45 47 50 52 59 60 61 63
53 01/12/2003,03 04 05 13 18 20 27 29 35 37 39 44 50 51 57 64 68 73 77 80
54 01/11/2003,05 11 14 16 20 21 23 24 31 43 44 46 54 61 63 66 68 73 78 80
55 01/10/2003,01 05 08 14 26 33 36 41 44 46 48 51 54 55 60 66 72 75 79 80
56 01/09/2003,02 05 09 17 18 19 20 24 25 27 33 37 38 39 41 42 45 71 72 80

```
57      01/08/2003,04 06 08 13 19 20 22 24 28 29 35 42 43 44 49 50 60 62 65 76
58      01/07/2003,04 05 06 12 15 27 30 33 34 39 41 45 53 64 65 66 67 72 73 78
59      01/06/2003,02 03 04 09 17 18 22 25 29 35 39 40 41 46 55 57 60 64 69 79
60      01/05/2003,12 15 16 19 22 25 26 27 30 36 39 43 44 58 59 61 67 68 73 78
61      01/04/2003,16 20 23 28 29 31 35 39 40 52 54 57 58 60 67 69 70 74 78 79
62      01/03/2003,06 07 08 10 12 15 18 19 21 25 32 38 40 44 47 51 53 64 75 80
63      01/02/2003,02 09 10 11 15 20 24 27 39 44 50 52 53 56 63 66 68 69 71 76
64      01/01/2003,01 03 15 16 17 25 28 30 33 43 46 48 55 58 60 61 67 70 75 79
65      12/31/2002,02 12 15 19 21 22 24 42 44 45 55 56 57 58 59 61 62 68 71 74
66      12/30/2002,01 02 08 11 16 26 28 32 34 35 38 40 45 46 55 57 60 61 64 71
67      12/29/2002,01 02 03 04 09 10 22 29 36 41 44 45 48 49 51 66 68 69 70 73
68      12/28/2002,04 06 07 09 11 14 15 17 19 31 32 40 44 47 52 55 60 65 70 78
69      12/27/2002,01 08 14 18 22 34 38 39 41 46 47 51 55 57 64 65 67 70 76 77
70      12/26/2002,03 04 17 20 21 24 25 42 49 51 52 56 62 64 65 68 72 73 75 78
71      12/24/2002,03 09 13 16 20 22 31 32 36 47 48 52 54 55 66 71 73 76 77 80
72      12/23/2002,03 07 10 15 17 25 34 36 38 39 42 44 58 64 66 70 73 74 77 78
73      12/22/2002,02 04 05 11 14 22 23 28 34 36 52 53 57 61 62 67 71 72 74 75
74      12/21/2002,03 06 07 09 18 22 27 30 32 33 38 58 64 69 70 72 75 77 79 80
75      12/20/2002,05 06 09 11 15 19 23 26 32 36 37 45 47 51 59 61 66 70 75 77
76      12/19/2002,08 10 14 15 16 18 19 29 31 34 35 39 50 55 59 64 65 66 74 79
77      12/18/2002,02 03 09 12 15 17 24 25 27 28 33 36 40 42 55 58 70 71 73 76
78      12/17/2002,02 03 04 18 22 24 26 32 41 48 49 52 58 67 70 71 73 75 77 79
79      12/16/2002,02 05 06 11 15 18 24 32 34 35 37 40 42 50 57 59 63 68 78 79
80      12/15/2002,04 05 06 15 16 23 32 33 42 43 44 48 51 60 64 66 69 73 74 79
81      12/14/2002,03 11 17 19 20 24 25 36 38 39 44 45 48 51 52 64 69 70 74 78
82      12/13/2002,03 06 07 08 09 15 18 21 23 24 26 34 38 40 49 51 60 69 70 76
83      12/12/2002,03 07 11 21 29 31 32 33 35 37 39 47 54 57 64 65 70 75 77 78
84      12/11/2002,01 08 14 20 24 26 28 34 35 38 40 42 45 47 60 61 66 70 73 80
85      12/10/2002,01 03 17 19 20 22 27 34 43 47 48 52 55 56 65 69 71 72 73 74
86      12/09/2002,07 11 12 23 28 29 35 37 42 43 45 46 53 55 58 60 64 65 71 75
87      12/08/2002,01 04 14 15 17 19 20 30 43 47 48 53 57 61 62 64 68 75 77 79
88      12/07/2002,03 06 15 16 19 20 23 24 25 37 41 45 52 57 58 64 69 71 75 77
89      12/06/2002,01 02 04 05 13 16 18 24 25 29 30 39 40 45 62 65 66 69 79 80
90      12/05/2002,10 17 18 24 27 29 31 37 40 44 47 51 52 54 55 59 63 66 71 80
91      12/04/2002,01 03 04 07 11 23 24 28 30 33 38 40 41 54 55 56 63 64 67 68
92      12/03/2002,02 04 06 08 13 18 21 22 36 42 44 50 53 59 60 61 64 67 68 78
93      12/02/2002,02 06 26 28 30 31 37 39 40 45 46 48 61 62 63 66 68 69 72 78
94      12/01/2002,12 15 16 22 26 31 33 39 42 43 45 46 47 48 52 53 64 69 71 73
95      11/30/2002,01 02 06 19 27 29 35 36 39 42 47 50 56 58 59 63 65 70 71 79
96      11/29/2002,06 09 17 19 20 22 23 24 32 33 40 55 62 66 70 72 73 77 78 80
97      11/28/2002,05 08 09 11 12 13 19 20 21 31 35 45 46 51 54 63 71 74 76 79
98      11/27/2002,03 04 10 18 20 23 24 34 37 38 41 42 44 45 47 48 51 57 63 80
99      11/26/2002,01 04 07 11 12 14 18 31 33 34 38 43 47 52 53 56 63 65 72 75
00      11/25/2002,01 06 12 14 17 20 21 23 30 33 34 37 44 49 55 62 67 73 74 76
01      11/24/2002,02 03 04 07 26 28 32 33 38 39 45 46 48 50 58 59 60 61 63 70
02      11/23/2002,02 04 06 13 14 16 20 24 27 44 46 52 56 58 59 61 62 72 76 78
03      11/22/2002,06 12 13 14 16 20 22 28 32 37 40 42 49 53 58 64 66 68 69 77
04      11/21/2002,01 03 05 21 22 25 33 38 40 43 44 49 56 57 63 64 68 69 77 78
05      11/20/2002,01 08 11 15 20 21 26 28 48 54 55 57 58 62 64 66 70 72 78 79
06      11/19/2002,05 06 08 14 15 26 27 28 36 44 45 49 53 54 59 61 69 72 79 80
07      11/18/2002,01 03 04 06 09 10 11 20 21 24 26 35 45 53 60 62 69 71 72 73
08      11/17/2002,05 07 09 11 12 13 14 17 22 24 26 31 47 48 54 57 67 71 76 80
09      11/16/2002,05 06 08 09 11 13 22 29 30 41 45 50 52 54 63 65 66 69 70 74
10      11/15/2002,08 13 14 15 21 25 26 32 33 35 36 38 39 53 59 68 70 72 76 80
```

11 11/14/2002,06 13 19 20 32 33 36 38 42 46 48 50 57 60 68 69 71 77 78 79
12 11/13/2002,04 06 08 16 18 20 32 35 36 48 49 53 54 57 60 63 64 68 71 78
13 11/12/2002,02 03 13 24 25 26 30 35 38 45 47 50 55 59 60 66 67 70 73 76
14 11/11/2002,02 07 08 15 17 25 26 30 37 39 40 41 53 54 60 61 68 70 74 77
15 11/10/2002,01 05 06 08 19 20 25 26 39 42 43 53 60 61 63 68 69 72 74 80
16 11/09/2002,01 05 06 08 12 15 16 17 19 20 21 33 38 42 48 49 51 64 71 72
17 11/08/2002,07 09 17 22 25 30 37 43 49 52 55 57 58 60 65 67 70 71 74 78
18 11/07/2002,01 11 12 13 15 18 21 24 25 26 29 39 42 46 49 53 55 72 75 77
19 11/06/2002,02 07 08 14 20 21 29 33 36 37 44 45 47 48 53 61 62 64 65 71
20 11/05/2002,08 12 18 24 25 26 32 36 43 45 51 53 58 59 60 67 72 73 76 80
21 11/04/2002,03 05 09 14 19 24 25 28 29 30 38 39 43 53 68 72 73 75 78 79
22 11/03/2002,05 08 13 19 35 38 39 45 46 48 51 54 57 61 65 67 76 77 78 79
23 11/02/2002,01 03 05 20 21 29 30 31 34 36 38 46 47 49 52 54 63 64 68 77
24 11/01/2002,14 15 23 26 30 33 38 43 44 48 58 59 62 66 69 70 72 73 74 80
25 10/31/2002,01 02 04 10 16 17 33 36 37 38 48 49 51 53 58 61 65 68 69 78
26 10/30/2002,01 03 04 06 07 10 13 16 18 23 37 39 41 44 45 53 58 71 76 77
27 10/29/2002,17 20 28 30 38 42 45 50 51 54 56 57 59 62 67 69 71 73 78 79
28 10/28/2002,01 04 08 09 18 23 26 37 43 45 48 52 53 55 65 68 72 73 75 80
29 10/27/2002,05 07 08 09 18 24 31 35 37 39 43 44 52 57 62 63 71 75 76 79
30 10/26/2002,02 05 11 13 17 18 19 26 27 29 33 38 46 50 53 58 60 69 72 75
31 10/25/2002,01 04 06 09 25 30 33 35 38 40 43 52 53 59 68 71 72 73 74 77
32 10/24/2002,06 07 08 15 16 17 18 19 22 24 26 27 28 52 57 61 68 71 76 79
33 10/23/2002,01 07 13 14 21 24 25 36 37 38 39 42 44 55 58 59 63 64 68 80
34 10/22/2002,02 05 12 15 17 22 32 35 43 44 47 49 51 56 58 65 68 70 72 80
35 10/21/2002,04 09 10 11 14 17 21 27 36 39 40 42 50 52 58 62 66 67 72 73
36 10/20/2002,02 03 09 11 18 23 24 30 37 40 41 44 46 53 55 57 59 66 68 70
37 10/19/2002,03 08 10 21 22 24 25 30 31 32 35 37 38 44 54 61 67 73 77 78
38 10/18/2002,06 10 12 23 31 35 36 38 39 41 47 50 55 58 60 67 68 72 75 80
39 10/17/2002,07 09 10 11 15 17 25 29 32 37 38 40 48 50 52 55 56 66 72 78
40 10/16/2002,09 10 13 17 18 19 21 35 38 49 57 58 59 64 65 68 71 72 79 80
41 10/15/2002,09 12 15 21 24 26 28 29 30 36 42 50 51 53 55 61 65 70 74 75
42 10/14/2002,03 08 09 17 23 26 27 34 36 39 42 48 56 57 60 67 68 70 74 76
43 10/13/2002,02 03 04 19 25 29 30 36 38 39 40 43 56 57 58 65 72 74 78 79
44 10/12/2002,02 03 07 18 20 24 27 40 41 42 44 46 47 49 54 58 66 68 73 78
45 10/11/2002,11 12 15 29 32 33 35 38 39 41 42 46 49 57 58 60 68 70 73 76
46 10/10/2002,01 07 09 10 13 14 16 19 23 26 30 35 44 51 64 66 72 73 74 76
47 10/09/2002,01 07 12 14 18 21 22 25 29 32 46 51 53 55 57 60 64 65 66 74
48 10/08/2002,07 13 16 18 23 27 31 34 35 39 41 48 50 52 57 62 68 71 77 80
49 10/07/2002,04 05 06 15 16 17 20 21 30 43 49 53 54 57 62 64 75 76 79 80
50 10/06/2002,03 05 07 12 22 23 28 29 30 31 33 40 45 50 52 56 65 68 77 78
51 10/05/2002,05 07 10 11 17 21 25 29 30 38 42 45 53 54 57 58 68 72 75 79
52 10/04/2002,03 05 14 19 25 31 40 43 52 54 56 59 62 66 67 68 69 70 74 80
53 10/03/2002,01 08 12 15 23 24 25 31 32 34 36 37 48 57 66 68 70 73 74 76
54 10/02/2002,02 08 10 11 17 18 22 25 34 35 41 53 54 58 59 61 62 69 70 78
55 10/01/2002,01 03 08 09 18 24 27 29 31 36 41 42 53 55 56 59 61 68 70 72
56 09/30/2002,02 10 15 16 20 21 26 33 36 37 39 48 49 53 55 62 63 68 70 73
57 09/29/2002,10 12 14 16 21 22 23 24 28 31 36 51 54 58 60 61 62 68 71 80
58 09/28/2002,02 09 15 19 22 24 31 32 38 41 43 45 47 58 59 62 70 72 76 79
59 09/27/2002,05 10 12 16 17 20 22 24 25 28 34 37 47 54 56 57 64 66 67 79
60 09/26/2002,03 04 12 13 15 19 23 29 30 35 45 46 51 56 57 59 65 69 73 77
61 09/25/2002,03 07 08 09 13 26 29 32 33 34 36 41 46 50 54 60 61 67 74 75
62 09/24/2002,03 06 07 09 24 32 33 35 36 41 43 47 53 57 58 63 64 68 71 80
63 09/23/2002,01 06 08 09 16 17 19 20 23 26 30 31 39 41 43 45 47 55 63 64
64 09/22/2002,01 03 10 11 15 20 24 27 31 36 37 38 41 47 49 50 59 61 62 78

65 09/21/2002,02 04 08 14 15 16 18 25 38 40 45 47 49 55 57 59 68 74 75 77
66 09/20/2002,01 05 06 07 09 14 18 19 25 29 34 44 52 57 60 61 62 71 73 76
67 09/19/2002,02 03 10 11 13 16 19 22 26 31 37 41 43 45 47 64 66 67 70 75
68 09/18/2002,01 10 12 13 19 25 29 30 34 35 40 42 44 57 58 59 71 73 74 77
69 09/17/2002,04 15 24 26 31 32 36 37 41 50 61 63 65 68 71 72 73 77 79 80
70 09/16/2002,02 06 13 14 18 25 26 28 29 35 39 44 45 50 51 56 59 70 74 76
71 09/15/2002,07 12 13 15 16 19 21 22 30 31 34 37 40 47 50 59 60 63 69 78
72 09/14/2002,03 07 09 10 11 12 13 16 17 19 22 34 36 40 45 51 52 53 66 71
73 09/13/2002,02 13 14 17 18 19 30 31 41 42 46 47 53 57 60 63 64 65 71 75
74 09/12/2002,04 06 07 17 20 23 25 26 36 39 41 42 45 46 53 54 67 73 76 80
75 09/11/2002,04 06 08 14 18 20 29 31 33 35 38 51 52 53 58 59 60 62 63 69
76 09/10/2002,01 13 20 23 25 26 38 39 40 41 43 44 46 48 49 53 55 57 69 74
77 09/09/2002,03 09 18 20 24 35 38 39 49 51 53 55 59 61 62 65 67 73 77 78
78 09/08/2002,05 06 07 12 15 16 20 28 31 35 43 47 49 56 64 70 72 73 74 75
79 09/07/2002,05 08 10 14 22 24 28 30 34 35 38 46 48 57 59 65 69 71 72 73
80 09/06/2002,02 03 10 13 16 18 24 29 30 31 37 40 52 54 56 57 58 65 78 79
81 09/05/2002,03 04 19 20 27 31 37 43 47 51 53 56 57 58 62 68 70 71 76 80
82 09/04/2002,02 04 07 09 11 13 17 18 29 41 47 54 58 61 62 69 70 75 78 80
83 09/03/2002,03 07 08 13 14 15 19 20 22 29 32 36 43 47 56 63 66 70 72 73
84 09/02/2002,05 08 10 16 18 27 31 36 38 45 46 47 51 52 55 63 64 69 71 74
85 09/01/2002,01 06 15 16 33 35 38 39 46 49 51 58 60 63 69 71 74 75 77 78
86 08/31/2002,08 10 12 17 18 21 22 34 36 38 40 46 51 52 53 67 68 76 77 78
87 08/30/2002,01 03 07 09 25 28 31 44 45 46 49 52 53 56 62 64 65 67 76 79
88 08/29/2002,02 04 18 24 25 29 30 32 34 45 48 49 52 53 61 65 67 69 79 80
89 08/28/2002,06 10 11 18 21 22 31 37 38 41 45 47 55 58 59 69 71 77 78 79
90 08/27/2002,02 04 08 16 21 28 31 35 37 38 44 48 50 59 61 65 66 68 73 77
91 08/26/2002,01 03 08 14 18 23 24 25 28 31 43 44 45 47 55 60 62 64 70 78
92 08/25/2002,08 10 13 17 26 33 35 36 40 43 45 49 50 54 62 71 73 74 75 77
93 08/24/2002,02 06 07 10 11 13 24 28 38 44 45 50 51 57 62 64 68 73 76 80
94 08/23/2002,02 06 07 08 16 18 19 32 36 40 47 54 56 57 61 65 69 73 75 78
95 08/22/2002,03 04 05 06 08 16 22 32 36 42 46 47 51 57 59 63 69 72 73 78
96 08/21/2002,01 04 07 17 19 20 24 25 31 32 38 43 44 50 58 59 64 67 68 80
97 08/20/2002,05 08 13 15 25 28 32 33 42 43 44 48 52 60 62 67 73 76 77 78
98 08/19/2002,06 08 14 16 19 26 28 29 35 36 37 45 47 52 58 59 63 68 78 79
99 08/18/2002,02 05 07 10 15 16 21 26 27 29 30 34 35 52 56 57 60 65 67 68
00 08/17/2002,06 16 21 23 26 40 43 44 47 50 52 54 56 61 63 67 69 72 73 74
01 08/16/2002,02 04 06 08 14 18 19 22 31 41 45 46 53 54 57 61 66 68 74 75
02 08/15/2002,09 18 21 23 36 37 41 43 49 51 53 58 65 68 69 70 74 76 77 78
03 08/14/2002,01 05 07 09 18 25 27 30 31 32 34 38 43 52 53 57 64 65 68 71
04 08/13/2002,06 10 24 25 27 28 29 32 33 34 36 37 48 52 54 58 63 65 75 79
05 08/12/2002,05 08 09 11 18 22 24 26 35 48 50 56 58 64 65 73 75 77 78 80
06 08/11/2002,07 12 14 16 18 21 31 34 35 37 39 43 45 49 56 60 65 68 75 77
07 08/10/2002,02 10 14 16 20 22 29 32 33 34 44 45 47 50 66 69 70 73 74 79
08 08/09/2002,04 06 08 09 11 16 19 26 27 33 37 38 45 47 48 49 63 66 76 80
09 08/08/2002,02 07 09 17 25 26 41 46 50 52 53 55 59 62 63 67 68 69 75 77
10 08/07/2002,03 12 25 30 32 45 46 51 54 57 58 60 61 65 66 74 75 77 79 80
11 08/06/2002,02 06 10 11 15 21 25 29 35 38 39 40 44 45 51 56 57 58 71 79
12 08/05/2002,01 08 09 12 22 23 26 29 33 34 35 41 44 66 69 70 71 74 77 78
13 08/04/2002,02 06 10 12 13 18 24 41 42 44 46 47 49 51 57 60 66 73 76 77
14 08/03/2002,01 04 05 06 13 21 24 37 38 45 46 47 48 53 59 60 62 73 74 75
15 08/02/2002,01 04 05 08 11 18 23 24 26 30 35 36 50 52 56 59 65 69 71 78
16 08/01/2002,02 09 11 15 18 21 22 23 24 27 33 39 40 53 54 55 58 59 69 78
17 07/31/2002,01 04 05 06 19 22 23 28 29 39 48 53 55 63 65 66 67 70 78 79
18 07/30/2002,04 05 06 08 23 28 33 36 38 39 41 44 45 47 56 58 61 69 73 74

```
19      07/29/2002,10 12 19 21 24 26 33 35 36 38 40 41 51 52 55 63 67 70 71 78
20      07/28/2002,02 08 09 19 20 24 25 26 30 34 38 48 51 54 60 67 68 71 75 78
21      07/27/2002,07 12 13 17 21 29 32 37 38 39 43 46 49 55 57 63 65 71 72 77
22      07/26/2002,01 07 08 14 19 22 25 34 35 40 44 51 53 60 64 67 73 74 75 76
23      07/25/2002,01 05 09 17 20 24 27 30 32 35 38 43 53 59 65 69 70 71 74 77
24      07/24/2002,04 07 11 14 17 21 25 28 32 34 35 39 44 45 59 60 69 70 76 77
25      07/23/2002,05 11 13 14 20 22 30 32 34 36 48 49 50 51 52 53 54 55 61 73
26      07/22/2002,02 03 05 06 09 10 16 32 41 50 52 54 61 65 67 68 73 74 78 79
27      07/21/2002,01 04 06 09 12 16 18 22 30 31 36 42 44 46 50 56 61 62 67 69
28      07/20/2002,04 09 16 17 18 21 32 37 39 46 47 52 58 61 66 69 71 73 77 78
29      07/19/2002,01 02 17 18 19 21 26 29 31 34 40 52 55 56 57 62 64 66 76 77
30      07/18/2002,01 11 12 13 15 24 25 35 37 44 46 49 51 54 62 72 76 78 79 80
31      07/17/2002,02 11 16 17 31 32 40 41 42 46 47 49 50 54 58 69 75 78 79 80
32      07/16/2002,01 02 10 13 21 28 30 32 37 38 39 41 43 49 51 65 67 70 73 79
33      07/15/2002,01 04 07 08 10 12 14 18 20 29 32 36 37 38 39 42 49 51 56 58
34      07/14/2002,01 03 08 13 17 22 23 30 32 33 44 46 47 49 54 57 63 68 72 76
35      07/13/2002,02 07 11 15 16 17 19 24 27 28 29 30 32 36 46 55 56 66 78 79
36      07/12/2002,26 32 34 39 40 45 52 53 56 58 60 61 63 64 65 67 73 75 77 79
37      07/11/2002,01 07 08 09 13 15 17 20 25 26 28 34 37 42 44 49 55 60 66 79
38      07/10/2002,02 10 17 26 38 39 40 43 48 50 51 52 62 64 66 67 68 74 75 80
39      07/09/2002,02 11 16 18 22 23 24 26 28 29 30 36 37 41 44 54 58 64 73 78
40      07/08/2002,04 07 26 27 28 29 30 31 41 48 53 55 58 61 65 66 69 73 77 79
41      07/07/2002,01 03 09 15 20 23 27 30 33 37 45 50 57 60 64 70 71 73 74 79
42      07/06/2002,06 11 15 17 23 34 35 38 44 52 55 56 61 63 64 66 69 70 77 78
43      07/05/2002,01 06 07 12 14 16 17 33 36 38 42 44 48 55 58 60 69 70 74 78
44      07/04/2002,06 09 11 23 24 26 36 39 43 46 51 53 55 62 65 68 71 73 78 80
45      07/03/2002,06 07 10 22 23 24 25 41 42 43 45 50 51 56 59 61 64 72 78 80
46      07/02/2002,01 03 04 08 14 16 23 27 31 34 37 40 42 50 58 62 69 72 74 80
47      07/01/2002,06 07 15 21 28 30 31 32 33 34 37 42 49 50 64 70 71 72 74 79
48      06/30/2002,02 07 13 15 18 25 26 27 31 33 36 38 39 41 46 51 57 62 65 75
49      06/29/2002,01 03 11 25 33 36 37 38 41 42 49 51 56 58 60 61 65 68 75 77
50      06/28/2002,02 08 15 16 19 26 27 30 31 33 35 39 49 53 54 65 66 68 71 76
51      06/27/2002,02 07 10 20 21 24 29 30 32 34 37 40 41 46 49 60 72 75 76 80
52      06/26/2002,01 09 12 22 28 30 33 35 36 40 41 42 43 46 49 53 55 69 73 76
53      06/25/2002,05 09 15 22 23 29 33 34 35 36 39 45 52 54 56 57 58 65 74 80
54      06/24/2002,02 09 13 18 21 23 25 31 32 44 46 47 49 52 57 65 67 68 73 77
55      06/23/2002,03 10 12 16 19 21 22 26 29 30 36 39 41 42 45 57 60 64 70 77
56      06/22/2002,08 09 18 20 23 24 25 27 28 30 31 37 41 47 50 51 54 61 62 65
57      06/21/2002,03 05 10 13 14 18 20 23 26 33 36 38 40 52 55 56 57 58 71 76
58      06/20/2002,03 07 11 15 20 22 26 32 40 44 45 48 54 59 62 68 69 73 75 76
59      06/19/2002,01 04 12 14 18 27 33 38 44 46 50 51 55 59 65 70 74 75 77 80
60      06/18/2002,01 04 05 11 20 21 28 31 40 41 42 56 58 64 67 68 70 71 75 78
61      06/17/2002,04 05 08 10 21 27 38 40 41 43 48 59 60 67 68 69 71 72 77 79
62      06/16/2002,02 03 12 21 22 23 29 33 44 45 48 51 57 58 61 65 68 69 71 80
63      06/15/2002,03 07 11 15 22 26 29 30 33 39 40 41 44 53 61 65 69 70 73 76
64      06/14/2002,05 08 10 12 13 23 26 28 32 33 36 46 55 65 68 70 76 78 79 80
65      06/13/2002,02 05 08 10 14 16 19 27 32 38 39 40 41 46 49 56 60 74 78 80
66      06/12/2002,06 08 10 11 12 13 14 19 28 30 33 39 41 52 55 60 65 69 73 76
67      06/11/2002,04 07 16 22 25 27 28 31 35 37 40 55 57 64 65 67 69 71 72 73
68      06/10/2002,07 08 10 14 18 19 20 24 29 34 39 44 48 56 68 73 74 75 77 80
69      06/09/2002,05 08 13 14 15 21 22 23 26 32 34 35 40 45 49 51 58 59 74 76
70      06/08/2002,04 05 07 11 14 17 20 22 28 43 50 51 52 54 57 62 67 69 71 75
71      06/07/2002,01 04 05 09 13 16 19 24 28 29 46 51 52 56 58 60 73 74 75 77
72      06/06/2002,01 03 04 07 09 24 26 31 33 36 47 51 56 58 64 72 75 76 77 79
```

```
73      06/05/2002,02 08 10 11 12 13 17 18 22 31 33 38 42 45 51 54 55 58 67 68
74      06/04/2002,07 09 15 20 26 32 41 42 47 54 58 62 63 65 68 69 70 76 78 79
75      06/03/2002,01 03 14 20 25 30 31 34 35 36 38 42 45 57 58 62 72 73 77 79
76      06/02/2002,03 06 14 24 26 30 31 39 47 49 51 54 59 65 67 68 69 72 77 78
77      06/01/2002,01 02 10 12 15 16 21 27 34 38 44 48 50 52 57 63 66 68 73 80
78      05/31/2002,04 05 09 11 14 19 22 26 40 51 54 63 64 66 68 71 75 77 78 80
79      05/30/2002,07 08 10 14 19 20 21 24 28 31 33 38 40 45 53 62 63 65 73 77
80      05/29/2002,04 05 08 10 19 20 22 31 32 35 42 43 49 50 51 54 55 57 60 68
81      05/28/2002,09 18 24 26 29 31 34 39 42 49 52 54 55 58 61 62 65 72 74 79
82      05/27/2002,02 03 07 08 09 13 18 29 30 32 38 51 54 60 62 64 65 71 72 76
83      05/26/2002,02 07 11 13 14 15 16 28 31 34 35 41 47 57 61 62 70 71 72 76
84      05/25/2002,01 02 06 07 08 09 10 12 22 32 33 35 41 44 50 65 66 72 73 74
85      05/24/2002,08 10 13 14 15 18 19 21 23 30 35 39 40 41 43 55 58 67 73 76
86      05/23/2002,01 04 12 19 20 23 24 25 28 29 32 37 57 62 72 73 75 76 77 78
87      05/22/2002,10 18 22 24 32 33 35 37 38 42 44 47 52 53 54 57 58 65 76 78
88      05/21/2002,07 10 17 28 34 35 37 38 42 44 45 46 49 53 57 62 67 69 71 77
89      05/20/2002,01 03 04 06 11 12 21 26 29 35 37 39 48 50 53 62 63 66 72 75
90      05/19/2002,04 05 08 11 13 33 34 37 39 40 44 46 50 55 56 62 67 71 73 80
91      05/18/2002,03 04 06 14 15 17 20 22 24 25 28 29 41 44 45 55 56 61 66 76
92      05/17/2002,01 04 12 14 15 19 23 30 31 33 37 38 40 43 48 59 63 65 69 73
93      05/16/2002,12 13 14 15 25 27 30 35 38 39 40 43 47 54 58 59 63 65 67 75
94      05/15/2002,05 08 10 11 12 16 19 22 25 31 33 34 36 40 56 65 70 72 74 78
95      05/14/2002,01 07 10 14 21 22 24 35 36 42 46 47 51 60 61 64 65 67 74 80
96      05/13/2002,01 03 09 14 20 21 22 23 37 43 48 50 51 53 55 57 58 60 76 79
97      05/12/2002,09 13 14 18 19 21 31 35 36 37 41 46 54 55 56 59 60 70 71 77
98      05/11/2002,03 09 10 11 15 21 32 34 39 40 41 47 48 49 52 59 61 70 73 80
99      05/10/2002,02 04 05 08 10 16 18 23 26 30 40 42 47 48 50 56 64 73 74 79
00      05/09/2002,09 10 11 12 14 17 18 20 22 30 37 42 45 48 51 54 63 74 76 80
01      05/08/2002,01 02 07 08 15 16 18 22 23 25 28 35 37 38 62 63 64 65 71 80
02      05/07/2002,06 16 17 20 21 24 34 37 47 49 50 53 57 60 61 62 65 74 75 79
03      05/06/2002,04 08 09 15 16 20 27 36 38 41 46 47 49 50 51 52 55 58 71 78
04      05/05/2002,10 13 22 24 28 30 31 33 38 39 46 48 56 60 66 69 70 71 75 76
05      05/04/2002,06 07 08 14 24 25 26 44 45 46 47 49 51 56 63 68 69 76 77 79
06      05/03/2002,02 07 14 15 19 23 26 30 32 35 40 41 51 61 62 64 72 73 77 80
07      05/02/2002,02 05 06 10 22 23 24 27 29 35 38 39 44 48 51 56 58 63 69 78
08      05/01/2002,02 09 10 11 26 27 36 38 41 45 49 53 55 57 59 64 71 73 74 79
09      04/30/2002,01 02 06 11 15 16 27 29 30 33 35 37 38 46 52 56 57 64 66 80
10      04/29/2002,05 13 16 18 22 27 31 32 33 40 45 47 50 52 55 71 73 74 77 79
11      04/28/2002,02 03 06 15 23 24 25 27 30 32 40 43 46 49 53 56 63 68 72 74
12      04/27/2002,02 04 07 16 23 28 29 31 34 35 42 44 49 50 51 55 57 63 65 80
13      04/26/2002,03 09 12 17 19 20 25 26 28 43 46 47 52 57 59 62 65 67 74 77
14      04/25/2002,01 05 08 09 10 14 15 23 38 44 47 49 50 51 53 59 64 65 74 79
15      04/24/2002,01 05 10 18 26 29 34 37 39 42 45 56 57 60 65 66 68 69 74 75
16      04/23/2002,02 08 10 11 12 22 23 31 34 37 43 45 48 51 53 54 62 63 64 65
17      04/22/2002,02 09 13 16 22 31 32 40 42 53 58 63 68 71 72 73 74 76 78 79
18      04/21/2002,03 09 10 13 23 24 26 29 30 34 39 44 55 56 68 69 74 75 76 79
19      04/20/2002,07 14 17 19 26 28 29 31 33 35 44 47 50 53 55 60 61 63 67 74
20      04/19/2002,07 09 17 18 21 26 27 33 37 40 43 44 45 46 51 59 60 70 75 80
21      04/18/2002,04 06 10 12 15 20 22 24 36 38 40 43 50 57 64 65 66 73 76 80
22      04/17/2002,09 11 20 23 26 29 32 35 39 41 42 51 56 62 63 64 69 72 73 74
23      04/16/2002,11 12 17 20 24 31 35 38 42 47 53 54 55 63 64 68 69 73 75 78
24      04/15/2002,04 09 10 14 17 30 36 38 40 41 43 49 51 53 61 62 65 68 71 73
25      04/14/2002,01 05 08 10 11 17 25 27 36 39 44 50 51 59 61 63 66 74 76 79
26      04/13/2002,04 09 15 18 19 23 25 30 34 38 43 50 51 53 58 62 63 67 68 73
```

```
27      04/12/2002,01 04 08 18 20 21 26 34 36 37 38 39 40 45 47 48 55 57 59 69
28      04/11/2002,01 03 06 07 12 16 21 30 31 33 41 51 55 56 57 58 72 73 76 77
29      04/10/2002,06 10 21 22 26 30 36 43 44 47 52 53 55 58 60 62 64 65 67 68
30      04/09/2002,09 11 20 22 28 36 44 48 50 51 52 53 57 58 64 65 66 67 68 71
31      04/08/2002,03 12 13 15 18 24 25 27 28 32 45 50 51 52 59 60 70 73 74 76
32      04/07/2002,02 06 07 08 23 33 36 37 40 41 42 45 51 52 55 58 63 68 70 71
33      04/06/2002,06 08 09 10 13 15 17 22 30 33 38 43 46 50 52 62 67 69 74 75
34      04/05/2002,03 09 10 11 12 16 21 26 31 32 33 41 47 48 50 55 66 67 69 79
35      04/04/2002,05 06 07 23 24 26 32 37 46 49 50 56 57 61 68 69 74 75 79 80
36      04/03/2002,08 10 16 17 19 20 25 35 38 39 41 42 50 56 59 61 64 68 74 79
37      04/02/2002,02 03 07 11 13 14 17 18 29 30 39 42 50 51 61 62 68 69 73 74
38      04/01/2002,02 05 12 16 24 31 32 40 41 42 46 47 55 59 60 61 69 73 76 79
39      03/31/2002,02 11 13 14 15 22 30 31 39 44 46 50 52 54 60 62 63 75 79 80
40      03/30/2002,01 08 09 10 20 21 22 29 32 35 39 42 44 49 53 61 62 71 77 79
41      03/29/2002,02 04 14 16 18 19 20 21 22 23 47 48 50 51 52 53 61 74 76 79
42      03/28/2002,07 13 15 16 19 25 27 28 29 39 40 41 43 46 58 59 63 67 69 75
43      03/27/2002,08 10 13 16 18 20 31 35 40 42 44 48 49 50 51 60 67 69 72 76
44      03/26/2002,02 05 06 09 11 13 19 24 35 36 43 45 48 52 53 65 69 70 71 77
45      03/25/2002,07 14 15 18 29 34 35 40 48 49 52 54 65 68 69 70 72 77 78 80
46      03/24/2002,01 02 07 11 13 19 22 23 30 32 37 38 45 46 56 59 62 68 74 77
47      03/23/2002,01 09 10 12 14 34 40 42 44 45 47 49 50 53 54 57 69 73 77 80
48      03/22/2002,01 04 14 15 16 17 22 30 42 43 44 45 49 51 61 62 68 69 70 76
49      03/21/2002,03 09 21 23 25 26 28 31 33 38 42 44 51 53 60 68 71 72 73 75
50      03/20/2002,01 06 09 12 21 22 30 32 33 36 39 43 47 49 51 56 58 64 65 74
51      03/19/2002,18 20 23 26 28 29 33 34 40 43 49 52 57 58 62 65 66 68 77 79
52      03/18/2002,02 08 11 12 15 22 23 24 30 31 34 39 53 57 62 63 67 69 71 76
53      03/17/2002,03 04 05 09 13 17 19 22 26 29 44 49 50 57 61 62 66 72 73 76
54      03/16/2002,03 05 09 10 12 20 21 22 25 28 35 36 46 48 52 53 55 58 64 71
55      03/15/2002,18 19 26 33 35 38 46 47 52 54 56 57 58 60 61 65 69 71 74 76
56      03/14/2002,05 07 11 15 19 20 21 24 27 28 30 32 39 40 44 48 55 56 79 80
57      03/13/2002,13 15 16 19 24 27 32 35 39 44 45 47 57 68 69 70 74 75 78 79
58      03/12/2002,03 05 10 15 24 26 31 32 41 43 45 46 47 50 60 65 66 68 73 80
59      03/11/2002,01 03 04 06 07 08 11 16 24 25 27 28 30 33 48 59 61 69 71 80
60      03/10/2002,04 05 08 14 16 18 23 25 27 45 47 51 54 58 61 66 69 72 77 78
61      03/09/2002,06 07 16 17 20 23 25 33 43 48 49 50 61 62 66 67 71 72 78 80
62      03/08/2002,04 06 09 11 20 22 26 29 48 49 52 53 54 55 62 63 73 78 79 80
63      03/07/2002,02 11 14 19 29 30 31 36 38 39 42 43 50 52 54 60 70 74 75 76
64      03/06/2002,01 02 03 06 07 16 22 31 32 34 39 41 45 50 58 63 65 70 76 77
65      03/05/2002,03 08 13 16 19 20 24 34 35 36 40 44 45 54 56 65 68 70 76 77
66      03/04/2002,15 16 19 20 21 25 29 35 39 48 49 53 55 69 72 74 76 77 78 80
67      03/03/2002,05 13 14 18 21 22 24 28 33 42 51 56 58 61 62 63 72 75 77 78
68      03/02/2002,06 08 11 15 17 20 22 28 33 36 38 39 43 45 48 51 53 67 73 75
69      03/01/2002,03 04 07 09 10 12 16 27 32 35 36 39 41 49 55 65 68 72 73 79
70      02/28/2002,02 03 14 20 21 26 27 32 33 36 47 49 51 55 56 57 63 70 77 79
71      02/27/2002,02 03 10 13 16 17 22 25 28 29 32 33 44 45 47 53 59 64 66 69
72      02/26/2002,01 07 09 11 12 18 25 26 29 30 44 56 59 60 63 66 68 70 75 77
73      02/25/2002,08 12 13 20 24 28 32 35 42 45 55 56 59 62 63 70 72 75 77 79
74      02/24/2002,01 05 08 10 12 14 20 21 24 26 27 37 39 41 42 43 52 56 58 78
75      02/23/2002,02 06 08 09 12 20 27 35 37 38 39 42 43 54 63 65 66 70 71 79
76      02/22/2002,06 12 23 24 26 28 29 31 33 34 35 40 41 44 46 54 61 63 76 78
77      02/21/2002,05 12 13 14 15 23 24 28 31 34 40 42 44 49 52 54 56 60 68 77
78      02/20/2002,01 03 11 14 15 20 29 34 44 47 49 53 55 56 63 67 68 69 71 74
79      02/19/2002,02 04 05 06 11 12 15 33 37 38 40 41 45 47 60 67 68 69 74 78
80      02/18/2002,09 10 14 17 18 20 24 30 32 40 41 42 44 54 58 62 65 77 79 80
```

```
81      02/17/2002,01 04 14 18 25 26 27 28 34 35 39 41 43 45 46 51 54 61 66 78
82      02/16/2002,04 10 21 22 29 35 40 43 48 51 56 57 58 60 61 63 67 69 70 78
83      02/15/2002,01 04 07 10 15 18 23 26 37 39 40 42 44 59 60 66 67 69 70 73
84      02/14/2002,01 04 13 15 21 23 35 36 47 49 50 53 54 55 56 57 59 65 69 78
85      02/13/2002,02 03 10 12 13 15 17 18 19 26 29 31 34 42 50 54 56 57 75 76
86      02/12/2002,01 02 08 10 13 21 22 23 26 33 42 44 45 52 56 58 62 65 69 74
87      02/11/2002,08 18 21 22 23 30 32 35 37 41 42 46 58 59 60 61 64 66 78 79
88      02/10/2002,03 04 08 09 12 16 20 29 36 38 49 51 52 60 62 67 69 75 78 80
89      02/09/2002,03 06 08 09 13 15 33 36 41 45 46 53 54 55 58 60 64 73 75 79
90      02/08/2002,01 03 05 12 15 20 23 27 32 35 40 43 52 53 59 61 66 68 73 74
91      02/07/2002,06 12 13 15 23 25 27 28 33 44 48 52 53 55 59 65 66 71 73 79
92      02/06/2002,02 11 14 17 22 24 27 32 37 41 48 53 58 66 68 71 73 74 75 76
93      02/05/2002,04 05 06 10 13 16 20 23 27 30 31 34 37 52 60 69 70 73 76 77
94      02/04/2002,02 14 15 20 23 24 26 28 32 39 41 47 49 53 54 55 56 61 63 65
95      02/03/2002,01 07 09 10 11 12 13 14 15 20 26 27 30 31 59 60 63 70 73 77
96      02/02/2002,01 03 04 12 14 23 25 27 31 33 36 41 48 52 56 58 65 66 71 76
97      02/01/2002,01 02 06 10 13 17 26 27 36 39 48 50 57 65 67 71 73 75 77 79
98      01/31/2002,02 04 13 17 22 23 27 36 38 46 49 50 55 56 57 62 64 66 68 77
99      01/30/2002,03 10 13 18 19 21 24 26 40 41 57 58 60 64 67 72 73 74 76 80
00      01/29/2002,02 03 04 05 11 14 24 29 30 32 36 47 50 55 58 62 64 65 67 78
01      01/28/2002,06 10 14 21 24 25 28 48 52 56 57 58 60 61 66 67 71 74 75 79
02      01/27/2002,02 03 04 07 15 16 23 35 41 45 50 52 56 57 59 62 68 72 77 80
03      01/26/2002,02 05 08 10 12 13 14 18 19 23 31 34 44 58 66 67 73 74 75 78
04      01/25/2002,01 03 05 08 13 17 19 20 22 31 32 34 36 42 43 47 52 53 67 70
05      01/24/2002,01 10 14 30 33 36 44 46 48 50 51 52 53 63 64 65 67 71 75 79
06      01/23/2002,09 12 13 17 21 26 27 28 44 46 56 67 68 70 71 73 74 75 77 80
07      01/22/2002,13 14 15 17 27 44 46 47 49 51 52 57 59 62 63 71 72 73 77 80
08      01/21/2002,06 07 10 15 16 21 23 29 35 38 40 53 56 58 64 69 75 77 78 79
09      01/20/2002,05 06 15 19 21 26 29 37 38 41 47 48 50 54 59 66 69 74 75 80
10      01/19/2002,04 05 10 12 15 24 25 26 28 29 32 33 36 47 56 58 70 71 73 77
11      01/18/2002,05 06 07 18 19 21 34 37 40 41 44 52 55 58 59 60 61 73 75 77
12      01/17/2002,01 05 10 11 17 20 21 25 26 30 36 37 44 55 58 65 66 68 69 73
13      01/16/2002,01 03 05 13 16 21 22 24 27 36 37 42 47 52 53 55 56 59 78 80
14      01/15/2002,01 02 04 16 18 19 22 26 27 29 30 32 38 40 47 49 54 57 59 80
15      01/14/2002,04 08 10 18 22 24 30 32 35 38 41 53 62 63 64 75 77 78 79 80
16      01/13/2002,02 04 08 17 18 22 26 27 30 31 35 39 40 41 47 48 54 71 77 78
17      01/12/2002,01 02 12 15 17 23 24 25 28 31 35 37 38 39 40 41 52 66 69 77
18      01/11/2002,01 03 07 13 18 20 21 34 41 42 46 47 55 57 59 65 70 75 76 79
19      01/10/2002,02 09 10 16 20 24 26 34 35 37 38 40 42 43 45 53 59 60 61 74
20      01/09/2002,02 06 12 14 15 19 24 29 30 32 33 37 40 43 47 69 76 77 78 79
21      01/08/2002,01 02 11 12 13 14 22 24 26 30 33 34 46 48 54 59 65 67 71 78
22      01/07/2002,01 02 08 10 11 17 18 21 34 41 53 54 58 61 66 70 71 73 74 75
23      01/06/2002,01 02 03 13 21 24 28 29 34 39 41 49 50 52 57 58 59 64 67 80
24      01/05/2002,14 18 20 23 24 26 27 34 38 42 46 49 50 52 57 60 64 69 77 79
25      01/04/2002,04 05 11 12 14 15 16 36 41 47 49 51 53 55 58 64 71 72 77 79
26      01/03/2002,02 07 13 18 19 21 23 26 27 40 41 43 44 45 52 54 67 68 74 75
27      01/02/2002,06 09 12 14 15 17 19 24 27 36 42 43 47 49 53 54 59 63 74 79
28      01/01/2002,05 06 07 09 10 23 26 27 28 35 41 48 52 55 58 61 65 69 76 80
29      12/31/2001,04 09 12 16 17 23 33 36 38 39 44 51 54 57 60 63 64 67 73 77
30      12/30/2001,03 04 08 17 37 40 41 44 52 57 58 59 60 63 64 66 72 75 78 79
31      12/29/2001,03 07 16 18 24 25 26 29 34 39 43 45 46 57 63 66 68 69 70 74
32      12/28/2001,06 07 12 13 14 29 31 32 33 34 38 41 46 49 53 56 58 70 74 79
33      12/27/2001,03 09 12 14 18 23 27 32 34 35 39 41 52 53 57 58 59 61 63 79
34      12/26/2001,04 05 08 12 13 14 16 19 21 27 30 44 50 53 59 60 61 62 68 80
```

```
12/24/2001,01 04 15 16 23 24 29 31 32 34 40 45 47 50 53 55 61 62 75 80
12/23/2001,02 03 07 11 13 15 16 25 33 35 36 40 41 42 49 52 53 59 77 80
12/22/2001,05 06 09 16 17 27 31 37 47 52 54 55 58 59 61 66 75 76 78 79
12/21/2001,01 08 13 14 15 17 19 24 31 36 37 38 47 51 59 69 71 75 76 80
12/20/2001,06 07 08 10 19 26 28 30 37 40 44 45 49 62 64 68 70 71 76 78
12/19/2001,01 02 10 11 16 20 23 27 28 34 38 41 46 49 63 75 76 78 79 80
12/18/2001,01 05 12 13 15 18 27 29 32 35 40 46 47 49 50 53 54 61 72 79
12/17/2001,01 06 12 15 17 29 30 34 44 49 50 51 52 53 61 64 67 72 77 80
12/16/2001,06 08 15 20 22 23 24 25 29 30 34 37 48 49 55 59 61 63 70 75
12/15/2001,04 14 18 22 24 26 33 35 37 41 46 47 53 56 60 68 74 77 78 80
12/14/2001,01 03 06 08 18 25 36 38 40 44 46 47 48 53 54 61 66 68 72 77
12/13/2001,02 03 05 08 20 37 41 42 45 46 49 51 52 58 61 66 67 70 75 80
12/12/2001,02 06 07 13 17 18 20 28 39 46 48 55 59 61 62 66 68 78 79 80
12/11/2001,10 14 15 19 25 27 28 29 32 33 36 37 40 46 54 56 60 62 72 78
12/10/2001,01 03 07 10 12 19 20 35 40 54 57 59 60 61 62 63 64 69 71 78
12/09/2001,08 13 18 20 25 32 35 36 41 47 50 52 54 57 61 62 73 76 77 78
12/08/2001,02 03 06 10 11 18 23 24 26 29 41 42 49 51 55 64 66 70 78 79
12/07/2001,12 13 14 17 21 27 32 34 36 41 46 49 60 61 65 66 68 70 73 78
12/06/2001,02 05 07 12 15 22 34 41 44 47 50 54 55 59 66 69 72 75 76 79
12/05/2001,01 06 10 12 14 16 18 20 22 52 59 61 63 64 66 67 68 73 77 78
12/04/2001,03 09 17 23 31 35 36 40 46 48 51 54 57 60 65 67 68 72 73 74
12/03/2001,02 03 06 09 12 14 16 22 24 28 36 37 39 42 44 53 56 60 67 73
12/02/2001,07 08 09 11 14 16 17 18 20 24 26 35 37 48 51 56 65 67 71 75
12/01/2001,07 09 12 18 20 23 29 30 31 36 38 40 45 59 66 70 71 74 76 77
11/30/2001,18 20 25 26 27 29 33 35 36 39 40 54 62 63 66 68 69 70 77 79
11/29/2001,07 09 10 16 23 30 35 38 40 45 52 53 57 58 60 61 62 67 68 70
11/28/2001,01 03 17 33 34 35 36 40 41 46 48 53 58 60 64 68 71 73 77 78
11/27/2001,05 09 20 22 24 25 28 32 52 53 54 58 59 69 72 73 74 76 77 79
11/26/2001,01 03 06 09 15 16 19 22 27 29 35 36 41 46 49 51 61 65 66 74
11/25/2001,11 14 16 22 27 32 34 49 52 54 56 57 59 60 61 67 69 71 78 80
11/24/2001,06 11 13 19 26 27 29 31 34 37 44 52 53 56 57 60 64 70 71 78
11/23/2001,03 12 16 18 23 38 42 46 51 55 57 58 61 63 64 66 67 70 72 74
11/22/2001,02 04 08 12 15 16 23 24 30 36 43 44 47 63 68 70 73 74 77 79
11/21/2001,02 03 19 22 27 35 36 38 40 45 48 50 58 64 65 67 71 76 77 80
11/20/2001,04 05 07 26 34 41 44 45 46 50 51 54 58 61 67 70 75 76 77 79
11/19/2001,01 02 08 09 10 17 18 20 24 25 26 27 45 50 52 54 55 65 70 77
11/18/2001,03 11 15 17 18 21 23 26 33 38 42 43 44 48 50 57 60 71 75 78
11/17/2001,03 06 09 12 25 31 34 36 42 48 54 57 59 61 63 64 70 72 73 78
11/16/2001,02 03 10 11 19 21 26 32 37 41 44 45 55 56 58 61 66 71 72 80
11/15/2001,01 05 14 16 20 26 27 29 30 33 34 45 48 54 57 61 62 65 67 71
11/14/2001,05 10 12 13 14 15 18 22 40 47 50 56 57 61 68 69 72 73 77 79
11/13/2001,01 04 09 12 14 16 21 22 24 29 31 34 35 43 61 65 67 70 71 75
11/12/2001,12 13 16 17 24 25 27 32 44 46 47 50 51 56 58 59 67 74 76 78
11/11/2001,04 18 19 23 26 27 28 31 39 40 41 43 45 49 50 54 69 71 72 79
11/10/2001,04 06 11 14 16 21 22 27 28 29 36 39 41 45 48 54 66 70 73 74
11/09/2001,02 06 09 10 14 16 20 28 29 35 36 45 46 52 59 65 69 70 71 76
11/08/2001,05 07 08 09 12 13 20 22 24 26 36 38 41 54 55 62 63 71 73 79
11/07/2001,01 02 09 21 23 25 33 34 42 46 47 49 50 63 67 71 74 75 78 79
11/06/2001,03 11 14 15 16 22 24 27 30 32 36 38 40 47 52 62 63 66 67 79
11/05/2001,02 05 11 27 28 29 30 33 35 37 42 43 56 57 63 64 65 66 67 78
11/04/2001,02 04 05 08 11 16 19 32 34 38 41 44 45 46 51 65 69 71 73 74
11/03/2001,01 12 15 16 20 32 33 38 39 42 44 47 48 49 50 52 70 74 77 78
11/02/2001,01 04 10 12 16 20 21 22 25 44 46 47 48 51 59 62 64 65 73 75
11/01/2001,05 06 11 13 19 28 29 30 37 38 43 46 51 53 54 56 60 66 68 69
```

```
89      10/31/2001,07 21 26 29 30 33 34 37 40 41 51 53 58 60 62 68 70 72 74 76
90      10/30/2001,05 09 16 19 21 27 28 33 37 38 39 44 50 51 52 68 69 73 74 77
91      10/29/2001,09 13 14 18 19 34 36 37 41 50 51 55 58 60 61 65 72 75 77 80
92      10/28/2001,01 06 09 12 15 17 27 28 29 39 44 50 51 55 56 57 60 66 77 79
93      10/27/2001,02 07 10 12 14 21 30 31 36 39 43 45 46 47 54 58 59 62 72 76
94      10/26/2001,01 02 03 04 06 08 18 21 22 23 33 38 40 48 49 51 53 54 64 71
95      10/25/2001,10 13 18 21 26 30 42 44 46 48 54 55 57 58 59 64 67 74 75 76
96      10/24/2001,01 04 07 09 15 18 23 24 33 37 42 47 48 62 65 66 71 72 76 79
97      10/23/2001,04 05 09 10 19 20 31 34 37 38 42 44 50 58 64 66 70 72 74 77
98      10/22/2001,06 15 16 17 18 21 22 30 31 32 34 41 43 47 49 57 73 74 79 80
99      10/21/2001,04 05 08 13 26 28 29 32 34 38 39 40 44 47 55 58 59 67 76 80
00      10/20/2001,02 04 09 14 18 21 26 27 30 42 46 53 55 56 62 63 67 69 74 75
01      10/19/2001,09 15 16 17 25 28 29 30 42 49 53 58 62 64 65 68 69 70 77 80
02      10/18/2001,10 13 25 31 36 37 38 39 40 41 44 46 52 58 60 65 71 73 75 76
03      10/17/2001,08 10 12 30 33 35 39 48 49 51 53 54 55 61 67 68 71 76 78 80
04      10/16/2001,04 09 10 20 24 27 30 32 40 44 45 46 47 59 62 66 70 73 74 78
05      10/15/2001,01 02 12 14 18 20 22 26 28 30 33 40 43 52 53 58 60 68 75 80
06      10/14/2001,01 02 04 05 08 09 13 16 26 28 29 35 36 39 43 45 46 49 65 76
07      10/13/2001,02 03 14 17 18 19 26 35 38 41 46 47 49 53 57 59 65 73 75 80
08      10/12/2001,06 07 11 15 29 32 34 40 42 45 49 53 63 66 67 68 73 75 76 79
09      10/11/2001,01 03 04 15 19 26 31 33 39 43 46 49 63 64 65 69 71 76 77 79
10      10/10/2001,03 04 16 17 18 20 23 24 27 35 36 39 41 42 45 47 57 64 66 75
11      10/09/2001,08 11 12 14 18 21 24 25 26 32 34 35 38 43 44 45 51 67 70 77
12      10/08/2001,03 04 09 11 16 17 18 29 31 33 34 38 42 47 48 58 62 63 69 71
13      10/07/2001,05 09 14 15 25 26 29 34 37 39 41 47 54 56 57 63 66 71 77 78
14      10/06/2001,02 03 11 12 14 15 16 17 20 23 27 33 35 39 45 46 52 58 77 80
15      10/05/2001,04 06 10 16 18 22 23 34 36 40 48 52 53 57 60 61 65 69 75 80
16      10/04/2001,02 05 07 09 12 15 18 20 22 26 30 33 36 37 39 41 48 50 59 79
17      10/03/2001,01 02 04 08 09 13 17 29 31 34 41 45 46 49 51 68 72 75 78 80
18      10/02/2001,13 14 20 24 28 30 32 33 34 40 42 44 52 55 56 61 65 68 77 78
19      10/01/2001,02 14 17 18 25 29 30 34 36 43 49 53 57 60 61 66 68 75 77 78
20      09/30/2001,05 08 12 16 17 28 30 43 44 47 55 57 61 62 64 65 68 76 77 80
21      09/29/2001,02 06 08 10 17 18 20 23 26 29 34 36 39 41 50 57 63 71 79 80
22      09/28/2001,01 02 03 05 11 14 22 25 38 39 42 49 58 61 63 65 73 77 79 80
23      09/27/2001,05 07 09 21 24 27 28 36 40 43 45 48 49 53 54 56 58 59 74 78
24      09/26/2001,02 08 11 14 21 26 29 41 42 46 51 52 56 58 61 63 67 69 76 80
25      09/25/2001,05 06 07 08 09 16 19 29 32 40 41 45 46 48 57 66 71 72 76 78
26      09/24/2001,14 18 23 33 34 38 42 44 46 47 49 56 57 58 62 67 68 71 73 74
27      09/23/2001,01 14 16 20 24 28 31 33 37 42 47 51 52 54 56 64 73 75 79 80
28      09/22/2001,01 02 06 08 12 18 23 24 26 29 30 33 34 44 46 56 61 65 71 80
29      09/21/2001,03 06 17 21 22 24 33 44 45 54 56 58 59 60 61 63 68 73 76 79
30      09/20/2001,05 14 24 25 26 31 34 35 37 38 44 45 50 54 63 65 70 72 74 79
31      09/19/2001,08 09 10 13 15 20 26 38 39 42 47 49 55 66 72 73 74 77 79 80
32      09/18/2001,01 02 06 09 10 13 30 35 36 40 41 44 45 57 58 63 64 71 73 79
33      09/17/2001,02 04 07 17 19 20 26 28 37 39 46 47 51 56 62 67 70 72 73 79
34      09/16/2001,01 20 23 33 35 39 40 41 43 44 45 46 49 55 56 57 61 63 65 78
35      09/15/2001,07 18 20 22 27 28 31 37 39 42 45 48 52 53 55 56 58 74 76 78
36      09/14/2001,03 06 07 09 10 17 19 21 24 31 33 35 38 45 53 57 61 62 73 80
37      09/13/2001,05 06 11 12 18 22 25 29 35 36 40 42 47 51 65 70 72 76 77 80
38      09/12/2001,02 03 04 06 28 29 34 36 38 39 43 45 46 57 65 68 69 73 77 80
39      09/11/2001,06 07 08 22 23 27 30 35 42 48 50 53 59 67 69 70 71 73 76 77
40      09/10/2001,02 03 06 11 22 25 31 39 40 46 47 52 67 68 74 75 76 77 78 80
41      09/09/2001,03 04 07 12 23 27 30 33 35 38 47 49 53 59 64 66 68 72 73 77
42      09/08/2001,04 08 11 17 23 26 27 29 38 51 54 55 59 60 62 67 68 73 77 78
```

```
43      09/07/2001,04 06 08 10 14 25 27 30 31 32 34 37 39 41 42 45 46 72 73 76
44      09/06/2001,01 03 11 12 24 33 39 51 54 58 61 63 64 66 72 74 75 76 78 79
45      09/05/2001,01 18 24 25 29 32 37 38 39 41 42 43 50 52 56 57 63 75 78 79
46      09/04/2001,03 07 13 16 17 28 29 33 34 36 44 50 62 64 69 70 71 74 76 80
47      09/03/2001,02 06 07 13 14 18 22 23 24 27 29 30 38 42 43 49 51 55 63 71
48      09/02/2001,01 02 05 09 10 19 30 32 34 40 42 43 53 56 58 59 69 70 73 77
49      09/01/2001,01 08 10 13 19 23 25 26 33 37 41 43 46 47 49 56 63 65 72 80
50      08/31/2001,03 06 12 16 22 24 25 26 28 30 34 47 51 52 54 57 59 62 65 79
51      08/30/2001,03 10 11 12 13 15 18 23 28 29 30 33 34 36 43 44 45 58 62 79
52      08/29/2001,05 10 16 27 33 34 37 38 39 40 46 48 49 52 54 63 67 70 74 79
53      08/28/2001,06 18 20 25 30 32 37 42 44 49 51 56 57 58 61 65 68 76 78 80
54      08/27/2001,04 07 11 13 18 20 24 26 30 39 40 43 46 52 54 58 59 60 65 78
55      08/26/2001,08 24 25 30 38 39 42 43 44 48 49 52 54 64 67 68 71 72 76 80
56      08/25/2001,01 02 04 11 16 17 18 21 27 31 34 35 42 50 62 68 71 72 75 79
57      08/24/2001,01 09 10 11 13 16 22 27 35 36 39 45 48 50 56 57 61 62 63 70
58      08/23/2001,05 06 08 11 12 18 20 21 25 27 31 36 38 44 59 66 71 73 75 76
59      08/22/2001,11 12 23 24 30 34 37 39 40 42 47 50 54 56 59 66 69 72 73 76
60      08/21/2001,03 07 13 15 20 21 26 31 35 42 44 45 47 56 60 69 71 75 78 79
61      08/20/2001,05 07 13 14 19 28 29 30 38 40 45 47 49 57 58 59 65 68 71 79
62      08/19/2001,01 08 12 17 22 29 30 44 46 47 51 53 56 60 61 62 64 73 79 80
63      08/18/2001,01 02 03 14 20 23 25 36 39 42 47 53 56 57 60 62 64 71 77 78
64      08/17/2001,02 04 10 14 19 20 21 23 31 32 38 39 46 51 54 60 61 70 72 80
65      08/16/2001,02 05 10 11 12 13 20 23 34 42 48 58 60 61 66 69 74 75 79 80
66      08/15/2001,02 04 08 10 14 20 24 26 30 36 41 50 52 56 58 61 63 70 71 78
67      08/14/2001,01 06 07 10 12 31 34 35 36 45 48 49 55 57 60 61 62 70 77 80
68      08/13/2001,02 05 15 23 32 36 39 43 44 45 46 47 51 52 54 59 70 72 76 79
69      08/12/2001,04 05 08 10 15 20 24 25 29 30 33 34 46 48 66 71 72 73 77 80
70      08/11/2001,06 08 09 11 19 20 25 29 32 33 35 40 44 49 52 63 68 76 77 79
71      08/10/2001,04 07 08 10 11 16 27 30 34 37 40 46 52 54 57 65 67 71 76 77
72      08/09/2001,01 04 11 14 17 32 36 38 39 46 53 59 63 65 67 68 71 72 75 76
73      08/08/2001,03 06 12 16 19 21 26 37 41 43 47 54 63 64 72 75 76 78 79 80
74      08/07/2001,02 04 07 26 31 32 34 38 47 48 50 58 60 68 73 74 76 77 78 80
75      08/06/2001,05 10 13 16 19 25 28 31 40 44 45 49 51 53 57 60 62 67 70 75
76      08/05/2001,03 04 05 09 12 14 15 21 25 34 35 36 50 54 59 60 61 67 71 75
77      08/04/2001,01 03 07 17 21 23 24 33 34 35 36 39 45 47 57 61 65 70 72 73
78      08/03/2001,01 14 15 20 25 33 37 46 49 50 53 54 56 57 59 67 73 75 78 79
79      08/02/2001,04 10 13 14 15 22 35 40 43 44 46 51 52 53 55 56 58 59 60 67
80      08/01/2001,02 03 04 05 07 10 12 15 24 30 31 36 39 43 44 54 59 67 68 69
81      07/31/2001,01 02 05 09 21 22 24 25 26 28 37 38 42 56 58 59 64 67 74 77
82      07/30/2001,03 05 13 14 17 28 34 35 38 40 41 43 50 54 58 64 65 67 73 75
83      07/29/2001,02 04 07 09 10 11 15 17 21 28 34 40 46 52 54 55 56 57 65 70
84      07/28/2001,05 15 16 30 32 37 44 46 47 49 51 53 55 59 62 64 67 70 74 77
85      07/27/2001,01 07 10 11 16 20 23 25 26 33 39 40 41 46 49 53 57 72 76 78
86      07/26/2001,02 05 08 17 24 26 28 31 38 42 44 46 56 58 63 64 68 71 72 77
87      07/25/2001,01 05 11 16 19 28 30 33 34 44 46 51 54 59 62 65 69 71 77 78
88      07/24/2001,01 04 18 22 24 33 35 36 38 43 45 49 53 59 65 66 67 68 74 77
89      07/23/2001,12 17 18 22 23 25 32 34 45 50 52 55 57 58 62 75 76 78 79 80
90      07/22/2001,01 03 07 10 31 34 36 43 48 49 51 52 56 58 63 67 68 71 76 80
91      07/21/2001,03 07 11 15 19 26 33 37 38 40 44 47 58 60 63 69 70 72 78 79
92      07/20/2001,06 09 13 14 19 22 35 40 45 47 48 49 50 56 65 68 72 74 76 77
93      07/19/2001,01 04 08 11 14 17 21 23 30 40 43 48 52 56 58 60 65 66 70 80
94      07/18/2001,01 22 29 33 34 41 42 45 47 52 53 59 62 64 69 74 75 76 77 78
95      07/17/2001,01 04 08 09 15 20 23 28 31 32 39 40 41 51 54 56 60 63 78 80
96      07/16/2001,07 08 20 21 23 30 32 39 41 43 48 56 57 59 61 62 63 71 73 80
```

```
97      07/15/2001,01 03 08 13 17 20 21 24 26 35 48 49 51 53 57 65 67 69 76 78
98      07/14/2001,09 12 13 14 17 27 30 31 36 40 41 43 46 56 59 61 66 67 72 79
99      07/13/2001,06 08 20 21 30 36 38 39 43 50 55 59 63 64 65 68 71 72 74 79
00      07/12/2001,01 02 03 07 09 13 15 17 25 26 39 41 42 44 47 52 59 61 67 71
01      07/11/2001,01 02 03 05 13 18 22 32 36 39 49 50 51 52 57 68 72 73 78 80
02      07/10/2001,02 03 11 19 20 26 32 33 39 44 50 51 53 62 65 66 71 72 73 74
03      07/09/2001,02 03 05 10 11 15 23 37 45 46 52 53 57 59 60 63 71 72 75 78
04      07/08/2001,01 15 18 20 26 30 31 32 34 37 43 44 46 47 49 58 60 62 67 68
05      07/07/2001,07 09 13 19 26 28 29 30 31 32 38 49 50 53 56 62 70 71 73 78
06      07/06/2001,03 09 12 15 20 24 35 40 43 49 52 55 58 59 62 67 68 76 78 79
07      07/05/2001,03 07 09 23 32 34 35 41 43 44 46 53 65 67 68 70 74 75 76 79
08      07/04/2001,01 02 06 21 27 29 32 33 37 38 46 54 56 59 67 69 70 74 75 76
09      07/03/2001,04 09 14 15 18 22 24 27 30 36 37 38 39 40 43 54 56 68 69 70
10      07/02/2001,02 16 17 22 30 32 33 39 40 41 45 46 52 56 59 61 67 70 73 76
11      07/01/2001,01 04 10 14 17 22 23 27 40 41 43 49 50 63 65 66 75 76 79 80
12      06/30/2001,08 15 19 21 25 26 29 36 40 47 49 51 52 56 57 58 60 67 71 79
13      06/29/2001,01 02 04 06 08 10 16 18 22 26 30 35 37 52 58 63 67 73 74 75
14      06/28/2001,01 02 04 06 11 15 16 17 19 20 24 34 35 43 50 55 60 64 69 71
15      06/27/2001,02 07 13 16 18 26 27 34 37 38 44 48 58 60 61 64 65 73 78 79
16      06/26/2001,04 09 12 13 14 15 18 20 27 28 32 34 37 44 52 56 59 64 72 79
17      06/25/2001,03 04 08 09 11 23 26 27 34 36 40 51 58 61 62 63 65 67 70 75
18      06/24/2001,02 05 06 08 12 13 15 18 26 33 38 45 47 48 50 64 67 68 74 76
19      06/23/2001,10 22 23 26 29 31 37 40 41 42 43 50 51 53 57 59 68 73 79 80
20      06/22/2001,04 06 10 14 18 21 22 24 27 30 40 46 54 58 68 71 72 75 78 80
21      06/21/2001,09 10 12 15 18 19 20 21 22 35 37 40 53 56 58 64 68 71 73 78
22      06/20/2001,01 02 03 09 11 17 23 25 26 32 35 41 42 49 62 67 69 75 76 79
23      06/19/2001,01 07 08 10 12 16 17 18 33 39 46 54 58 65 66 67 68 71 74 76
24      06/18/2001,01 11 14 19 20 24 39 47 50 52 53 54 55 58 71 72 74 76 78 80
25      06/17/2001,01 10 12 15 16 19 24 34 36 40 46 50 51 55 59 64 71 72 74 80
26      06/16/2001,07 09 24 32 37 38 43 45 46 49 50 54 55 57 60 61 62 64 71 78
27      06/15/2001,04 05 06 08 17 21 27 28 29 38 46 49 55 56 65 66 68 73 77 80
28      06/14/2001,02 03 06 09 12 16 19 30 31 32 33 37 40 49 50 63 71 73 77 80
29      06/13/2001,12 15 20 31 35 36 40 41 46 48 51 52 53 54 57 58 60 62 72 75
30      06/12/2001,02 03 04 08 16 20 22 26 27 40 46 50 53 56 61 62 63 69 75 79
31      06/11/2001,06 07 08 18 20 23 26 28 29 38 39 41 44 48 54 55 60 64 72 79
32      06/10/2001,03 05 07 10 11 20 22 23 24 27 30 53 59 65 66 70 72 73 75 79
33      06/09/2001,05 09 15 16 18 20 31 36 46 50 52 59 64 65 69 70 74 76 78 80
34      06/08/2001,05 07 09 11 14 15 17 20 23 30 37 39 42 47 55 57 65 68 73 80
35      06/07/2001,02 04 05 13 19 20 24 26 30 33 43 45 52 55 56 66 69 76 79 80
36      06/06/2001,01 03 05 06 13 14 24 26 30 33 43 48 53 56 61 63 67 71 76 79
37      06/05/2001,04 16 17 26 29 34 37 40 42 43 45 48 50 57 65 66 68 73 77 78
38      06/04/2001,02 03 06 23 29 30 33 37 40 41 55 56 58 59 63 70 74 77 78 79
39      06/03/2001,07 10 11 13 16 23 28 30 35 37 42 46 49 51 54 56 57 59 60 74
40      06/02/2001,01 04 06 10 12 21 25 28 33 36 39 42 56 57 63 64 68 71 74 77
41      06/01/2001,02 16 17 20 24 26 31 36 42 43 47 49 53 56 61 67 70 72 75 80
42      05/31/2001,02 14 20 23 25 28 31 32 34 42 43 49 50 52 55 58 66 75 76 80
43      05/30/2001,09 13 15 18 21 22 28 29 30 32 35 40 44 49 62 66 67 68 74 75
44      05/29/2001,01 03 04 06 07 08 09 15 21 31 51 53 54 62 70 72 74 76 77 79
45      05/28/2001,03 08 09 13 15 16 17 33 35 41 44 49 53 61 62 63 67 71 78 79
46      05/27/2001,02 03 14 15 25 36 37 39 40 42 43 53 55 59 63 67 69 70 73 75
47      05/26/2001,03 06 14 17 22 24 25 29 32 33 45 50 56 58 60 63 66 71 72 76
48      05/25/2001,11 22 26 27 29 30 31 33 37 41 42 47 51 52 53 60 61 65 70 80
49      05/24/2001,02 06 08 09 10 13 16 30 34 40 41 45 47 50 52 53 55 59 79 80
50      05/23/2001,02 06 15 19 23 27 36 37 45 48 51 53 57 58 63 70 71 72 75 79
```

```
51      05/22/2001,02 03 21 24 27 29 33 35 38 39 47 48 51 59 60 62 67 73 75 76
52      05/21/2001,07 08 10 12 21 28 32 37 39 40 50 56 60 61 62 63 67 71 73 75
53      05/20/2001,08 09 12 15 16 19 31 40 42 47 49 51 59 60 61 66 67 74 75 77
54      05/19/2001,07 12 15 17 19 20 24 27 28 31 38 41 51 53 55 57 63 65 75 80
55      05/18/2001,06 08 11 13 19 20 32 34 36 44 46 47 49 50 51 57 61 63 64 78
56      05/17/2001,03 07 13 15 16 19 23 27 36 39 40 43 56 60 61 65 66 74 75 77
57      05/16/2001,07 11 12 13 19 20 22 33 36 39 44 46 51 55 58 61 70 72 73 78
58      05/15/2001,05 06 08 16 18 20 25 38 39 45 47 50 60 61 69 72 73 76 77 79
59      05/14/2001,02 03 09 12 14 18 22 24 27 35 39 45 49 54 58 59 61 64 66 71
60      05/13/2001,02 04 06 07 12 18 19 22 33 36 47 48 49 51 61 62 63 66 74 76
61      05/12/2001,01 04 05 07 09 11 14 24 29 34 39 45 48 60 64 66 72 73 77 78
62      05/11/2001,03 20 23 28 29 30 31 34 35 37 39 45 53 56 59 64 65 70 75 79
63      05/10/2001,01 03 04 08 14 19 26 28 30 43 48 52 53 57 64 66 70 72 78 79
64      05/09/2001,03 12 17 22 23 24 30 32 36 38 39 43 50 51 54 59 63 66 70 78
65      05/08/2001,02 05 06 09 24 35 39 41 42 43 44 50 62 64 66 67 70 71 73 75
66      05/07/2001,03 04 06 09 11 12 20 34 35 38 45 48 59 62 63 66 73 78 79 80
67      05/06/2001,03 10 13 16 17 20 39 45 47 51 52 55 56 60 61 62 64 71 72 73
68      05/05/2001,05 06 09 12 14 15 18 19 33 35 36 40 42 50 58 59 61 73 75 76
69      05/04/2001,01 03 08 09 21 25 26 27 29 33 34 38 42 44 49 52 59 65 67 75
70      05/03/2001,06 07 08 17 26 28 29 38 39 42 44 48 49 52 53 60 64 73 76 77
71      05/02/2001,03 06 08 13 19 20 24 30 31 39 43 53 54 55 62 69 70 71 73 79
72      05/01/2001,05 10 11 12 14 18 21 25 37 39 40 42 45 52 54 57 68 70 75 78
73      04/30/2001,14 16 17 18 24 25 26 30 31 32 37 41 43 44 57 63 66 70 72 76
74      04/29/2001,03 05 07 08 10 13 23 24 28 36 38 45 46 53 56 58 60 67 71 72
75      04/28/2001,01 02 08 18 19 24 33 34 42 43 48 51 53 64 66 67 68 71 72 80
76      04/27/2001,01 02 09 10 11 14 17 19 28 44 46 50 52 53 54 56 57 63 67 68
77      04/26/2001,08 16 17 18 21 22 24 26 33 34 36 40 44 47 50 53 55 59 67 75
78      04/25/2001,05 06 08 09 15 20 21 26 29 34 38 39 52 62 66 67 68 71 72 75
79      04/24/2001,03 08 09 12 17 19 21 22 23 26 35 44 46 48 51 58 66 67 72 76
80      04/23/2001,04 05 11 16 20 24 26 27 28 33 34 42 43 45 52 53 63 72 76 77
81      04/22/2001,02 17 18 20 27 28 43 49 53 57 60 61 64 67 68 70 77 78 79 80
82      04/21/2001,01 10 12 17 21 29 30 34 40 44 54 56 60 62 66 72 73 74 75 77
83      04/20/2001,01 04 09 10 14 21 25 33 36 38 43 44 45 50 55 63 68 71 74 80
84      04/19/2001,01 09 17 31 32 35 36 44 46 49 50 57 62 63 66 68 70 76 77 80
85      04/18/2001,04 08 09 21 22 26 28 32 33 34 35 51 52 60 64 66 69 72 75 79
86      04/17/2001,02 08 09 10 11 21 25 26 29 31 33 34 35 42 50 59 62 67 71 72
87      04/16/2001,05 09 10 12 19 21 22 24 30 34 43 50 53 56 58 61 64 69 78 79
88      04/15/2001,08 09 13 15 18 19 26 31 33 34 41 44 49 55 56 65 69 71 73 77
89      04/14/2001,02 12 15 18 19 20 29 30 37 40 41 44 46 49 55 70 74 76 78 80
90      04/13/2001,02 03 05 07 09 19 20 25 36 37 42 43 56 60 64 69 70 71 76 79
91      04/12/2001,03 04 05 06 07 17 21 22 23 28 35 40 44 48 53 54 58 64 68 79
92      04/11/2001,02 05 08 09 12 13 15 21 24 30 31 39 53 56 64 71 74 75 77 78
93      04/10/2001,04 06 11 12 14 22 27 29 31 43 48 51 52 55 60 64 66 67 68 72
94      04/09/2001,04 05 06 11 21 23 24 30 35 39 44 51 56 59 62 63 64 69 71 73
95      04/08/2001,18 24 25 27 29 31 32 36 38 49 50 52 59 60 61 62 66 68 69 77
96      04/07/2001,06 09 18 21 22 26 27 33 34 37 38 42 49 53 55 56 68 70 71 79
97      04/06/2001,03 09 15 17 23 28 29 31 32 34 35 37 41 45 48 49 58 70 75 79
98      04/05/2001,01 06 21 22 29 33 35 39 41 42 43 48 50 51 54 60 61 66 67 77
99      04/04/2001,03 20 22 23 26 33 37 41 44 49 52 53 55 59 62 64 73 74 75 76
00      04/03/2001,02 09 10 13 18 19 21 22 24 30 32 34 46 48 49 60 61 64 76 78
01      04/02/2001,04 08 10 13 16 20 24 25 30 32 37 39 41 45 48 49 50 57 62 73
02      04/01/2001,05 09 13 28 32 33 37 40 41 47 51 53 59 60 65 69 71 75 78 79
03      03/31/2001,02 03 05 06 07 08 09 12 14 20 24 25 27 30 52 53 62 69 70 79
04      03/30/2001,03 09 11 17 20 21 35 40 41 42 48 50 51 57 58 66 67 71 76 80
```

```
05      03/29/2001,09 10 17 21 29 32 36 44 46 52 54 55 56 59 63 68 71 72 79 80
06      03/28/2001,01 09 10 12 14 15 26 36 38 44 48 50 53 57 58 62 68 75 79 80
07      03/27/2001,08 10 14 16 18 23 24 28 31 34 35 46 47 56 57 61 62 66 70 80
08      03/26/2001,01 04 08 14 19 20 28 32 41 46 48 49 50 52 53 56 59 62 68 69
09      03/25/2001,03 09 11 12 13 24 34 35 38 39 50 51 55 58 60 61 69 72 75 79
10      03/24/2001,02 14 15 17 22 27 30 37 43 45 47 50 51 60 62 66 72 73 77 79
11      03/23/2001,03 05 12 19 26 32 35 37 42 43 45 46 51 52 59 61 63 66 72 77
12      03/22/2001,07 09 15 18 27 29 37 39 46 47 53 55 59 60 66 70 74 75 76 80
13      03/21/2001,02 05 09 12 15 17 19 39 40 41 44 45 47 48 50 59 63 71 74 76
14      03/20/2001,01 02 04 06 08 14 22 33 34 35 39 41 42 43 44 54 59 64 70 74
15      03/19/2001,06 08 11 13 14 19 20 25 26 29 33 37 38 43 58 59 66 70 72 74
16      03/18/2001,01 03 06 11 17 20 32 38 39 47 55 56 57 61 62 64 69 73 74 79
17      03/17/2001,06 07 08 14 15 16 25 28 32 41 46 55 56 57 59 60 64 67 72 74
18      03/16/2001,03 06 07 08 12 16 17 23 31 32 38 40 50 54 61 69 71 72 73 76
19      03/15/2001,04 08 09 15 20 21 27 30 36 37 38 44 52 54 55 56 64 65 70 76
20      03/14/2001,05 06 08 09 11 15 24 28 29 34 35 41 42 51 54 59 64 68 71 80
21      03/13/2001,14 18 20 24 26 28 30 47 49 56 57 62 63 64 65 66 71 73 74 77
22      03/12/2001,03 07 08 13 18 24 25 30 33 36 41 42 51 54 60 61 64 65 73 80
23      03/11/2001,12 18 19 21 24 27 29 36 39 48 51 56 57 58 68 71 72 74 75 78
24      03/10/2001,06 08 14 15 16 20 25 38 39 44 45 46 51 60 61 68 72 74 76 77
25      03/09/2001,03 06 09 11 20 28 38 46 54 55 56 60 62 64 68 72 73 74 77 80
26      03/08/2001,10 12 15 18 22 25 26 27 32 35 42 45 49 52 58 59 61 64 76 78
27      03/07/2001,03 06 07 08 09 15 27 29 30 39 42 46 54 58 59 62 65 68 70 74
28      03/06/2001,05 07 10 12 15 16 18 28 36 39 40 51 52 57 62 63 66 68 74 80
29      03/05/2001,07 08 14 21 25 31 34 35 51 52 53 56 58 68 69 71 73 75 76 80
30      03/04/2001,02 06 10 14 17 21 25 31 38 40 42 43 45 50 51 56 58 59 65 79
31      03/03/2001,03 04 16 21 24 25 28 38 40 44 46 52 54 56 59 63 65 75 76 77
32      03/02/2001,02 12 18 22 24 25 26 32 36 37 49 51 52 53 61 66 69 75 79 80
33      03/01/2001,05 12 14 16 24 26 27 29 35 51 52 54 60 62 66 71 72 73 75 78
34      02/28/2001,03 06 08 09 10 19 26 29 33 34 35 36 42 46 49 64 67 69 70 79
35      02/27/2001,03 04 09 10 14 19 23 26 29 33 39 41 43 46 48 57 68 74 77 78
36      02/26/2001,03 10 14 15 17 22 23 26 29 37 38 44 46 50 52 60 61 62 73 77
37      02/25/2001,03 05 13 15 16 20 27 30 32 33 39 41 42 44 46 52 59 62 72 77
38      02/24/2001,06 09 11 21 26 29 31 32 35 42 43 44 48 50 53 57 72 75 76 78
39      02/23/2001,01 07 08 10 21 25 31 33 39 42 44 45 46 47 49 62 66 78 79 80
40      02/22/2001,07 10 14 18 19 23 24 27 28 29 44 46 47 51 57 59 65 67 69 77
41      02/21/2001,05 09 13 17 18 21 22 29 32 34 43 49 56 57 63 64 65 69 71 74
42      02/20/2001,07 10 12 13 20 22 23 26 32 35 37 40 54 55 60 62 63 73 75 79
43      02/19/2001,01 02 03 16 18 21 24 26 27 30 33 44 50 55 58 67 70 73 77 80
44      02/18/2001,01 04 10 12 15 16 19 26 32 33 37 42 43 44 57 67 69 71 74 79
45      02/17/2001,07 09 10 14 20 22 23 29 51 54 56 58 59 63 65 68 72 74 78 80
46      02/16/2001,06 08 10 11 16 19 20 23 35 41 43 47 48 58 61 65 70 74 75 77
47      02/15/2001,01 03 04 07 18 25 31 34 36 37 39 40 45 56 57 59 64 66 72 76
48      02/14/2001,10 13 14 19 20 23 26 37 42 50 54 55 57 59 63 67 68 72 74 79
49      02/13/2001,01 02 04 10 11 25 27 34 41 46 47 51 52 57 59 61 65 68 75 79
50      02/12/2001,11 16 23 24 33 34 35 42 54 61 67 71 72 73 74 75 77 78 79 80
51      02/11/2001,04 06 10 15 21 24 29 31 36 43 47 55 57 63 64 72 74 76 77 78
52      02/10/2001,03 06 13 17 22 25 38 39 47 51 54 56 57 60 62 65 66 70 77 80
53      02/09/2001,01 02 04 05 07 19 20 22 23 25 34 43 45 46 50 56 57 60 66 78
54      02/08/2001,01 04 12 20 21 25 26 29 31 37 40 44 50 55 56 62 63 64 68 76
55      02/07/2001,02 05 11 14 32 34 37 38 40 42 47 58 64 66 72 73 74 75 76 77
56      02/06/2001,03 21 22 24 26 27 30 33 34 37 40 41 49 51 53 54 58 62 73 78
57      02/05/2001,07 08 09 11 13 21 32 34 43 44 45 53 54 56 62 63 68 69 75 80
58      02/04/2001,02 03 08 13 14 15 17 18 24 26 27 41 45 50 51 53 56 63 64 78
```

```
59      02/03/2001,04 07 13 14 18 21 28 38 46 47 55 58 60 61 65 66 71 72 74 76
60      02/02/2001,07 14 15 19 23 25 31 34 36 39 41 55 57 58 59 63 65 68 71 72
61      02/01/2001,11 14 16 17 22 25 30 31 33 34 36 37 41 42 44 48 57 70 75 80
62      01/31/2001,03 08 15 19 21 23 27 38 45 46 53 54 55 57 64 65 66 71 74 80
63      01/30/2001,05 07 08 09 10 11 14 34 40 42 43 45 48 52 54 61 67 68 70 78
64      01/29/2001,02 04 07 11 23 26 29 30 32 33 35 38 39 41 46 57 63 65 70 78
65      01/28/2001,01 02 06 08 09 10 22 24 26 28 29 32 42 45 62 67 70 71 73 79
66      01/27/2001,02 03 06 10 11 14 15 16 23 25 26 31 32 47 51 64 71 72 73 78
67      01/26/2001,06 09 11 18 20 21 23 24 27 29 31 36 43 45 53 56 58 60 66 74
68      01/25/2001,04 06 10 12 14 22 26 30 32 33 35 40 46 47 50 55 63 65 67 74
69      01/24/2001,02 03 04 07 09 14 17 20 25 30 32 35 36 45 48 55 58 63 66 71
70      01/23/2001,04 11 13 15 18 20 23 24 31 38 39 40 42 43 50 51 52 53 59 74
71      01/22/2001,02 06 14 15 16 19 28 35 37 47 48 50 52 53 57 59 62 63 65 77
72      01/21/2001,01 02 26 31 32 33 35 36 44 45 50 51 53 59 68 69 71 76 77 78
73      01/20/2001,07 10 12 22 23 24 25 31 32 38 40 45 48 53 54 60 65 66 68 69
74      01/19/2001,02 05 12 16 19 31 34 36 37 40 43 44 58 60 61 63 70 71 72 75
75      01/18/2001,05 06 22 25 26 27 35 44 47 55 59 61 64 65 66 68 69 70 72 74
76      01/17/2001,01 07 10 15 21 22 30 33 39 45 55 58 60 62 65 69 70 71 79 80
77      01/16/2001,01 07 09 12 13 18 21 22 25 32 36 38 39 47 60 62 64 68 73 74
78      01/15/2001,06 18 19 21 23 27 34 37 38 39 47 60 64 66 67 69 70 72 74 80
79      01/14/2001,04 05 06 07 21 22 27 28 33 35 41 42 46 54 60 63 70 78 79 80
80      01/13/2001,07 09 12 21 27 33 36 41 42 45 49 54 59 60 64 67 71 75 77 78
81      01/12/2001,07 08 10 11 15 20 21 23 24 25 39 40 42 43 48 50 62 68 70 74
82      01/11/2001,01 03 08 10 16 17 24 34 35 37 48 50 52 53 56 60 66 67 76 79
83      01/10/2001,08 10 16 17 23 26 27 30 38 39 40 48 56 62 63 64 67 68 74 77
84      01/09/2001,12 13 15 17 24 25 30 33 34 36 42 43 49 53 54 68 72 73 74 80
85      01/08/2001,03 04 10 12 13 18 23 24 31 41 43 47 54 57 59 63 67 69 74 79
86      01/07/2001,08 09 12 31 33 34 35 43 49 50 53 54 63 67 72 74 75 76 78 79
87      01/06/2001,02 07 08 09 14 18 22 30 31 35 39 42 47 53 61 64 68 70 73 79
88      01/05/2001,01 05 08 09 12 16 21 25 27 29 32 33 42 45 48 65 67 71 74 76
89      01/04/2001,01 09 19 20 23 27 29 31 36 38 41 46 54 62 69 70 76 77 78 80
90      01/03/2001,05 07 08 10 16 22 27 30 32 34 36 44 47 48 51 62 72 73 74 75
91      01/02/2001,01 02 15 19 21 24 29 31 32 33 35 39 41 44 55 59 69 73 78 80
92      01/01/2001,08 14 16 18 31 33 36 37 42 45 48 50 52 54 58 67 69 71 76 77
93      12/31/2000,02 03 04 05 08 14 15 17 29 33 45 46 51 55 58 64 69 70 73 80
94      12/30/2000,03 06 12 15 16 19 21 22 31 32 36 43 49 51 54 63 64 68 78 79
95      12/29/2000,20 22 24 26 27 28 29 36 38 39 47 54 60 65 66 67 73 76 78 79
96      12/28/2000,04 09 11 13 15 18 19 22 24 31 35 36 44 51 54 59 63 67 74 78
97      12/27/2000,01 02 14 24 26 32 33 38 39 41 42 44 52 54 58 59 64 65 68 80
98      12/26/2000,01 08 13 14 15 16 21 23 28 29 33 34 43 51 56 59 61 62 78 80
99      12/24/2000,02 03 09 11 12 18 22 32 37 38 41 59 61 62 63 67 70 74 77 78
00      12/23/2000,02 03 06 07 09 10 12 22 29 32 37 43 45 51 53 57 64 70 75 77
01      12/22/2000,04 05 21 22 31 32 33 34 36 38 40 41 45 48 57 59 70 72 75 80
02      12/21/2000,14 18 21 22 23 24 28 32 35 37 45 48 49 52 55 56 61 62 67 79
03      12/20/2000,01 02 03 08 15 19 22 26 29 32 34 40 46 50 54 58 61 69 71 74
04      12/19/2000,03 06 07 11 13 16 18 29 30 31 38 39 47 51 54 56 68 71 72 80
05      12/18/2000,01 02 03 05 07 11 12 16 21 34 39 43 50 60 61 66 67 69 77 79
06      12/17/2000,19 20 24 26 27 30 37 42 43 46 47 50 53 54 56 57 59 61 64 67
07      12/16/2000,09 23 36 38 40 45 46 49 50 51 57 58 59 64 65 68 71 72 73 80
08      12/15/2000,03 05 07 08 12 14 20 27 28 29 32 35 45 46 65 66 70 73 74 80
09      12/14/2000,01 08 09 12 18 19 20 27 34 49 54 57 61 64 66 71 72 74 75 76
10      12/13/2000,12 14 16 18 24 33 36 37 38 43 45 47 49 54 56 60 65 66 67 76
11      12/12/2000,06 07 13 16 31 33 36 39 49 57 61 63 68 69 70 72 73 75 78 79
12      12/11/2000,05 07 15 19 22 24 31 32 36 38 41 44 45 49 51 66 67 71 73 74
```

```
13      12/10/2000,03 06 13 18 23 25 37 42 50 57 59 60 61 66 72 74 75 77 78 80
14      12/09/2000,01 04 06 09 10 20 23 31 33 34 35 42 43 49 52 54 64 70 74 79
15      12/08/2000,02 05 06 10 12 25 26 30 31 34 40 46 50 51 52 57 63 65 69 77
16      12/07/2000,08 11 14 15 18 20 22 23 24 36 39 41 43 47 54 56 60 63 64 69
17      12/06/2000,01 03 13 17 18 26 41 47 50 53 54 57 58 60 62 66 70 74 78 80
18      12/05/2000,02 08 12 14 16 18 21 24 25 27 35 40 53 58 60 67 68 71 77 78
19      12/04/2000,05 09 12 13 14 15 21 22 23 27 29 31 32 34 42 43 44 53 57 63
20      12/03/2000,01 20 21 22 23 25 31 34 38 39 49 52 57 65 66 67 68 69 70 74
21      12/02/2000,01 08 14 16 22 27 34 35 39 46 49 52 55 57 59 64 70 74 78 79
22      12/01/2000,01 02 06 13 21 23 31 36 40 41 42 43 48 52 54 60 67 71 78 79
23      11/30/2000,05 11 13 19 23 26 29 32 36 37 38 40 52 53 58 62 65 69 70 72
24      11/29/2000,05 11 12 16 18 24 27 28 30 38 40 42 45 47 53 54 62 65 72 76
25      11/28/2000,03 04 10 12 15 22 25 32 36 37 46 49 50 52 53 56 67 68 69 76
26      11/27/2000,01 11 15 19 20 23 25 34 36 41 42 45 51 52 57 58 66 67 72 76
27      11/26/2000,02 05 06 12 13 14 15 16 20 21 30 32 34 38 43 45 47 49 61 65
28      11/25/2000,01 02 06 07 16 20 22 33 34 36 40 41 51 52 61 63 64 65 67 76
29      11/24/2000,01 06 09 13 15 17 18 22 23 30 33 35 39 40 59 63 70 73 76 80
30      11/23/2000,02 08 20 23 25 26 31 34 37 38 39 47 49 50 54 61 62 68 74 76
31      11/22/2000,06 09 12 18 19 27 28 33 35 46 50 54 60 61 63 66 67 69 76 78
32      11/21/2000,05 13 17 22 31 35 36 39 40 43 44 45 47 52 53 54 56 57 75 79
33      11/20/2000,01 05 08 20 21 25 36 37 38 40 41 45 49 51 57 58 60 69 70 79
34      11/19/2000,06 10 15 16 17 21 28 30 33 38 41 44 47 50 52 62 65 69 74 76
35      11/18/2000,01 02 07 09 10 15 16 18 19 39 42 49 60 64 69 73 75 76 77 80
36      11/17/2000,01 04 09 11 12 13 16 20 22 31 43 49 51 54 61 64 72 75 76 77
37      11/16/2000,08 09 11 18 28 29 36 38 39 40 41 45 52 55 58 60 73 75 76 80
38      11/15/2000,06 08 09 16 24 25 26 36 37 41 42 45 55 59 61 65 67 71 74 77
39      11/14/2000,05 09 10 15 18 20 22 26 29 30 32 38 42 63 64 65 68 71 72 78
40      11/13/2000,03 04 06 08 26 27 32 35 39 41 45 48 49 51 52 54 58 59 69 72
41      11/12/2000,04 05 10 12 15 17 18 22 28 34 40 42 55 57 58 61 69 70 73 75
42      11/11/2000,08 11 12 13 14 17 19 23 24 28 29 31 34 42 44 56 62 65 67 74
43      11/10/2000,05 07 08 10 13 17 21 26 30 44 52 55 59 61 62 71 73 74 76 77
44      11/09/2000,02 03 05 08 09 18 29 33 36 37 38 39 47 56 58 59 68 73 74 80
45      11/08/2000,06 08 10 11 17 24 36 38 39 42 49 52 53 56 60 61 63 65 68 80
46      11/07/2000,02 12 14 16 21 24 27 29 30 37 41 45 46 49 62 64 66 68 70 77
47      11/06/2000,05 14 15 18 22 37 38 46 47 50 53 55 58 61 62 64 69 70 73 80
48      11/05/2000,02 06 07 10 11 13 18 21 23 27 31 34 49 52 53 61 63 64 65 71
49      11/04/2000,01 08 09 17 18 21 25 29 32 37 38 39 41 49 56 66 67 69 78 79
50      11/03/2000,03 04 09 10 12 14 16 28 29 43 44 46 55 56 57 63 73 74 77 80
51      11/02/2000,03 05 09 15 16 22 28 30 37 38 47 54 55 58 59 66 67 72 73 80
52      11/01/2000,05 10 16 23 28 31 35 47 52 53 59 60 61 62 67 68 69 70 72 73
53      10/31/2000,02 04 07 10 11 19 25 33 34 36 45 56 58 59 62 64 73 74 77 78
54      10/30/2000,01 04 08 10 14 15 16 20 22 25 28 35 39 41 44 50 52 53 54 65
55      10/29/2000,02 05 10 14 17 19 21 22 25 26 28 30 37 45 53 57 60 61 74 77
56      10/28/2000,02 03 05 06 09 16 26 28 48 49 50 51 54 57 62 64 69 72 75 78
57      10/27/2000,11 19 21 22 27 28 29 32 33 36 38 43 45 49 51 63 67 71 78 79
58      10/26/2000,02 12 16 21 27 30 52 55 58 59 65 66 72 73 74 75 76 77 78 80
59      10/25/2000,11 16 20 37 38 39 40 41 42 43 44 47 50 53 58 61 70 74 75 80
60      10/24/2000,01 02 05 06 08 09 16 19 22 27 28 30 36 41 43 49 64 65 76 79
61      10/23/2000,01 05 13 17 24 27 30 37 42 44 46 51 53 60 63 64 65 70 71 72
62      10/22/2000,02 04 09 12 14 23 25 28 31 33 34 35 38 43 45 46 49 51 72 78
63      10/21/2000,01 05 11 13 16 17 24 28 29 36 37 44 45 50 55 64 69 71 74 75
64      10/20/2000,07 11 14 16 24 26 28 29 38 39 41 49 50 51 55 66 69 74 76 78
65      10/19/2000,02 09 11 12 16 18 19 22 24 33 34 36 37 41 42 47 49 54 69 80
66      10/18/2000,02 09 14 15 18 22 24 27 31 34 36 40 42 47 51 55 59 61 68 73
```

```
67      10/17/2000,02 06 10 12 18 24 29 30 34 40 46 50 52 53 57 65 66 67 68 75
68      10/16/2000,05 11 12 13 15 29 34 35 45 49 54 56 59 60 64 67 68 75 78 80
69      10/15/2000,02 05 07 31 32 35 39 40 43 48 51 53 54 55 56 57 61 62 73 79
70      10/14/2000,06 11 17 18 21 28 32 33 37 45 48 49 53 54 56 73 75 78 79 80
71      10/13/2000,03 04 06 13 15 18 19 23 24 27 37 39 44 50 51 55 57 60 67 74
72      10/12/2000,04 11 19 20 23 29 30 31 33 37 39 42 50 54 63 69 75 76 79 80
73      10/11/2000,02 04 06 08 13 19 21 36 40 44 47 51 62 71 72 73 74 75 77 79
74      10/10/2000,03 05 06 08 16 21 24 25 29 30 38 44 50 54 56 59 60 63 71 75
75      10/09/2000,01 05 07 12 15 17 26 33 39 41 44 47 49 50 52 58 63 66 77 79
76      10/08/2000,01 02 05 06 09 15 23 28 31 38 45 46 54 55 59 60 62 74 75 79
77      10/07/2000,02 03 04 08 12 13 21 33 37 38 47 56 59 60 64 68 71 74 77 79
78      10/06/2000,03 06 10 11 12 16 21 22 28 34 47 48 49 57 58 62 65 71 77 79
79      10/05/2000,03 06 08 09 12 17 25 33 35 37 39 43 45 50 55 60 61 64 66 71
80      10/04/2000,04 10 11 16 22 30 37 53 54 55 59 62 63 65 68 72 73 74 75 77
81      10/03/2000,10 16 18 19 22 36 40 43 45 48 49 53 54 56 59 64 67 76 77 79
82      10/02/2000,06 09 12 20 24 26 35 38 41 45 49 51 52 55 58 64 67 75 77 80
83      10/01/2000,10 12 18 20 22 23 24 25 26 35 37 52 54 55 61 63 65 68 71 75
84      09/30/2000,07 13 16 18 19 20 26 31 34 35 41 45 54 55 56 66 68 74 76 78
85      09/29/2000,03 06 08 12 13 16 17 24 25 28 30 31 40 46 49 54 58 64 70 79
86      09/28/2000,01 07 09 14 26 27 28 30 34 38 40 43 46 52 54 65 66 68 70 76
87      09/27/2000,05 13 17 21 22 23 25 26 29 35 36 46 53 56 57 63 66 69 76 78
88      09/26/2000,02 03 04 10 13 15 20 40 43 44 46 47 51 61 69 70 73 76 79 80
89      09/25/2000,01 02 08 11 14 28 29 30 32 34 41 45 48 50 55 56 63 64 65 66
90      09/24/2000,01 02 19 22 27 30 32 34 37 40 41 48 50 58 65 70 72 74 75 80
91      09/23/2000,03 04 10 12 15 16 28 43 45 48 49 50 52 53 58 60 61 64 65 75
92      09/22/2000,01 03 08 10 13 25 29 34 40 44 49 52 54 57 59 65 70 73 74 77
93      09/21/2000,01 03 04 05 06 07 09 19 20 23 27 36 45 46 47 49 53 57 61 72
94      09/20/2000,03 04 05 23 25 26 33 34 38 39 40 44 46 50 51 52 53 66 67 69
95      09/19/2000,02 06 09 22 28 29 30 34 37 41 46 49 54 58 67 70 71 75 79 80
96      09/18/2000,04 06 10 12 15 16 22 31 32 34 36 41 42 44 46 52 60 62 64 75
97      09/17/2000,03 05 07 18 20 21 23 25 29 32 35 37 38 43 45 52 53 55 65 67
98      09/16/2000,01 02 04 11 12 18 20 22 26 37 38 49 52 53 54 61 65 68 72 74
99      09/15/2000,03 06 08 15 17 19 24 46 47 49 51 54 58 59 61 62 70 74 77 78
00      09/14/2000,06 08 17 18 20 22 28 29 33 34 38 44 46 58 60 64 66 68 70 72
01      09/13/2000,01 02 04 05 08 09 15 16 20 33 43 53 55 56 59 62 73 75 77 79
02      09/12/2000,03 04 06 08 09 15 26 30 34 38 39 40 50 63 64 70 73 75 78 79
03      09/11/2000,06 07 14 21 23 29 35 36 38 41 50 51 53 64 65 66 68 71 77 79
04      09/10/2000,01 02 04 05 10 30 31 33 38 39 42 46 52 56 68 69 72 74 79 80
05      09/09/2000,02 09 14 15 16 24 25 29 33 36 41 49 50 53 57 58 66 70 72 73
06      09/08/2000,08 10 11 15 24 28 37 45 47 50 53 56 62 64 65 66 68 73 76 77
07      09/07/2000,01 04 16 21 29 30 33 36 37 41 44 49 54 55 56 59 65 70 73 78
08      09/06/2000,06 07 10 13 27 30 31 35 38 39 41 42 44 54 60 61 63 69 74 76
09      09/05/2000,05 06 08 09 30 31 36 43 45 46 47 52 53 55 56 61 63 64 70 80
10      09/04/2000,10 11 17 18 19 25 27 29 37 39 40 45 46 51 56 57 62 74 75 80
11      09/03/2000,01 02 06 09 10 14 15 33 38 43 47 52 54 60 69 72 74 76 77 80
12      09/02/2000,02 07 09 12 17 19 24 29 46 49 57 60 65 68 71 73 74 76 77 79
13      09/01/2000,05 06 10 11 12 15 26 27 34 38 40 46 52 58 59 60 62 63 64 67
14      08/31/2000,04 09 12 13 14 23 24 28 38 44 55 56 58 59 62 65 68 72 73 78
15      08/30/2000,04 08 09 11 13 14 16 17 22 27 30 33 39 48 50 57 61 66 67 79
16      08/29/2000,01 03 06 11 13 21 23 27 34 36 45 55 59 64 65 69 74 76 77 80
17      08/28/2000,08 12 13 14 15 25 28 36 38 40 42 43 44 49 51 52 58 71 76 78
18      08/27/2000,01 13 19 28 29 34 37 38 39 45 53 56 63 64 65 70 74 77 79 80
19      08/26/2000,03 07 14 16 31 32 36 39 42 43 55 56 58 70 72 73 75 76 77 78
20      08/25/2000,05 13 14 16 18 22 23 25 27 30 31 36 41 46 49 56 60 62 79 80
```

```
08/24/2000,02 06 08 11 14 19 23 24 26 29 33 35 41 46 56 58 59 60 66 74
08/23/2000,01 15 17 18 20 23 31 35 40 41 43 48 49 53 56 64 73 76 78 80
08/22/2000,09 11 12 16 17 19 20 22 26 29 32 33 34 40 43 45 46 53 66 79
08/21/2000,03 04 06 14 24 26 27 30 33 35 38 42 46 49 57 67 71 72 74 78
08/20/2000,02 03 07 09 11 14 16 28 29 30 31 32 54 55 62 65 66 71 74 79
08/19/2000,05 06 10 12 20 22 29 32 33 39 40 41 46 49 58 59 61 63 68 75
08/18/2000,01 18 19 20 28 33 34 39 41 42 43 48 50 59 65 67 75 76 77 78
08/17/2000,08 10 11 12 18 28 32 33 36 39 43 50 51 53 60 63 64 69 77 79
08/16/2000,05 08 09 12 16 22 30 34 35 37 53 59 62 63 65 71 75 76 77 79
08/15/2000,06 07 10 15 20 23 24 26 30 37 38 40 43 51 52 53 57 69 77 79
08/14/2000,07 13 17 19 26 27 30 31 35 44 47 49 53 54 57 62 72 75 77 79
08/13/2000,12 15 18 19 20 27 29 31 35 37 40 43 44 51 58 60 63 69 72 80
08/12/2000,08 12 24 25 27 30 32 34 39 40 41 50 51 56 61 66 67 68 69 78
08/11/2000,02 09 11 13 23 29 35 37 41 44 48 51 53 56 58 62 63 64 65 74
08/10/2000,07 21 23 26 28 29 30 31 32 34 40 43 48 52 56 60 62 74 76 80
08/09/2000,03 06 09 13 20 21 38 43 44 46 48 49 54 59 64 66 71 75 77 78
08/08/2000,03 08 12 17 20 22 25 28 29 31 32 45 55 57 60 66 67 69 70 77
08/07/2000,05 10 12 17 21 23 27 30 31 45 46 47 54 55 56 59 69 76 77 78
08/06/2000,04 10 15 24 25 26 31 33 38 40 43 51 53 57 59 60 61 68 73 79
08/05/2000,05 07 09 24 26 28 30 32 39 41 46 52 54 55 57 68 70 72 73 78
08/04/2000,06 07 10 17 19 34 46 51 52 55 56 58 59 64 66 67 71 76 78 80
08/03/2000,02 06 18 27 33 37 39 42 46 47 52 56 57 58 63 68 73 75 78 80
08/02/2000,01 13 17 19 21 26 27 28 29 31 39 42 50 53 57 59 61 75 79 80
08/01/2000,03 04 06 13 29 31 34 39 42 45 48 52 55 56 58 62 63 69 70 79
07/31/2000,03 05 07 11 13 16 22 29 32 33 40 47 54 62 64 66 76 77 78 80
07/30/2000,02 07 09 10 13 20 23 29 41 46 47 54 55 57 61 62 66 67 68 73
07/29/2000,02 03 06 12 17 30 34 35 38 40 43 44 48 53 61 63 67 71 78 80
07/28/2000,01 06 08 09 10 15 19 25 39 42 43 48 49 50 57 58 59 75 77 80
07/27/2000,01 02 04 22 26 29 31 32 36 37 39 41 43 46 50 55 58 64 67 71
07/26/2000,05 06 13 17 22 26 27 30 31 34 44 58 60 61 64 68 74 75 78 80
07/25/2000,01 02 05 07 08 13 24 29 36 38 48 49 56 59 64 66 67 75 79 80
07/24/2000,01 02 07 08 09 10 15 24 27 33 35 36 50 52 53 57 60 66 67 73
07/23/2000,03 06 08 16 19 22 26 29 30 36 43 52 57 58 60 62 66 72 73 77
07/22/2000,06 07 08 12 14 18 21 25 29 30 31 32 38 43 59 61 63 64 67 68
07/21/2000,10 13 16 17 18 19 29 33 34 35 38 39 43 45 46 47 51 52 53 68
07/20/2000,01 06 08 09 10 11 13 18 19 22 29 36 42 43 48 51 59 67 71 80
07/19/2000,03 04 05 06 08 13 14 20 21 27 33 37 47 58 59 62 64 67 76 77
07/18/2000,07 23 25 26 30 32 36 42 47 49 50 52 57 59 69 70 76 77 78 80
07/17/2000,02 05 15 34 39 42 45 49 50 54 58 61 63 66 68 73 74 78 79 80
07/16/2000,02 11 12 15 18 29 32 37 41 44 51 56 62 63 67 68 69 71 72 76
07/15/2000,02 05 06 16 21 25 33 37 43 45 48 51 58 60 62 64 65 71 73 74
07/14/2000,06 11 17 22 25 28 29 31 41 42 56 57 58 59 66 68 74 76 79 80
07/13/2000,01 07 08 11 19 26 31 32 37 38 44 47 48 56 58 60 62 63 71 73
07/12/2000,04 05 13 18 21 25 29 34 46 50 52 54 63 66 69 70 74 75 76 80
07/11/2000,01 04 10 14 15 21 28 30 35 36 44 46 47 48 49 51 53 67 73 74
07/10/2000,03 04 09 10 11 13 15 29 32 37 38 42 50 56 58 59 61 63 65 75
07/09/2000,06 08 11 14 17 18 23 29 32 34 38 39 40 53 59 65 71 73 76 80
07/08/2000,08 09 10 12 15 17 18 19 25 26 29 30 32 38 48 49 57 62 67 79
07/07/2000,03 04 08 10 18 19 21 25 26 33 34 36 41 42 43 64 68 75 76 78
07/06/2000,01 10 11 14 16 17 19 25 32 33 34 35 48 54 61 65 67 74 76 77
07/05/2000,01 04 05 07 11 13 14 18 20 22 23 25 29 30 32 36 37 49 51 66
07/04/2000,04 10 12 17 21 30 34 36 37 40 44 45 46 52 53 62 63 65 68 76
07/03/2000,02 05 06 07 17 19 20 22 24 31 38 40 45 57 61 63 65 72 74 78
07/02/2000,03 07 09 10 11 15 16 22 27 28 45 47 59 60 72 73 74 75 76 77
```

```
75      07/01/2000,02 12 17 21 24 38 40 42 44 46 49 51 59 60 61 62 69 70 72 78
76      06/30/2000,03 07 12 18 20 22 27 47 48 49 50 52 55 58 72 73 74 75 76 78
77      06/29/2000,08 09 13 22 23 24 26 29 30 31 36 37 38 39 40 49 59 62 67 73
78      06/28/2000,01 05 07 09 15 21 23 26 36 41 42 48 53 54 55 58 65 73 76 80
79      06/27/2000,06 10 14 18 20 22 24 31 37 40 41 46 58 61 62 64 65 72 73 75
80      06/26/2000,05 10 11 13 14 15 19 26 32 33 34 41 50 51 52 58 60 65 67 80
81      06/25/2000,05 08 14 18 23 26 32 40 43 48 54 55 57 62 63 65 69 74 77 78
82      06/24/2000,06 11 12 14 15 22 26 31 37 39 49 50 51 52 56 57 59 70 78 79
83      06/23/2000,02 04 07 11 12 20 26 27 33 35 45 48 54 57 61 67 71 73 75 80
84      06/22/2000,01 04 12 16 22 28 29 36 40 41 42 43 45 46 51 59 61 65 69 74
85      06/21/2000,03 06 08 09 14 15 18 21 25 36 45 50 56 57 59 61 62 69 70 79
86      06/20/2000,05 08 09 11 13 15 19 21 23 24 36 44 52 54 61 68 69 70 75 77
87      06/19/2000,03 12 20 21 23 27 40 41 43 44 47 53 55 56 58 65 66 70 78 79
88      06/18/2000,01 02 06 16 21 23 27 34 36 38 42 44 45 52 59 63 64 65 66 67
89      06/17/2000,02 05 08 11 14 19 27 29 31 34 36 45 49 52 53 56 57 68 75 77
90      06/16/2000,04 05 10 19 20 21 26 27 33 35 42 44 45 46 49 57 62 69 74 77
91      06/15/2000,06 07 10 14 16 18 19 20 26 27 29 41 51 59 65 66 68 71 72 78
92      06/14/2000,02 07 13 15 16 21 22 28 33 34 35 37 44 52 53 59 61 66 69 78
93      06/13/2000,10 14 18 21 26 31 32 33 35 36 39 44 48 53 56 62 65 71 75 79
94      06/12/2000,01 02 12 13 19 23 24 27 34 39 42 46 49 50 56 57 67 69 71 80
95      06/11/2000,01 05 08 10 12 16 23 24 34 37 47 48 50 51 52 57 62 68 72 74
96      06/10/2000,01 04 08 13 16 19 26 31 32 36 37 43 45 47 49 51 63 66 72 74
97      06/09/2000,02 13 15 25 26 31 33 35 40 46 48 54 59 65 67 69 74 75 77 78
98      06/08/2000,01 03 04 08 12 13 16 21 23 29 33 34 47 48 52 60 72 73 78 79
99      06/07/2000,02 04 08 10 13 23 28 32 34 35 42 50 51 54 58 59 63 71 77 80
00      06/06/2000,06 07 12 16 21 24 26 27 28 31 37 44 47 54 57 66 70 78 79 80
01      06/05/2000,02 03 05 12 18 22 28 31 39 41 48 49 59 63 69 70 71 73 75 79
02      06/04/2000,01 02 03 04 11 17 18 20 23 27 31 36 38 43 53 56 58 69 73 76
03      06/03/2000,03 04 10 13 15 22 24 25 33 35 38 42 49 56 61 63 71 75 76 78
04      06/02/2000,03 05 07 08 09 24 28 36 42 44 52 53 54 58 62 67 69 71 74 80
05      06/01/2000,05 06 07 08 12 15 18 19 21 24 28 29 31 32 47 52 54 56 65 66
06      05/31/2000,02 06 14 18 20 22 24 33 38 39 42 44 48 52 56 57 64 65 74 78
07      05/30/2000,03 18 19 27 35 37 47 48 51 53 54 59 61 68 69 70 72 74 79 80
08      05/29/2000,03 05 06 11 15 23 24 28 30 31 34 37 40 48 49 51 55 57 63 77
09      05/28/2000,03 04 05 06 07 12 15 17 21 22 23 28 30 31 35 37 40 59 65 68
10      05/27/2000,01 02 09 10 12 18 20 24 30 38 42 46 51 52 55 62 63 65 70 78
11      05/26/2000,02 08 09 10 18 19 28 31 33 37 39 43 45 46 52 55 58 59 66 69
12      05/25/2000,02 09 12 13 17 20 23 24 33 36 39 42 43 49 52 58 66 70 71 76
13      05/24/2000,11 17 29 30 34 36 43 46 49 51 52 54 61 62 64 65 67 68 72 73
14      05/23/2000,11 14 17 18 22 24 36 39 43 45 46 54 55 59 66 73 76 77 78 79
15      05/22/2000,01 13 15 26 32 35 39 46 50 53 54 57 62 68 69 70 71 74 77 78
16      05/21/2000,04 06 07 10 11 14 22 24 27 31 38 39 46 52 56 61 62 74 76 77
17      05/20/2000,01 04 10 20 28 32 33 41 46 47 52 56 57 60 63 64 65 74 75 77
18      05/19/2000,10 17 22 30 32 39 43 45 47 49 58 61 66 67 72 73 74 75 77 80
19      05/18/2000,01 02 09 10 12 13 14 17 21 23 29 41 43 47 52 62 65 67 71 72
20      05/17/2000,02 08 12 13 17 22 23 25 26 30 38 50 56 58 59 63 64 67 72 77
21      05/16/2000,06 11 13 14 21 26 28 32 34 37 44 45 49 54 55 58 63 69 71 75
22      05/15/2000,09 16 17 23 25 27 29 31 43 45 54 56 58 59 63 64 70 71 74 77
23      05/14/2000,05 10 11 19 28 35 38 42 43 47 52 53 54 55 61 64 65 70 71 74
24      05/13/2000,03 04 07 23 28 34 36 39 40 43 46 52 53 55 62 64 68 70 79 80
25      05/12/2000,01 08 09 15 18 19 24 26 29 34 40 45 46 50 53 56 59 67 71 72
26      05/11/2000,01 02 04 09 25 31 36 37 38 44 47 51 53 56 59 60 62 71 75 78
27      05/10/2000,09 10 18 20 27 36 39 42 45 47 53 62 65 66 67 69 71 75 76 78
28      05/09/2000,04 07 12 20 21 22 25 26 27 31 35 36 46 58 64 71 73 74 75 77
```

```
29      05/08/2000,05 06 08 10 17 18 19 20 23 37 46 51 54 59 60 61 62 72 74 75
30      05/07/2000,05 09 13 21 26 27 28 35 38 43 51 53 56 61 62 65 67 69 74 75
31      05/06/2000,05 06 08 12 13 16 29 33 35 43 44 46 60 65 67 72 75 78 79 80
32      05/05/2000,02 04 05 07 08 12 18 22 27 28 32 38 50 54 68 69 70 71 78 79
33      05/04/2000,03 06 34 37 38 41 44 46 47 50 51 53 56 58 64 68 70 71 75 80
34      05/03/2000,06 09 10 25 32 37 39 40 41 42 48 52 55 58 59 65 66 77 78 80
35      05/02/2000,01 04 09 11 12 13 15 16 19 39 42 45 49 51 56 60 66 67 75 80
36      05/01/2000,05 10 13 16 18 28 29 30 37 43 44 49 54 60 63 64 68 69 73 79
37      04/30/2000,02 03 05 06 08 09 15 17 19 22 35 36 40 41 43 44 52 63 66 77
38      04/29/2000,04 14 15 20 25 29 33 37 39 41 44 45 52 54 58 62 69 72 75 76
39      04/28/2000,01 03 04 05 06 13 15 29 32 43 45 46 48 54 57 60 64 69 78 80
40      04/27/2000,01 04 06 07 09 17 19 20 22 23 24 30 36 37 39 54 56 71 72 78
41      04/26/2000,05 09 10 11 17 22 26 36 44 46 48 55 59 63 64 66 70 72 74 75
42      04/25/2000,02 03 04 05 10 11 13 14 17 19 22 24 26 27 28 36 43 51 66 79
43      04/24/2000,01 04 05 06 11 13 17 21 23 26 28 37 44 55 57 58 64 70 73 79
44      04/23/2000,02 08 12 14 19 22 24 25 26 28 35 36 45 48 57 60 66 73 77 80
45      04/22/2000,10 14 16 17 20 22 26 30 42 43 49 60 63 65 66 72 73 74 76 78
46      04/21/2000,01 04 09 10 14 15 17 22 25 27 41 52 55 56 57 62 67 74 76 79
47      04/20/2000,03 10 16 17 18 24 25 30 37 39 41 44 46 55 60 62 65 74 75 79
48      04/19/2000,02 05 08 13 18 26 27 31 35 36 43 47 50 53 56 59 62 63 64 76
49      04/18/2000,02 09 10 12 13 16 18 21 22 25 30 39 42 46 47 53 66 70 72 74
50      04/17/2000,07 08 15 21 24 26 28 31 33 34 43 44 49 51 57 62 72 73 77 79
51      04/16/2000,01 08 09 15 20 22 28 37 41 42 55 56 58 63 64 70 73 76 77 78
52      04/15/2000,03 05 06 08 11 13 18 22 30 32 37 43 44 49 51 52 61 62 64 74
53      04/14/2000,05 06 16 21 28 32 33 37 42 43 46 52 54 57 60 65 67 68 73 75
54      04/13/2000,01 02 09 12 26 31 32 36 46 47 49 51 52 54 56 57 59 62 65 73
55      04/12/2000,01 04 05 14 19 24 27 28 39 41 45 51 54 56 58 70 73 74 77 79
56      04/11/2000,07 08 10 23 24 25 32 34 38 47 48 50 54 59 61 66 72 75 76 80
57      04/10/2000,02 04 06 17 19 20 32 33 42 43 44 46 49 57 61 64 69 74 79 80
58      04/09/2000,02 08 14 15 19 21 25 32 39 41 45 51 53 58 59 60 64 72 73 76
59      04/08/2000,05 14 18 19 31 34 35 37 39 40 45 47 49 52 53 54 56 60 73 79
60      04/07/2000,02 03 05 12 17 19 20 26 29 32 34 40 41 44 48 55 58 65 67 74
61      04/06/2000,10 13 14 16 25 26 28 30 32 37 38 41 42 45 46 52 57 59 74 76
62      04/05/2000,10 12 14 18 20 21 24 29 30 31 34 47 49 54 56 64 68 70 77 79
63      04/04/2000,01 06 08 10 12 19 29 36 41 42 44 45 50 52 55 63 64 65 68 75
64      04/03/2000,02 05 07 09 11 12 13 15 26 28 35 41 44 48 49 52 56 60 67 70
65      04/02/2000,01 02 07 10 11 13 14 15 17 29 46 49 50 51 55 61 67 68 71 77
66      04/01/2000,02 08 10 13 16 18 19 21 27 42 43 44 45 48 50 60 61 63 75 78
67      03/31/2000,01 02 03 05 11 17 32 34 36 42 43 44 45 46 51 55 59 72 74 79
68      03/30/2000,02 08 10 16 17 18 19 23 35 38 39 40 43 55 58 61 71 72 76 79
69      03/29/2000,02 03 04 07 14 16 18 21 22 36 40 46 53 54 55 59 62 63 66 74
70      03/28/2000,02 11 16 18 24 25 35 36 40 42 44 45 54 56 58 61 63 69 74 79
71      03/27/2000,02 04 13 16 20 24 32 38 42 50 52 57 58 60 68 70 74 75 76 78
72      03/26/2000,03 07 12 22 23 26 30 35 36 43 45 50 61 63 65 73 75 76 78 80
73      03/25/2000,05 07 08 12 16 29 31 36 37 50 52 55 59 61 63 64 65 67 70 80
74      03/24/2000,10 12 16 26 34 35 37 38 43 47 52 58 71 73 74 75 76 77 78 80
75      03/23/2000,08 09 13 14 17 23 24 25 27 31 40 42 44 52 53 57 58 64 79 80
76      03/22/2000,02 05 07 08 13 15 21 24 27 40 41 42 48 50 54 56 63 68 75 76
77      03/21/2000,01 11 12 14 17 20 23 35 41 47 49 54 60 66 67 68 69 73 74 79
78      03/20/2000,01 03 04 05 15 21 25 26 34 42 45 46 47 58 59 61 62 64 74 80
79      03/19/2000,13 17 18 21 23 24 30 36 37 42 43 53 57 58 60 69 75 76 77 78
80      03/18/2000,05 06 08 18 22 30 37 39 42 43 48 49 51 55 57 62 67 73 75 78
81      03/17/2000,01 05 07 10 11 16 17 18 22 24 29 32 35 38 43 46 53 55 76 78
82      03/16/2000,01 04 05 06 07 11 15 21 24 26 28 29 31 41 47 53 63 65 77 80
```

```
83      03/15/2000,03 04 12 16 18 21 26 27 29 31 32 38 42 43 48 58 64 66 73 78
84      03/14/2000,06 22 37 41 43 49 50 52 54 55 57 62 65 67 73 74 75 76 77 80
85      03/13/2000,04 07 09 13 16 19 20 28 29 36 38 41 45 50 56 66 68 71 72 80
86      03/12/2000,03 06 07 09 11 14 19 25 30 32 36 37 38 40 48 49 55 66 69 75
87      03/11/2000,11 13 18 29 30 32 34 35 36 38 40 41 43 45 50 53 57 69 71 72
88      03/10/2000,01 02 08 10 15 20 21 22 31 39 40 41 47 49 55 57 58 63 79 80
89      03/09/2000,02 07 12 15 19 20 25 26 28 33 37 40 41 45 50 62 64 69 70 76
90      03/08/2000,03 07 19 20 26 31 32 33 35 37 38 39 52 56 59 63 64 67 73 76
91      03/07/2000,06 12 13 16 17 20 22 30 31 34 38 43 51 53 58 65 67 69 73 76
92      03/06/2000,01 03 10 16 20 24 26 27 28 29 32 39 49 50 51 53 56 64 74 77
93      03/05/2000,06 19 24 27 29 32 35 38 41 42 46 48 50 52 53 58 67 69 77 79
94      03/04/2000,03 05 06 07 09 11 14 15 19 23 25 31 33 35 39 46 59 65 70 79
95      03/03/2000,03 12 13 22 28 29 32 33 37 44 48 52 55 58 60 61 67 68 78 80
96      03/02/2000,02 04 07 11 13 14 16 18 26 30 36 45 46 47 48 56 66 67 74 80
97      03/01/2000,03 11 13 14 20 26 27 30 37 42 43 46 48 49 53 54 57 59 70 79
98      02/29/2000,06 09 12 13 19 24 27 29 38 44 46 47 51 52 53 57 63 68 77 80
99      02/28/2000,09 11 20 22 28 30 37 41 42 46 49 50 52 55 57 60 61 66 71 79
00      02/27/2000,02 09 15 16 21 23 33 34 39 42 51 54 64 65 67 69 73 74 75 78
01      02/26/2000,06 09 13 14 15 17 23 24 26 30 39 45 49 50 51 53 57 58 75 79
02      02/25/2000,06 12 16 23 26 28 34 36 38 44 45 52 55 61 65 66 69 77 78 80
03      02/24/2000,02 03 16 18 28 31 34 38 41 43 48 50 51 55 56 57 58 64 68 71
04      02/23/2000,01 06 11 12 14 20 25 33 37 46 50 51 52 57 59 68 75 76 78 79
05      02/22/2000,03 06 11 15 18 24 26 29 32 39 40 46 51 58 60 63 72 74 78 79
06      02/21/2000,08 09 10 13 21 31 37 41 45 50 54 55 57 62 63 65 69 70 73 80
07      02/20/2000,03 05 07 09 10 13 15 21 25 32 34 35 41 42 46 54 56 68 79 80
08      02/19/2000,01 03 07 09 17 24 29 30 35 37 40 47 55 59 67 68 69 70 77 80
09      02/18/2000,08 10 12 13 14 17 24 30 33 36 45 46 58 59 62 63 68 69 74 77
10      02/17/2000,02 03 04 06 07 14 30 32 34 37 39 40 41 42 47 48 54 59 71 76
11      02/16/2000,02 03 07 08 16 18 31 32 34 37 38 39 41 44 45 47 57 59 62 74
12      02/15/2000,02 03 10 21 22 23 25 28 41 43 44 45 48 56 64 69 71 72 73 79
13      02/14/2000,02 03 05 10 15 22 26 29 31 35 36 40 50 57 62 66 68 69 74 78
14      02/13/2000,03 04 10 11 13 15 16 19 22 29 32 37 39 44 46 58 72 73 76 80
15      02/12/2000,08 11 19 20 22 29 41 42 43 50 51 53 61 65 69 71 72 77 79 80
16      02/11/2000,04 06 13 19 20 21 27 28 29 33 34 36 38 53 55 58 60 63 65 68
17      02/10/2000,07 08 20 25 26 37 41 43 47 52 53 58 59 63 65 68 69 70 77 79
18      02/09/2000,01 02 03 05 09 14 26 36 39 42 49 50 53 59 61 63 65 70 74 77
19      02/08/2000,04 07 09 10 11 12 19 22 26 29 30 31 34 40 48 56 58 66 74 80
20      02/07/2000,16 20 23 24 29 31 39 41 43 46 49 51 52 53 60 64 65 70 73 78
21      02/06/2000,02 07 13 15 18 20 22 23 24 31 44 45 48 55 58 61 62 64 69 80
22      02/05/2000,01 02 03 05 07 11 19 25 31 34 35 40 41 45 55 59 65 69 75 77
23      02/04/2000,03 04 07 12 16 17 18 20 22 23 24 38 45 49 52 54 55 58 74 80
24      02/03/2000,01 02 06 09 22 26 27 32 41 43 45 50 53 56 60 61 65 74 75 80
25      02/02/2000,06 07 12 25 29 32 34 35 41 50 52 54 57 61 63 67 71 73 75 78
26      02/01/2000,07 08 10 16 24 29 33 34 40 41 43 44 45 49 51 53 57 69 73 77
27      01/31/2000,05 11 12 13 15 23 33 34 35 41 53 55 63 65 69 70 77 78 79 80
28      01/30/2000,04 10 12 15 19 22 29 32 41 42 45 50 56 57 58 68 72 75 78 80
29      01/29/2000,02 05 08 09 13 16 17 23 26 29 32 34 38 45 49 50 61 65 66 78
30      01/28/2000,02 09 11 24 30 32 33 34 36 38 48 49 52 53 62 64 67 70 72 76
31      01/27/2000,06 08 10 12 14 16 31 32 33 36 40 46 50 56 61 63 70 76 77 78
32      01/26/2000,02 13 18 21 22 23 25 28 33 34 36 37 38 46 49 51 68 71 75 76
33      01/25/2000,02 05 14 21 28 29 31 32 34 38 42 48 54 55 61 63 66 67 73 80
34      01/24/2000,01 12 13 16 25 30 32 33 39 47 48 51 55 56 58 60 69 70 71 72
35      01/23/2000,01 03 06 08 26 29 32 37 41 47 50 52 54 56 57 59 64 69 77 80
36      01/22/2000,02 03 07 09 13 18 23 31 32 34 35 38 42 48 50 55 58 74 78 79
```

```
01/21/2000,01 09 10 11 13 15 21 22 28 31 35 42 45 52 54 60 61 63 67 75
01/20/2000,02 04 05 07 11 15 18 19 23 26 34 39 43 48 53 64 65 73 78 80
01/19/2000,02 08 11 14 15 17 19 22 26 32 33 41 47 63 64 65 68 70 73 76
01/18/2000,03 12 13 25 27 28 35 36 50 54 56 57 59 64 66 68 69 73 76 77
01/17/2000,04 05 06 17 18 25 26 31 32 37 51 52 58 59 60 63 71 75 77 80
01/16/2000,03 05 08 11 16 17 19 20 33 42 45 61 62 63 70 71 72 75 79 80
01/15/2000,07 08 15 21 25 26 34 38 40 41 44 48 51 54 55 67 69 72 76 78
01/14/2000,15 16 22 34 36 37 38 41 43 47 49 50 51 53 57 61 68 69 70 76
01/13/2000,03 05 06 08 09 12 13 15 16 18 19 22 23 31 36 38 52 54 58 63
01/12/2000,05 06 10 11 18 25 26 34 38 40 41 43 52 58 65 67 70 73 76 80
01/11/2000,04 05 10 14 15 21 22 24 27 30 36 43 54 57 58 62 66 69 71 74
01/10/2000,05 09 10 11 14 19 20 24 33 37 39 41 51 58 66 71 73 74 76 79
01/09/2000,01 04 06 12 13 14 27 32 36 38 39 42 43 47 48 49 58 62 64 76
01/08/2000,01 03 05 06 07 08 09 11 22 24 31 34 39 41 50 52 57 58 60 67
01/07/2000,08 12 14 19 28 31 33 34 37 40 41 44 45 57 58 63 67 68 70 80
01/06/2000,06 15 22 23 26 29 31 32 40 41 49 50 55 57 58 61 62 64 65 79
01/05/2000,08 13 16 17 19 20 30 31 33 35 45 46 54 56 66 70 72 74 76 78
01/04/2000,02 09 16 19 21 26 35 37 40 48 49 52 58 59 60 63 64 69 73 77
01/03/2000,03 04 05 07 16 17 18 22 29 31 35 37 39 42 60 62 68 72 75 76
01/02/2000,01 04 05 11 12 14 22 26 38 41 47 50 54 55 63 67 69 75 76 78
01/01/2000,05 12 15 17 18 24 27 34 36 37 38 39 45 51 56 60 62 67 71 80
12/31/1999,06 08 11 13 16 17 19 24 32 48 52 53 58 59 64 70 71 73 77 78
12/30/1999,12 13 15 17 21 27 29 34 36 37 38 42 54 55 57 61 70 72 75 79
12/29/1999,01 11 15 22 24 30 32 35 37 38 40 41 42 46 49 60 66 70 72 77
12/28/1999,04 07 10 12 16 18 23 32 36 40 46 49 59 60 63 65 69 75 78 80
12/27/1999,02 07 18 21 23 27 31 34 37 39 40 46 50 54 55 59 64 69 72 73
12/26/1999,06 10 14 18 19 25 30 46 50 52 53 54 59 60 62 63 64 65 69 70
12/24/1999,04 10 17 18 22 25 30 44 50 51 56 57 58 59 69 71 72 73 75 77
12/23/1999,11 15 24 26 27 31 34 37 38 40 41 44 47 54 57 68 69 70 72 77
12/22/1999,03 06 09 10 20 25 26 48 50 51 53 55 56 57 59 66 70 72 73 80
12/21/1999,04 15 17 23 30 32 33 42 44 48 51 53 54 55 56 64 65 69 77 80
12/20/1999,03 04 11 13 15 16 20 21 30 34 40 45 49 50 61 62 63 72 76 79
12/19/1999,16 19 21 22 24 25 26 29 31 34 40 54 56 59 61 63 70 72 73 78
12/18/1999,06 10 12 16 20 21 25 31 32 38 39 41 42 43 45 51 53 59 64 70
12/17/1999,04 07 09 15 19 22 23 25 32 45 47 52 56 63 64 65 67 69 71 76
12/16/1999,01 10 11 19 22 32 36 37 38 42 44 45 46 53 54 59 61 65 69 80
12/15/1999,02 07 09 23 25 29 38 39 43 48 49 52 54 58 64 68 71 76 78 80
12/14/1999,01 09 10 18 19 21 22 23 28 34 35 42 50 52 55 57 63 64 67 79
12/13/1999,05 25 30 35 37 38 39 43 44 46 50 53 61 68 69 71 72 73 75 79
12/12/1999,01 07 09 11 28 30 34 38 46 47 49 51 60 61 65 67 69 70 72 73
12/11/1999,01 13 15 17 20 21 28 46 50 52 53 55 56 58 65 67 71 73 78 79
12/10/1999,02 10 11 25 26 31 32 35 41 49 51 54 56 64 65 68 69 72 74 78
12/09/1999,01 04 06 19 23 26 27 28 30 32 38 39 42 44 45 48 57 63 70 74
12/08/1999,06 12 15 18 22 31 34 40 42 43 44 53 54 59 60 63 64 72 73 75
12/07/1999,01 03 10 14 15 19 20 21 25 27 29 32 51 57 65 67 73 74 76 79
12/06/1999,07 11 12 22 23 28 29 38 42 43 46 48 49 50 58 61 67 68 79 80
12/05/1999,06 07 11 12 13 14 16 23 28 42 45 48 52 58 59 66 71 77 78 79
12/04/1999,01 03 15 16 20 22 25 37 39 47 50 52 53 57 62 66 68 75 78 79
12/03/1999,02 05 06 07 09 11 20 31 32 34 36 43 45 52 54 58 60 63 65 77
12/02/1999,02 03 04 05 06 16 18 27 28 30 33 34 51 52 58 61 63 64 70 80
12/01/1999,02 06 08 10 16 17 18 20 21 27 36 38 40 41 47 54 59 61 69 80
11/30/1999,03 05 07 21 24 29 32 41 44 47 52 56 63 66 68 69 71 75 77 80
11/29/1999,06 08 10 11 13 21 24 25 26 27 32 33 40 46 48 58 63 65 73 80
11/28/1999,01 07 10 11 14 25 27 30 38 39 47 53 54 55 62 68 71 72 74 80
```

```
91      11/27/1999,03 05 12 15 17 24 27 32 33 34 37 40 41 45 50 55 63 68 70 71
92      11/26/1999,07 12 19 27 32 34 36 37 40 42 43 46 50 54 56 57 67 69 72 77
93      11/25/1999,03 05 13 14 15 17 18 21 23 25 27 33 36 43 46 57 58 63 74 80
94      11/24/1999,03 14 15 16 18 23 24 28 29 40 42 44 46 47 62 65 70 74 76 79
95      11/23/1999,08 09 19 21 31 32 36 37 38 46 47 50 52 55 64 66 67 69 71 79
96      11/22/1999,03 04 08 09 12 14 15 23 42 47 49 51 53 55 57 60 62 63 73 75
97      11/21/1999,05 10 12 19 21 25 29 36 40 41 44 45 46 52 56 58 60 69 72 80
98      11/20/1999,01 04 07 08 09 15 17 19 28 35 36 37 44 47 49 52 57 60 69 79
99      11/19/1999,05 11 16 19 20 21 25 29 33 42 43 44 49 51 52 57 71 74 75 77
00      11/18/1999,02 04 09 16 22 26 27 30 31 33 34 36 44 51 52 53 54 59 69 72
01      11/17/1999,07 09 10 12 14 19 24 30 31 32 42 53 54 55 56 59 63 68 70 75
02      11/16/1999,03 04 05 06 08 12 30 31 34 35 37 43 44 48 52 54 55 57 63 72
03      11/15/1999,02 10 11 14 15 28 30 37 43 44 45 50 59 64 68 69 71 72 75 76
04      11/14/1999,01 04 07 15 16 21 39 40 45 49 51 55 57 58 62 65 71 72 75 76
05      11/13/1999,03 07 14 19 25 26 27 29 35 40 42 43 44 47 53 61 71 72 73 80
06      11/12/1999,06 07 08 09 10 11 18 23 25 26 31 35 38 43 57 60 68 70 76 80
07      11/11/1999,02 08 09 14 23 25 26 28 31 33 39 58 59 66 68 69 73 74 75 78
08      11/10/1999,04 07 14 15 20 23 28 30 32 34 43 48 49 54 55 60 62 63 64 76
09      11/09/1999,03 04 05 08 16 17 18 21 24 32 37 41 42 51 54 60 61 73 77 80
10      11/08/1999,02 05 09 13 16 20 24 30 35 43 44 50 51 58 59 65 69 72 73 77
11      11/07/1999,01 02 03 05 15 16 17 20 22 28 30 31 39 44 46 64 65 73 79 80
12      11/06/1999,04 10 16 19 20 25 26 30 31 33 35 42 49 54 56 75 76 78 79 80
13      11/05/1999,01 03 04 08 10 12 20 28 29 46 50 55 58 62 66 71 74 75 78 80
14      11/04/1999,04 05 06 08 11 14 21 24 32 41 46 48 49 54 61 66 69 74 77 79
15      11/03/1999,01 02 06 09 13 23 24 28 32 33 35 43 48 53 64 71 73 75 77 79
16      11/02/1999,01 03 06 11 14 34 38 43 44 45 46 49 50 52 54 57 58 60 66 69
17      11/01/1999,02 14 16 17 22 27 28 33 36 45 47 48 52 54 55 61 65 68 71 79
18      10/31/1999,01 02 17 18 21 24 26 31 34 37 38 43 47 54 55 56 66 69 71 72
19      10/30/1999,02 04 11 15 16 17 20 30 35 47 48 51 55 57 63 65 73 74 79 80
20      10/29/1999,05 08 12 18 21 22 26 32 41 42 48 53 54 57 58 61 63 64 68 73
21      10/28/1999,07 09 11 13 14 19 22 25 28 30 37 38 44 47 51 58 59 68 69 79
22      10/27/1999,05 09 12 15 18 19 20 22 24 27 38 40 41 45 47 58 68 76 79 80
23      10/26/1999,03 06 12 25 26 30 32 35 36 37 45 47 51 57 65 67 69 72 74 79
24      10/25/1999,04 07 14 16 18 22 24 25 28 33 39 45 48 55 57 61 65 76 78 80
25      10/24/1999,13 14 19 22 24 26 28 29 31 33 37 51 54 55 64 67 70 76 78 80
26      10/23/1999,01 08 22 29 30 34 35 41 46 48 50 52 57 59 63 65 66 67 73 75
27      10/22/1999,01 02 14 15 17 20 21 23 24 27 28 29 30 36 45 49 52 60 65 76
28      10/21/1999,02 04 10 11 12 13 14 21 25 29 32 33 34 38 61 62 65 67 74 77
29      10/20/1999,04 05 07 11 13 18 21 26 28 29 37 38 40 55 56 60 61 67 78 79
30      10/19/1999,11 13 16 18 20 30 37 39 40 42 44 48 56 57 64 65 66 70 74 79
31      10/18/1999,01 02 06 10 14 17 19 21 37 44 45 53 57 61 63 64 66 72 73 74
32      10/17/1999,11 21 26 32 34 35 39 41 46 47 48 50 54 58 65 67 72 73 78 80
33      10/16/1999,01 02 08 16 20 21 27 35 39 40 42 48 51 53 61 64 70 71 74 77
34      10/15/1999,03 07 11 12 27 33 39 40 41 49 51 52 54 55 59 63 64 69 72 73
35      10/14/1999,01 02 16 17 21 30 35 36 37 38 48 53 57 58 64 65 69 74 78 79
36      10/13/1999,02 06 07 10 16 19 22 24 29 30 39 43 54 57 63 68 75 76 77 78
37      10/12/1999,04 10 23 26 29 39 48 49 50 52 54 55 57 63 64 70 72 76 78 80
38      10/11/1999,01 05 07 08 09 14 18 28 29 33 41 43 50 53 54 56 58 63 70 75
39      10/10/1999,01 03 04 08 10 14 16 23 28 29 31 42 45 49 57 58 68 69 76 77
40      10/09/1999,06 14 15 16 22 30 34 39 42 44 45 48 50 52 55 60 66 67 72 75
41      10/08/1999,02 03 04 09 10 13 16 20 21 25 34 37 40 45 46 48 49 65 66 76
42      10/07/1999,01 02 07 09 11 15 17 22 24 27 30 39 43 44 46 49 60 76 78 79
43      10/06/1999,02 04 08 19 25 26 29 31 32 34 35 42 60 66 69 71 72 76 77 80
44      10/05/1999,03 04 06 14 18 20 22 29 31 32 33 35 39 51 52 53 55 61 73 78
```

```
10/04/1999,02 07 11 12 17 29 31 39 41 42 46 50 52 55 68 70 71 75 76 80
10/03/1999,02 05 11 14 15 22 27 30 36 44 46 47 49 53 58 62 63 72 73 74
10/02/1999,04 07 10 12 21 26 27 31 32 35 41 47 50 54 55 58 60 67 79 80
10/01/1999,01 06 08 13 19 21 25 26 27 33 35 36 40 42 44 49 59 67 71 75
09/30/1999,01 05 07 10 14 19 28 38 40 46 48 49 50 51 52 53 56 73 75 78
09/29/1999,02 06 12 17 18 25 26 34 38 40 41 48 60 61 62 64 65 66 67 76
09/28/1999,01 02 03 05 08 09 13 14 17 30 36 41 47 57 64 68 70 71 72 80
09/27/1999,08 10 12 13 15 22 29 31 35 51 53 54 56 57 58 66 71 73 75 76
09/26/1999,02 08 11 14 15 19 22 30 32 36 40 44 46 47 57 60 65 67 72 74
09/25/1999,01 05 11 16 18 19 21 24 29 35 36 43 47 49 54 55 57 66 69 74
09/24/1999,03 06 16 22 24 25 26 28 32 33 36 39 45 54 63 66 67 68 78 80
09/23/1999,03 12 19 20 22 23 24 25 29 30 31 32 39 42 43 49 55 66 70 71
09/22/1999,01 09 10 11 12 13 21 25 27 33 36 44 46 49 50 53 61 68 70 75
09/21/1999,11 12 15 17 18 24 38 45 49 53 54 58 61 63 64 65 67 69 76 79
09/20/1999,09 10 12 22 23 27 29 30 32 36 44 45 51 54 59 62 69 70 73 80
09/19/1999,02 03 13 18 20 30 34 36 45 46 52 55 56 61 62 68 71 74 77 78
09/18/1999,06 07 08 11 16 20 23 31 32 42 53 55 58 63 64 67 69 74 78 79
09/17/1999,05 07 08 15 20 21 28 34 39 41 51 55 56 57 60 63 65 70 75 78
09/16/1999,02 05 15 16 17 19 22 25 27 31 37 40 41 49 51 57 59 63 70 76
09/15/1999,01 04 11 12 13 15 22 23 29 30 37 40 41 61 64 65 66 69 76 77
09/14/1999,03 05 07 12 17 18 22 25 29 41 42 46 48 52 56 59 64 66 71 80
09/13/1999,12 13 15 21 22 23 25 26 33 34 36 38 40 46 50 56 58 59 60 80
09/12/1999,01 02 05 07 10 11 18 20 33 36 37 41 48 50 55 60 61 68 73 78
09/11/1999,03 04 16 26 27 30 31 36 41 44 45 49 54 55 57 63 65 70 71 77
09/10/1999,04 05 06 08 09 14 25 36 40 41 42 43 45 46 54 56 57 70 74 75
09/09/1999,12 15 16 20 24 25 26 32 38 45 46 48 53 57 58 65 67 69 74 75
09/08/1999,02 05 06 16 18 24 26 27 30 39 51 54 56 57 62 63 68 69 71 80
09/07/1999,01 05 13 15 17 23 25 29 33 34 37 47 55 60 67 71 76 77 78 79
09/06/1999,08 12 13 17 19 26 30 35 37 38 39 43 49 53 58 61 71 72 77 78
09/05/1999,01 10 13 23 31 33 34 41 42 47 49 57 61 64 66 68 71 73 77 80
09/04/1999,05 09 14 17 22 30 31 34 37 49 51 54 58 62 65 72 74 77 79 80
09/03/1999,02 05 07 11 12 15 18 26 34 36 39 40 42 43 49 55 57 59 76 79
09/02/1999,03 07 11 15 16 22 26 34 37 38 47 53 54 55 63 64 66 74 75 76
09/01/1999,10 11 18 25 31 32 34 40 43 45 47 55 59 61 63 65 76 77 79 80
08/31/1999,05 14 15 16 17 20 21 24 26 28 31 34 43 51 58 61 62 64 65 67
08/30/1999,05 06 08 12 18 27 33 35 36 37 43 45 47 51 52 54 67 74 76 77
08/29/1999,11 14 16 19 20 21 36 39 40 44 46 48 52 53 55 56 60 62 63 75
08/28/1999,10 15 19 21 27 31 40 42 46 48 51 58 59 61 67 68 70 73 79 80
08/27/1999,05 09 14 20 25 28 36 39 43 50 51 54 57 58 66 71 75 78 79 80
08/26/1999,06 10 19 23 29 30 34 40 45 46 47 49 58 63 65 66 68 70 73 75
08/25/1999,02 03 11 18 19 21 23 25 26 27 29 39 40 47 54 60 67 71 74 78
08/24/1999,01 02 07 16 22 27 28 29 30 38 39 40 46 47 48 49 51 56 59 65
08/23/1999,03 08 09 11 13 17 18 19 20 28 41 49 54 66 68 71 72 75 79 80
08/22/1999,06 09 18 21 29 34 37 42 47 53 54 59 65 67 68 73 74 75 76 79
08/21/1999,01 02 06 08 10 11 16 32 33 35 41 44 58 63 64 70 71 72 73 75
08/20/1999,02 10 12 13 22 23 25 30 31 33 37 44 49 58 60 63 65 66 67 72
08/19/1999,07 13 16 18 19 21 22 24 25 26 33 34 35 36 37 38 46 52 65 78
08/18/1999,02 03 05 08 09 13 19 24 34 37 38 50 54 56 57 60 66 75 77 80
08/17/1999,08 12 18 22 23 24 27 29 30 31 34 39 44 48 58 72 74 76 77 78
08/16/1999,01 10 14 19 24 28 31 32 35 37 40 42 47 52 55 62 64 73 76 77
08/15/1999,01 02 10 20 22 23 29 36 44 45 51 52 57 58 65 66 69 74 75 80
08/14/1999,03 08 10 15 19 21 22 23 24 37 38 41 48 49 51 52 54 58 71 72
08/13/1999,14 16 17 21 25 30 31 42 43 44 48 50 56 57 59 63 67 74 77 79
08/12/1999,14 16 17 22 23 27 43 44 45 51 52 54 58 60 63 66 70 71 72 74
```

```
99      08/11/1999,03 04 08 15 16 18 20 25 27 28 32 37 39 53 56 59 62 74 75 79
00      08/10/1999,02 13 15 21 29 40 41 45 51 52 54 58 60 61 63 65 71 74 76 77
01      08/09/1999,04 05 08 12 23 24 30 31 36 42 43 46 56 57 58 60 62 70 71 80
02      08/08/1999,03 09 14 19 24 25 26 30 34 40 41 42 52 54 55 58 61 68 70 72
03      08/07/1999,04 05 06 10 12 14 22 23 26 28 31 38 41 49 54 58 64 74 78 80
04      08/06/1999,04 16 18 22 32 34 37 42 43 51 52 53 59 61 66 67 70 73 74 76
05      08/05/1999,01 03 07 12 16 19 24 33 37 40 44 48 58 59 60 63 67 76 79 80
06      08/04/1999,03 06 07 11 12 13 14 18 19 22 26 35 51 52 58 66 69 76 77 78
07      08/03/1999,06 08 13 16 19 20 23 24 25 26 34 36 39 43 48 61 62 64 70 78
08      08/02/1999,05 06 10 16 17 25 33 37 43 44 46 48 50 58 59 61 66 70 74 75
09      08/01/1999,08 09 15 19 21 23 24 26 27 30 37 38 47 60 62 66 69 70 72 74
10      07/31/1999,04 08 12 13 14 18 20 28 32 36 41 43 53 57 60 62 63 64 68 77
11      07/30/1999,01 02 04 06 14 18 24 28 36 41 48 51 53 54 62 63 64 73 78 80
12      07/29/1999,02 04 07 09 12 14 19 22 28 39 41 47 53 54 57 62 64 66 77 78
13      07/28/1999,02 05 06 09 17 18 19 21 26 31 35 40 42 47 52 56 64 66 71 79
14      07/27/1999,09 13 14 15 24 25 28 32 37 40 41 43 47 48 58 61 62 66 72 76
15      07/26/1999,02 07 08 13 15 16 25 30 31 39 42 43 45 48 50 58 60 62 66 70
16      07/25/1999,02 09 12 20 21 24 36 39 41 43 45 47 53 60 61 65 66 71 72 76
17      07/24/1999,01 07 10 17 20 21 32 36 44 45 52 55 62 65 67 68 69 72 73 78
18      07/23/1999,04 05 09 15 17 19 33 34 37 49 54 56 58 62 64 70 72 77 79 80
19      07/22/1999,03 08 26 28 29 33 36 43 47 50 52 59 60 63 64 65 67 71 73 75
20      07/21/1999,06 13 15 19 20 45 49 57 59 62 65 66 67 69 70 75 77 78 79 80
21      07/20/1999,02 12 17 18 21 23 24 29 30 36 44 53 54 59 62 70 72 73 76 80
22      07/19/1999,01 03 04 07 13 15 18 19 23 34 41 42 49 52 54 62 74 76 77 79
23      07/18/1999,01 06 08 15 16 19 31 34 42 44 47 52 54 59 69 72 73 74 77 80
24      07/17/1999,13 20 22 23 24 25 27 32 37 39 40 52 53 58 59 61 62 65 69 78
25      07/16/1999,01 03 14 23 31 39 44 48 54 57 60 63 64 65 68 69 70 71 72 77
26      07/15/1999,01 02 03 11 19 22 27 29 31 32 34 43 55 56 60 65 66 74 77 80
27      07/14/1999,05 10 11 12 13 15 22 37 38 44 46 48 49 51 59 63 65 68 72 77
28      07/13/1999,04 08 09 11 14 18 21 25 27 42 47 48 59 61 62 63 68 73 75 77
29      07/12/1999,13 14 18 23 24 28 34 35 41 48 49 56 57 61 63 64 66 74 75 77
30      07/11/1999,04 10 14 18 19 20 28 41 42 44 48 51 58 63 70 72 75 77 78 79
31      07/10/1999,02 17 18 20 22 24 27 29 33 34 35 41 42 43 45 47 55 56 76 79
32      07/09/1999,01 05 08 09 12 18 22 27 34 35 54 56 59 64 68 73 75 76 79 80
33      07/08/1999,01 04 09 10 13 18 27 29 31 34 43 44 54 60 61 69 70 71 73 76
34      07/07/1999,05 07 08 10 11 13 18 19 28 35 39 44 46 60 61 62 63 64 68 80
35      07/06/1999,05 08 12 15 17 18 19 29 31 35 36 45 50 51 54 58 59 72 77 78
36      07/05/1999,03 04 05 08 09 11 17 18 19 21 24 30 32 36 53 57 68 70 72 76
37      07/04/1999,04 09 13 15 16 21 25 28 37 41 44 47 50 53 54 62 67 73 74 79
38      07/03/1999,04 15 16 19 21 23 32 36 41 43 45 48 55 57 63 65 66 70 72 74
39      07/02/1999,02 11 13 16 21 25 29 35 37 39 41 42 44 49 50 51 63 65 68 74
40      07/01/1999,13 14 15 17 20 27 32 42 44 48 50 53 60 61 62 66 68 71 74 80
41      06/30/1999,06 16 18 19 21 29 30 32 33 35 48 52 55 58 61 66 70 71 73 79
42      06/29/1999,01 06 07 10 18 28 31 32 35 38 42 46 47 48 50 51 52 70 76 80
43      06/28/1999,04 06 11 16 18 20 21 29 40 46 47 49 53 56 60 61 64 71 73 75
44      06/27/1999,02 10 12 16 23 26 32 33 37 43 46 53 55 57 58 62 68 69 75 78
45      06/26/1999,04 14 15 17 18 22 24 30 31 40 43 45 48 57 64 66 68 71 74 79
46      06/25/1999,01 03 06 09 12 14 21 23 28 36 39 42 45 52 53 56 64 65 70 79
47      06/24/1999,01 08 25 27 31 35 36 42 44 45 46 49 52 55 58 64 66 70 71 74
48      06/23/1999,02 03 05 15 16 26 30 41 43 48 49 52 56 58 60 61 65 68 72 75
49      06/22/1999,03 04 09 13 15 31 34 40 45 54 57 58 61 67 72 73 74 75 77 78
50      06/21/1999,01 05 12 17 19 26 32 35 36 41 46 47 48 49 57 64 68 71 72 79
51      06/20/1999,04 05 07 10 11 17 18 19 33 35 38 40 44 45 51 68 71 73 75 78
52      06/19/1999,04 07 10 13 19 21 27 34 36 37 38 40 41 43 44 48 51 55 56 60
```

```
53      06/18/1999,09 11 14 16 19 21 23 25 27 29 30 31 33 36 41 45 51 54 60 80
54      06/17/1999,02 10 11 14 21 24 26 38 39 42 44 45 53 56 64 70 74 75 76 77
55      06/16/1999,15 16 21 26 28 30 34 41 44 47 52 53 57 59 66 74 75 77 78 80
56      06/15/1999,03 06 09 10 18 20 21 27 34 39 41 44 45 48 49 59 61 65 68 71
57      06/14/1999,04 09 10 14 16 21 22 26 28 30 38 42 43 48 53 60 67 72 73 80
58      06/13/1999,01 04 10 11 12 14 18 30 31 32 33 41 48 51 52 56 66 68 69 78
59      06/12/1999,01 09 10 14 17 21 31 38 44 47 53 57 58 61 68 69 70 75 78 80
60      06/11/1999,02 03 04 09 10 12 24 25 26 29 30 36 41 43 44 49 53 63 66 74
61      06/10/1999,05 12 19 24 29 36 39 43 48 49 51 54 55 57 63 64 70 73 77 80
62      06/09/1999,02 10 13 16 28 29 31 41 43 44 49 51 53 54 61 64 69 70 71 74
63      06/08/1999,17 18 25 27 31 33 36 37 42 48 51 55 56 57 58 63 65 69 72 74
64      06/07/1999,02 06 12 25 26 32 33 37 40 42 43 44 46 48 50 55 63 65 69 77
65      06/06/1999,02 09 10 13 14 15 16 18 26 34 36 45 52 53 54 60 66 74 76 78
66      06/05/1999,01 03 07 19 20 22 27 36 39 41 43 47 48 51 57 59 63 65 68 69
67      06/04/1999,02 05 12 13 16 17 18 19 26 29 30 38 43 45 54 61 71 72 75 78
68      06/03/1999,07 12 20 21 25 28 29 35 37 50 54 57 60 62 63 64 70 71 78 80
69      06/02/1999,02 03 10 17 25 26 32 41 44 48 51 58 61 64 65 66 69 71 73 76
70      06/01/1999,01 08 11 14 15 19 22 24 29 34 39 48 52 53 54 59 64 74 78 79
71      05/31/1999,06 07 08 12 18 19 23 30 44 47 48 49 55 56 65 67 70 71 73 75
72      05/30/1999,01 03 06 07 11 12 19 28 30 38 42 44 45 47 54 56 63 64 67 69
73      05/29/1999,02 05 07 10 17 30 33 34 42 43 44 48 58 61 64 70 74 78 79 80
74      05/28/1999,06 07 13 16 17 18 19 22 23 27 29 36 40 42 47 50 55 63 66 78
75      05/27/1999,07 08 13 17 20 30 37 39 40 43 48 51 60 61 66 69 70 71 76 79
76      05/26/1999,02 04 05 09 11 13 15 16 20 24 27 38 44 50 52 53 64 72 75 77
77      05/25/1999,03 07 13 19 21 25 26 27 29 37 40 43 47 53 58 62 65 67 73 75
78      05/24/1999,03 04 08 14 22 31 33 35 39 43 45 50 55 57 62 66 68 70 75 76
79      05/23/1999,10 24 25 26 28 38 40 51 55 56 58 60 64 65 66 67 73 77 78 80
80      05/22/1999,05 14 19 23 25 27 29 33 41 45 49 55 57 58 62 63 70 71 75 77
81      05/21/1999,02 07 09 13 15 16 17 21 24 36 40 47 50 54 56 59 65 73 77 79
82      05/20/1999,03 04 07 12 13 18 28 30 33 36 42 43 46 49 53 62 65 66 70 71
83      05/19/1999,03 06 09 10 12 14 15 23 31 35 43 48 55 56 57 62 64 73 75 76
84      05/18/1999,04 05 11 12 23 24 26 27 28 51 54 58 60 63 64 65 68 75 77 78
85      05/17/1999,01 02 03 09 12 13 20 21 22 23 28 32 35 37 56 68 71 77 78 79
86      05/16/1999,01 03 09 12 14 16 24 28 30 31 32 33 39 41 44 46 52 60 71 72
87      05/15/1999,01 02 03 06 15 17 19 21 28 31 32 43 44 45 48 50 67 68 69 70
88      05/14/1999,02 07 08 16 23 24 32 37 38 41 47 54 56 57 58 60 64 70 72 75
89      05/13/1999,01 07 10 18 21 23 27 28 38 45 46 47 49 52 54 64 67 73 78 80
90      05/12/1999,01 02 05 06 09 10 18 21 28 34 35 41 46 53 59 64 66 74 75 78
91      05/11/1999,03 10 12 20 21 23 27 28 29 30 31 32 41 51 53 54 69 70 78 80
92      05/10/1999,08 10 17 26 33 36 40 41 44 45 47 50 51 53 54 55 58 69 70 76
93      05/09/1999,01 04 06 11 12 18 20 28 34 40 42 43 44 46 47 50 60 64 79 80
94      05/08/1999,01 08 10 13 22 25 26 28 32 34 39 40 45 47 50 54 56 57 63 64
95      05/07/1999,02 03 08 09 10 11 17 22 30 31 35 38 47 51 60 65 67 69 77 80
96      05/06/1999,01 02 05 08 17 20 22 24 25 28 29 30 32 43 47 56 61 63 68 71
97      05/05/1999,01 05 10 15 19 21 22 24 28 30 35 41 47 48 49 58 64 73 77 78
98      05/04/1999,01 06 14 20 22 24 25 26 30 31 32 40 43 48 51 56 63 66 71 77
99      05/03/1999,02 05 10 22 26 30 35 37 40 44 45 47 50 51 53 66 68 72 75 76
00      05/02/1999,07 11 21 23 24 28 38 43 45 48 51 56 65 66 67 68 71 75 77 79
01      05/01/1999,13 16 17 23 27 29 30 37 38 47 50 52 62 63 64 66 68 74 77 79
02      04/30/1999,06 09 11 12 18 25 26 29 35 37 46 51 53 56 57 62 64 72 74 79
03      04/29/1999,05 08 11 18 23 28 34 35 45 46 48 52 58 64 65 66 67 68 72 80
04      04/28/1999,02 07 11 15 23 24 25 29 33 36 37 40 41 43 49 58 61 67 79 80
05      04/27/1999,06 07 09 11 12 15 20 21 23 28 30 37 42 47 48 50 58 62 68 79
06      04/26/1999,10 13 17 21 23 36 37 41 44 49 52 55 62 63 69 71 72 73 75 76
```

07 04/25/1999,02 03 04 06 08 14 16 18 25 28 33 39 44 45 47 49 68 71 75 79
08 04/24/1999,02 05 07 12 13 14 20 24 27 29 33 36 41 50 54 55 56 59 65 77
09 04/23/1999,07 08 11 16 23 29 34 44 49 51 52 56 57 60 68 69 70 71 77 79
10 04/22/1999,08 10 14 20 21 22 23 31 40 44 48 49 55 56 59 66 71 75 79 80
11 04/21/1999,05 13 14 15 18 24 31 33 34 35 36 37 38 40 45 51 56 64 65 66
12 04/20/1999,03 07 13 14 15 18 20 26 28 29 31 39 43 44 47 54 55 57 71 80
13 04/19/1999,01 09 10 12 13 14 22 23 33 42 46 51 52 61 67 71 72 75 76 77
14 04/18/1999,05 07 14 17 18 20 22 33 36 37 42 43 46 47 59 60 65 68 71 79
15 04/17/1999,01 03 06 11 14 16 17 25 28 29 33 36 37 38 54 59 69 73 76 77
16 04/16/1999,03 04 05 06 11 25 28 34 40 41 44 50 53 55 68 71 73 74 76 79
17 04/15/1999,02 07 12 14 20 21 24 40 41 46 48 49 51 52 56 61 68 70 71 73
18 04/14/1999,01 03 04 09 11 17 23 25 33 35 48 55 60 62 63 68 70 75 78 79
19 04/13/1999,03 05 09 12 17 24 25 28 31 35 36 37 40 46 50 54 59 64 74 78
20 04/12/1999,03 06 08 12 14 15 16 21 33 40 43 52 53 54 56 58 61 67 69 79
21 04/11/1999,06 09 12 13 15 16 17 19 24 25 29 32 35 39 52 55 62 76 78 80
22 04/10/1999,01 02 13 16 24 25 27 32 33 37 45 48 50 51 55 65 67 71 72 75
23 04/09/1999,09 12 18 22 25 27 28 31 38 39 42 46 49 51 59 61 63 64 76 79
24 04/08/1999,04 07 08 16 17 18 20 23 33 34 43 46 48 49 55 56 59 68 69 75
25 04/07/1999,06 07 18 19 22 23 31 32 33 42 43 44 47 52 54 55 57 58 65 70
26 04/06/1999,05 11 15 18 21 22 23 26 30 31 40 42 51 54 58 61 64 65 73 74
27 04/05/1999,03 05 06 08 11 15 16 26 30 31 34 35 42 52 60 63 64 66 70 79
28 04/04/1999,17 21 24 27 29 32 34 38 40 41 46 49 50 52 59 64 70 71 75 77
29 04/03/1999,01 04 05 08 10 18 19 21 22 33 34 35 38 41 51 56 69 71 72 79
30 04/02/1999,02 10 17 21 22 28 29 32 40 49 52 53 56 57 60 62 63 67 75 77
31 04/01/1999,01 09 11 14 22 31 35 36 38 39 43 46 52 53 57 58 59 61 71 79
32 03/31/1999,09 12 14 15 20 27 29 32 33 44 45 51 54 55 59 60 65 66 67 72
33 03/30/1999,12 13 16 39 42 43 46 49 53 55 58 59 60 61 62 70 72 76 77 78
34 03/29/1999,03 08 09 14 29 31 37 39 42 45 48 56 57 58 67 69 71 75 79 80
35 03/28/1999,02 08 19 20 21 28 34 39 40 44 46 54 59 65 66 67 71 72 73 75
36 03/27/1999,05 06 09 19 23 35 37 38 43 45 48 50 52 55 60 62 63 66 70 75
37 03/26/1999,01 03 07 10 21 28 31 37 39 41 42 43 49 51 58 61 65 67 76 78
38 03/25/1999,01 06 10 16 18 20 25 26 33 39 50 54 56 59 63 68 69 70 75 78
39 03/24/1999,08 11 14 17 18 20 23 36 38 39 40 44 45 48 50 56 58 68 70 72
40 03/23/1999,09 12 13 18 21 24 25 32 34 40 47 49 51 52 53 58 62 63 68 74
41 03/22/1999,01 07 14 18 24 25 26 29 45 48 50 55 61 63 68 69 73 76 78 80
42 03/21/1999,02 03 09 10 17 18 21 29 34 35 36 37 40 43 44 45 47 59 62 70
43 03/20/1999,03 05 09 10 19 21 29 30 36 39 42 53 59 60 61 64 65 78 79 80
44 03/19/1999,01 02 05 11 14 19 29 35 43 44 54 55 59 60 61 62 68 70 79 80
45 03/18/1999,08 14 15 17 20 28 30 35 37 45 49 50 53 54 56 58 65 67 68 80
46 03/17/1999,03 06 14 29 33 34 36 40 41 44 49 50 53 56 62 63 64 66 70 80
47 03/16/1999,06 13 17 19 20 24 26 28 29 30 33 51 56 57 59 61 62 66 73 80
48 03/15/1999,02 03 08 10 11 16 17 19 25 27 28 29 32 35 39 48 53 55 79 80
49 03/14/1999,02 04 12 15 20 21 25 26 31 33 35 37 41 63 65 66 70 73 75 77
50 03/13/1999,02 03 06 14 17 19 28 31 34 35 45 48 49 55 57 65 68 72 74 76
51 03/12/1999,01 05 10 11 27 28 31 33 39 41 49 51 56 59 63 64 65 70 73 76
52 03/11/1999,08 09 11 14 16 18 22 23 26 28 32 36 49 55 58 62 67 69 73 78
53 03/10/1999,01 04 10 14 21 25 28 33 41 42 45 46 47 49 60 67 74 76 78 80
54 03/09/1999,02 04 07 16 21 23 29 33 39 40 41 45 49 66 68 69 72 75 76 77
55 03/08/1999,07 13 14 15 25 28 29 35 36 44 48 49 57 58 65 68 73 75 78 80
56 03/07/1999,02 03 05 06 10 14 20 25 28 31 34 38 41 47 53 57 60 69 73 80
57 03/06/1999,04 06 07 09 12 14 26 30 35 36 37 39 42 54 56 58 61 62 65 77
58 03/05/1999,03 05 09 10 12 13 18 23 24 37 47 48 53 55 58 59 69 71 77 79
59 03/04/1999,11 15 17 19 27 32 37 38 40 50 53 55 57 59 66 68 70 75 76 79
60 03/03/1999,03 04 08 12 17 18 19 22 24 25 37 39 41 55 59 60 61 68 74 80

```
61      03/02/1999,06 09 14 21 34 46 51 54 61 64 65 66 67 69 70 74 76 77 78 80
62      03/01/1999,01 04 07 15 18 22 34 41 42 44 48 56 58 60 63 74 75 78 79 80
63      02/28/1999,01 03 04 11 12 14 18 19 25 26 38 43 44 50 55 66 68 75 78 79
64      02/27/1999,01 03 05 06 13 20 21 28 29 33 34 35 40 46 49 51 53 63 67 73
65      02/26/1999,06 08 09 11 13 24 25 40 41 42 43 44 45 49 52 53 56 63 70 76
66      02/25/1999,02 03 11 13 21 24 25 32 38 43 47 56 57 59 64 65 73 75 77 79
67      02/24/1999,07 10 14 16 22 23 29 40 43 48 49 51 55 56 58 63 65 75 76 78
68      02/23/1999,01 07 10 15 16 19 25 28 29 36 46 51 52 58 61 64 68 70 74 79
69      02/22/1999,01 02 06 10 12 14 16 21 22 31 32 38 47 50 51 57 64 66 68 70
70      02/21/1999,01 05 07 08 10 16 22 28 36 37 40 46 47 52 56 58 67 76 77 78
71      02/20/1999,07 09 10 11 25 26 28 29 30 36 37 38 49 63 65 66 71 72 74 77
72      02/19/1999,02 04 12 13 15 30 32 38 39 43 45 46 52 53 61 67 73 76 77 78
73      02/18/1999,02 08 09 12 18 30 31 33 41 45 46 51 52 62 63 65 66 67 79 80
74      02/17/1999,02 06 09 11 14 17 27 35 38 41 42 43 46 48 52 55 59 65 73 76
75      02/16/1999,03 10 13 15 17 24 25 29 34 38 39 46 47 50 54 57 61 62 77 78
76      02/15/1999,06 16 17 19 22 31 33 34 36 37 50 55 56 63 66 67 69 73 74 79
77      02/14/1999,01 10 16 18 33 38 46 49 51 54 58 59 66 68 72 76 77 78 79 80
78      02/13/1999,15 25 26 31 33 37 38 40 48 49 53 54 55 57 58 61 69 72 75 79
79      02/12/1999,02 07 12 18 19 21 22 30 35 39 42 44 50 51 54 64 67 69 72 80
80      02/11/1999,02 03 04 07 14 16 20 21 22 28 35 44 51 54 60 62 63 68 69 80
81      02/10/1999,06 11 15 17 20 24 25 27 38 39 44 46 48 55 57 63 69 70 72 74
82      02/09/1999,04 12 14 16 20 22 23 26 31 34 37 39 55 56 59 62 64 68 72 74
83      02/08/1999,01 04 05 11 15 21 24 32 33 34 41 43 45 49 54 58 64 69 75 77
84      02/07/1999,18 19 22 24 25 27 29 34 35 37 41 47 48 51 52 53 61 78 79 80
85      02/06/1999,02 07 16 19 22 26 27 28 33 34 35 39 49 54 63 64 67 70 78 80
86      02/05/1999,02 04 08 13 19 20 21 24 32 33 34 43 44 50 51 52 53 74 78 80
87      02/04/1999,02 07 16 18 21 27 29 30 35 36 37 46 52 57 58 64 65 69 70 71
88      02/03/1999,05 07 12 17 18 22 23 24 25 27 30 33 45 47 49 51 54 57 75 77
89      02/02/1999,04 10 15 17 20 24 25 26 27 29 36 45 52 55 60 66 68 70 76 78
90      02/01/1999,04 10 13 25 26 28 31 34 39 40 43 45 54 56 60 69 70 72 73 77
91      01/31/1999,04 08 09 16 18 26 33 34 35 36 46 47 54 56 59 64 68 70 75 79
92      01/30/1999,01 04 16 17 19 31 33 36 39 42 44 46 49 56 57 59 61 65 68 76
93      01/29/1999,10 12 15 21 24 25 27 36 38 40 41 50 51 52 54 62 64 71 75 79
94      01/28/1999,03 08 10 19 20 24 25 33 34 35 38 40 43 47 55 62 63 70 71 77
95      01/27/1999,05 07 27 30 31 33 36 40 47 48 50 57 62 64 66 72 73 74 76 79
96      01/26/1999,05 08 15 16 17 19 20 21 30 31 36 41 53 60 61 62 64 65 66 68
97      01/25/1999,02 04 06 09 10 11 17 24 31 38 43 44 49 52 55 70 71 76 78 80
98      01/24/1999,07 13 14 15 23 25 26 30 33 44 45 47 48 52 58 60 64 70 77 78
99      01/23/1999,09 12 13 18 24 26 27 33 35 36 56 57 58 61 69 71 72 73 77 79
00      01/22/1999,04 05 10 12 15 16 19 25 29 32 34 36 51 55 65 69 72 76 77 78
01      01/21/1999,02 04 08 12 13 37 40 42 43 45 47 52 58 61 62 65 70 72 75 79
02      01/20/1999,11 13 15 17 20 22 24 26 28 35 36 37 45 47 50 53 58 60 63 73
03      01/19/1999,01 12 13 23 31 35 39 45 46 47 49 52 53 56 61 67 71 74 75 79
04      01/18/1999,02 03 07 09 13 15 16 17 21 23 26 30 32 41 52 54 56 64 67 76
05      01/17/1999,03 04 10 11 12 16 17 18 24 33 36 37 38 40 49 52 56 62 69 76
06      01/16/1999,02 04 14 17 18 19 23 27 46 51 59 62 63 67 68 73 74 75 76 77
07      01/15/1999,03 04 06 08 10 11 20 23 25 37 44 58 59 64 65 68 69 70 75 77
08      01/14/1999,01 02 07 08 11 14 16 18 30 31 45 56 57 60 61 63 68 75 79 80
09      01/13/1999,04 07 08 10 15 17 18 20 22 23 26 31 34 36 39 43 46 55 57 67
10      01/12/1999,04 07 14 16 19 25 37 38 39 43 44 47 53 59 63 65 67 73 75 79
11      01/11/1999,01 10 14 15 21 26 27 28 29 45 46 51 54 55 58 60 62 75 77 79
12      01/10/1999,01 04 05 08 10 12 13 16 20 23 29 30 36 41 46 50 58 64 76 77
13      01/09/1999,01 02 03 04 05 14 18 29 30 36 39 40 50 52 57 61 62 69 71 79
14      01/08/1999,06 08 13 14 23 26 31 33 35 36 39 41 44 46 56 59 77 78 79 80
```

```
01/07/1999,02 06 08 11 22 27 28 37 40 45 47 48 50 57 59 62 64 69 71 80
01/06/1999,05 12 13 15 19 20 24 28 30 31 35 36 46 54 55 57 64 67 72 76
01/05/1999,01 04 07 13 16 17 22 23 36 41 42 48 49 55 57 69 71 74 79 80
01/04/1999,07 08 13 19 25 33 40 44 45 46 47 48 54 55 59 63 65 69 79 80
01/03/1999,02 03 11 16 25 30 34 37 42 49 58 59 60 63 64 65 66 72 74 76
01/02/1999,01 03 15 16 17 18 32 34 35 36 38 49 51 54 57 63 67 70 72 80
01/01/1999,05 09 12 16 24 25 27 28 31 37 39 40 42 51 53 56 66 67 76 78
12/31/1998,09 16 20 21 24 29 32 34 36 37 40 44 49 56 61 62 65 67 71 73
12/30/1998,03 05 13 18 24 27 29 39 40 48 55 56 58 61 63 66 69 70 74 77
12/29/1998,04 16 18 20 22 25 26 27 30 32 35 42 44 54 69 70 72 73 74 79
12/28/1998,01 05 08 09 10 13 14 18 25 28 34 35 38 39 40 48 58 69 74 80
12/27/1998,01 08 16 19 22 23 27 28 29 30 41 43 46 47 49 50 54 55 56 67
12/26/1998,01 02 06 07 08 26 28 33 38 39 46 50 55 56 58 60 64 72 75 79
12/24/1998,03 05 09 17 27 31 32 33 39 44 46 47 50 51 53 56 60 68 78 79
12/23/1998,06 08 09 10 13 15 18 24 26 31 33 39 42 43 50 51 55 57 67 72
12/22/1998,03 07 08 15 24 25 27 29 34 37 38 42 43 45 49 56 64 69 72 80
12/21/1998,03 04 05 10 19 27 31 42 44 47 54 57 62 65 66 67 70 71 72 76
12/20/1998,02 16 21 23 27 31 34 37 38 43 49 53 58 66 68 72 73 77 79 80
12/19/1998,02 05 08 09 21 34 36 39 41 44 46 47 50 57 59 63 64 69 70 73
12/18/1998,06 07 13 14 16 20 26 27 29 32 34 44 56 58 60 61 68 70 75 80
12/17/1998,01 07 13 14 16 17 18 19 23 29 32 38 40 43 57 58 60 68 71 76
12/16/1998,03 04 08 11 13 14 21 24 30 31 35 36 39 40 41 44 51 58 62 70
12/15/1998,02 03 08 09 12 19 21 26 28 32 36 38 52 54 55 56 57 62 65 80
12/14/1998,01 04 06 11 14 25 31 35 39 47 50 53 57 63 66 68 71 72 78 80
12/13/1998,02 10 13 18 22 27 31 43 45 47 48 52 54 58 59 61 69 73 77 80
12/12/1998,02 07 13 14 15 26 30 33 37 40 41 44 48 49 51 54 68 75 78 79
12/11/1998,01 05 07 10 13 17 27 32 34 51 52 53 54 55 62 64 65 72 74 79
12/10/1998,15 19 20 23 24 27 28 36 37 39 42 44 45 47 50 61 62 63 65 78
12/09/1998,01 06 07 08 09 11 13 19 34 39 51 54 56 59 63 64 69 70 71 79
12/08/1998,10 13 16 17 18 25 28 30 33 35 39 40 45 51 54 64 67 77 78 80
12/07/1998,01 06 09 11 16 18 19 22 24 29 33 35 40 48 49 53 66 72 73 77
12/06/1998,07 11 13 23 28 30 31 34 43 45 48 53 57 58 60 63 64 70 77 79
12/05/1998,05 09 10 12 15 17 19 29 30 38 39 41 51 53 55 57 67 70 74 76
12/04/1998,16 19 20 25 27 31 33 36 37 44 45 51 55 56 57 65 69 73 79 80
12/03/1998,02 05 07 11 22 25 26 27 33 45 47 49 50 62 63 65 66 67 79 80
12/02/1998,03 08 10 12 14 16 18 24 26 31 35 44 47 56 58 59 67 73 75 76
12/01/1998,06 07 10 14 22 23 28 33 36 38 42 46 47 49 56 58 60 66 72 79
11/30/1998,04 06 16 31 32 35 37 38 39 41 42 43 44 47 50 52 61 64 67 76
11/29/1998,02 05 11 13 19 21 22 30 33 34 41 42 44 48 49 57 60 61 75 78
11/28/1998,05 09 17 22 29 35 38 44 47 50 60 62 65 66 67 69 70 71 74 75
11/27/1998,02 03 04 05 07 08 10 12 28 30 35 40 47 48 57 60 64 67 73 78
11/26/1998,04 08 17 18 26 27 28 29 34 36 41 42 45 47 48 50 52 58 59 66
11/25/1998,05 10 22 24 27 33 34 39 51 52 54 56 58 60 61 62 66 69 73 80
11/24/1998,04 08 09 15 17 18 22 29 30 37 39 46 47 57 58 65 71 74 75 77
11/23/1998,01 07 10 11 12 15 28 30 31 34 40 41 44 46 50 54 59 66 69 75
11/22/1998,01 03 04 08 11 14 21 22 29 31 39 46 47 54 56 57 59 67 72 78
11/21/1998,01 04 09 11 20 23 26 34 40 41 42 44 48 51 53 56 61 74 75 79
11/20/1998,06 08 13 14 22 24 27 28 32 37 38 40 50 52 57 60 72 73 78 79
11/19/1998,02 17 18 28 33 38 39 41 43 45 48 60 64 70 71 72 73 74 77 79
11/18/1998,01 04 09 18 20 26 27 28 29 31 33 36 40 42 44 61 63 67 68 70
11/17/1998,02 13 18 27 33 34 38 39 46 47 49 50 55 56 64 66 67 69 74 76
11/16/1998,01 06 07 08 14 26 27 33 38 41 43 50 52 56 59 64 68 72 74 79
11/15/1998,01 02 03 11 14 15 21 25 26 27 28 36 44 47 55 59 63 66 73 80
11/14/1998,01 02 05 10 12 17 26 28 29 30 31 32 33 35 38 50 56 74 76 79
```

```
69      11/13/1998,06 07 08 10 16 18 20 22 25 26 28 41 43 44 58 62 67 70 76 79
70      11/12/1998,02 05 07 08 10 12 13 16 28 35 36 43 46 57 63 66 70 72 74 78
71      11/11/1998,04 05 09 13 24 32 35 40 42 44 45 48 50 56 62 68 71 74 77 79
72      11/10/1998,06 10 12 13 14 17 22 27 33 36 42 43 44 45 49 50 56 58 70 71
73      11/09/1998,04 11 14 15 23 25 30 32 33 37 38 44 48 50 51 57 59 63 64 72
74      11/08/1998,13 16 22 27 30 32 35 39 43 45 47 48 52 55 56 59 68 71 73 80
75      11/07/1998,02 09 10 12 25 27 30 32 36 37 41 45 47 50 52 60 64 69 76 79
76      11/06/1998,04 06 16 17 29 30 33 36 37 41 51 57 60 65 66 68 71 75 76 80
77      11/05/1998,01 06 16 18 25 37 47 52 54 56 58 60 63 68 69 72 73 75 77 79
78      11/04/1998,01 02 04 06 14 24 26 28 30 33 41 44 45 47 51 54 59 69 78 80
79      11/03/1998,11 15 17 22 30 32 33 38 40 44 47 49 50 52 57 59 63 65 74 75
80      11/02/1998,05 09 10 13 14 22 33 34 35 36 43 49 53 55 62 64 67 72 74 78
81      11/01/1998,01 05 07 10 12 16 20 23 25 26 36 38 40 42 43 48 58 67 69 74
82      10/31/1998,04 06 12 23 24 31 38 39 47 48 51 52 56 59 61 62 66 75 77 78
83      10/30/1998,06 11 13 20 21 33 35 41 42 43 44 50 52 63 64 68 71 72 74 75
84      10/29/1998,08 10 11 13 21 25 32 33 37 38 49 52 58 62 63 67 72 76 78 80
85      10/28/1998,01 04 05 13 16 19 20 26 28 31 32 45 60 61 62 63 65 68 73 76
86      10/27/1998,05 07 09 12 19 22 24 27 29 30 33 34 41 46 52 55 58 61 67 71
87      10/26/1998,06 14 15 16 23 26 28 30 37 38 44 46 47 49 56 65 69 74 76 78
88      10/25/1998,14 15 21 22 27 29 30 35 36 43 51 52 57 58 60 61 67 70 72 73
89      10/24/1998,01 02 19 24 26 27 33 35 43 45 48 62 63 65 66 67 69 74 76 79
90      10/23/1998,01 05 10 14 17 19 31 32 34 36 44 46 51 55 57 64 69 75 76 78
91      10/22/1998,08 10 21 23 24 26 27 31 33 34 39 40 42 43 46 53 55 68 78 80
92      10/21/1998,11 16 17 32 34 36 38 39 40 43 46 49 50 55 57 62 71 72 76 78
93      10/20/1998,01 02 05 07 08 15 27 28 34 40 46 51 53 58 60 61 68 69 70 79
94      10/19/1998,03 08 10 11 12 19 27 28 29 35 37 40 42 47 55 57 58 60 68 69
95      10/18/1998,01 03 05 06 09 11 13 15 16 28 30 37 38 40 51 63 66 68 74 80
96      10/17/1998,06 19 22 23 24 26 30 38 39 44 45 49 52 56 59 61 67 69 73 75
97      10/16/1998,01 07 13 18 19 23 24 26 27 35 37 40 47 49 63 67 73 76 77 79
98      10/15/1998,02 03 09 10 11 16 29 32 34 35 41 43 44 47 55 60 63 68 69 76
99      10/14/1998,11 14 15 33 34 35 37 39 40 42 45 50 53 57 62 68 69 71 72 75
00      10/13/1998,06 08 15 20 24 28 29 32 42 43 48 51 54 55 59 61 65 68 71 76
01      10/12/1998,02 07 09 14 17 19 21 23 26 31 34 35 37 39 49 59 61 66 70 78
02      10/11/1998,03 04 05 09 21 22 24 38 39 40 49 54 56 62 65 69 70 71 72 80
03      10/10/1998,02 06 08 09 11 12 17 19 25 28 30 35 37 43 48 51 53 57 60 68
04      10/09/1998,03 04 12 15 17 21 23 26 28 29 31 32 38 41 44 45 58 61 62 80
05      10/08/1998,10 13 17 18 22 23 25 29 31 36 42 43 47 52 53 55 57 61 68 79
06      10/07/1998,05 13 17 21 24 28 29 32 35 37 47 49 50 53 60 61 64 67 68 79
07      10/06/1998,04 14 17 18 20 25 28 34 36 38 44 45 47 55 58 66 74 76 77 79
08      10/05/1998,03 04 08 09 14 18 23 25 27 35 40 42 43 46 48 49 60 64 75 77
09      10/04/1998,02 05 09 10 13 18 30 33 41 51 54 61 64 67 68 69 70 71 72 76
10      10/03/1998,01 03 06 10 14 15 16 17 19 20 26 31 35 39 50 53 58 63 76 78
11      10/02/1998,07 09 20 25 27 29 32 35 42 43 44 47 57 61 64 70 74 75 79 80
12      10/01/1998,01 10 12 19 21 24 26 33 37 39 41 43 55 61 63 69 70 77 79 80
13      09/30/1998,03 08 15 23 26 33 34 39 42 47 56 59 60 61 66 70 71 76 79 80
14      09/29/1998,01 03 09 18 21 32 34 37 45 48 50 52 57 58 63 64 65 67 72 79
15      09/28/1998,10 12 20 22 23 24 27 29 32 37 41 44 45 47 53 57 58 67 72 79
16      09/27/1998,01 02 03 04 05 18 23 25 32 34 35 39 40 42 45 47 48 57 58 77
17      09/26/1998,04 05 06 08 14 15 17 24 32 35 38 47 48 49 57 68 71 72 74 80
18      09/25/1998,02 04 05 07 08 15 19 22 25 31 33 35 36 39 42 46 52 56 64 67
19      09/24/1998,02 15 24 26 29 37 39 40 44 45 49 50 53 57 58 60 71 74 76 79
20      09/23/1998,11 14 15 16 18 21 23 29 31 36 37 43 44 49 61 66 68 72 75 77
21      09/22/1998,04 08 09 12 15 18 20 22 23 29 40 42 46 52 54 57 62 75 77 80
22      09/21/1998,01 03 05 06 07 09 11 20 22 24 28 33 41 48 52 53 56 57 61 72
```

```
23      09/20/1998,04 05 07 12 15 22 26 27 28 36 43 44 45 46 47 52 53 55 75 76
24      09/19/1998,01 03 07 08 09 13 14 26 31 32 33 36 41 43 45 46 47 63 64 70
25      09/18/1998,02 06 15 22 26 31 33 35 43 44 47 48 60 62 63 67 71 74 78 79
26      09/17/1998,04 12 14 18 27 31 34 35 37 39 44 49 54 61 65 67 76 77 79 80
27      09/16/1998,02 03 05 09 12 13 22 23 26 29 32 34 47 54 59 61 65 72 73 77
28      09/15/1998,02 05 08 10 12 14 16 18 21 23 34 36 48 49 58 60 62 64 70 76
29      09/14/1998,05 08 12 16 23 24 29 32 39 43 48 56 58 65 67 69 70 71 73 74
30      09/13/1998,01 04 07 10 13 17 18 19 21 22 26 28 30 32 43 52 72 74 76 80
31      09/12/1998,05 09 22 28 30 32 33 35 36 41 44 47 48 53 57 62 66 73 76 77
32      09/11/1998,01 03 04 06 08 09 12 22 27 28 29 33 35 41 50 55 66 69 75 77
33      09/10/1998,02 07 13 15 17 18 19 23 27 29 36 39 40 42 43 44 47 65 68 80
34      09/09/1998,05 09 10 11 14 20 21 33 40 43 51 61 62 63 66 68 70 73 74 77
35      09/08/1998,05 07 11 14 18 20 23 34 35 46 52 55 59 60 62 64 66 67 77 80
36      09/07/1998,03 04 06 08 10 19 20 23 24 36 38 40 43 44 45 57 58 61 65 80
37      09/06/1998,01 04 17 18 20 29 30 35 36 37 38 39 43 46 49 65 73 74 75 80
38      09/05/1998,03 05 07 12 17 20 21 23 27 28 31 37 41 45 50 55 58 59 73 74
39      09/04/1998,09 20 23 26 27 28 34 37 39 41 43 44 45 49 56 63 65 71 72 78
40      09/03/1998,06 09 18 25 28 31 34 42 43 49 52 54 58 61 64 66 71 72 79 80
41      09/02/1998,03 06 08 11 12 17 18 19 22 31 34 35 36 38 46 47 51 56 60 61
42      09/01/1998,01 07 08 10 14 18 22 25 30 32 38 39 42 43 44 46 59 63 65 67
43      08/31/1998,08 15 16 17 24 32 35 36 38 39 46 54 57 61 65 66 68 70 72 74
44      08/30/1998,02 07 11 12 25 26 30 32 38 40 43 50 57 58 59 63 66 76 77 80
45      08/29/1998,08 09 10 14 16 17 20 26 30 33 36 41 48 58 62 63 64 65 69 72
46      08/28/1998,02 03 06 13 24 27 29 30 33 37 39 40 44 48 49 51 52 65 76 78
47      08/27/1998,05 09 11 14 17 19 22 34 39 42 44 53 55 58 59 62 67 71 73 77
48      08/26/1998,02 08 13 19 21 24 33 37 47 51 52 54 56 61 63 67 72 73 76 80
49      08/25/1998,06 15 25 29 33 38 41 43 46 51 54 58 59 62 67 69 70 72 74 79
50      08/24/1998,02 03 08 12 14 15 24 25 28 38 43 46 47 49 50 59 67 69 70 74
51      08/23/1998,03 06 16 17 25 27 30 32 33 35 37 38 44 47 50 51 56 62 64 75
52      08/22/1998,07 08 09 13 14 18 20 33 34 38 44 48 49 54 59 60 66 76 77 80
53      08/21/1998,07 15 16 19 20 23 25 34 41 44 47 49 50 51 65 75 76 77 78 79
54      08/20/1998,10 17 19 20 21 30 41 49 50 54 58 60 61 66 68 71 72 76 78 80
55      08/19/1998,01 04 06 09 14 16 17 22 23 26 36 50 54 55 56 62 63 66 70 74
56      08/18/1998,01 12 13 16 17 18 22 30 37 41 44 49 51 54 57 58 65 67 70 80
57      08/17/1998,02 07 12 27 31 37 39 41 45 51 56 57 61 62 67 68 71 76 77 78
58      08/16/1998,03 13 15 21 29 32 37 45 46 48 57 58 59 65 66 67 68 74 76 80
59      08/15/1998,05 07 10 11 13 16 20 23 28 30 31 32 35 36 42 56 61 65 69 79
60      08/14/1998,01 02 03 10 14 20 22 25 26 35 41 45 49 54 56 61 63 65 67 77
61      08/13/1998,02 07 09 10 11 15 23 24 27 28 37 39 41 42 45 46 69 71 73 78
62      08/12/1998,07 14 18 27 28 30 32 36 41 42 46 47 49 50 51 61 62 68 76 80
63      08/11/1998,04 09 10 18 23 27 30 31 35 38 48 49 52 58 61 64 65 66 75 76
64      08/10/1998,05 09 10 18 19 20 24 26 27 36 43 46 52 61 62 64 71 72 76 79
65      08/09/1998,03 04 08 14 18 20 27 28 29 38 40 42 43 47 57 58 68 70 74 79
66      08/08/1998,01 08 09 17 19 26 28 31 43 45 47 52 54 58 62 73 74 75 77 80
67      08/07/1998,06 07 08 12 13 15 28 33 35 47 52 60 63 65 70 71 75 78 79 80
68      08/06/1998,02 03 06 08 09 11 17 19 20 21 22 24 25 32 35 36 42 46 63 69
69      08/05/1998,01 03 04 06 08 10 13 15 17 20 35 36 38 39 41 51 52 55 56 73
70      08/04/1998,03 07 12 14 27 31 32 34 40 41 42 43 45 47 49 55 59 64 66 78
71      08/03/1998,02 07 08 28 30 31 35 39 43 45 56 61 65 68 72 74 75 76 77 78
72      08/02/1998,05 12 13 29 35 37 39 41 44 45 48 50 52 56 57 60 68 74 78 79
73      08/01/1998,06 13 14 16 20 21 26 29 36 43 44 49 55 63 65 67 69 70 71 74
74      07/31/1998,01 03 12 17 21 23 24 30 38 43 45 50 53 55 57 60 61 63 64 71
75      07/30/1998,02 04 05 06 12 16 19 30 36 38 43 50 51 56 64 65 72 74 76 80
76      07/29/1998,02 03 10 13 16 19 32 33 34 38 47 55 59 61 66 68 69 74 77 78
```

```
77      07/28/1998,08 10 14 15 17 21 22 24 26 34 43 46 47 49 53 56 65 67 78 79
78      07/27/1998,06 08 09 13 17 19 20 30 39 42 60 64 65 67 69 70 74 76 77 78
79      07/26/1998,01 03 04 13 16 18 25 28 29 32 35 46 53 54 57 58 60 63 66 74
80      07/25/1998,02 04 05 07 10 18 22 24 25 27 36 37 46 52 55 58 59 64 69 73
81      07/24/1998,07 13 22 25 34 43 44 45 50 51 52 53 57 61 62 63 68 75 79 80
82      07/23/1998,02 07 08 17 20 32 47 48 50 52 53 56 59 62 66 68 72 73 79 80
83      07/22/1998,11 15 19 20 27 35 36 37 40 41 44 45 46 47 49 58 68 73 77 80
84      07/21/1998,07 08 09 15 16 18 21 22 37 40 41 47 51 52 55 56 65 67 71 77
85      07/20/1998,02 03 04 05 07 08 14 20 21 34 40 46 49 51 59 61 68 69 72 74
86      07/19/1998,03 06 08 11 18 21 25 28 29 36 37 40 49 52 58 60 62 66 68 73
87      07/18/1998,03 07 11 12 14 17 28 30 31 32 33 35 39 41 43 50 58 61 65 67
88      07/17/1998,04 06 09 20 22 27 28 30 31 39 40 41 46 56 58 62 68 69 79 80
89      07/16/1998,01 05 06 09 11 26 35 36 39 42 43 46 47 55 60 62 70 75 77 80
90      07/15/1998,04 12 13 17 24 25 29 39 40 47 48 49 51 52 55 57 61 66 72 77
91      07/14/1998,03 17 20 23 27 28 29 36 38 39 42 48 50 52 62 65 69 78 79 80
92      07/13/1998,07 10 11 14 15 16 21 24 25 30 32 34 38 54 55 63 64 65 71 76
93      07/12/1998,02 07 09 10 16 22 29 33 34 35 38 39 42 46 48 52 62 73 74 77
94      07/11/1998,06 09 14 18 20 26 28 29 32 33 37 46 51 55 60 62 64 65 78 80
95      07/10/1998,01 03 04 12 13 14 23 27 30 31 35 37 38 48 50 55 60 64 72 77
96      07/09/1998,16 18 19 22 24 31 33 36 38 44 46 56 60 61 62 63 64 72 76 79
97      07/08/1998,01 08 11 21 25 28 31 36 37 50 51 53 54 59 62 69 70 71 73 77
98      07/07/1998,06 11 16 17 21 23 26 27 33 34 46 48 49 54 57 58 65 71 72 80
99      07/06/1998,05 07 08 12 13 22 24 26 40 43 45 46 47 53 56 62 64 67 70 74
00      07/05/1998,06 09 11 17 21 25 31 35 37 41 44 45 52 57 59 63 65 66 70 76
01      07/04/1998,01 10 13 14 20 29 30 32 36 39 40 41 53 54 56 57 58 59 71 79
02      07/03/1998,02 05 06 13 17 19 24 37 41 42 48 50 53 63 64 67 70 71 76 78
03      07/02/1998,06 10 21 30 31 33 34 36 37 52 54 55 57 58 65 66 67 68 70 79
04      07/01/1998,01 04 13 14 18 21 25 28 32 34 42 44 48 52 58 70 72 73 75 79
05      06/30/1998,03 04 08 09 10 13 15 20 22 25 28 35 42 46 68 72 73 75 78 79
06      06/29/1998,05 08 11 16 20 24 32 42 44 49 54 58 63 65 66 68 69 70 71 80
07      06/28/1998,08 10 11 12 17 23 34 36 39 44 45 47 52 54 58 63 64 70 73 76
08      06/27/1998,05 12 13 16 20 21 33 38 39 45 46 55 56 57 58 59 65 66 70 77
09      06/26/1998,02 08 14 16 23 30 34 35 38 42 43 48 55 56 57 67 73 77 78 80
10      06/25/1998,03 10 13 15 18 19 23 36 37 39 41 42 46 48 54 59 61 66 77 78
11      06/24/1998,01 02 12 16 20 21 23 27 34 36 40 42 46 55 63 65 67 69 70 73
12      06/23/1998,17 19 20 21 23 25 27 28 30 34 36 39 47 50 58 59 61 62 70 78
13      06/22/1998,01 12 15 18 23 24 25 30 41 44 50 54 57 58 60 64 66 68 74 77
14      06/21/1998,01 03 04 07 08 13 15 21 28 34 35 43 52 58 64 70 71 73 78 80
15      06/20/1998,02 05 08 12 14 18 23 26 27 31 35 37 45 46 51 52 64 74 75 80
16      06/19/1998,10 11 18 19 21 28 29 36 44 47 49 50 51 56 57 65 68 73 76 78
17      06/18/1998,03 06 11 13 21 27 37 41 44 46 48 49 52 59 63 66 70 75 78 79
18      06/17/1998,02 06 10 16 17 23 25 26 31 38 43 48 51 52 53 55 64 67 70 77
19      06/16/1998,03 05 09 10 14 20 30 34 35 36 39 43 45 54 58 61 64 66 73 75
20      06/15/1998,05 06 14 16 17 23 24 25 30 34 36 42 43 46 53 55 63 67 73 76
21      06/14/1998,05 08 10 11 13 16 18 19 23 31 32 34 37 51 54 57 71 72 74 76
22      06/13/1998,09 10 18 21 25 27 33 34 35 36 42 43 48 51 54 55 58 63 72 77
23      06/12/1998,04 13 15 19 26 30 32 33 34 35 40 41 44 50 53 56 61 72 74 78
24      06/11/1998,01 05 06 12 27 34 40 41 43 44 47 50 51 53 54 56 63 70 76 78
25      06/10/1998,01 13 27 29 30 35 41 47 48 49 50 52 54 57 65 66 74 75 77 79
26      06/09/1998,04 07 22 25 28 29 33 34 39 49 51 53 58 60 63 67 72 73 74 80
27      06/08/1998,08 09 12 18 19 23 25 27 28 38 40 41 45 47 54 56 62 72 77 78
28      06/07/1998,04 07 14 15 24 28 35 45 53 54 56 57 63 67 69 70 71 74 75 80
29      06/06/1998,02 03 05 07 13 14 16 17 18 20 27 34 36 46 47 51 56 58 76 79
30      06/05/1998,13 20 25 27 28 33 37 44 45 48 50 51 54 55 56 59 66 69 73 74
```

```
06/04/1998,10 12 17 18 20 22 23 24 25 27 35 41 51 54 56 57 58 67 72 76
06/03/1998,01 02 18 22 23 24 26 30 32 35 36 45 55 56 60 66 68 69 76 80
06/02/1998,04 08 18 19 21 22 23 33 34 36 37 39 42 48 55 56 62 69 77 80
06/01/1998,01 05 12 13 18 19 22 23 25 32 33 37 40 56 58 61 64 74 79 80
05/31/1998,10 11 23 24 26 34 37 41 42 50 51 52 57 60 63 64 68 70 74 76
05/30/1998,04 09 11 12 14 17 29 30 37 38 42 44 48 54 56 64 70 71 74 80
05/29/1998,02 10 11 12 21 24 25 28 34 44 45 46 52 57 59 60 67 74 79 80
05/28/1998,02 03 15 16 21 28 29 30 33 53 55 56 58 59 60 61 70 71 74 77
05/27/1998,01 03 05 06 07 09 15 19 20 23 31 39 52 56 57 58 60 66 72 75
05/26/1998,03 09 14 15 17 21 23 25 27 31 37 43 51 52 58 59 73 76 78 80
05/25/1998,03 05 07 08 10 17 20 22 28 30 39 45 53 54 61 65 66 68 70 72
05/24/1998,06 08 11 24 31 32 33 34 39 43 44 48 50 51 53 58 68 69 72 75
05/23/1998,03 04 12 16 27 28 29 39 40 41 43 46 54 61 63 65 69 72 75 77
05/22/1998,06 07 12 14 18 24 29 32 33 40 43 47 49 50 51 57 64 69 71 73
05/21/1998,01 05 10 11 16 24 26 27 29 30 38 45 50 53 60 69 71 73 75 76
05/20/1998,03 12 19 21 22 24 25 26 27 31 32 37 40 45 51 53 55 61 65 78
05/19/1998,05 06 08 10 12 26 27 38 39 42 54 58 63 66 68 70 77 78 79 80
05/18/1998,08 11 12 15 19 20 23 27 33 37 41 57 58 63 64 69 71 72 77 78
05/17/1998,09 11 13 16 17 22 29 30 33 34 36 39 42 48 52 54 59 64 65 79
05/16/1998,03 05 09 10 11 17 21 35 37 41 43 47 51 58 62 64 67 72 74 75
05/15/1998,03 12 17 32 33 41 43 44 45 46 49 55 56 58 65 66 71 73 78 79
05/14/1998,01 03 04 09 10 23 25 34 42 43 47 49 53 54 56 59 64 75 77 79
05/13/1998,01 04 11 16 17 20 25 27 35 39 49 54 61 63 66 74 75 76 77 79
05/12/1998,03 05 09 11 16 22 26 27 28 31 47 48 50 52 55 58 62 67 69 70
05/11/1998,01 03 08 17 20 21 24 25 26 28 32 33 41 45 46 63 69 70 75 76
05/10/1998,05 06 08 14 20 26 32 37 39 40 41 44 61 69 71 73 74 77 79 80
05/09/1998,05 07 10 12 21 22 31 41 46 49 50 54 61 72 73 76 77 78 79 80
05/08/1998,02 05 07 15 22 27 28 30 35 39 41 43 46 62 64 69 71 73 77 80
05/07/1998,05 10 20 27 31 33 38 41 42 47 50 52 57 58 59 67 70 71 79 80
05/06/1998,14 15 16 19 21 25 27 28 29 31 39 42 44 54 57 62 64 65 68 73
05/05/1998,03 06 07 08 09 11 12 22 23 31 32 37 43 47 55 59 69 71 73 79
05/04/1998,04 05 08 24 26 28 35 39 43 45 46 48 53 56 63 67 70 75 76 77
05/03/1998,10 12 15 18 19 24 28 29 48 49 50 51 58 61 67 69 70 71 78 80
05/02/1998,03 04 08 15 16 23 24 35 36 40 46 48 52 60 61 66 67 68 69 75
05/01/1998,03 07 08 12 22 23 27 29 30 33 34 40 43 48 49 63 70 71 76 77
04/30/1998,03 06 11 15 16 19 25 27 30 32 33 35 43 45 49 52 53 61 65 76
04/29/1998,01 03 07 08 11 13 15 31 37 42 43 46 48 49 55 56 59 70 76 80
04/28/1998,02 06 08 15 16 19 26 29 32 35 37 38 41 50 51 54 56 67 72 79
04/27/1998,04 12 25 26 30 34 39 45 47 48 49 52 58 60 61 70 73 74 75 78
04/26/1998,01 04 05 10 14 16 18 22 26 28 31 33 35 40 45 53 54 55 64 79
04/25/1998,09 20 22 30 32 33 34 35 39 42 44 46 47 51 57 60 67 71 74 75
04/24/1998,08 12 14 18 21 22 31 34 40 42 51 52 61 62 66 75 76 77 78 79
04/23/1998,04 14 19 25 28 37 41 43 47 48 51 52 59 60 66 67 69 73 76 79
04/22/1998,03 05 07 09 11 16 17 22 25 27 29 35 45 50 54 61 65 71 75 77
04/21/1998,07 09 11 24 29 33 34 47 49 50 51 53 56 58 63 65 67 68 71 76
04/20/1998,02 04 07 08 10 12 16 18 19 24 31 32 45 51 65 66 70 73 78 79
04/19/1998,01 02 14 16 24 30 32 41 43 45 47 49 50 52 58 60 69 74 76 79
04/18/1998,01 08 12 14 16 17 25 26 31 34 40 48 49 52 62 64 68 76 77 79
04/17/1998,05 07 08 10 11 14 15 16 27 45 53 56 62 63 70 71 72 75 76 79
04/16/1998,04 10 17 18 21 22 23 32 35 36 44 46 51 54 55 58 64 65 70 73
04/15/1998,05 06 12 14 22 23 25 30 36 41 45 46 57 65 67 68 71 74 75 78
04/14/1998,01 03 09 10 20 27 32 35 43 57 62 63 64 66 69 70 72 77 78 80
04/13/1998,04 07 14 15 18 19 21 22 24 31 39 40 41 46 58 61 69 70 72 78
04/12/1998,01 03 04 11 15 20 22 31 33 34 38 41 47 54 65 66 67 74 75 78
```

```
85      04/11/1998,02 05 06 11 13 15 17 23 25 26 42 47 48 49 50 51 53 57 68 72
86      04/10/1998,03 07 08 10 11 14 15 16 27 29 30 31 46 48 56 57 59 68 69 78
87      04/09/1998,01 02 04 09 10 11 12 23 25 29 33 35 37 39 43 49 55 67 69 77
88      04/08/1998,07 10 13 14 16 21 24 30 31 38 39 41 42 44 55 56 71 72 74 78
89      04/07/1998,01 03 05 23 32 34 39 43 45 52 57 63 67 73 74 75 76 78 79 80
90      04/06/1998,12 19 21 26 27 28 35 36 37 38 41 44 46 50 53 59 64 68 70 73
91      04/05/1998,03 07 10 13 18 23 31 33 34 36 37 40 41 47 49 52 59 67 77 80
92      04/04/1998,03 04 06 08 28 30 32 34 39 42 44 52 55 56 60 62 68 71 75 78
93      04/03/1998,05 10 14 19 23 24 26 27 31 32 45 46 47 48 54 55 57 64 71 77
94      04/02/1998,08 09 12 14 21 25 29 38 42 44 48 57 58 61 65 68 69 76 77 79
95      04/01/1998,02 07 08 10 16 18 19 27 28 38 40 46 48 50 56 60 67 69 73 77
96      03/31/1998,01 06 09 10 13 15 27 33 41 45 47 48 50 52 54 61 63 70 75 76
97      03/30/1998,04 06 15 19 21 27 30 40 46 48 59 61 63 64 67 71 74 76 78 79
98      03/29/1998,12 13 18 25 26 28 29 30 35 37 49 53 54 55 62 68 74 75 77 78
99      03/28/1998,04 11 18 22 23 24 28 39 42 48 50 52 56 64 65 69 71 76 77 78
00      03/27/1998,02 08 13 14 23 24 26 30 39 41 42 43 59 64 66 68 70 74 75 78
01      03/26/1998,01 02 07 18 19 25 28 29 35 37 38 39 40 46 49 51 60 68 69 72
02      03/25/1998,07 10 11 18 19 21 23 30 39 41 48 49 54 58 59 65 66 70 74 77
03      03/24/1998,10 14 16 19 28 35 42 47 48 52 55 58 59 64 67 69 72 73 77 78
04      03/23/1998,01 02 08 16 17 20 21 28 31 37 41 43 44 49 52 55 72 76 79 80
05      03/22/1998,03 07 13 17 27 28 30 38 41 42 46 48 49 50 62 63 69 70 71 72
06      03/21/1998,04 05 06 08 13 14 15 16 21 24 33 34 44 59 62 71 73 74 75 77
07      03/20/1998,01 06 15 18 20 23 30 31 40 45 49 51 52 53 55 56 57 59 74 75
08      03/19/1998,07 17 21 25 27 30 32 39 43 48 51 58 59 63 65 66 70 72 75 80
09      03/18/1998,01 02 09 10 16 22 24 28 29 38 39 41 43 45 50 53 56 60 75 79
10      03/17/1998,03 10 13 14 17 19 26 32 34 36 37 39 45 51 52 58 65 68 72 74
11      03/16/1998,04 09 14 15 18 19 23 33 34 36 38 39 40 51 57 63 69 70 73 74
12      03/15/1998,01 02 03 06 08 14 15 24 25 27 28 30 31 35 43 58 68 70 71 79
13      03/14/1998,01 03 11 13 19 20 25 31 34 36 47 53 57 58 65 67 68 70 79 80
14      03/13/1998,04 16 18 23 28 29 35 39 45 50 51 52 55 56 57 63 65 67 73 80
15      03/12/1998,03 10 18 21 30 32 33 35 38 39 42 43 52 60 61 71 72 73 75 78
16      03/11/1998,01 04 08 09 18 19 22 28 32 33 34 39 46 48 50 51 59 67 70 78
17      03/10/1998,02 13 24 25 26 28 30 32 38 41 43 49 65 66 68 70 73 75 76 78
18      03/09/1998,09 11 13 17 22 23 33 35 43 46 48 57 62 67 68 69 70 72 73 78
19      03/08/1998,01 04 05 09 14 17 19 21 27 29 34 36 45 50 53 63 64 72 74 75
20      03/07/1998,03 04 10 15 16 20 22 27 34 38 40 41 47 51 52 55 57 63 67 73
21      03/06/1998,13 14 18 19 21 25 26 27 30 34 37 41 42 46 47 57 60 68 71 78
22      03/05/1998,05 09 10 11 12 15 16 19 26 28 36 41 42 44 46 49 53 59 65 71
23      03/04/1998,05 14 16 17 20 22 29 31 32 35 43 45 56 57 61 66 74 76 77 79
24      03/03/1998,01 12 20 23 30 32 35 39 41 42 44 46 50 51 54 60 65 66 71 77
25      03/02/1998,04 07 09 17 19 26 27 28 29 38 51 52 53 60 61 64 65 68 72 77
26      03/01/1998,01 03 07 11 14 21 24 28 42 45 48 49 50 51 56 59 66 67 74 75
27      02/28/1998,01 10 15 17 21 24 39 41 46 48 50 52 57 60 64 67 71 72 78 79
28      02/27/1998,02 04 17 19 22 24 28 29 30 33 42 46 51 53 56 60 66 68 75 80
29      02/26/1998,02 04 10 16 18 21 24 28 30 38 40 44 45 63 66 68 70 73 77 79
30      02/25/1998,01 05 06 10 12 18 24 26 28 32 34 37 44 50 52 53 65 69 74 75
31      02/24/1998,03 04 05 07 08 09 10 16 20 23 26 27 28 35 36 48 50 57 63 78
32      02/23/1998,09 12 13 16 18 19 26 31 35 41 43 47 51 54 56 58 64 67 70 71
33      02/22/1998,04 06 15 17 19 21 24 25 27 28 38 40 54 60 62 64 70 74 77 79
34      02/21/1998,01 04 14 16 17 21 30 31 34 35 39 42 43 46 52 56 57 59 62 64
35      02/20/1998,03 05 11 12 14 22 24 30 34 35 39 44 51 52 54 62 65 66 73 78
36      02/19/1998,02 17 20 22 26 34 39 40 41 51 52 53 56 60 63 67 68 69 75 76
37      02/18/1998,01 06 07 11 22 27 35 36 47 52 55 57 58 59 63 66 68 70 72 80
38      02/17/1998,03 13 15 18 23 24 29 43 44 47 51 53 58 59 60 63 64 72 73 79
```

```
02/16/1998,06 08 09 10 14 19 21 23 24 27 28 32 36 43 44 45 51 59 68 74
02/15/1998,05 06 10 13 15 17 22 28 32 37 40 43 49 56 60 65 67 74 76 80
02/14/1998,02 05 12 13 15 16 19 21 36 37 52 53 56 57 63 64 65 72 77 80
02/13/1998,05 06 07 09 10 16 21 36 42 43 47 48 49 56 66 67 68 77 78 80
02/12/1998,02 03 04 14 17 23 27 29 31 36 43 46 47 54 55 59 63 66 67 78
02/11/1998,03 06 08 11 13 16 17 20 21 24 25 32 36 50 52 55 67 69 75 78
02/10/1998,01 06 09 11 17 23 25 26 37 49 53 56 59 60 62 65 66 68 70 72
02/09/1998,04 10 13 24 29 31 34 40 47 49 51 55 60 64 65 70 73 75 76 79
02/08/1998,07 15 16 28 30 36 40 42 45 48 49 55 60 61 62 63 69 70 74 77
02/07/1998,03 04 05 06 15 16 19 22 26 27 30 32 39 44 45 57 62 65 70 77
02/06/1998,03 10 14 17 23 24 38 43 44 53 55 57 60 62 63 64 65 67 71 79
02/05/1998,03 04 10 13 14 25 26 34 35 36 48 49 54 55 59 64 65 67 70 77
02/04/1998,01 02 03 05 06 10 18 22 27 32 38 41 43 46 48 50 57 58 64 77
02/03/1998,02 06 08 11 20 24 34 36 39 44 47 48 51 52 55 56 68 74 75 76
02/02/1998,03 10 12 15 17 24 28 29 34 39 41 44 46 47 52 53 59 66 68 78
02/01/1998,04 13 19 22 26 27 35 37 50 51 56 57 58 62 63 68 72 73 75 76
01/31/1998,01 03 04 06 14 21 24 27 29 30 35 37 38 51 53 57 62 66 72 74
01/30/1998,05 09 13 16 18 23 29 30 32 35 36 39 42 44 45 48 55 58 70 80
01/29/1998,04 07 14 15 20 21 29 31 50 51 53 55 56 58 60 65 69 71 75 78
01/28/1998,01 03 04 12 17 18 19 23 25 29 34 38 40 41 45 52 58 60 70 78
01/27/1998,03 09 14 18 19 21 23 31 33 34 44 46 47 50 56 68 74 77 78 80
01/26/1998,05 06 08 09 26 30 35 37 39 41 43 50 54 55 60 63 66 69 79 80
01/25/1998,01 02 14 15 18 22 25 29 42 44 54 58 60 62 66 67 68 70 74 77
01/24/1998,02 07 10 11 12 13 17 22 25 26 29 31 33 37 39 41 45 46 61 75
01/23/1998,02 04 05 06 12 13 17 22 24 27 38 45 46 47 51 62 67 74 75 80
01/22/1998,01 10 12 14 18 21 23 33 43 44 50 54 59 61 65 67 68 71 75 76
01/21/1998,08 09 10 11 12 14 15 16 36 38 40 46 54 56 58 60 61 64 70 74
01/20/1998,05 06 08 11 12 18 19 28 34 35 36 41 54 61 64 67 68 70 75 80
01/19/1998,11 12 14 15 19 23 34 39 40 46 47 49 52 55 58 60 61 67 72 75
01/18/1998,02 08 10 18 21 26 32 36 40 43 53 54 58 60 61 62 66 71 73 77
01/17/1998,02 04 20 22 24 26 28 34 36 41 44 46 48 50 51 54 57 62 75 78
01/16/1998,03 04 07 09 10 12 13 19 26 43 45 46 51 53 57 66 72 76 77 80
01/15/1998,02 03 04 05 08 11 19 22 30 31 33 41 50 53 54 62 66 69 75 76
01/14/1998,03 09 11 19 23 25 28 30 35 45 46 47 57 62 63 68 72 73 76 80
01/13/1998,05 16 18 19 20 26 28 34 36 41 48 51 54 59 60 61 63 77 78 80
01/12/1998,02 03 08 09 12 21 23 28 33 34 40 41 47 48 61 63 66 68 70 72
01/11/1998,03 12 16 17 19 22 23 30 31 37 42 45 47 51 52 61 66 72 75 80
01/10/1998,04 07 12 22 23 27 32 37 39 40 44 46 53 61 62 63 69 71 76 78
01/09/1998,04 05 08 11 12 20 25 27 28 30 35 38 51 55 56 62 68 70 76 78
01/08/1998,06 07 11 12 15 22 24 26 29 32 36 37 50 51 54 62 68 71 78 79
01/07/1998,03 06 10 11 12 14 18 22 35 40 46 50 56 59 60 61 64 71 78 79
01/06/1998,01 17 19 26 36 38 39 43 44 50 52 56 60 62 64 66 67 70 72 76
01/05/1998,01 09 10 15 16 18 28 33 35 38 39 49 51 52 56 58 61 70 78 79
01/04/1998,04 06 07 19 21 26 29 31 37 42 47 52 54 56 61 65 69 70 72 73
01/03/1998,01 10 12 13 15 19 21 23 26 28 29 32 38 39 45 51 52 53 61 71
01/02/1998,02 08 13 15 16 18 24 32 35 41 42 44 45 50 51 58 62 66 69 75
01/01/1998,15 22 27 34 37 38 39 43 46 47 52 56 59 62 63 65 70 74 78 79
12/31/1997,01 06 09 14 21 25 30 34 37 44 47 52 54 60 63 64 65 67 70 79
12/30/1997,05 15 16 26 27 32 34 43 45 53 60 62 69 71 72 73 76 77 79 80
12/29/1997,01 04 06 13 14 16 17 20 22 23 29 32 33 41 53 55 58 70 71 75
12/28/1997,04 06 10 12 13 21 34 45 51 55 56 57 58 63 65 69 72 73 77 78
12/27/1997,01 06 08 12 21 32 35 49 50 56 59 60 62 64 65 67 70 71 72 78
12/26/1997,11 23 34 35 36 37 40 41 43 46 51 55 61 62 65 66 72 73 75 77
12/24/1997,07 09 11 12 23 24 31 34 36 38 40 47 53 60 63 66 69 70 76 80
```

```
93      12/23/1997,01 08 11 13 25 26 31 35 36 49 50 52 53 57 59 60 61 65 77 78
94      12/22/1997,01 02 04 10 14 19 21 23 34 35 37 42 44 50 57 62 73 75 77 80
95      12/21/1997,02 04 13 22 28 30 32 33 36 41 53 54 55 58 61 71 72 75 77 78
96      12/20/1997,02 06 07 12 14 22 30 36 37 39 40 42 57 58 61 64 66 69 72 79
97      12/19/1997,05 20 21 23 26 29 33 34 36 44 47 50 57 58 64 66 70 72 76 78
98      12/18/1997,08 10 15 20 22 30 39 42 43 46 48 54 57 58 59 60 65 70 74 80
99      12/17/1997,02 05 07 09 13 19 21 23 25 31 35 36 37 45 50 61 62 73 74 79
00      12/16/1997,02 03 07 10 13 17 20 22 25 27 31 33 43 45 57 69 73 74 79 80
01      12/15/1997,02 04 05 06 18 22 25 27 34 35 53 57 60 64 66 67 71 72 73 75
02      12/14/1997,02 07 08 11 15 22 29 30 34 38 39 41 44 49 61 63 64 71 73 78
03      12/13/1997,05 11 14 15 16 17 19 22 25 27 37 44 45 47 54 57 60 62 74 77
04      12/12/1997,01 04 09 18 23 25 26 27 39 43 44 46 49 54 56 65 67 71 72 76
05      12/11/1997,03 05 06 17 22 25 26 29 30 33 35 42 44 45 49 52 56 60 76 79
06      12/10/1997,03 06 07 09 15 17 18 23 33 39 41 56 58 67 69 70 72 73 74 77
07      12/09/1997,03 05 07 09 10 12 14 19 23 26 28 30 32 39 43 47 66 67 68 72
08      12/08/1997,01 02 06 12 17 19 26 34 35 39 45 51 61 62 64 65 66 67 70 71
09      12/07/1997,01 02 08 12 13 15 19 20 34 36 45 54 56 64 66 73 75 76 78 80
10      12/06/1997,06 09 17 18 21 24 30 33 41 46 47 50 52 53 54 55 60 63 68 77
11      12/05/1997,12 18 22 23 26 28 29 30 36 37 39 46 47 49 56 58 60 71 74 77
12      12/04/1997,02 06 08 13 14 17 18 19 22 25 38 47 50 52 58 60 66 68 70 80
13      12/03/1997,05 12 13 16 20 23 28 30 32 38 39 40 46 47 53 55 57 61 73 80
14      12/02/1997,10 12 25 26 29 30 35 36 38 40 45 49 52 57 64 65 66 71 76 78
15      12/01/1997,04 07 17 19 24 25 27 29 31 37 39 45 47 51 53 60 61 63 70 72
16      11/30/1997,02 06 08 11 19 20 32 36 38 40 44 45 48 50 51 59 60 71 72 77
17      11/29/1997,11 15 16 22 23 26 29 32 35 37 39 42 50 53 60 63 67 70 74 80
18      11/28/1997,01 06 21 26 33 35 40 42 44 47 48 50 56 64 66 68 71 73 79 80
19      11/27/1997,01 14 19 20 22 24 30 39 52 53 57 60 63 68 71 73 74 77 78 80
20      11/26/1997,05 06 19 20 23 32 37 40 41 49 51 52 58 59 60 63 64 66 68 79
21      11/25/1997,01 04 09 13 17 18 19 21 26 31 41 45 46 47 48 51 53 59 60 79
22      11/24/1997,07 09 10 19 23 27 28 34 42 44 46 48 49 53 63 64 69 73 77 80
23      11/23/1997,01 04 07 08 10 17 18 21 30 44 47 51 58 59 65 72 73 74 75 80
24      11/22/1997,02 07 08 20 27 32 36 37 40 44 49 52 53 54 58 62 63 66 69 71
25      11/21/1997,04 05 06 22 23 26 29 32 41 48 50 62 66 69 70 72 76 77 79 80
26      11/20/1997,03 07 08 09 22 23 36 42 43 47 49 53 54 59 60 63 64 70 72 74
27      11/19/1997,04 09 10 16 18 20 24 33 43 47 48 54 58 60 62 64 66 68 70 78
28      11/18/1997,08 10 11 12 15 19 20 23 26 31 35 36 40 45 61 62 63 75 76 79
29      11/17/1997,08 09 14 19 21 27 28 29 31 33 35 38 45 55 57 63 73 74 78 80
30      11/16/1997,02 03 04 05 08 14 17 19 26 28 36 40 52 54 56 59 60 62 63 66
31      11/15/1997,04 09 10 16 20 28 30 33 34 35 38 45 53 54 56 58 62 65 69 72
32      11/14/1997,04 05 06 09 15 19 20 21 26 32 37 38 46 48 49 52 54 65 68 74
33      11/13/1997,03 08 12 13 14 21 24 26 27 28 29 34 39 43 53 54 57 64 72 77
34      11/12/1997,05 11 12 14 23 25 32 35 36 37 38 48 54 61 64 67 68 69 71 75
35      11/11/1997,02 11 15 17 21 23 30 31 34 37 40 41 51 54 59 60 62 69 72 75
36      11/10/1997,01 03 07 19 23 24 26 28 29 31 35 39 46 56 58 62 72 74 76 78
37      11/09/1997,07 09 10 13 19 20 24 38 39 45 52 53 55 57 58 59 62 64 66 68
38      11/08/1997,09 11 17 24 26 29 30 31 33 44 45 49 51 53 64 66 70 71 72 77
39      11/07/1997,01 02 07 11 12 15 16 24 26 35 39 42 44 51 59 60 61 68 75 79
40      11/06/1997,02 10 14 15 18 19 20 28 33 35 38 39 44 45 53 55 59 66 74 79
41      11/05/1997,01 02 03 05 10 17 24 29 31 36 39 42 54 58 64 65 69 71 76 79
42      11/04/1997,02 05 06 09 10 14 16 22 32 33 36 37 43 48 50 51 56 59 69 71
43      11/03/1997,02 15 20 23 25 26 29 37 38 39 40 43 45 49 52 55 58 60 71 76
44      11/02/1997,02 08 10 13 16 18 21 22 26 27 36 39 43 45 61 62 67 69 70 77
45      11/01/1997,03 05 06 08 17 19 24 27 35 37 39 43 44 45 57 58 63 72 74 77
46      10/31/1997,01 04 05 06 10 18 20 26 27 29 36 39 40 43 48 53 56 59 62 73
```

```
47      10/30/1997,02 04 09 19 21 29 33 36 42 43 46 49 56 58 65 67 76 77 78 79
48      10/29/1997,05 10 18 27 29 32 33 35 53 61 62 63 64 65 66 67 68 71 74 78
49      10/28/1997,01 04 09 10 14 19 20 29 33 34 35 37 44 46 49 50 66 67 72 76
50      10/27/1997,02 04 05 16 17 19 23 32 33 49 50 51 58 66 67 69 70 74 79 80
51      10/26/1997,01 05 12 14 16 17 24 25 33 35 38 43 44 46 52 56 63 66 67 69
52      10/25/1997,01 05 07 10 13 21 26 35 36 43 56 60 61 64 66 68 69 75 76 80
53      10/24/1997,01 05 07 10 12 13 15 20 31 32 33 51 56 60 61 65 68 71 72 80
54      10/23/1997,01 10 11 17 24 26 29 30 33 34 36 46 54 57 58 62 67 71 74 78
55      10/22/1997,01 05 06 07 10 18 19 26 28 31 33 35 42 46 53 54 56 58 74 77
56      10/21/1997,01 02 03 04 17 21 25 28 31 35 44 46 47 49 54 66 69 76 78 79
57      10/20/1997,05 07 08 22 23 24 35 36 40 47 50 57 58 61 62 63 66 74 78 80
58      10/19/1997,06 12 13 16 27 28 34 38 39 43 51 58 61 62 63 69 74 75 78 80
59      10/18/1997,01 15 21 27 29 33 38 43 47 56 59 60 61 63 66 70 71 74 76 80
60      10/17/1997,02 12 16 20 22 25 26 27 28 40 41 48 49 56 61 67 68 69 70 79
61      10/16/1997,02 19 26 27 34 35 39 41 42 51 52 53 59 62 67 72 73 77 79 80
62      10/15/1997,01 03 10 12 16 17 24 28 34 43 49 52 57 58 65 67 69 74 75 78
63      10/14/1997,12 13 15 25 34 35 37 39 40 43 44 54 55 59 62 63 68 69 72 74
64      10/13/1997,01 07 08 13 17 20 25 26 35 40 41 42 43 47 64 67 68 69 70 75
65      10/12/1997,01 02 11 18 20 21 29 43 47 49 51 56 58 60 64 68 76 78 79 80
66      10/11/1997,01 07 08 14 20 21 32 42 44 52 55 56 57 61 70 71 72 74 77 78
67      10/10/1997,01 05 06 07 11 12 35 36 42 44 46 47 48 49 52 56 60 69 71 76
68      10/09/1997,02 03 04 07 20 24 26 29 32 36 39 41 42 44 54 59 73 76 77 79
69      10/08/1997,01 03 05 08 16 23 25 35 36 40 44 46 47 48 49 55 57 59 66 67
70      10/07/1997,01 02 03 12 14 15 28 32 43 44 46 47 53 54 65 68 76 77 78 79
71      10/06/1997,02 05 07 11 12 15 18 37 41 44 49 50 51 65 67 69 71 77 78 79
72      10/05/1997,05 09 17 19 31 32 37 38 42 43 44 46 49 52 53 68 77 78 79 80
73      10/04/1997,01 08 09 11 12 17 26 27 33 39 41 47 48 49 52 54 57 61 63 75
74      10/03/1997,06 07 09 10 20 22 24 25 27 29 40 45 46 49 58 61 65 66 67 73
75      10/02/1997,01 03 11 12 14 15 16 24 28 31 34 37 38 39 40 50 51 57 69 75
76      10/01/1997,05 09 13 18 20 25 33 35 36 39 47 49 53 57 59 64 65 67 68 72
77      09/30/1997,01 03 11 12 15 16 21 24 28 30 31 47 50 51 52 55 61 62 70 76
78      09/29/1997,01 09 10 14 15 19 20 23 29 33 36 42 49 50 51 56 62 74 78 80
79      09/28/1997,01 02 07 11 12 13 17 20 22 23 24 27 39 49 55 61 66 71 74 79
80      09/27/1997,01 02 08 13 16 17 24 33 34 35 40 50 60 62 65 66 68 69 77 80
81      09/26/1997,02 03 05 07 15 17 18 19 20 23 24 25 27 32 37 39 41 44 55 64
82      09/25/1997,01 03 05 11 14 21 22 25 34 36 37 45 47 49 50 57 65 67 73 76
83      09/24/1997,03 09 12 15 22 23 24 26 32 36 40 41 43 45 48 52 63 76 77 80
84      09/23/1997,01 04 14 21 22 23 24 35 41 43 47 48 52 64 66 73 74 75 76 80
85      09/22/1997,10 22 26 28 34 35 38 40 43 45 47 50 52 54 56 59 63 67 74 80
86      09/21/1997,03 13 26 35 38 40 43 47 49 51 53 57 58 65 66 67 69 72 76 78
87      09/20/1997,06 07 11 13 21 26 31 32 33 35 43 48 49 51 57 61 65 73 74 77
88      09/19/1997,04 08 17 24 25 27 30 31 33 38 39 40 41 44 49 55 59 60 61 71
89      09/18/1997,03 05 17 21 22 28 29 34 35 37 40 46 48 50 56 62 73 75 76 78
90      09/17/1997,06 09 14 17 23 25 27 32 38 41 42 50 52 58 60 62 64 67 73 78
91      09/16/1997,01 04 06 12 17 20 22 31 32 35 36 39 43 48 50 52 59 65 67 80
92      09/15/1997,03 05 06 22 24 25 27 32 33 35 41 42 43 44 47 50 60 71 75 80
93      09/14/1997,10 15 16 20 23 27 29 36 39 46 48 50 51 55 56 63 65 71 78 80
94      09/13/1997,01 02 03 06 19 22 27 28 29 30 35 37 39 45 50 60 65 72 76 78
95      09/12/1997,04 05 14 16 19 21 26 30 31 32 39 40 43 52 55 56 76 77 78 80
96      09/11/1997,01 08 17 21 28 31 32 43 47 51 54 55 60 65 66 67 68 69 72 73
97      09/10/1997,04 05 06 08 11 15 25 31 32 33 35 36 39 45 56 62 66 73 74 76
98      09/09/1997,01 10 12 19 22 24 33 35 37 40 42 49 50 56 60 61 65 71 73 80
99      09/08/1997,02 06 10 11 12 16 21 25 26 27 33 42 43 48 49 55 65 69 72 80
00      09/07/1997,02 03 08 15 19 20 25 38 42 43 45 47 48 49 56 62 65 69 70 75
```

01 09/06/1997,08 12 13 21 23 27 38 39 43 44 49 59 60 62 63 64 66 68 69 79
02 09/05/1997,01 09 12 15 31 34 41 42 43 46 47 50 60 61 64 69 72 78 79 80
03 09/04/1997,01 04 07 10 16 19 31 33 36 37 39 46 48 49 63 69 71 72 73 79
04 09/03/1997,03 11 19 27 30 34 35 38 40 44 45 46 51 54 57 63 72 75 76 78
05 09/02/1997,02 08 09 15 19 24 25 32 34 48 52 55 58 69 70 71 73 74 76 77
06 09/01/1997,04 16 19 21 22 25 29 36 41 43 46 50 54 56 58 63 65 67 70 80
07 08/31/1997,02 07 10 11 18 28 34 35 38 41 44 47 61 64 66 67 69 75 78 80
08 08/30/1997,04 09 17 23 26 31 32 37 40 44 45 46 47 48 49 55 60 69 72 78
09 08/29/1997,07 08 10 14 16 17 18 20 23 31 34 36 46 47 48 52 64 67 68 78
10 08/28/1997,03 10 12 13 23 33 34 35 42 46 49 56 57 58 63 69 75 77 79 80
11 08/27/1997,05 07 13 15 27 29 40 44 46 50 53 55 56 61 62 66 70 74 75 79
12 08/26/1997,02 03 09 16 18 19 24 32 35 38 39 42 47 54 56 57 65 68 78 80
13 08/25/1997,04 11 12 18 22 23 24 34 35 41 46 53 56 59 65 68 69 70 76 79
14 08/24/1997,01 06 23 24 30 33 38 42 46 49 50 51 53 56 58 59 62 71 75 80
15 08/23/1997,01 03 06 15 17 18 20 27 30 36 40 49 50 51 53 54 56 58 72 73
16 08/22/1997,07 08 17 18 30 31 35 38 44 47 53 57 62 68 69 70 72 75 77 78
17 08/21/1997,06 07 16 19 24 26 29 33 39 43 44 45 54 60 62 64 67 70 71 80
18 08/20/1997,01 06 16 29 30 31 36 40 41 43 48 54 56 59 61 65 72 73 76 78
19 08/19/1997,04 08 10 13 17 19 21 27 37 39 45 47 51 55 63 64 65 67 74 76
20 08/18/1997,03 08 11 15 22 25 26 28 33 44 48 53 60 61 62 65 67 71 74 78
21 08/17/1997,04 11 21 23 24 28 34 35 39 40 42 50 51 52 53 54 58 61 66 73
22 08/16/1997,04 10 13 15 17 22 26 31 46 52 55 56 61 62 63 64 70 73 75 79
23 08/15/1997,01 05 09 11 13 15 17 21 26 29 30 33 36 44 50 51 53 65 69 71
24 08/14/1997,03 04 14 21 26 27 32 33 37 45 51 53 56 58 59 60 63 70 76 77
25 08/13/1997,04 06 08 15 24 25 31 35 36 37 41 42 44 45 47 64 69 70 72 76
26 08/12/1997,01 02 05 06 07 09 15 18 20 27 31 36 37 41 47 56 57 61 70 75
27 08/11/1997,02 04 05 07 10 14 16 19 24 25 32 34 42 58 65 67 68 69 76 80
28 08/10/1997,01 02 03 04 06 08 10 22 24 28 47 51 62 64 67 69 70 72 78 79
29 08/09/1997,03 05 16 18 23 26 30 31 32 39 41 43 45 46 47 51 59 65 69 79
30 08/08/1997,03 07 10 14 16 17 28 34 37 46 51 54 56 57 61 65 69 74 77 79
31 08/07/1997,02 04 13 23 25 27 30 31 37 40 42 44 46 50 54 58 59 63 67 79
32 08/06/1997,04 08 09 11 13 14 16 19 25 32 39 40 43 46 48 51 54 58 59 71
33 08/05/1997,01 10 11 15 20 24 28 31 33 40 42 46 49 51 53 54 70 71 75 77
34 08/04/1997,02 03 06 15 29 30 36 39 41 46 47 48 50 51 52 58 61 67 73 80
35 08/03/1997,06 10 15 22 23 25 29 31 32 36 47 51 55 57 59 60 63 64 74 79
36 08/02/1997,02 05 14 18 22 23 26 28 29 37 46 50 53 60 61 64 65 71 72 73
37 08/01/1997,02 05 08 09 10 14 22 23 31 32 34 38 39 41 42 47 52 54 59 78
38 07/31/1997,10 13 19 22 27 31 32 36 37 39 43 45 56 59 60 63 73 75 79 80
39 07/30/1997,06 16 17 18 19 20 22 24 25 27 32 38 50 51 53 56 67 68 74 75
40 07/29/1997,01 05 11 14 17 23 31 32 34 35 39 43 51 52 53 58 72 74 76 79
41 07/28/1997,02 04 06 13 16 19 21 23 33 39 42 43 49 54 55 57 63 75 79 80
42 07/27/1997,04 05 08 10 18 20 21 25 27 29 41 47 51 54 55 57 58 65 75 77
43 07/26/1997,03 07 12 17 18 26 32 33 37 38 44 45 49 53 54 57 58 61 76 80
44 07/25/1997,01 03 11 19 20 24 44 45 47 49 52 55 57 60 62 69 71 78 79 80
45 07/24/1997,09 12 16 18 20 35 42 44 45 49 52 55 58 62 67 72 73 77 78 79
46 07/23/1997,09 10 14 18 21 24 25 31 32 39 40 41 44 45 48 50 62 69 71 79
47 07/22/1997,07 15 19 21 24 32 33 36 38 43 48 51 55 65 68 73 74 76 77 79
48 07/21/1997,01 02 04 05 09 11 16 22 26 27 29 36 51 61 67 69 70 72 73 76
49 07/20/1997,02 03 10 12 17 21 22 24 36 51 53 54 55 59 68 69 70 72 73 80
50 07/19/1997,02 04 07 10 17 21 27 30 32 33 34 39 40 42 49 56 66 69 72 73
51 07/18/1997,05 06 14 24 28 36 43 50 56 57 58 61 62 65 69 71 72 73 76 78
52 07/17/1997,08 09 11 14 23 28 34 38 43 45 47 52 58 59 62 63 66 76 78 80
53 07/16/1997,01 02 08 14 26 28 38 40 41 42 49 55 62 63 64 65 67 72 74 76
54 07/15/1997,05 07 10 24 31 37 40 52 53 60 61 63 64 65 66 71 74 77 78 79

```
55      07/14/1997,02 08 09 13 14 15 18 22 24 32 37 41 47 50 54 56 72 76 77 80
56      07/13/1997,10 13 15 17 33 37 42 44 47 53 54 55 58 59 62 69 72 77 79 80
57      07/12/1997,03 04 11 13 14 15 19 24 34 43 44 47 49 51 56 60 66 67 70 73
58      07/11/1997,06 08 19 26 29 30 32 35 40 41 53 54 56 60 62 66 71 72 73 76
59      07/10/1997,02 06 07 10 12 13 14 19 21 27 33 41 44 45 46 49 52 67 73 75
60      07/09/1997,03 11 13 15 19 22 30 35 36 38 39 43 47 52 56 57 58 60 61 78
61      07/08/1997,01 02 05 08 10 13 16 17 19 20 24 42 43 53 55 57 67 69 73 80
62      07/07/1997,02 08 09 10 12 14 15 18 31 38 41 45 46 57 59 66 67 69 76 80
63      07/06/1997,02 09 13 15 21 24 25 26 27 29 31 34 35 37 46 52 54 72 74 78
64      07/05/1997,02 17 22 23 25 28 29 41 50 51 52 53 56 60 61 62 63 68 74 78
65      07/04/1997,01 09 17 21 22 36 39 40 46 50 59 61 62 65 67 68 69 71 73 77
66      07/03/1997,07 08 15 21 29 32 41 43 44 48 49 60 61 64 66 67 72 73 74 76
67      07/02/1997,01 02 15 17 26 27 29 31 33 36 45 46 49 51 57 59 60 66 74 79
68      07/01/1997,03 08 10 13 19 20 21 40 42 47 48 50 51 55 58 59 64 72 74 79
69      06/30/1997,01 03 07 20 27 28 32 33 38 43 48 50 51 56 58 62 67 70 72 73
70      06/29/1997,04 06 13 14 20 22 24 25 31 34 39 43 48 49 50 57 61 63 64 66
71      06/28/1997,03 15 18 19 23 28 31 36 41 47 51 52 53 60 62 64 68 70 75 76
72      06/27/1997,12 16 21 22 28 29 30 36 49 52 53 55 58 64 67 70 72 75 76 77
73      06/26/1997,01 03 10 21 22 28 29 30 32 33 40 44 58 62 65 69 73 76 77 78
74      06/25/1997,04 10 11 14 15 16 18 31 33 34 38 45 46 50 61 62 68 70 76 79
75      06/24/1997,01 07 09 12 15 22 27 30 35 40 41 42 44 48 54 60 61 68 70 78
76      06/23/1997,01 05 11 21 22 23 30 33 34 36 37 38 46 49 51 60 66 69 74 77
77      06/22/1997,01 11 14 17 19 21 26 28 33 39 40 41 43 46 47 49 57 61 73 80
78      06/21/1997,02 05 06 09 12 14 15 16 21 39 43 46 48 52 56 57 59 61 68 80
79      06/20/1997,03 08 09 11 18 21 22 26 27 31 37 39 41 52 56 58 66 71 77 78
80      06/19/1997,04 07 09 16 25 26 27 31 39 44 45 47 48 58 62 69 72 73 74 76
81      06/18/1997,04 06 07 10 12 20 21 24 38 44 47 49 50 55 57 60 62 65 67 68
82      06/17/1997,06 09 10 17 18 19 23 28 29 34 35 39 55 58 70 74 77 78 79 80
83      06/16/1997,05 07 18 19 22 24 26 27 28 29 44 45 48 53 63 67 70 71 72 74
84      06/15/1997,04 08 09 13 21 22 24 33 39 40 51 53 55 56 61 66 71 73 75 78
85      06/14/1997,07 09 12 16 28 30 31 45 49 52 55 59 62 65 68 71 74 76 77 80
86      06/13/1997,02 06 11 12 18 31 37 38 39 40 43 44 47 54 58 61 63 65 70 76
87      06/12/1997,02 08 09 12 19 23 26 27 28 39 42 43 48 49 51 57 66 71 72 78
88      06/11/1997,08 12 13 14 22 27 37 39 41 49 57 59 62 63 66 68 74 76 79 80
89      06/10/1997,16 19 34 38 39 43 45 49 53 56 57 60 63 66 68 69 70 71 73 74
90      06/09/1997,09 10 14 16 21 24 29 30 36 37 43 51 52 53 55 56 67 72 76 79
91      06/08/1997,02 04 05 10 13 14 25 29 30 32 45 46 47 48 56 61 66 69 74 79
92      06/07/1997,02 04 08 15 18 20 27 28 32 33 36 38 41 49 51 66 67 71 73 80
93      06/06/1997,11 17 20 21 24 31 32 34 36 38 46 49 54 55 57 58 61 63 64 78
94      06/05/1997,07 09 13 17 23 24 28 29 30 33 37 44 53 55 56 62 64 71 73 80
95      06/04/1997,04 05 06 14 15 18 23 30 32 35 38 39 40 58 59 65 70 72 75 76
96      06/03/1997,01 02 14 25 28 29 30 32 34 38 39 46 48 53 55 57 58 60 69 76
97      06/02/1997,03 07 08 10 14 16 17 20 21 26 33 34 37 40 44 52 53 58 62 70
98      06/01/1997,01 08 09 11 16 19 24 31 32 35 39 45 48 52 57 65 67 68 73 76
99      05/31/1997,04 08 12 18 19 20 25 27 33 36 39 43 47 50 51 66 70 72 73 75
00      05/30/1997,08 16 17 20 21 25 31 32 35 37 39 47 48 54 60 61 62 63 64 72
01      05/29/1997,03 15 21 25 26 33 40 45 50 51 53 60 65 66 67 70 74 76 77 78
02      05/28/1997,01 08 14 15 16 21 35 37 39 45 51 54 61 64 65 71 72 73 74 78
03      05/27/1997,05 15 16 17 18 19 28 30 40 41 50 54 55 58 64 65 66 68 75 79
04      05/26/1997,10 11 12 15 24 25 28 33 36 40 41 47 50 51 53 54 59 61 68 71
05      05/25/1997,02 21 23 24 37 38 39 44 49 50 51 53 55 58 61 64 71 73 78 79
06      05/24/1997,05 08 19 20 26 31 33 36 40 41 44 45 48 51 57 61 68 74 77 78
07      05/23/1997,04 07 16 18 26 28 34 41 45 51 54 58 62 68 72 73 74 76 77 78
08      05/22/1997,05 06 12 14 21 27 31 32 34 37 42 44 53 64 65 66 67 68 69 77
```

09 05/21/1997,03 06 08 09 12 15 17 19 20 23 24 27 31 32 34 35 38 41 75 77
10 05/20/1997,03 08 09 15 19 21 22 27 30 39 43 44 54 59 61 62 68 76 77 80
11 05/19/1997,01 08 09 10 12 13 16 20 21 23 24 28 32 37 38 48 63 69 71 75
12 05/18/1997,04 06 08 12 15 18 19 23 27 34 35 36 37 41 47 49 53 63 71 72
13 05/17/1997,02 08 18 21 27 32 36 37 38 48 51 56 64 68 70 71 72 76 77 80
14 05/16/1997,01 04 05 07 11 16 21 23 33 36 42 43 45 46 50 55 60 66 76 78
15 05/15/1997,02 03 11 12 19 31 35 40 43 51 52 54 57 60 64 70 73 76 77 79
16 05/14/1997,04 06 08 09 14 17 21 24 26 27 31 36 44 47 48 50 60 65 68 74
17 05/13/1997,01 03 04 12 15 17 19 20 27 34 39 40 50 53 54 62 65 66 68 73
18 05/12/1997,04 19 21 28 31 36 37 39 41 45 52 53 55 57 61 63 65 68 74 78
19 05/11/1997,03 12 17 20 22 23 25 26 28 29 38 47 52 53 61 62 65 67 76 80
20 05/10/1997,03 15 17 20 21 26 31 33 35 44 55 57 64 65 67 72 73 76 77 80
21 05/09/1997,01 08 16 19 21 30 31 35 36 37 38 40 41 45 54 62 67 68 69 78
22 05/08/1997,01 02 03 10 11 12 16 17 19 24 28 35 37 39 52 54 57 73 76 77
23 05/07/1997,04 05 08 10 12 19 28 39 40 41 49 50 54 57 59 60 63 64 70 79
24 05/06/1997,01 03 06 09 12 14 17 22 27 33 46 48 50 54 59 70 72 75 78 79
25 05/05/1997,02 08 20 24 26 34 35 36 39 40 46 47 49 52 54 60 61 62 68 71
26 05/04/1997,01 02 04 07 11 17 23 27 33 34 35 37 51 55 60 66 67 78 79 80
27 05/03/1997,11 12 19 24 25 26 30 33 34 39 40 47 48 50 58 59 61 69 71 78
28 05/02/1997,05 06 09 13 15 16 19 20 22 27 31 32 33 37 39 51 53 62 76 78
29 05/01/1997,03 05 09 10 12 15 20 25 31 32 39 40 44 48 55 56 68 69 71 74
30 04/30/1997,01 03 07 10 11 13 15 19 22 25 36 38 46 50 54 60 65 69 73 75
31 04/29/1997,01 09 10 25 30 32 35 42 48 49 51 52 54 58 60 69 71 73 75 77
32 04/28/1997,02 06 17 22 29 31 32 40 45 48 50 57 61 62 63 68 71 73 76 80
33 04/27/1997,08 09 14 15 17 19 28 29 30 32 43 44 46 49 61 64 68 69 75 80
34 04/26/1997,02 05 09 10 12 13 18 22 31 33 36 39 46 49 52 54 62 63 67 73
35 04/25/1997,02 04 09 13 19 30 37 42 44 46 48 49 52 53 59 62 68 74 76 79
36 04/24/1997,03 08 12 15 16 22 27 33 34 35 40 45 47 50 54 55 62 69 73 79
37 04/23/1997,02 04 07 10 16 23 25 28 31 33 36 37 52 55 61 66 72 74 77 80
38 04/22/1997,05 06 07 08 13 16 23 32 40 46 51 54 57 60 61 65 69 74 75 77
39 04/21/1997,03 07 14 18 23 25 28 30 31 32 36 38 51 54 59 61 63 64 69 70
40 04/20/1997,03 08 13 16 18 19 24 26 41 42 52 55 63 64 65 67 71 75 76 78
41 04/19/1997,04 06 09 12 16 17 24 26 28 32 34 35 37 40 42 49 56 59 64 78
42 04/18/1997,05 06 10 17 18 19 20 23 33 39 41 46 51 57 60 65 66 67 72 73
43 04/17/1997,04 16 17 18 19 21 27 28 33 34 46 50 61 65 67 69 71 78 79 80
44 04/16/1997,02 03 08 10 11 15 19 23 25 26 29 31 42 44 46 59 62 71 72 76
45 04/15/1997,01 03 04 05 17 19 25 26 33 35 37 38 42 43 45 48 52 56 75 80
46 04/14/1997,02 06 07 09 13 15 19 28 30 35 37 40 42 50 51 52 63 65 66 72
47 04/13/1997,02 08 17 19 28 29 30 31 35 37 44 46 47 52 64 66 69 70 75 77
48 04/12/1997,01 03 06 10 12 15 17 24 25 27 35 37 39 44 47 53 58 61 63 79
49 04/11/1997,07 12 13 16 23 28 34 36 37 41 43 44 49 53 55 57 65 67 70 80
50 04/10/1997,05 10 12 16 26 27 29 32 40 43 50 51 52 66 72 73 74 75 77 79
51 04/09/1997,05 09 14 17 20 21 26 28 31 33 38 43 47 48 49 60 64 78 79 80
52 04/08/1997,03 07 15 20 21 23 30 33 35 36 37 38 45 50 61 67 70 74 75 76
53 04/07/1997,02 04 10 16 19 20 26 27 32 33 43 49 53 56 58 63 64 70 76 78
54 04/06/1997,01 08 11 12 15 20 25 26 28 38 44 49 51 58 61 64 66 67 72 77
55 04/05/1997,02 04 11 13 16 18 23 30 31 34 42 45 49 51 53 70 72 73 75 77
56 04/04/1997,08 10 15 16 17 18 24 25 26 27 34 56 60 64 70 71 75 77 78 80
57 04/03/1997,03 18 19 20 26 27 30 31 32 35 36 38 40 45 54 66 67 75 79 80
58 04/02/1997,06 12 14 19 21 29 30 32 35 45 46 48 51 53 64 69 70 72 76 78
59 04/01/1997,03 11 12 13 26 29 35 37 40 41 46 55 57 59 65 66 67 73 74 76
60 03/31/1997,01 09 14 19 22 23 31 33 35 37 40 42 52 55 59 67 69 72 74 78
61 03/30/1997,09 13 16 17 23 32 33 37 38 40 41 42 50 57 62 71 72 73 76 78
62 03/29/1997,03 04 17 18 19 26 30 32 35 37 59 60 61 62 65 66 69 72 77 79

```
63      03/28/1997,04 06 08 09 21 23 27 33 35 36 41 45 47 52 53 64 70 71 76 80
64      03/27/1997,02 07 09 12 15 17 23 25 29 33 34 42 45 47 48 49 55 60 72 80
65      03/26/1997,01 12 14 18 21 27 35 40 42 45 53 59 60 62 63 66 67 72 74 79
66      03/25/1997,06 09 12 14 19 22 30 31 42 44 46 58 59 61 64 68 69 72 75 76
67      03/24/1997,11 13 19 26 28 36 42 46 53 56 57 58 59 62 64 66 67 70 77 78
68      03/23/1997,01 02 03 05 09 15 17 18 19 22 29 30 34 43 48 55 59 61 64 79
69      03/22/1997,03 05 07 10 11 12 14 16 18 26 34 37 40 44 50 57 69 70 75 76
70      03/21/1997,03 04 18 19 28 31 36 46 52 53 55 56 64 65 67 68 75 78 79 80
71      03/20/1997,03 13 16 18 21 22 30 31 43 48 49 50 51 58 61 63 66 69 70 73
72      03/19/1997,01 08 15 16 18 19 24 30 32 35 48 50 53 57 62 68 71 72 74 75
73      03/18/1997,03 04 07 08 14 18 19 20 25 30 44 46 47 54 58 59 60 66 70 80
74      03/17/1997,03 07 18 23 27 28 38 39 40 44 48 53 54 57 60 63 69 71 77 78
75      03/16/1997,05 10 12 13 19 20 21 25 27 32 33 36 38 39 48 53 55 60 65 72
76      03/15/1997,03 04 05 13 14 15 26 33 34 36 44 46 49 50 52 57 61 66 68 72
77      03/14/1997,12 14 17 21 25 27 32 39 41 42 46 47 51 58 62 65 69 74 77 79
78      03/13/1997,03 06 15 17 22 23 24 32 35 39 40 41 44 48 58 59 63 70 75 76
79      03/12/1997,01 07 10 11 12 24 33 35 37 44 47 48 49 50 53 54 64 66 75 76
80      03/11/1997,02 03 04 06 13 15 17 26 30 41 46 48 51 52 55 59 67 71 73 78
81      03/10/1997,03 11 13 15 16 19 20 24 35 37 42 45 48 52 55 63 64 77 78 79
82      03/09/1997,08 10 15 16 18 23 24 36 38 41 45 46 47 49 66 70 73 76 77 80
83      03/08/1997,02 13 15 16 17 22 24 28 38 47 48 57 58 59 62 63 64 70 74 80
84      03/07/1997,10 20 27 29 31 34 36 40 49 50 53 55 57 59 61 65 68 69 72 75
85      03/06/1997,09 14 17 18 20 27 30 32 39 45 50 51 52 53 56 57 61 62 66 74
86      03/05/1997,01 08 19 20 26 27 29 31 35 38 41 44 45 49 50 51 55 62 68 76
87      03/04/1997,01 03 06 09 10 12 19 25 26 35 37 47 52 53 55 56 58 61 71 78
88      03/03/1997,03 09 12 27 29 32 39 41 43 44 53 55 57 61 63 66 70 72 75 78
89      03/02/1997,13 14 23 24 27 43 44 48 51 52 54 56 58 59 60 64 75 77 78 79
90      03/01/1997,11 14 16 17 26 27 29 33 34 37 40 46 47 50 58 60 61 67 68 79
91      02/28/1997,01 04 12 14 16 21 25 31 32 36 37 41 45 46 48 59 61 68 72 79
92      02/27/1997,05 08 09 11 17 26 31 35 39 40 41 44 47 54 59 61 62 74 75 78
93      02/26/1997,03 08 14 17 26 29 31 33 37 52 56 57 58 59 62 67 68 73 76 79
94      02/25/1997,11 18 19 20 21 22 25 28 29 30 36 47 49 51 55 59 62 66 68 73
95      02/24/1997,02 06 07 14 18 19 25 27 30 33 37 38 42 52 56 58 60 66 67 75
96      02/23/1997,03 07 14 20 24 25 27 30 34 36 41 50 52 56 60 63 69 71 72 79
97      02/22/1997,03 08 10 18 23 24 28 31 34 41 42 52 53 54 55 57 62 65 68 78
98      02/21/1997,04 06 07 12 17 19 21 23 25 34 35 41 43 45 56 57 59 61 70 76
99      02/20/1997,02 03 04 05 20 21 24 27 28 31 36 42 45 52 54 64 69 73 77 79
00      02/19/1997,01 04 10 14 15 16 18 21 23 24 28 33 40 45 62 63 64 67 71 77
01      02/18/1997,10 11 17 24 30 33 39 40 45 50 51 54 58 61 63 66 72 74 75 78
02      02/17/1997,02 06 10 13 16 22 23 24 27 31 37 39 41 46 48 52 61 66 68 80
03      02/16/1997,03 04 12 16 21 28 31 46 48 50 51 53 57 59 61 67 70 71 77 79
04      02/15/1997,01 08 09 14 16 21 26 35 38 39 41 43 46 53 54 59 61 62 68 74
05      02/14/1997,07 12 13 22 29 30 31 32 33 36 39 40 41 42 52 55 59 72 73 76
06      02/13/1997,04 10 11 15 20 26 35 40 42 50 52 55 60 64 66 72 73 76 77 78
07      02/12/1997,02 05 09 12 15 17 26 31 32 37 38 50 53 54 58 65 72 76 77 79
08      02/11/1997,01 03 06 09 10 11 17 21 23 24 31 32 35 41 42 44 53 57 59 62
09      02/10/1997,03 08 09 16 23 29 30 32 38 39 42 50 52 57 59 68 73 75 76 77
10      02/09/1997,07 08 12 16 21 24 26 29 31 39 40 45 48 52 59 62 65 71 75 79
11      02/08/1997,08 11 16 18 19 25 34 36 37 39 50 53 56 63 65 67 70 73 75 79
12      02/07/1997,02 12 13 18 21 25 27 31 32 34 36 42 49 50 55 57 66 67 75 76
13      02/06/1997,13 14 15 17 19 20 24 36 40 44 48 51 55 67 68 70 71 76 78 79
14      02/05/1997,01 05 06 10 14 15 18 26 27 28 29 30 32 49 52 55 63 69 76 78
15      02/04/1997,05 07 10 11 13 17 20 23 26 28 30 32 48 55 58 60 69 70 74 80
16      02/03/1997,02 07 09 11 12 18 20 25 28 30 35 46 47 49 50 53 55 64 67 71
```

```
17      02/02/1997,05 09 18 21 22 26 27 31 34 38 41 43 44 50 55 58 59 66 76 80
18      02/01/1997,01 04 05 10 12 13 16 20 28 34 39 41 43 44 45 47 68 71 75 79
19      01/31/1997,03 05 09 12 18 19 24 25 30 32 40 45 46 47 50 52 53 57 77 79
20      01/30/1997,06 07 09 13 16 18 19 20 25 29 31 34 36 37 39 62 65 66 73 74
21      01/29/1997,10 19 20 23 27 33 37 38 39 40 41 42 49 52 54 56 65 66 71 80
22      01/28/1997,03 05 06 10 21 27 29 30 32 37 39 49 50 53 54 56 59 64 65 77
23      01/27/1997,12 13 15 18 21 22 26 32 34 40 46 52 59 60 61 67 69 73 77 80
24      01/26/1997,04 05 07 17 22 24 30 37 39 45 55 60 62 64 65 73 74 76 79 80
25      01/25/1997,07 08 09 10 16 25 26 27 28 29 36 40 46 51 53 58 65 68 79 80
26      01/24/1997,04 06 08 10 14 16 22 24 30 32 38 41 45 49 50 53 57 69 76 79
27      01/23/1997,06 09 10 13 21 22 30 38 42 45 46 52 53 56 62 63 71 72 77 79
28      01/22/1997,01 07 09 13 18 21 27 29 33 35 36 37 50 51 53 56 57 66 70 72
29      01/21/1997,11 13 14 16 19 20 25 30 33 39 41 43 47 48 60 61 65 67 70 71
30      01/20/1997,01 07 08 12 16 18 23 24 27 29 30 31 44 46 49 50 54 62 63 72
31      01/19/1997,02 08 14 32 34 37 40 42 47 48 49 50 58 59 63 65 69 71 74 76
32      01/18/1997,03 04 25 29 40 43 47 48 53 56 58 60 61 64 65 71 73 74 75 78
33      01/17/1997,02 08 09 10 16 24 29 34 36 37 40 43 51 54 56 62 64 68 71 79
34      01/16/1997,03 05 12 20 24 31 34 38 41 43 51 53 58 60 68 69 70 71 76 78
35      01/15/1997,04 06 15 18 19 22 25 28 31 33 47 49 55 56 57 58 59 64 67 68
36      01/14/1997,01 09 13 15 20 26 29 31 33 41 44 51 55 56 57 63 64 70 72 76
37      01/13/1997,04 11 13 24 25 27 28 30 32 39 52 57 58 61 62 67 70 72 76 77
38      01/12/1997,01 08 16 17 20 28 33 36 40 44 51 54 57 58 62 66 68 69 71 78
39      01/11/1997,10 12 13 15 17 20 28 31 37 39 47 49 51 56 63 64 65 66 69 76
40      01/10/1997,03 04 05 15 23 26 27 32 48 53 54 58 59 60 62 69 73 77 79 80
41      01/09/1997,02 05 20 21 41 42 43 44 45 52 54 55 59 61 62 67 69 77 78 80
42      01/08/1997,03 14 15 17 18 24 29 30 34 35 37 40 44 47 48 55 66 68 75 77
43      01/07/1997,01 03 08 12 14 18 21 22 24 26 27 47 50 56 57 62 63 69 71 73
44      01/06/1997,02 09 12 13 16 21 22 26 28 29 30 31 37 39 43 45 50 62 63 72
45      01/05/1997,03 04 06 07 14 24 34 40 43 47 49 54 55 56 66 67 68 70 76 78
46      01/04/1997,02 03 04 10 14 15 20 24 40 46 49 50 51 57 60 61 62 69 73 79
47      01/03/1997,03 04 08 15 26 30 32 33 36 43 47 48 49 57 60 63 67 68 74 75
48      01/02/1997,01 09 13 16 23 30 37 45 47 48 59 61 63 70 71 73 76 77 78 80
49      01/01/1997,03 05 07 17 19 20 22 29 36 37 41 42 46 49 53 57 61 67 75 79
50      12/31/1996,08 10 13 14 15 16 31 33 35 41 43 47 52 54 61 62 63 64 67 74
51      12/30/1996,04 06 11 13 26 28 30 31 33 34 38 39 40 57 59 63 68 70 73 74
52      12/29/1996,01 07 09 12 17 18 20 23 24 30 31 36 59 60 64 66 68 72 74 79
53      12/28/1996,09 11 12 14 16 17 30 33 47 48 50 56 57 60 61 63 71 72 75 78
54      12/27/1996,04 06 08 11 13 16 21 25 27 29 30 31 35 37 43 45 50 57 63 76
55      12/26/1996,01 16 21 22 25 28 30 32 33 36 47 54 55 57 60 61 68 70 71 75
56      12/24/1996,01 09 12 13 14 17 22 33 38 39 41 43 45 46 57 58 61 62 72 78
57      12/23/1996,03 05 06 07 14 15 19 23 27 31 38 39 47 48 50 53 57 64 71 72
58      12/22/1996,04 07 14 15 17 18 24 27 33 35 37 41 45 48 63 69 72 73 74 75
59      12/21/1996,02 03 07 08 10 18 21 26 30 42 43 46 50 54 58 61 71 72 74 76
60      12/20/1996,01 02 07 13 21 22 26 27 38 39 42 45 48 49 58 61 64 68 72 74
61      12/19/1996,06 09 13 16 19 20 24 32 33 35 36 37 38 40 52 54 55 57 65 75
62      12/18/1996,05 09 10 16 20 23 25 30 32 41 44 45 52 54 57 63 64 73 74 78
63      12/17/1996,03 05 06 10 21 24 26 28 29 38 44 52 55 57 62 63 64 71 76 77
64      12/16/1996,05 08 16 19 24 32 33 39 46 49 52 59 63 64 65 66 69 70 71 77
65      12/15/1996,04 09 12 17 19 26 28 34 35 38 41 47 48 51 56 61 67 70 72 77
66      12/14/1996,02 08 09 10 15 22 26 28 32 33 39 47 51 52 54 57 59 72 74 76
67      12/13/1996,04 06 08 11 12 16 18 21 22 26 30 44 47 48 50 55 58 61 78 79
68      12/12/1996,05 06 07 09 12 14 15 17 22 29 48 51 52 54 56 63 66 67 68 69
69      12/11/1996,07 14 16 17 19 23 34 35 37 38 45 46 51 54 55 58 62 63 64 72
70      12/10/1996,01 04 08 17 21 22 28 30 33 39 43 44 49 61 64 67 71 74 77 78
```

```
71      12/09/1996,02 08 09 10 11 20 24 25 30 33 40 41 42 48 49 58 59 63 64 77
72      12/08/1996,01 12 24 27 34 35 39 50 51 54 55 56 59 61 66 67 69 73 76 79
73      12/07/1996,02 09 11 13 16 19 20 24 29 31 33 34 35 45 56 59 60 63 64 65
74      12/06/1996,01 08 15 21 22 23 30 32 46 48 50 53 55 58 59 60 62 67 73 75
75      12/05/1996,02 03 07 11 17 23 30 31 37 40 47 50 56 58 61 66 68 73 76 77
76      12/04/1996,03 07 12 20 23 25 26 29 30 31 32 33 34 49 52 68 69 74 75 80
77      12/03/1996,01 06 07 18 22 27 30 31 36 39 42 49 57 63 64 67 71 72 77 80
78      12/02/1996,01 04 14 17 26 28 33 35 41 47 48 51 52 56 57 59 61 76 78 79
79      12/01/1996,07 16 24 26 27 28 31 32 33 36 37 52 54 56 61 65 70 75 76 79
80      11/30/1996,02 09 10 11 15 16 17 18 19 22 38 43 45 48 55 63 64 74 76 80
81      11/29/1996,01 02 04 05 08 11 16 17 27 28 37 39 50 53 55 57 58 64 70 80
82      11/28/1996,02 06 07 13 20 22 23 29 42 62 64 66 69 71 72 73 74 76 79 80
83      11/27/1996,01 06 07 08 12 28 31 34 44 45 46 50 57 60 61 63 65 66 78 80
84      11/26/1996,06 07 08 09 14 15 23 29 33 34 40 45 49 54 56 59 63 71 74 78
85      11/25/1996,04 07 13 15 20 28 32 33 36 37 40 41 51 55 58 70 71 74 76 78
86      11/24/1996,04 05 07 17 24 33 35 43 44 45 46 53 54 55 60 61 63 68 73 75
87      11/23/1996,02 05 16 17 18 20 28 32 36 45 47 54 57 60 61 66 69 73 79 80
88      11/22/1996,16 18 19 21 25 27 32 38 42 44 46 52 55 56 57 58 60 61 70 80
89      11/21/1996,02 03 17 20 26 32 38 41 42 50 52 55 58 60 64 68 72 76 79 80
90      11/20/1996,10 17 18 20 27 32 33 38 46 47 54 56 60 62 67 73 74 77 79 80
91      11/19/1996,05 08 10 14 15 17 20 27 34 41 47 49 50 51 53 63 66 70 77 79
92      11/18/1996,01 02 06 07 14 15 16 23 27 28 36 40 52 53 54 59 62 68 74 78
93      11/17/1996,03 08 12 14 19 28 31 33 41 43 47 51 62 63 66 70 73 76 78 80
94      11/16/1996,02 05 07 11 21 23 26 28 34 37 38 39 46 51 57 60 65 66 70 80
95      11/15/1996,07 09 22 26 27 31 32 35 36 38 40 44 48 51 53 59 67 72 73 79
96      11/14/1996,06 07 11 16 24 28 29 35 39 40 41 44 55 59 63 65 70 72 78 79
97      11/13/1996,01 02 03 09 11 17 26 28 31 34 41 42 45 50 51 53 59 60 70 73
98      11/12/1996,01 12 14 15 24 25 28 31 32 33 35 52 53 57 58 60 64 65 75 77
99      11/11/1996,01 04 05 07 16 18 23 24 25 26 32 46 54 55 59 67 69 74 79 80
00      11/10/1996,01 02 04 08 13 21 31 32 33 37 43 46 52 53 63 70 71 73 75 79
01      11/09/1996,04 09 13 14 15 18 30 35 42 49 52 53 55 57 58 62 71 73 78 79
02      11/08/1996,05 14 15 26 27 28 31 32 43 47 49 51 53 55 61 62 66 72 75 79
03      11/07/1996,01 06 07 10 14 22 29 30 31 41 42 44 50 55 56 60 71 73 79 80
04      11/06/1996,01 02 05 09 19 26 27 29 40 44 53 54 55 56 61 62 68 70 76 80
05      11/05/1996,01 02 03 04 16 22 23 25 34 36 41 43 48 50 51 57 58 62 65 73
06      11/04/1996,03 06 08 09 10 11 20 27 34 37 43 45 50 54 61 62 69 76 78 80
07      11/03/1996,01 02 11 12 19 24 25 27 28 35 36 37 39 42 44 60 66 67 70 79
08      11/02/1996,03 04 06 07 08 12 15 23 31 33 36 37 49 50 51 52 56 69 76 80
09      11/01/1996,03 04 05 13 17 18 21 23 27 29 34 42 45 47 50 56 75 76 77 79
10      10/31/1996,02 03 05 06 09 10 15 16 28 32 43 51 55 60 61 63 68 69 70 74
11      10/30/1996,04 05 08 17 20 21 24 25 28 34 35 37 39 46 49 54 62 67 69 79
12      10/29/1996,04 06 08 15 16 21 25 29 34 35 38 44 46 61 62 64 67 69 74 80
13      10/28/1996,03 09 17 25 26 28 32 42 43 44 45 47 50 53 55 63 72 75 77 79
14      10/27/1996,02 04 05 10 11 14 16 21 37 38 43 47 50 51 57 60 63 64 76 77
15      10/26/1996,02 04 06 15 18 19 29 34 43 44 46 48 54 55 65 68 70 74 75 77
16      10/25/1996,01 03 14 18 22 24 26 28 35 36 40 49 51 52 53 67 75 77 79 80
17      10/24/1996,11 17 18 20 24 25 27 28 30 32 35 42 46 50 55 59 60 70 73 79
18      10/23/1996,03 06 08 11 14 15 17 25 32 37 38 40 42 49 50 51 52 57 71 72
19      10/22/1996,04 09 11 13 17 21 24 26 33 42 44 46 48 53 57 61 62 68 72 80
20      10/21/1996,05 09 14 20 21 22 29 34 38 39 43 47 48 52 58 62 72 76 77 80
21      10/20/1996,02 05 07 13 14 17 22 24 29 30 44 51 56 60 61 70 71 72 78 80
22      10/19/1996,05 07 11 13 17 23 28 29 32 34 35 40 42 44 45 49 52 57 61 74
23      10/18/1996,01 02 19 20 21 22 24 32 33 35 36 38 44 50 55 62 63 65 72 73
24      10/17/1996,04 05 07 13 31 34 42 43 47 49 51 53 55 62 63 68 73 78 79 80
```

```
10/16/1996,08 10 12 13 17 23 24 37 46 47 48 51 53 59 62 63 69 72 78 79
10/15/1996,06 10 12 25 27 28 33 39 41 44 46 47 49 50 51 53 59 60 66 75
10/14/1996,06 07 10 33 37 39 45 52 53 55 59 60 62 67 68 69 70 71 74 77
10/13/1996,02 04 05 10 11 15 18 23 28 34 35 36 39 40 45 51 52 59 69 76
10/12/1996,02 04 05 16 19 24 25 26 29 32 33 41 44 48 51 57 65 66 72 77
10/11/1996,05 07 16 19 30 32 35 45 49 50 51 53 55 57 59 60 62 70 73 77
10/10/1996,06 08 09 16 17 20 21 32 36 41 47 48 50 51 55 57 60 72 77 79
10/09/1996,02 17 20 22 27 28 29 35 37 38 41 42 45 52 58 60 67 71 72 76
10/08/1996,03 05 07 09 11 19 21 25 27 29 36 41 55 61 63 66 69 73 75 76
10/07/1996,07 13 14 16 18 21 25 31 33 38 42 52 56 57 66 71 73 76 79 80
10/06/1996,03 04 06 09 10 15 25 27 32 33 37 44 49 51 54 55 65 67 68 73
10/05/1996,10 14 17 23 29 31 35 37 38 39 43 46 48 51 53 61 62 67 75 78
10/04/1996,01 03 08 19 20 26 28 34 35 40 43 45 46 55 57 61 66 70 75 77
10/03/1996,01 02 07 10 18 23 26 27 29 31 41 45 47 57 59 68 70 71 74 78
10/02/1996,01 06 07 09 14 15 16 19 21 22 32 49 51 57 58 59 64 72 79 80
10/01/1996,02 04 09 10 14 15 17 18 20 21 25 29 46 57 61 68 71 72 74 79
09/30/1996,01 02 04 15 19 23 25 31 37 39 42 44 47 54 59 60 61 62 63 78
09/29/1996,02 03 09 17 24 28 29 33 35 42 52 54 55 56 61 62 66 69 76 80
09/28/1996,05 06 09 13 16 19 22 24 35 38 42 45 46 51 60 62 64 67 75 80
09/27/1996,01 05 10 13 15 20 21 25 30 43 45 47 50 54 56 68 69 70 74 80
09/26/1996,01 14 15 19 20 22 30 33 40 45 46 51 59 62 64 66 70 71 75 77
09/25/1996,01 08 18 21 25 31 34 35 37 39 43 47 51 53 59 63 69 75 76 77
09/24/1996,01 11 18 19 24 32 37 40 46 47 51 54 55 56 58 65 69 73 75 80
09/23/1996,07 09 14 15 17 21 22 28 29 36 39 43 44 45 48 57 58 59 60 73
09/22/1996,02 04 09 12 14 17 24 30 32 36 37 38 42 44 56 57 70 73 74 76
09/21/1996,05 07 10 11 15 20 23 30 33 35 38 42 44 47 65 69 71 74 75 76
09/20/1996,19 22 24 29 38 42 46 49 50 52 55 60 62 63 64 67 70 72 73 75
09/19/1996,08 14 15 21 22 28 29 31 39 43 53 54 57 58 67 71 73 76 78 79
09/18/1996,06 08 10 11 13 15 22 23 33 35 40 43 44 46 57 64 67 70 76 77
09/17/1996,06 08 26 28 31 35 36 38 45 47 50 52 56 60 61 70 74 75 77 78
09/16/1996,01 02 24 25 26 30 32 33 40 41 42 44 51 55 63 65 68 71 72 77
09/15/1996,02 13 15 18 19 20 22 29 32 33 37 42 45 46 51 52 55 60 63 68
09/14/1996,06 08 17 19 24 27 36 43 51 52 53 55 61 63 64 65 67 73 74 80
09/13/1996,06 07 09 14 19 20 22 25 29 38 41 42 43 55 56 68 70 72 74 76
09/12/1996,03 05 07 12 21 22 24 27 32 35 36 39 47 57 60 63 69 74 78 80
09/11/1996,04 05 11 13 14 16 26 31 35 38 39 54 55 58 65 67 71 74 77 80
09/10/1996,05 06 10 13 14 16 17 19 25 32 33 39 45 48 54 57 59 60 73 79
09/09/1996,06 07 08 10 15 16 18 31 35 37 40 42 46 49 61 65 67 76 78 80
09/08/1996,10 11 12 20 23 24 28 29 30 33 43 50 53 56 63 65 67 73 74 76
09/07/1996,09 10 19 22 23 27 35 36 40 46 47 50 57 58 62 63 74 75 77 79
09/06/1996,05 10 22 23 24 28 29 30 35 37 39 40 44 45 53 57 61 65 69 77
09/05/1996,06 11 13 15 19 20 23 24 25 27 29 39 45 49 63 64 67 68 73 76
09/04/1996,01 07 08 09 35 37 39 42 43 50 51 55 59 69 70 72 74 76 77 78
09/03/1996,10 19 21 22 23 27 29 30 35 36 44 49 55 58 66 67 69 75 78 80
09/02/1996,07 11 12 14 20 23 26 28 40 42 47 57 59 62 71 72 73 74 78 79
09/01/1996,04 07 08 09 18 20 24 28 29 32 45 49 51 61 63 64 65 66 67 74
08/31/1996,01 18 20 21 23 27 28 30 34 36 40 47 48 51 57 59 62 69 71 77
08/30/1996,12 13 14 16 19 24 33 37 40 45 46 48 52 60 63 64 66 70 71 79
08/29/1996,01 05 12 16 18 20 21 30 33 35 36 38 40 57 60 62 65 71 73 76
08/28/1996,09 13 14 15 17 21 22 24 36 39 44 55 63 65 67 71 74 75 76 79
08/27/1996,03 10 12 14 16 21 35 37 38 40 45 46 49 61 62 69 75 76 77 79
08/26/1996,02 05 10 15 21 22 27 28 30 32 37 43 55 60 63 66 67 73 76 77
08/25/1996,03 08 14 17 25 29 34 35 41 47 49 54 59 60 64 66 70 71 72 77
08/24/1996,02 06 08 10 15 31 33 37 40 42 44 48 50 55 56 57 60 64 68 70
```

```
79      08/23/1996,02 06 10 11 13 19 23 41 42 51 52 55 60 62 66 69 71 75 76 79
80      08/22/1996,04 06 08 09 12 14 21 26 32 33 39 41 45 53 56 60 67 70 75 76
81      08/21/1996,03 05 06 08 12 20 26 33 34 41 45 46 53 56 57 63 70 76 77 78
82      08/20/1996,11 12 13 18 22 23 27 29 33 40 42 46 47 48 58 62 64 65 67 74
83      08/19/1996,01 07 12 13 20 23 28 29 32 37 39 49 52 55 64 65 66 72 76 78
84      08/18/1996,06 14 15 17 21 28 35 37 38 39 40 41 45 52 54 58 60 62 72 76
85      08/17/1996,02 05 09 10 12 14 16 22 23 31 32 37 41 42 48 52 55 61 63 66
86      08/16/1996,01 04 09 18 21 28 33 38 40 41 45 46 47 48 52 56 58 73 76 79
87      08/15/1996,01 02 04 14 22 23 26 27 36 40 41 42 44 47 50 58 69 71 74 79
88      08/14/1996,03 11 17 18 23 24 28 30 46 49 51 52 54 56 58 60 61 62 76 77
89      08/13/1996,01 06 07 11 13 15 22 28 29 33 35 43 46 57 63 67 75 76 77 80
90      08/12/1996,03 04 08 11 13 16 18 30 33 35 36 42 48 50 54 64 68 75 79 80
91      08/11/1996,01 03 10 16 18 19 22 23 24 26 34 47 49 50 52 54 55 63 79 80
92      08/10/1996,01 06 07 13 14 17 20 30 35 39 40 46 52 55 65 68 74 76 78 79
93      08/09/1996,04 05 07 14 22 23 25 27 32 34 38 41 46 47 50 53 55 63 72 79
94      08/08/1996,04 07 09 13 15 16 24 30 31 41 42 44 45 49 50 62 63 67 71 77
95      08/07/1996,02 06 07 10 13 15 16 20 24 29 33 35 39 42 50 51 55 60 61 68
96      08/06/1996,01 04 09 15 18 19 21 24 28 32 34 37 39 45 53 54 62 64 70 75
97      08/05/1996,09 14 15 17 22 26 32 35 42 48 53 54 55 56 63 69 70 71 76 78
98      08/04/1996,14 17 20 24 28 34 35 42 44 45 48 49 51 55 61 70 76 78 79 80
99      08/03/1996,02 05 16 18 19 20 21 26 31 32 34 39 40 52 53 55 59 72 75 78
00      08/02/1996,03 10 14 15 22 24 28 33 35 37 40 41 47 48 51 57 60 65 75 76
01      08/01/1996,02 03 11 14 17 22 27 33 36 37 41 46 47 48 52 54 60 66 67 73
02      07/31/1996,03 11 16 18 19 23 28 41 42 52 53 58 60 68 70 71 72 74 78 80
03      07/30/1996,02 06 10 13 14 15 20 25 29 31 33 38 45 46 63 64 69 71 77 80
04      07/29/1996,02 03 07 12 16 26 45 47 48 49 51 52 59 60 62 64 67 74 76 80
05      07/28/1996,01 02 06 11 12 13 15 18 20 35 39 40 41 48 49 54 55 64 75 78
06      07/27/1996,03 06 07 08 10 18 19 21 23 31 33 35 41 44 46 58 59 66 69 70
07      07/26/1996,04 06 23 25 28 29 31 34 35 36 38 42 46 47 52 53 69 70 71 72
08      07/25/1996,03 10 17 19 20 26 29 31 36 39 40 47 51 52 64 66 72 75 78 80
09      07/24/1996,06 11 14 17 19 20 29 30 32 33 38 39 44 45 54 55 61 64 67 74
10      07/23/1996,02 05 10 12 13 19 20 26 33 36 41 43 44 48 53 57 63 65 71 72
11      07/22/1996,04 09 11 16 17 21 34 38 41 50 53 58 63 64 66 70 75 76 78 80
12      07/21/1996,02 03 04 11 21 22 23 25 38 39 47 50 61 62 63 64 69 72 75 76
13      07/20/1996,01 07 09 11 12 14 16 21 27 37 43 45 48 49 54 64 68 69 75 80
14      07/19/1996,01 03 07 10 13 17 25 26 28 32 36 39 43 47 52 54 59 63 75 76
15      07/18/1996,02 04 11 12 13 17 19 31 32 36 37 39 48 64 65 68 70 71 74 78
16      07/17/1996,03 08 15 29 33 39 41 43 55 57 60 61 67 69 70 71 72 76 77 80
17      07/16/1996,06 10 19 21 25 32 35 38 39 42 43 48 51 60 64 65 67 70 76 79
18      07/15/1996,07 08 10 13 17 24 25 27 36 38 41 45 50 58 66 69 70 71 72 78
19      07/14/1996,03 05 09 10 13 29 31 40 42 45 46 50 53 60 64 65 69 71 74 76
20      07/13/1996,03 09 13 15 22 24 27 28 32 37 39 42 43 50 51 59 60 63 64 71
21      07/12/1996,07 13 14 15 18 22 25 31 34 35 40 41 42 43 44 45 50 69 77 79
22      07/11/1996,01 02 03 07 12 13 17 20 24 28 34 35 39 47 51 54 56 62 69 73
23      07/10/1996,02 03 07 09 15 18 23 26 29 36 39 44 48 61 66 71 72 76 77 80
24      07/09/1996,06 11 26 30 33 34 36 39 41 48 49 51 53 54 62 64 65 69 76 80
25      07/08/1996,03 04 06 07 15 17 20 34 45 47 55 56 61 65 66 68 69 72 75 78
26      07/07/1996,08 09 11 17 20 23 31 33 46 48 52 53 56 57 60 68 72 75 76 77
27      07/06/1996,03 04 10 12 13 23 27 30 31 32 36 43 46 55 56 60 70 74 76 80
28      07/05/1996,07 09 10 11 13 16 18 24 30 31 32 35 45 58 62 63 64 67 70 77
29      07/04/1996,04 10 14 17 22 24 27 28 30 39 40 42 48 50 55 66 68 69 73 76
30      07/03/1996,01 04 09 12 18 19 21 27 31 42 44 48 51 55 57 66 67 69 70 74
31      07/02/1996,04 16 17 23 26 30 34 35 36 38 43 44 47 55 56 59 61 67 69 72
32      07/01/1996,06 07 08 19 20 30 31 38 41 43 44 45 48 56 64 66 70 72 77 79
```

```
33      06/30/1996,02 04 06 08 15 18 24 25 27 29 35 36 39 50 52 69 72 73 74 78
34      06/29/1996,07 08 09 21 22 28 29 33 41 42 47 52 55 56 62 64 70 76 78 79
35      06/28/1996,01 07 11 16 25 27 29 37 48 49 50 53 54 56 57 61 65 75 78 80
36      06/27/1996,04 15 20 23 25 26 32 35 38 45 48 49 54 58 65 66 67 68 77 79
37      06/26/1996,01 11 12 14 16 17 24 32 39 40 57 61 63 70 71 73 75 77 79 80
38      06/25/1996,06 12 17 21 25 32 33 41 43 44 46 47 57 59 60 64 66 70 76 79
39      06/24/1996,05 09 10 12 18 20 25 31 33 35 43 45 46 49 52 58 59 64 67 69
40      06/23/1996,02 07 10 16 19 21 24 26 29 34 36 37 42 46 62 64 69 70 74 76
41      06/22/1996,06 10 15 16 26 32 38 45 48 50 52 53 58 61 63 70 74 75 78 80
42      06/21/1996,02 05 07 08 25 27 28 35 38 42 54 63 64 65 67 70 72 73 74 77
43      06/20/1996,03 05 07 17 18 19 21 27 37 46 47 54 58 59 60 64 70 71 72 78
44      06/19/1996,02 06 08 17 27 33 36 41 43 44 49 51 52 56 57 63 66 67 75 78
45      06/18/1996,01 08 09 10 14 15 16 19 22 25 26 27 35 40 48 53 59 65 71 75
46      06/17/1996,02 05 06 09 11 16 17 22 27 29 35 41 44 48 51 55 60 64 70 80
47      06/16/1996,03 04 05 09 11 16 18 20 23 24 35 41 42 59 60 64 69 73 77 79
48      06/15/1996,01 02 05 06 07 08 23 24 26 29 31 38 47 56 58 59 63 65 73 78
49      06/14/1996,08 10 11 16 20 21 28 31 33 37 43 45 46 50 60 61 64 65 67 74
50      06/13/1996,01 02 07 12 13 14 16 19 20 23 29 42 48 49 50 52 55 62 71 80
51      06/12/1996,02 04 05 08 11 17 19 21 28 34 42 50 51 52 54 55 68 72 74 76
52      06/11/1996,02 03 05 06 14 18 23 27 33 34 43 47 52 53 54 58 65 66 70 72
53      06/10/1996,03 07 12 15 19 22 29 30 36 52 55 60 61 62 64 66 67 71 75 79
54      06/09/1996,05 10 15 18 19 21 24 28 40 47 48 58 63 64 66 69 71 72 73 74
55      06/08/1996,02 03 04 05 10 21 33 35 38 39 40 43 47 54 57 58 64 68 74 78
56      06/07/1996,07 11 16 32 37 40 43 51 52 53 54 55 60 61 71 72 73 74 77 78
57      06/06/1996,06 10 11 12 16 22 28 29 31 37 39 41 42 52 54 55 58 63 66 71
58      06/05/1996,02 03 06 07 11 12 14 16 17 31 35 38 40 42 43 44 47 57 77 80
59      06/04/1996,02 04 07 09 12 13 16 25 26 35 41 47 51 52 53 58 61 69 71 75
60      06/03/1996,03 06 11 26 27 28 30 31 33 40 47 50 52 56 58 59 60 62 68 77
61      06/02/1996,02 07 15 23 26 29 39 42 50 54 56 57 58 59 60 65 68 71 74 75
62      06/01/1996,02 12 15 21 23 25 32 34 43 50 51 58 59 63 67 68 69 75 77 78
63      05/31/1996,05 07 08 11 14 15 19 21 29 32 33 34 37 41 51 53 65 74 76 78
64      05/30/1996,03 07 11 17 20 22 23 29 30 35 39 46 51 55 56 63 64 67 71 72
65      05/29/1996,03 09 12 18 19 20 23 28 29 30 37 41 42 45 53 57 63 66 70 71
66      05/28/1996,04 06 10 12 14 22 24 30 34 38 39 42 44 48 49 50 56 57 62 78
67      05/27/1996,08 11 14 19 22 24 31 36 37 39 41 46 49 54 55 56 62 67 68 79
68      05/26/1996,07 08 11 14 24 25 28 29 30 36 43 45 47 48 54 57 66 68 70 77
69      05/25/1996,15 16 18 23 25 27 31 36 37 45 46 54 55 64 68 69 70 71 72 78
70      05/24/1996,10 16 17 21 24 31 33 35 39 41 48 50 54 56 61 65 68 71 74 78
71      05/23/1996,01 02 06 09 10 11 20 22 24 27 28 36 47 48 49 52 54 67 75 77
72      05/22/1996,05 08 09 10 11 15 18 21 23 28 30 31 44 45 50 51 57 61 64 73
73      05/21/1996,03 04 06 07 08 11 21 22 26 31 33 34 37 53 57 58 62 65 70 71
74      05/20/1996,06 10 15 19 20 21 29 33 36 43 49 61 62 66 67 68 70 72 77 80
75      05/19/1996,02 03 07 12 16 18 19 20 21 28 41 43 54 55 56 57 58 60 76 79
76      05/18/1996,03 11 12 16 18 22 26 27 29 39 40 47 49 50 51 53 54 59 65 76
77      05/17/1996,03 06 08 18 27 29 38 43 48 53 54 58 60 63 66 70 73 74 78 79
78      05/16/1996,03 08 10 19 25 26 29 30 33 37 42 44 47 48 51 52 62 66 70 79
79      05/15/1996,01 12 13 14 19 23 27 32 33 41 51 52 54 56 59 60 62 74 77 80
80      05/14/1996,01 05 18 22 26 28 30 37 38 41 43 46 56 57 67 70 71 75 76 77
81      05/13/1996,14 16 19 22 23 30 31 32 34 42 53 57 61 66 67 68 69 73 77 78
82      05/12/1996,04 08 10 12 14 21 24 25 28 43 44 45 47 50 51 54 55 70 76 77
83      05/11/1996,10 13 22 24 27 28 39 42 43 44 45 53 54 57 58 60 62 65 76 79
84      05/10/1996,02 12 15 17 18 20 22 25 31 32 34 36 37 42 54 56 58 67 76 80
85      05/09/1996,01 02 04 05 09 11 24 25 27 28 33 35 40 44 46 49 53 55 66 67
86      05/08/1996,04 05 07 10 11 17 23 26 33 36 38 48 49 51 52 55 56 57 60 67
```

```
87      05/07/1996,06 08 13 14 20 24 36 38 39 41 52 54 55 64 65 66 69 73 76 77
88      05/06/1996,04 08 10 17 30 31 35 36 41 45 48 50 51 58 62 63 65 66 67 74
89      05/05/1996,01 03 09 12 18 25 34 38 39 49 57 62 63 66 67 71 74 75 77 78
90      05/04/1996,03 09 11 12 28 29 33 34 35 37 38 54 62 64 65 68 69 72 79 80
91      05/03/1996,01 07 14 17 18 22 24 26 31 35 37 45 50 52 54 58 64 66 67 77
92      05/02/1996,03 06 13 15 16 24 29 32 33 37 38 45 52 55 58 67 73 75 76 78
93      05/01/1996,01 08 12 14 15 34 36 38 39 46 52 54 58 59 62 68 71 75 77 78
94      04/30/1996,03 09 11 13 17 21 26 31 32 36 37 38 53 54 57 59 61 64 66 69
95      04/29/1996,02 06 16 17 19 26 35 36 38 47 55 58 60 61 64 68 70 71 72 78
96      04/28/1996,09 16 20 21 22 24 25 26 31 37 39 46 52 53 63 64 65 68 70 77
97      04/27/1996,02 11 12 15 17 27 30 32 34 41 42 47 49 52 56 61 63 68 69 76
98      04/26/1996,05 07 09 14 15 18 19 27 31 32 33 34 39 41 43 44 48 55 64 73
99      04/25/1996,01 03 09 14 17 21 24 27 28 40 46 49 52 55 63 64 65 69 71 78
00      04/24/1996,07 08 10 12 14 22 24 29 32 36 37 52 54 55 57 58 60 70 77 80
01      04/23/1996,01 13 14 19 24 27 30 33 34 37 43 44 48 50 51 58 61 66 70 73
02      04/22/1996,10 14 16 18 32 36 37 38 46 47 53 55 56 58 61 62 65 68 69 78
03      04/21/1996,01 06 11 13 16 17 18 19 23 30 31 35 40 43 51 54 61 64 71 80
04      04/20/1996,03 04 06 13 23 26 34 37 42 53 55 59 61 64 65 69 70 71 73 80
05      04/19/1996,06 12 16 18 24 30 31 34 46 48 52 56 61 67 69 70 73 74 75 79
06      04/18/1996,01 02 06 09 11 19 36 39 45 48 50 52 64 65 68 69 71 72 76 79
07      04/17/1996,02 04 10 11 27 32 35 37 44 45 46 47 48 55 57 61 62 64 68 77
08      04/16/1996,12 15 24 26 28 30 35 37 38 44 45 46 47 49 52 64 68 70 72 78
09      04/15/1996,12 23 28 32 37 38 42 51 53 62 63 64 66 67 70 71 74 75 76 78
10      04/14/1996,06 13 14 16 21 22 29 30 33 36 42 43 44 51 52 55 62 70 72 75
11      04/13/1996,03 10 16 18 25 28 38 39 40 44 48 51 52 59 61 63 67 69 73 74
12      04/12/1996,01 09 10 18 32 36 38 41 43 44 49 53 58 62 63 66 67 70 71 74
13      04/11/1996,01 03 09 10 11 12 13 14 16 17 18 26 35 51 54 55 60 65 70 71
14      04/10/1996,02 10 11 17 20 21 23 29 31 39 41 44 47 58 66 68 74 75 76 80
15      04/09/1996,02 09 10 11 13 16 18 22 29 30 31 34 36 46 48 49 55 64 70 79
16      04/08/1996,03 04 05 11 14 20 22 24 38 39 42 49 53 54 61 62 69 72 77 79
17      04/07/1996,09 10 11 16 20 22 27 28 33 35 37 47 51 53 54 66 70 72 73 79
18      04/06/1996,05 06 09 14 19 23 24 27 30 33 34 36 43 54 58 62 64 67 71 78
19      04/05/1996,01 11 17 18 29 33 37 38 43 45 47 50 53 54 58 63 65 71 74 80
20      04/04/1996,03 11 15 26 35 38 39 40 42 43 45 46 49 55 58 59 63 67 70 75
21      04/03/1996,02 04 07 09 12 17 19 21 24 27 33 34 35 36 37 38 39 48 55 75
22      04/02/1996,09 12 15 20 27 32 34 43 48 51 53 55 57 62 63 65 69 72 75 79
23      04/01/1996,05 06 07 09 11 13 19 20 24 26 27 33 37 38 46 47 48 54 60 78
24      03/31/1996,02 03 06 08 11 25 29 35 40 41 48 57 59 60 61 64 65 66 71 80
25      03/30/1996,02 12 13 14 15 18 24 25 26 31 36 46 48 50 55 57 61 68 77 78
26      03/29/1996,04 15 16 19 27 30 33 38 39 42 46 53 54 55 63 64 65 66 72 80
27      03/28/1996,01 03 05 14 17 21 23 25 30 39 42 50 52 59 61 62 66 74 76 79
28      03/27/1996,02 09 14 15 33 39 41 47 48 49 53 61 62 65 66 69 71 73 75 80
29      03/26/1996,03 05 14 15 18 23 29 31 33 35 37 39 45 50 57 61 63 66 70 80
30      03/25/1996,04 09 15 16 20 34 35 40 50 52 61 62 63 64 65 66 67 68 69 79
31      03/24/1996,03 08 24 26 28 33 38 39 43 48 49 50 51 55 56 57 59 76 77 80
32      03/23/1996,01 06 10 13 15 24 25 27 31 34 36 37 45 58 66 67 68 69 70 78
33      03/22/1996,02 16 18 24 26 27 28 29 32 37 43 44 52 54 55 67 69 71 72 80
34      03/21/1996,03 05 09 12 15 21 31 32 44 45 55 58 61 65 66 67 70 71 74 79
35      03/20/1996,09 22 23 24 28 36 38 41 43 45 48 49 54 55 59 71 73 77 78 80
36      03/19/1996,09 14 15 17 18 21 27 28 29 33 38 45 48 55 58 59 60 61 65 69
37      03/18/1996,02 04 06 12 16 17 18 19 21 27 28 33 35 37 51 60 69 72 75 80
38      03/17/1996,05 14 22 32 34 35 37 39 42 45 50 51 57 58 60 65 66 67 77 78
39      03/16/1996,02 03 04 10 11 15 17 18 21 22 32 34 37 38 39 41 48 51 64 69
40      03/15/1996,03 09 10 13 20 22 31 36 39 40 47 52 57 60 61 66 76 77 79 80
```

```
41      03/14/1996,17 24 25 29 32 33 38 40 41 44 48 50 53 54 58 61 69 75 76 77
42      03/13/1996,06 15 18 27 28 30 36 39 44 45 46 53 55 58 60 61 65 71 72 73
43      03/12/1996,04 07 13 14 15 16 21 24 35 37 39 40 45 56 59 60 63 69 70 73
44      03/11/1996,05 07 11 13 16 20 22 24 35 41 45 48 54 64 66 68 71 76 79 80
45      03/10/1996,01 09 11 18 20 23 24 30 31 32 35 38 41 45 50 61 64 66 69 76
46      03/09/1996,01 11 16 19 27 28 29 30 31 33 40 42 48 54 56 58 62 75 78 80
47      03/08/1996,02 04 08 10 11 15 17 32 38 42 45 51 52 56 57 60 61 63 64 69
48      03/07/1996,01 03 06 08 10 11 12 20 22 23 24 32 46 48 56 60 68 75 77 79
49      03/06/1996,06 08 16 27 31 32 34 38 43 49 52 57 58 60 64 68 70 71 72 77
50      03/05/1996,06 07 08 11 13 17 26 28 35 37 38 49 54 57 59 60 61 62 65 72
51      03/04/1996,07 12 13 16 20 21 30 31 34 48 49 50 51 53 57 59 65 68 75 79
52      03/03/1996,04 05 06 12 15 20 22 30 32 40 43 44 55 57 58 59 60 62 65 79
53      03/02/1996,01 06 07 10 21 24 29 33 36 40 44 45 46 51 54 61 75 76 78 79
54      03/01/1996,07 11 16 19 28 30 35 36 39 41 43 46 50 51 53 58 59 60 70 75
55      02/29/1996,05 07 17 22 25 30 31 32 43 45 54 55 57 58 59 62 66 69 72 80
56      02/28/1996,04 06 09 18 19 24 25 28 32 38 46 48 49 59 60 61 65 66 69 79
57      02/27/1996,01 02 06 15 17 20 23 30 34 35 43 47 52 53 58 60 65 67 70 72
58      02/26/1996,01 07 11 13 20 33 34 35 38 45 49 54 56 58 59 60 62 63 67 77
59      02/25/1996,02 03 11 16 21 27 28 31 39 43 51 56 57 58 59 62 65 66 75 78
60      02/24/1996,03 04 24 27 29 32 36 38 39 44 46 47 52 54 61 69 71 72 75 80
61      02/23/1996,03 13 14 15 24 27 29 39 42 44 55 59 64 65 67 69 74 75 78 79
62      02/22/1996,02 05 10 13 14 16 22 25 29 30 38 41 42 55 58 66 68 70 72 73
63      02/21/1996,01 02 07 20 21 27 28 30 31 33 36 37 43 45 51 63 66 70 77 80
64      02/20/1996,05 12 14 19 28 29 33 35 36 42 43 46 47 50 52 53 54 57 62 73
65      02/19/1996,02 04 05 06 08 10 20 35 43 44 47 53 54 58 61 67 69 71 77 78
66      02/18/1996,04 13 17 19 22 26 30 35 52 55 59 60 61 66 68 69 70 72 75 76
67      02/17/1996,08 11 19 25 26 27 31 39 47 49 52 53 55 57 59 62 69 73 76 80
68      02/16/1996,03 04 05 08 10 12 16 22 23 24 30 35 37 49 56 57 66 70 71 76
69      02/15/1996,03 11 17 18 19 25 30 32 33 37 38 48 50 52 62 63 66 70 75 79
70      02/14/1996,01 02 10 11 12 15 17 23 26 32 33 39 46 54 55 58 62 65 70 79
71      02/13/1996,10 11 18 22 33 39 43 44 50 53 57 60 61 63 65 66 71 76 78 80
72      02/12/1996,04 08 13 17 24 33 36 40 43 49 50 51 52 57 61 63 65 67 70 79
73      02/11/1996,01 02 05 06 07 08 15 18 22 27 28 33 34 35 46 60 65 72 76 77
74      02/10/1996,01 03 05 10 11 15 22 28 29 36 41 48 49 57 61 63 64 68 73 80
75      02/09/1996,08 11 13 18 25 31 33 37 38 40 41 44 50 52 53 58 59 61 63 65
76      02/08/1996,04 06 07 10 14 18 20 25 27 30 37 39 40 45 46 50 58 62 74 80
77      02/07/1996,01 05 08 10 13 14 17 18 22 23 43 44 48 50 65 72 74 75 76 78
78      02/06/1996,15 19 21 22 39 41 43 44 46 48 52 60 63 65 66 68 69 70 74 77
79      02/05/1996,06 07 10 12 20 21 25 37 39 42 50 55 59 61 64 66 70 71 75 79
80      02/04/1996,17 18 22 26 30 35 36 39 42 43 47 48 53 55 56 67 72 77 78 80
81      02/03/1996,01 05 11 20 26 31 42 47 51 53 54 57 58 61 63 65 69 72 75 79
82      02/02/1996,02 04 18 19 20 21 22 23 27 34 37 39 42 53 54 62 71 72 75 77
83      02/01/1996,05 06 07 09 23 26 28 29 31 32 38 40 41 44 46 56 57 61 63 67
84      01/31/1996,04 05 06 07 08 11 12 15 17 23 34 36 38 41 47 57 70 72 75 80
85      01/30/1996,01 02 03 10 17 18 20 21 22 28 29 36 45 47 48 57 67 72 77 80
86      01/29/1996,10 16 17 20 23 24 29 32 33 37 38 48 51 53 56 59 61 63 67 72
87      01/28/1996,01 06 13 15 25 28 29 35 36 37 42 45 46 57 58 62 65 71 72 76
88      01/27/1996,05 10 15 19 20 21 23 24 30 32 34 40 46 48 52 61 65 69 71 74
89      01/26/1996,10 15 16 17 20 23 30 34 36 40 41 50 53 55 61 66 68 75 78 79
90      01/25/1996,02 03 05 10 16 18 19 20 34 35 39 44 49 52 53 54 63 65 74 77
91      01/24/1996,01 04 05 12 17 18 21 34 40 41 42 46 48 49 50 53 57 58 63 67
92      01/23/1996,03 07 10 17 21 22 23 24 38 39 43 44 45 52 62 64 72 73 78 79
93      01/22/1996,01 04 06 08 13 28 36 40 44 45 48 50 53 57 63 68 69 70 76 78
94      01/21/1996,09 13 15 18 30 35 37 39 44 45 46 50 53 58 59 60 63 70 73 79
```

```
95      01/20/1996,02 05 06 16 20 22 25 27 29 32 36 50 53 57 59 66 71 75 78 79
96      01/19/1996,10 11 12 20 22 23 24 29 31 32 35 37 38 56 57 59 61 62 73 76
97      01/18/1996,04 05 06 08 13 16 18 19 24 33 39 40 42 43 47 57 59 60 64 78
98      01/17/1996,05 09 10 13 15 17 23 28 29 30 35 37 40 43 45 48 51 55 58 61
99      01/16/1996,01 02 04 12 14 21 24 40 43 45 46 50 51 59 63 65 66 72 75 79
00      01/15/1996,03 05 07 16 17 24 25 28 30 34 35 39 43 44 53 57 58 62 66 79
01      01/14/1996,05 10 16 21 29 31 32 33 36 39 42 47 54 57 60 61 64 66 69 70
02      01/13/1996,02 08 10 12 14 19 24 26 32 37 39 40 44 46 52 54 61 69 72 73
03      01/12/1996,01 04 05 12 18 26 28 38 42 43 47 48 50 62 65 66 73 75 78 80
04      01/11/1996,06 07 15 19 20 23 24 28 34 37 40 45 46 52 56 60 66 69 76 80
05      01/10/1996,05 07 12 14 19 25 28 30 35 41 49 54 55 57 62 68 71 72 74 78
06      01/09/1996,18 20 22 27 28 31 33 38 40 43 52 58 60 61 62 63 70 74 75 77
07      01/08/1996,02 04 12 13 20 22 24 25 28 29 35 38 41 46 50 57 60 69 73 77
08      01/07/1996,05 07 08 09 11 13 19 23 26 30 32 36 40 50 54 57 61 65 70 76
09      01/06/1996,02 07 08 17 25 32 34 43 52 56 63 67 70 72 73 74 75 77 78 79
10      01/05/1996,01 06 10 11 15 19 20 21 22 32 41 49 53 56 62 71 73 74 79 80
11      01/04/1996,06 11 20 26 28 33 34 36 37 39 42 44 45 55 60 64 65 67 76 80
12      01/03/1996,02 04 08 15 19 20 25 27 31 41 44 49 51 52 59 66 67 73 76 77
13      01/02/1996,04 09 10 29 30 32 35 38 41 42 48 49 50 62 64 68 69 72 76 78
14      01/01/1996,02 06 07 11 14 17 18 20 30 34 36 41 43 52 60 61 70 74 75 76
15      12/31/1995,01 03 09 13 14 15 17 21 31 34 37 38 39 43 47 50 66 69 75 77
16      12/30/1995,03 07 08 12 19 26 28 34 35 40 46 55 56 60 63 67 70 73 77 79
17      12/29/1995,04 08 09 16 17 20 22 29 33 39 43 47 48 49 54 59 63 76 78 79
18      12/28/1995,05 07 08 16 20 22 26 29 31 32 43 49 58 59 64 67 68 69 78 79
19      12/27/1995,03 07 08 15 20 24 26 28 32 40 45 61 62 63 67 70 71 73 77 80
20      12/26/1995,01 03 04 15 24 28 34 41 45 49 50 51 63 64 65 67 71 72 74 80
21      12/24/1995,06 11 12 13 16 19 22 24 25 26 32 36 48 58 63 64 65 67 74 80
22      12/23/1995,04 06 09 21 25 26 27 29 32 36 37 40 44 47 49 58 59 62 66 72
23      12/22/1995,03 07 14 15 18 19 20 21 22 41 42 43 50 51 54 55 65 71 74 80
24      12/21/1995,02 03 05 14 18 20 27 35 38 42 45 46 47 52 62 63 71 74 75 76
25      12/20/1995,02 09 14 27 28 34 35 47 51 55 58 61 62 64 65 68 72 73 77 80
26      12/19/1995,04 09 14 15 18 19 21 30 35 45 46 48 55 58 60 66 70 74 75 80
27      12/18/1995,15 18 21 30 31 34 38 41 42 44 48 50 51 52 57 58 67 74 78 79
28      12/17/1995,01 13 18 27 30 31 39 41 42 43 45 46 49 50 51 58 61 65 68 79
29      12/16/1995,01 02 08 19 20 26 27 30 33 34 41 45 55 57 59 60 62 63 64 66
30      12/15/1995,01 04 07 13 17 22 26 32 34 35 40 43 44 48 54 55 64 69 72 76
31      12/14/1995,01 02 10 13 26 27 40 44 46 47 50 55 56 59 61 62 68 71 74 79
32      12/13/1995,03 09 10 12 14 34 37 39 43 44 46 49 51 59 60 62 68 74 75 76
33      12/12/1995,11 14 17 19 23 30 35 38 40 44 49 51 52 56 57 61 69 72 77 80
34      12/11/1995,02 06 09 23 26 31 33 42 46 52 56 57 58 60 62 65 66 68 73 76
35      12/10/1995,08 14 17 19 21 31 38 41 43 45 48 49 56 61 62 66 69 72 76 78
36      12/09/1995,04 09 11 26 28 33 36 37 44 45 47 49 52 53 58 64 66 68 71 78
37      12/08/1995,01 06 08 11 27 29 35 38 40 42 45 51 53 57 61 62 65 66 70 76
38      12/07/1995,03 07 09 11 22 27 28 31 33 38 39 45 46 47 48 52 58 64 66 75
39      12/06/1995,04 08 13 15 17 18 19 27 32 33 34 47 54 55 56 65 69 71 73 75
40      12/05/1995,01 08 10 14 21 28 29 36 37 42 45 46 49 51 52 59 61 66 77 80
41      12/04/1995,02 04 05 21 22 27 31 37 42 44 51 54 57 63 66 69 73 76 77 80
42      12/03/1995,06 10 14 19 24 27 28 30 36 41 42 45 46 47 51 55 57 62 63 77
43      12/02/1995,01 15 21 27 30 34 42 43 44 48 51 52 56 59 61 64 65 67 71 73
44      12/01/1995,02 12 13 15 19 23 28 29 33 38 48 51 63 65 66 67 68 69 72 74
45      11/30/1995,07 12 16 18 20 21 27 29 31 36 41 45 61 62 65 66 67 73 74 77
46      11/29/1995,05 06 07 14 15 16 23 30 34 42 44 49 54 56 61 62 70 75 76 78
47      11/28/1995,01 11 18 20 21 26 32 33 34 35 47 49 52 58 62 66 72 76 78 79
48      11/27/1995,05 17 34 40 42 45 48 50 54 55 62 64 65 66 69 70 74 76 78 79
```

```
49      11/26/1995,02 04 07 09 26 30 33 35 38 39 48 49 52 54 57 58 59 61 69 76
50      11/25/1995,14 15 18 19 23 25 26 27 28 30 35 36 51 53 57 60 61 67 77 79
51      11/24/1995,05 08 09 14 15 18 21 28 29 31 34 36 38 43 48 50 60 71 75 77
52      11/23/1995,05 08 14 15 18 26 31 39 47 50 51 52 53 54 61 63 70 74 76 80
53      11/22/1995,10 11 15 19 23 26 30 35 36 37 39 41 42 47 48 49 57 64 71 77
54      11/21/1995,01 04 11 18 26 27 31 34 35 44 50 54 55 61 63 67 68 74 76 78
55      11/20/1995,04 05 06 09 16 28 31 33 34 36 40 41 42 43 45 51 53 70 75 78
56      11/19/1995,11 19 20 21 23 31 35 36 38 39 44 47 50 52 60 61 70 75 76 80
57      11/18/1995,06 10 11 13 16 19 22 24 35 42 48 49 51 63 66 67 71 73 75 80
58      11/17/1995,09 15 16 20 34 39 43 47 52 53 59 66 70 72 73 74 76 78 79 80
59      11/16/1995,03 04 06 07 17 19 21 24 26 34 37 39 42 44 48 51 57 67 70 79
60      11/15/1995,01 05 08 09 14 19 21 36 37 44 45 51 52 57 60 71 73 75 76 77
61      11/14/1995,04 05 06 10 21 25 27 33 34 43 46 48 49 50 54 55 63 65 69 75
62      11/13/1995,03 09 14 16 21 24 32 37 46 47 51 52 53 55 58 61 63 75 78 80
63      11/12/1995,14 18 23 27 29 30 32 36 39 41 45 46 52 56 59 69 70 73 78 80
64      11/11/1995,03 04 11 12 13 15 21 23 32 34 37 41 46 47 48 58 67 70 77 79
65      11/10/1995,07 22 23 26 28 29 32 34 36 39 47 49 52 58 60 71 73 74 77 80
66      11/09/1995,03 07 10 13 19 21 24 26 30 32 38 40 44 47 48 50 56 68 72 80
67      11/08/1995,06 09 16 17 18 23 24 25 29 32 39 44 45 46 51 62 63 69 72 80
68      11/07/1995,06 08 09 14 16 17 22 23 25 27 32 35 38 41 49 52 59 67 69 79
69      11/06/1995,02 03 14 20 28 29 30 31 33 39 41 47 56 58 60 63 64 67 76 80
70      11/05/1995,05 06 09 17 19 20 27 30 35 38 43 44 49 51 53 54 67 72 76 77
71      11/04/1995,01 04 05 06 08 13 14 16 24 30 32 41 47 54 55 56 59 62 73 79
72      11/03/1995,07 15 17 18 21 22 30 33 41 42 43 46 47 55 58 65 66 71 72 74
73      11/02/1995,06 07 19 20 22 25 28 29 31 40 42 43 48 49 50 58 68 72 78 79
74      11/01/1995,03 06 07 08 14 16 20 25 27 30 31 35 43 48 50 54 58 61 62 79
75      10/31/1995,07 10 14 19 20 30 38 39 40 45 48 52 60 62 66 73 74 75 77 80
76      10/30/1995,03 06 08 13 16 18 21 22 34 37 42 45 47 53 58 59 63 64 68 78
77      10/29/1995,04 09 10 16 18 20 26 28 32 34 40 44 45 46 51 54 63 67 73 80
78      10/28/1995,02 09 17 23 27 28 29 32 36 37 40 43 54 56 58 61 66 73 75 77
79      10/27/1995,05 12 13 16 17 23 24 27 28 31 41 43 44 47 53 60 71 72 73 79
80      10/26/1995,01 09 13 15 17 18 31 35 37 40 41 43 44 45 48 51 56 70 77 78
81      10/25/1995,01 09 14 16 17 21 23 24 34 39 40 46 48 57 60 61 67 70 71 79
82      10/24/1995,03 07 08 17 23 24 28 34 35 41 45 48 57 62 68 69 76 78 79 80
83      10/23/1995,05 17 19 24 34 37 39 40 41 43 49 50 57 59 63 66 70 71 78 79
84      10/22/1995,03 16 17 27 28 31 33 34 38 43 45 47 52 58 59 61 64 67 73 74
85      10/21/1995,05 09 13 14 16 20 23 28 37 45 46 47 56 61 64 68 70 72 75 78
86      10/20/1995,01 06 07 10 12 16 18 23 25 31 41 47 49 54 58 59 60 66 76 80
87      10/19/1995,05 07 08 14 15 17 20 21 22 27 31 34 35 37 40 44 55 56 61 69
88      10/18/1995,04 05 09 12 13 16 18 26 37 41 50 55 57 65 66 67 68 69 75 80
89      10/17/1995,01 03 05 09 12 15 18 31 32 42 43 44 51 56 57 66 67 71 72 78
90      10/16/1995,01 07 08 15 18 20 24 27 29 34 53 55 56 57 62 67 68 73 78 80
91      10/15/1995,01 02 03 08 26 27 32 42 44 47 51 54 55 58 61 65 73 75 77 78
92      10/14/1995,20 25 26 27 29 31 32 34 40 48 55 56 60 62 63 70 73 74 75 80
93      10/13/1995,03 04 07 20 22 32 35 36 38 39 42 43 51 52 58 65 66 68 72 75
94      10/12/1995,06 13 14 18 23 27 29 33 37 38 45 48 49 52 55 57 60 63 77 80
95      10/11/1995,01 02 05 06 07 16 21 29 42 44 46 49 50 51 52 58 60 61 64 78
96      10/10/1995,01 12 15 18 26 27 31 36 39 41 43 53 58 60 63 64 72 74 77 78
97      10/09/1995,02 18 25 26 30 32 34 36 37 40 43 45 48 54 57 63 67 74 77 78
98      10/08/1995,02 06 11 16 19 22 32 41 42 44 49 52 53 55 60 61 64 65 73 80
99      10/07/1995,01 10 11 12 16 19 24 28 30 32 45 49 50 54 57 66 68 71 72 75
00      10/06/1995,01 03 08 14 17 22 23 24 27 31 33 35 44 46 48 51 61 68 73 78
01      10/05/1995,01 03 04 11 16 19 20 21 31 32 39 42 54 63 66 72 76 77 78 80
02      10/04/1995,08 09 11 16 17 22 25 29 32 34 42 43 46 53 56 59 62 64 73 74
```

```
03      10/03/1995,02 07 12 13 15 16 18 19 31 33 38 41 50 52 55 60 63 72 75 79
04      10/02/1995,01 02 03 07 08 12 24 27 28 46 50 58 59 62 64 68 69 74 75 79
05      10/01/1995,01 02 03 04 09 11 14 30 34 45 54 55 56 59 65 67 70 72 73 74
06      09/30/1995,07 11 15 16 20 21 26 35 46 50 52 54 56 61 62 63 69 72 75 77
07      09/29/1995,04 11 26 30 34 36 40 42 46 48 49 54 59 61 62 63 65 72 73 79
08      09/28/1995,08 09 10 15 25 28 34 35 37 40 42 47 53 54 55 57 65 71 75 78
09      09/27/1995,03 10 17 21 24 26 30 37 40 41 46 48 50 51 54 62 64 72 77 78
10      09/26/1995,01 04 05 14 17 32 33 36 39 40 51 52 56 58 59 62 63 66 67 77
11      09/25/1995,03 05 08 23 24 30 32 36 40 42 44 48 53 59 61 69 70 72 74 75
12      09/24/1995,02 03 11 15 18 27 40 41 49 52 55 57 58 60 61 67 69 70 71 79
13      09/23/1995,02 09 13 14 18 19 22 24 27 33 34 36 37 45 47 54 70 75 76 77
14      09/22/1995,01 05 06 16 18 22 30 31 33 35 37 39 42 46 47 51 52 56 73 80
15      09/21/1995,08 10 11 16 20 30 31 41 43 45 47 51 53 56 62 66 71 75 77 80
16      09/20/1995,07 09 14 15 18 19 23 24 25 28 35 45 48 52 58 59 60 66 67 74
17      09/19/1995,04 05 15 24 27 36 43 45 46 50 53 58 61 63 67 68 72 74 79 80
18      09/18/1995,02 09 15 19 20 25 27 28 35 38 39 43 45 46 47 57 60 63 74 80
19      09/17/1995,01 05 11 13 14 19 23 29 30 34 40 47 48 61 63 65 68 73 74 79
20      09/16/1995,02 04 07 08 14 21 22 25 26 32 36 40 43 45 59 62 66 72 74 79
21      09/15/1995,04 07 09 13 18 20 24 26 31 38 42 45 48 49 54 60 65 74 79 80
22      09/14/1995,03 05 07 16 17 22 29 31 37 40 42 49 51 54 58 59 67 69 70 72
23      09/13/1995,01 10 15 17 18 23 26 28 30 33 34 35 37 42 44 50 54 61 74 77
24      09/12/1995,03 06 12 17 26 27 34 37 40 46 50 55 56 61 65 67 69 70 73 78
25      09/11/1995,01 05 06 09 11 12 14 18 21 24 28 31 33 45 50 58 66 68 74 79
26      09/10/1995,01 04 05 06 11 15 21 22 32 39 45 51 53 56 61 63 70 72 73 78
27      09/09/1995,01 05 11 13 14 23 30 31 39 45 53 56 61 62 63 66 67 72 76 79
28      09/08/1995,01 05 11 15 21 27 42 44 46 48 53 58 61 63 67 72 73 77 79 80
29      09/07/1995,06 07 09 15 16 18 24 33 34 36 37 46 55 56 61 62 66 67 68 69
30      09/06/1995,09 11 17 28 29 30 32 36 42 46 47 49 51 54 58 62 68 74 75 78
31      09/05/1995,11 12 13 14 16 19 21 26 33 34 35 47 49 54 55 66 69 71 74 78
32      09/04/1995,04 07 11 12 16 26 30 35 39 45 46 49 51 57 59 61 62 65 69 79
33      09/03/1995,09 11 12 14 33 35 38 40 41 47 48 49 52 55 56 57 70 72 73 78
34      09/02/1995,05 08 13 23 24 26 30 31 33 34 36 47 49 53 55 57 58 65 67 80
35      09/01/1995,01 03 04 11 13 16 17 20 26 28 32 35 41 62 65 67 68 74 77 79
36      08/31/1995,02 04 10 11 16 20 22 23 32 33 34 36 38 50 60 63 72 74 77 79
37      08/30/1995,03 09 12 18 23 28 41 46 48 49 50 52 55 59 66 69 70 74 76 78
38      08/29/1995,03 06 07 09 12 19 23 24 38 39 48 50 56 57 65 73 74 76 78 79
39      08/28/1995,01 12 14 23 26 29 31 38 43 48 49 50 52 61 63 65 66 72 76 78
40      08/27/1995,08 09 20 24 28 36 41 46 50 59 60 61 65 66 68 69 74 75 76 80
41      08/26/1995,10 11 16 18 19 24 29 31 33 34 35 38 40 42 43 54 59 65 69 77
42      08/25/1995,07 08 14 17 18 20 21 25 31 40 41 42 45 49 51 61 64 65 67 68
43      08/24/1995,03 09 10 11 16 18 20 23 26 28 31 32 35 39 42 44 66 67 73 76
44      08/23/1995,01 02 10 11 12 13 15 24 25 32 37 39 48 50 56 58 62 63 69 70
45      08/22/1995,01 11 12 13 20 25 28 31 33 37 46 48 49 55 58 61 69 70 72 80
46      08/21/1995,02 16 18 20 21 25 29 37 38 40 47 48 50 51 61 62 64 70 74 79
47      08/20/1995,03 21 24 32 41 42 44 47 50 51 53 56 58 59 62 67 70 72 78 80
48      08/19/1995,03 04 06 07 12 16 17 19 22 27 28 35 37 47 56 62 64 70 71 72
49      08/18/1995,04 06 07 09 13 15 22 27 30 33 34 43 50 60 61 64 68 72 75 80
50      08/17/1995,01 03 04 05 08 15 16 20 26 27 28 39 40 45 46 47 55 70 75 78
51      08/16/1995,03 07 09 10 13 24 28 32 33 35 36 41 42 48 51 58 69 70 72 75
52      08/15/1995,01 03 04 06 24 27 34 36 37 39 40 42 46 53 56 64 68 69 71 79
53      08/14/1995,03 06 15 21 23 26 30 34 37 46 49 52 53 58 61 62 65 70 75 79
54      08/13/1995,05 06 08 10 15 16 19 21 30 38 39 41 44 48 49 51 55 60 70 73
55      08/12/1995,02 05 16 17 19 21 29 45 46 49 51 54 56 57 61 62 63 65 67 80
56      08/11/1995,09 13 15 24 25 27 28 30 35 36 40 41 43 53 58 59 61 64 66 76
```

```
57      08/10/1995,03 05 07 11 14 15 19 20 28 31 37 39 49 56 57 65 67 70 72 76
58      08/09/1995,02 05 11 12 15 18 25 26 37 42 44 48 49 50 61 63 64 71 74 78
59      08/08/1995,01 05 10 14 20 24 31 38 40 41 45 47 50 63 66 69 70 71 73 80
60      08/07/1995,03 05 08 09 11 12 18 27 29 32 35 36 42 47 49 50 68 72 73 77
61      08/06/1995,02 05 07 12 16 20 31 32 33 35 43 44 52 53 66 67 69 73 74 76
62      08/05/1995,02 04 10 11 13 14 24 25 32 34 37 42 48 50 52 57 58 63 65 74
63      08/04/1995,08 10 13 16 18 21 22 24 27 33 34 35 40 44 49 62 65 69 74 78
64      08/03/1995,01 03 07 09 13 14 17 18 21 29 35 42 43 44 50 54 55 64 76 79
65      08/02/1995,02 05 06 15 17 22 23 27 32 34 36 37 41 43 52 57 66 68 69 76
66      08/01/1995,06 08 09 11 17 19 22 23 25 26 32 36 39 52 55 58 61 66 73 80
67      07/31/1995,01 04 12 13 16 21 24 32 36 37 39 40 45 52 53 54 66 75 76 77
68      07/30/1995,02 08 17 20 27 30 32 37 40 53 58 62 63 64 65 68 70 71 76 79
69      07/29/1995,01 06 09 17 20 21 28 32 37 41 45 47 53 59 64 66 71 72 77 80
70      07/28/1995,03 07 09 15 16 27 35 39 53 54 55 57 61 62 63 65 66 67 68 80
71      07/27/1995,04 06 10 13 14 19 23 26 27 30 39 43 44 47 49 56 58 66 70 80
72      07/26/1995,02 05 08 10 13 14 19 21 23 27 36 39 42 53 54 56 62 72 74 80
73      07/25/1995,02 03 04 07 08 10 16 18 21 24 25 28 33 37 50 53 58 60 61 63
74      07/24/1995,05 07 09 12 13 22 26 27 30 41 47 49 50 51 64 74 75 76 77 78
75      07/23/1995,01 02 16 21 31 35 38 48 49 50 51 55 58 61 68 70 71 74 76 77
76      07/22/1995,01 05 06 08 16 30 34 35 36 42 46 48 50 51 52 59 60 62 63 73
77      07/21/1995,02 05 07 15 16 22 25 32 34 41 43 49 52 58 59 65 72 74 78 80
78      07/20/1995,06 09 10 14 22 27 30 38 39 45 46 48 49 54 62 63 65 72 76 79
79      07/19/1995,02 03 07 15 20 22 28 32 33 35 37 48 49 50 55 56 63 66 69 73
80      07/18/1995,01 06 16 17 19 30 31 34 37 38 42 43 45 50 51 58 67 68 70 77
81      07/17/1995,01 02 03 05 07 14 24 25 29 35 40 43 52 53 63 67 70 73 77 79
82      07/16/1995,03 07 23 25 26 29 34 39 40 43 44 45 48 53 57 62 65 68 72 78
83      07/15/1995,04 05 06 10 16 19 20 22 23 28 29 31 32 33 45 52 56 59 62 76
84      07/14/1995,03 05 07 14 15 22 35 38 44 46 49 53 57 60 63 64 70 75 78 80
85      07/13/1995,02 06 13 14 16 20 23 37 40 47 51 55 56 58 63 72 73 77 79 80
86      07/12/1995,01 03 04 05 09 13 16 25 34 35 37 40 50 51 52 59 70 71 78 80
87      07/11/1995,02 06 08 09 12 16 22 23 25 30 35 37 39 40 45 54 57 60 77 80
88      07/10/1995,10 11 12 18 29 40 45 48 50 52 56 58 59 65 66 67 70 73 74 80
89      07/09/1995,01 04 07 09 11 12 17 18 19 20 23 33 42 43 51 55 61 68 72 77
90      07/08/1995,06 10 13 14 15 19 23 24 27 28 38 50 53 56 57 62 64 66 67 76
91      07/07/1995,05 08 09 14 19 21 23 25 39 41 47 48 55 58 59 60 61 72 74 80
92      07/06/1995,08 15 18 24 27 28 31 34 37 42 46 47 51 52 53 57 64 65 72 73
93      07/05/1995,10 11 12 13 15 18 23 25 26 37 38 39 41 45 51 53 57 59 65 80
94      07/04/1995,06 08 13 17 20 27 29 32 37 38 43 55 56 58 61 68 69 72 74 76
95      07/03/1995,16 18 23 24 29 32 36 37 42 45 46 48 49 61 69 73 75 77 78 79
96      07/02/1995,02 11 16 17 18 23 31 38 40 45 53 54 55 66 67 69 71 72 74 76
97      07/01/1995,03 04 09 10 14 15 20 21 25 31 36 39 42 44 46 57 59 60 77 80
98      06/30/1995,05 15 16 17 18 22 26 27 31 33 35 36 53 56 60 61 67 71 72 74
99      06/29/1995,04 10 19 20 22 25 29 34 38 41 50 53 55 62 63 65 75 76 79 80
00      06/28/1995,01 02 03 07 13 15 17 21 30 32 35 36 37 42 44 55 60 61 63 77
01      06/27/1995,02 04 05 06 17 19 30 39 42 43 48 51 54 62 64 67 71 74 75 77
02      06/26/1995,02 03 05 07 08 16 20 36 38 42 51 52 53 54 56 58 60 67 71 74
03      06/25/1995,03 04 07 16 23 24 29 39 43 47 54 56 59 60 61 64 65 66 77 79
04      06/24/1995,02 06 07 13 16 19 20 23 24 30 37 38 47 50 54 59 61 68 73 78
05      06/23/1995,01 02 03 16 17 19 22 25 31 36 45 54 55 57 58 59 64 67 68 75
06      06/22/1995,11 12 16 21 23 24 31 35 36 37 38 40 41 48 51 61 72 75 76 78
07      06/21/1995,06 13 15 18 35 36 39 40 42 43 45 50 52 55 56 57 59 65 66 78
08      06/20/1995,10 12 14 16 18 19 20 22 24 26 35 36 38 45 50 53 62 66 71 80
09      06/19/1995,07 09 10 12 13 14 19 25 35 37 43 45 53 55 56 62 68 69 74 78
10      06/18/1995,07 21 24 26 27 28 30 31 32 34 43 48 55 57 60 65 66 69 75 76
```

```
11      06/17/1995,01 02 05 06 12 20 21 22 25 28 31 32 37 38 44 45 46 56 70 80
12      06/16/1995,01 04 11 13 19 20 27 28 30 32 33 38 40 41 62 63 64 70 73 74
13      06/15/1995,07 11 27 29 33 35 39 43 45 47 49 50 51 58 60 65 68 70 71 76
14      06/14/1995,03 07 10 14 23 24 25 26 31 39 43 45 48 50 53 56 58 63 66 76
15      06/13/1995,08 09 15 16 21 24 28 40 43 44 46 47 49 52 62 70 71 77 79 80
16      06/12/1995,01 04 12 14 17 21 25 29 32 33 39 41 46 54 64 65 68 75 76 80
17      06/11/1995,01 03 05 12 18 19 22 23 25 28 33 35 39 50 56 60 62 63 76 80
18      06/10/1995,01 15 16 17 20 25 26 28 32 34 36 40 41 48 61 65 69 72 75 78
19      06/09/1995,04 10 28 32 43 45 48 50 52 55 58 60 62 64 66 67 68 72 73 79
20      06/08/1995,06 12 13 16 26 30 33 34 47 49 54 57 59 60 63 64 66 67 76 80
21      06/07/1995,02 09 13 15 20 22 26 29 30 35 38 42 49 55 61 67 69 70 71 80
22      06/06/1995,08 10 11 21 24 33 36 40 42 43 52 53 55 62 65 67 69 73 74 77
23      06/05/1995,01 03 11 14 19 20 25 29 35 36 37 38 44 49 52 57 62 65 73 78
24      06/04/1995,01 02 05 09 11 14 15 21 28 30 31 35 38 43 45 51 57 66 67 75
25      06/03/1995,03 05 18 24 25 27 31 33 35 37 41 49 51 52 54 60 63 68 74 76
26      06/02/1995,10 16 21 22 25 26 28 29 34 39 40 41 42 44 58 60 61 62 71 77
27      06/01/1995,02 03 09 19 22 28 31 34 42 47 48 51 52 55 59 62 64 71 79 80
28      05/31/1995,10 12 17 24 25 28 32 33 36 38 42 48 49 52 56 58 72 73 75 76
29      05/30/1995,15 16 20 21 26 28 33 37 43 51 55 59 62 64 66 67 72 75 77 78
30      05/29/1995,02 12 15 18 19 25 26 27 29 39 41 44 57 69 70 74 75 77 78 80
31      05/28/1995,01 02 07 08 09 10 11 12 13 31 32 42 45 51 53 62 65 66 72 73
32      05/27/1995,03 08 09 12 16 23 24 26 34 35 41 42 53 54 57 58 62 64 67 78
33      05/26/1995,02 03 06 26 29 31 35 36 38 42 44 45 47 49 50 57 61 65 68 71
34      05/25/1995,04 05 12 24 26 30 32 35 37 38 41 52 57 61 65 66 73 74 77 78
35      05/24/1995,01 05 06 07 08 09 23 24 29 36 40 50 51 55 57 60 67 73 77 78
36      05/23/1995,03 06 13 18 20 24 29 30 33 34 36 41 42 47 48 51 60 68 71 72
37      05/22/1995,05 06 07 09 14 19 21 25 27 29 31 33 36 39 44 48 51 52 57 58
38      05/21/1995,01 02 04 06 13 18 20 23 26 29 33 36 40 46 55 61 62 71 75 78
39      05/20/1995,04 11 12 13 25 26 28 29 38 43 45 49 57 60 63 65 66 67 68 70
40      05/19/1995,03 04 05 07 11 17 18 19 28 29 36 38 46 58 60 67 69 70 76 78
41      05/18/1995,03 06 09 13 15 21 25 32 37 47 48 49 50 53 60 62 66 69 70 73
42      05/17/1995,12 22 24 25 26 32 34 39 40 41 42 51 56 59 60 68 71 72 76 79
43      05/16/1995,03 13 14 20 21 24 27 28 31 32 39 41 48 54 55 61 69 70 71 78
44      05/15/1995,02 07 09 11 18 20 21 24 25 29 33 36 37 42 43 49 56 64 68 77
45      05/14/1995,11 15 23 24 39 42 44 45 49 51 53 54 56 57 67 71 73 74 76 79
46      05/13/1995,06 11 12 17 18 20 28 31 32 35 38 40 41 54 57 58 60 62 67 70
47      05/12/1995,01 02 04 06 08 15 20 21 23 24 34 35 36 52 58 63 68 70 72 75
48      05/11/1995,07 09 12 13 15 17 19 21 22 24 26 34 35 38 40 51 52 53 59 68
49      05/10/1995,01 04 09 11 14 19 20 31 37 42 45 46 48 51 54 59 61 63 75 80
50      05/09/1995,02 09 13 15 20 21 24 28 30 33 35 38 41 45 49 57 62 71 73 78
51      05/08/1995,01 02 15 18 23 29 31 34 36 40 42 44 46 48 50 61 67 76 78 79
52      05/07/1995,01 02 07 08 10 12 16 21 27 29 32 33 37 39 43 47 60 61 67 79
53      05/06/1995,04 05 06 13 20 25 29 30 32 33 35 43 46 58 62 67 69 73 75 78
54      05/05/1995,07 14 17 20 25 26 34 37 39 43 44 46 52 60 66 71 73 74 76 80
55      05/04/1995,02 13 19 25 27 42 44 50 51 53 54 55 59 61 62 63 65 71 76 80
56      05/03/1995,10 16 18 19 22 23 26 33 36 38 40 41 44 45 57 58 65 68 73 77
57      05/02/1995,10 13 15 21 25 26 27 30 31 32 37 44 53 59 62 69 73 74 76 79
58      05/01/1995,01 06 10 13 17 19 21 25 30 38 39 41 43 53 55 56 57 59 65 73
59      04/30/1995,07 14 16 21 22 28 30 31 37 38 44 45 46 49 63 70 71 75 76 80
60      04/29/1995,01 03 06 12 20 22 26 28 38 41 46 47 49 51 56 57 60 70 74 76
61      04/28/1995,01 07 10 11 17 26 27 28 35 37 50 51 52 53 60 61 64 68 70 74
62      04/27/1995,03 04 08 14 16 17 18 30 32 37 39 47 54 60 61 65 67 73 78 80
63      04/26/1995,03 04 12 15 16 18 25 26 29 32 45 49 50 56 58 66 68 69 71 80
64      04/25/1995,09 16 19 21 22 23 25 27 29 36 41 42 46 53 55 59 61 68 70 77
```

```
65      04/24/1995,02 03 05 13 16 18 19 23 24 25 27 43 47 51 52 55 62 64 69 80
66      04/23/1995,04 07 19 21 23 26 33 34 36 46 47 50 52 54 58 59 64 76 78 80
67      04/22/1995,01 07 13 14 16 17 19 21 22 30 39 43 48 52 55 57 58 59 62 77
68      04/21/1995,02 03 09 10 15 22 27 30 32 36 44 47 49 52 56 57 70 76 78 79
69      04/20/1995,03 09 13 16 19 21 22 26 28 34 36 39 44 54 57 59 65 66 69 79
70      04/19/1995,08 10 13 14 18 20 28 33 35 36 45 46 50 55 64 65 70 71 76 80
71      04/18/1995,02 04 07 09 12 16 17 18 19 28 35 38 40 53 54 57 58 60 72 80
72      04/17/1995,08 09 13 15 16 21 25 29 30 33 37 38 39 40 41 46 50 58 63 72
73      04/16/1995,09 10 13 17 22 35 39 41 44 45 47 51 59 60 61 62 63 68 71 76
74      04/15/1995,03 04 08 10 15 19 20 25 36 37 49 50 51 53 54 62 68 74 77 79
75      04/14/1995,06 13 18 24 25 26 45 47 49 52 53 54 56 61 62 64 67 73 74 77
76      04/13/1995,05 06 08 12 19 22 24 32 40 41 42 45 47 52 64 65 72 73 74 78
77      04/12/1995,01 16 21 27 30 33 35 36 38 41 43 46 48 51 64 67 71 76 78 80
78      04/11/1995,02 13 17 18 23 24 25 34 36 38 40 45 46 48 50 53 59 60 65 80
79      04/10/1995,02 09 12 14 17 20 23 27 29 33 34 44 51 61 62 64 67 68 69 74
80      04/09/1995,04 13 14 21 23 29 37 38 43 45 55 56 58 63 68 70 71 72 73 78
81      04/08/1995,06 08 12 15 17 21 25 26 31 32 35 51 52 56 64 66 68 71 76 78
82      04/07/1995,04 07 08 13 19 21 24 29 35 42 44 48 49 53 54 62 65 68 71 76
83      04/06/1995,01 05 08 10 11 20 22 27 28 40 43 50 51 52 60 63 64 71 75 79
84      04/05/1995,05 08 10 12 21 27 29 32 33 41 43 44 45 50 55 57 59 71 72 73
85      04/04/1995,01 02 04 12 19 21 27 32 33 40 44 48 49 53 56 59 63 71 77 79
86      04/03/1995,01 03 10 11 14 16 21 26 27 29 30 31 33 49 55 57 58 60 69 72
87      04/02/1995,01 02 06 08 09 16 20 22 28 34 38 40 44 45 50 56 62 67 71 78
88      04/01/1995,11 18 21 24 26 34 36 47 49 50 55 67 68 69 70 73 75 76 77 78
89      03/31/1995,15 16 17 18 20 21 23 28 29 30 33 34 38 52 60 61 62 69 71 72
90      03/30/1995,01 02 03 09 10 14 21 33 45 49 50 51 53 54 58 60 68 73 75 77
91      03/29/1995,04 05 10 16 21 28 29 34 36 37 38 41 43 46 47 52 54 64 75 79
92      03/28/1995,02 03 04 06 09 10 17 19 28 33 38 45 47 52 55 64 73 74 75 79
93      03/27/1995,04 10 11 22 25 28 31 37 38 40 47 53 57 60 62 65 68 70 72 77
94      03/26/1995,05 11 13 15 21 25 26 28 36 37 39 41 50 51 60 61 65 69 72 76
95      03/25/1995,03 04 14 18 25 26 27 30 31 36 46 56 60 62 68 71 73 74 75 80
96      03/24/1995,01 04 08 09 14 15 27 29 30 34 36 41 46 58 62 65 68 69 70 78
97      03/23/1995,02 07 13 16 18 19 30 35 39 41 51 54 58 64 67 68 69 75 76 77
98      03/22/1995,10 14 20 25 26 28 33 34 35 42 45 52 59 61 66 70 71 76 79 80
99      03/21/1995,03 22 23 25 28 31 37 41 42 43 45 48 50 56 58 63 68 73 75 79
00      03/20/1995,05 10 11 19 21 26 31 36 38 39 41 47 49 51 52 53 59 62 76 79
01      03/19/1995,02 03 10 15 17 18 23 26 35 38 39 43 48 56 60 64 71 76 78 79
02      03/18/1995,02 05 11 12 13 19 23 25 30 34 37 43 45 50 53 63 69 72 74 78
03      03/17/1995,04 13 14 20 22 27 28 36 39 43 46 54 56 57 60 61 65 67 71 76
04      03/16/1995,03 06 07 11 17 18 20 23 29 31 34 46 59 62 64 68 72 76 77 80
05      03/15/1995,01 07 08 09 17 19 26 28 33 37 44 47 51 55 57 59 68 69 73 78
06      03/14/1995,02 05 09 13 15 16 18 20 25 32 34 41 48 49 54 69 72 74 75 80
07      03/13/1995,06 09 10 14 29 30 32 37 38 40 41 43 50 51 52 60 64 65 66 73
08      03/12/1995,10 11 20 23 27 31 35 36 40 43 49 52 55 61 66 69 70 72 77 80
09      03/11/1995,04 06 07 14 16 23 30 32 37 39 41 50 59 60 62 63 71 72 74 76
10      03/10/1995,01 03 09 10 14 15 18 22 24 27 33 39 44 52 57 58 61 65 68 74
11      03/09/1995,11 36 38 39 41 42 43 45 46 49 51 53 55 56 57 66 68 72 75 80
12      03/08/1995,04 08 12 19 20 31 37 38 39 45 46 47 48 51 59 64 67 70 74 79
13      03/07/1995,04 05 18 20 27 30 31 33 38 44 47 48 51 53 55 56 59 60 69 76
14      03/06/1995,03 05 14 23 31 32 36 37 40 41 43 52 60 64 65 69 73 74 76 79
15      03/05/1995,02 04 06 25 29 30 33 36 38 39 42 47 48 49 56 59 61 73 76 79
16      03/04/1995,03 04 05 06 07 09 13 22 24 25 30 35 36 38 45 46 61 73 76 79
17      03/03/1995,03 04 11 13 25 26 27 28 31 34 47 54 55 57 59 61 69 70 73 80
18      03/02/1995,01 02 03 06 07 13 14 15 18 23 24 25 39 51 52 61 62 64 68 79
```

```
19      03/01/1995,03 06 07 10 20 29 30 31 40 44 49 53 56 57 68 69 71 72 73 80
20      02/28/1995,05 06 13 16 19 23 28 29 33 37 42 45 46 48 54 57 66 69 71 80
21      02/27/1995,01 05 12 13 14 17 20 23 27 31 32 33 35 37 45 46 51 59 64 77
22      02/26/1995,01 02 03 04 11 17 20 31 33 34 36 38 53 57 59 60 64 65 67 76
23      02/25/1995,04 07 15 16 18 21 22 23 26 28 30 39 40 51 55 57 63 66 73 75
24      02/24/1995,01 07 09 12 17 19 20 26 34 38 42 45 46 59 60 66 68 71 72 77
25      02/23/1995,04 05 06 07 09 10 13 15 16 26 28 33 44 48 50 51 58 68 73 74
26      02/22/1995,06 10 19 21 25 31 46 49 50 51 57 58 61 64 70 74 75 77 79 80
27      02/21/1995,02 04 08 13 14 23 28 33 36 41 51 54 58 59 61 66 67 72 74 76
28      02/20/1995,06 14 17 18 23 26 27 44 45 47 48 51 52 53 60 61 66 72 73 80
29      02/19/1995,03 07 08 09 11 17 23 25 26 34 35 41 45 46 47 53 54 55 66 73
30      02/18/1995,06 10 12 17 18 22 24 28 29 34 36 45 51 53 59 60 61 64 72 80
31      02/17/1995,01 03 04 07 10 15 19 21 28 29 34 36 39 47 52 57 68 69 72 76
32      02/16/1995,12 13 22 24 25 34 36 39 41 42 43 48 50 56 58 61 67 71 74 76
33      02/15/1995,12 13 15 16 18 32 39 43 46 49 51 53 54 56 65 66 67 68 72 74
34      02/14/1995,03 07 15 18 21 23 24 26 45 48 53 61 62 64 65 66 70 76 77 78
35      02/13/1995,01 03 04 07 09 12 21 23 30 37 39 44 47 48 53 58 59 60 62 79
36      02/12/1995,05 08 09 13 17 19 22 32 44 51 53 55 60 61 63 66 69 70 71 74
37      02/11/1995,01 02 03 07 22 27 28 30 40 44 51 53 59 61 62 66 68 71 76 77
38      02/10/1995,01 06 08 13 15 21 27 30 39 41 42 45 46 51 56 61 64 66 70 74
39      02/09/1995,02 03 04 11 12 13 14 15 21 35 36 38 47 56 57 63 65 66 69 73
40      02/08/1995,01 03 15 20 30 36 37 39 41 45 53 58 59 64 66 68 72 74 75 79
41      02/07/1995,06 11 17 25 28 29 30 33 38 41 42 52 53 58 62 66 69 70 71 75
42      02/06/1995,01 05 06 11 13 16 18 21 23 26 27 33 47 55 58 59 64 66 76 79
43      02/05/1995,02 07 10 12 14 15 17 20 21 36 41 46 49 54 55 56 64 65 66 70
44      02/04/1995,01 05 06 10 13 22 32 33 34 35 36 37 41 47 48 60 61 72 75 76
45      02/03/1995,01 02 03 04 08 10 11 15 16 20 25 37 41 42 45 48 50 56 58 75
46      02/02/1995,04 06 12 13 18 20 21 25 29 30 32 45 57 58 59 63 72 75 77 79
47      02/01/1995,02 03 10 11 12 23 29 42 43 48 49 52 58 59 68 69 71 73 77 79
48      01/31/1995,05 09 15 16 17 24 33 34 36 37 39 42 47 53 62 65 68 70 77 78
49      01/30/1995,01 04 08 14 17 22 25 35 37 38 39 42 49 52 53 55 60 62 71 77
50      01/29/1995,16 19 21 23 27 32 35 36 37 39 42 45 49 50 51 54 56 60 68 71
51      01/28/1995,01 07 13 16 17 18 25 26 27 28 29 31 45 46 51 55 56 72 74 77
52      01/27/1995,02 06 18 21 24 34 35 36 37 38 39 43 49 56 58 59 60 62 76 79
53      01/26/1995,01 05 10 11 16 19 22 24 29 31 36 37 44 47 50 56 67 69 70 76
54      01/25/1995,01 02 03 10 12 15 17 22 28 33 50 54 58 59 61 63 68 73 76 78
55      01/24/1995,01 02 05 09 11 14 21 35 37 38 40 43 45 49 56 57 66 67 76 77
56      01/23/1995,03 04 07 08 18 19 22 34 36 37 38 47 49 51 53 56 61 73 76 78
57      01/22/1995,01 02 04 10 12 13 29 31 33 42 44 46 47 54 57 63 70 71 72 76
58      01/21/1995,02 03 11 13 14 20 27 32 34 38 40 43 45 51 52 58 64 69 70 80
59      01/20/1995,06 07 13 17 26 38 44 53 54 55 56 57 61 64 68 71 76 77 78 79
60      01/19/1995,05 06 07 08 09 10 11 12 22 27 35 40 42 47 48 49 56 66 70 72
61      01/18/1995,05 08 10 17 24 28 33 39 47 48 52 53 57 59 63 65 72 75 77 78
62      01/17/1995,05 12 15 16 19 26 27 28 29 30 37 41 47 49 50 57 59 67 68 74
63      01/16/1995,03 06 07 11 14 19 24 27 29 32 34 36 44 51 54 58 59 61 70 73
64      01/15/1995,05 07 12 13 14 16 18 36 48 49 51 54 57 65 67 68 69 72 73 79
65      01/14/1995,04 06 08 11 16 17 19 27 29 34 35 39 51 53 60 62 63 66 71 72
66      01/13/1995,03 11 17 21 27 29 30 31 32 38 46 57 58 60 63 69 70 71 72 73
67      01/12/1995,02 03 04 08 14 19 33 36 41 43 51 54 56 59 65 69 70 72 77 80
68      01/11/1995,01 14 16 17 20 24 26 28 34 37 40 42 44 55 60 61 65 73 75 80
69      01/10/1995,04 07 09 12 21 22 23 39 40 43 45 49 51 53 55 56 69 71 76 78
70      01/09/1995,01 04 14 15 16 19 21 23 27 32 39 40 42 50 51 52 55 59 69 80
71      01/08/1995,10 12 13 14 15 33 39 44 45 46 47 51 56 58 60 64 66 72 79 80
72      01/07/1995,01 04 07 09 11 12 16 20 21 23 26 29 30 37 43 46 56 66 78 80
```

```
73      01/06/1995,11 17 20 26 27 28 32 35 36 39 42 47 57 58 60 64 74 75 77 78
74      01/05/1995,10 14 19 23 25 29 33 35 37 41 43 50 53 58 60 61 72 75 77 79
75      01/04/1995,01 10 17 22 25 26 33 35 37 38 45 46 49 60 62 65 68 70 72 79
76      01/03/1995,08 10 16 24 25 27 32 34 36 40 44 45 50 52 53 54 60 63 64 70
77      01/02/1995,03 05 10 11 14 16 17 28 29 30 33 35 38 41 56 57 63 69 74 80
78      01/01/1995,04 11 14 20 21 22 24 27 29 33 36 38 39 46 50 54 57 58 74 77
79      12/31/1994,13 18 25 27 29 30 35 38 39 40 42 51 52 53 57 61 63 64 70 79
80      12/30/1994,03 07 08 11 12 14 15 26 27 31 35 39 45 47 51 55 72 75 78 79
81      12/29/1994,01 02 03 05 10 12 23 32 34 37 40 44 53 55 57 58 63 64 68 75
82      12/28/1994,03 05 08 09 14 16 17 19 20 28 31 48 52 55 56 57 58 65 73 76
83      12/27/1994,01 09 14 18 20 24 26 30 33 36 38 50 53 56 60 62 66 67 68 76
84      12/26/1994,03 10 11 15 19 23 25 26 30 35 41 45 50 60 62 64 67 68 71 74
85      12/24/1994,12 17 21 22 25 29 30 31 33 37 39 43 51 52 53 54 56 57 79 80
86      12/23/1994,02 03 05 08 09 10 13 15 33 45 46 56 58 61 63 66 73 76 77 79
87      12/22/1994,03 12 16 25 32 34 35 40 42 45 46 47 49 52 59 61 63 72 73 76
88      12/21/1994,02 12 18 21 22 29 32 43 46 47 48 55 60 64 68 72 74 76 77 78
89      12/20/1994,03 06 10 17 19 28 29 35 41 43 45 47 50 51 54 57 61 66 70 77
90      12/19/1994,02 07 19 30 34 35 38 40 46 47 49 54 55 57 62 65 67 75 77 79
91      12/18/1994,01 03 08 25 26 28 34 35 37 39 40 41 42 44 46 48 50 62 66 76
92      12/17/1994,04 11 17 23 26 27 29 30 34 36 37 42 57 60 61 66 71 72 74 80
93      12/16/1994,02 03 05 06 14 26 33 36 41 50 57 61 62 68 69 72 73 75 76 77
94      12/15/1994,02 05 07 16 18 21 23 24 33 45 47 53 55 57 58 64 66 70 72 75
95      12/14/1994,09 13 14 16 18 21 24 26 30 33 40 42 46 53 61 62 68 76 77 79
96      12/13/1994,04 18 21 26 27 28 35 37 40 47 49 50 62 65 68 69 73 76 79 80
97      12/12/1994,03 14 16 18 19 27 32 35 36 39 41 43 46 52 58 67 71 78 79 80
98      12/11/1994,02 17 19 20 24 36 41 42 44 45 47 53 54 58 62 65 66 74 76 78
99      12/10/1994,02 03 04 06 17 37 39 41 43 49 51 52 55 58 60 62 65 68 74 79
00      12/09/1994,02 08 12 15 22 25 33 34 35 41 46 51 56 59 60 65 67 70 77 80
01      12/08/1994,09 11 12 13 19 36 37 40 41 42 43 46 48 54 68 69 70 72 79 80
02      12/07/1994,11 20 25 31 36 39 46 47 49 51 52 53 59 62 63 67 68 72 73 80
03      12/06/1994,02 08 11 15 17 21 25 29 30 32 37 41 52 54 56 61 62 64 79 80
04      12/05/1994,05 13 18 20 22 26 30 32 37 40 46 47 49 51 53 56 64 67 77 79
05      12/04/1994,03 04 08 12 13 15 19 24 26 27 31 32 36 40 42 49 57 62 70 73
06      12/03/1994,05 06 09 11 14 22 30 33 35 42 43 44 50 53 55 62 71 72 73 79
07      12/02/1994,01 04 09 13 22 26 27 44 50 55 59 61 62 68 74 75 76 77 79 80
08      12/01/1994,05 06 10 15 19 20 27 30 35 45 48 49 50 52 55 62 63 68 78 80
09      11/30/1994,02 07 11 15 17 18 26 28 30 37 38 41 44 58 66 71 73 75 78 80
10      11/29/1994,03 06 09 13 15 17 19 23 24 25 27 33 48 51 69 71 73 75 77 78
11      11/28/1994,07 10 13 14 19 21 29 30 35 42 47 48 51 52 53 59 60 66 68 74
12      11/27/1994,05 06 09 15 17 22 24 25 39 40 41 42 44 45 48 67 69 71 75 76
13      11/26/1994,06 08 15 21 23 26 27 34 35 41 44 46 47 53 54 55 63 66 76 77
14      11/25/1994,03 04 06 07 15 20 22 23 27 28 29 33 34 36 38 41 67 72 74 77
15      11/24/1994,01 03 04 05 22 24 30 33 40 43 45 46 51 58 65 67 69 72 78 79
16      11/23/1994,06 07 08 11 15 16 19 28 30 53 60 61 62 63 66 67 70 73 78 79
17      11/22/1994,02 05 06 12 17 28 33 34 40 41 44 46 53 57 60 62 69 71 75 79
18      11/21/1994,01 02 15 18 20 22 23 31 32 35 36 41 43 49 51 57 60 62 63 79
19      11/20/1994,01 04 07 10 13 24 27 36 37 51 53 56 57 60 62 66 68 69 70 78
20      11/19/1994,05 13 14 18 20 23 27 29 30 34 52 58 63 66 68 70 71 72 74 76
21      11/18/1994,05 23 27 32 34 36 38 41 44 50 59 60 61 64 65 66 71 74 75 79
22      11/17/1994,04 07 11 23 26 27 28 32 33 36 43 45 47 50 52 63 68 70 71 78
23      11/16/1994,03 04 05 13 19 24 28 41 42 46 50 54 55 63 65 66 68 73 76 78
24      11/15/1994,02 03 04 12 13 17 22 29 32 39 47 49 57 63 64 65 69 70 73 78
25      11/14/1994,09 11 13 20 25 26 29 32 35 40 41 42 49 50 52 62 67 70 73 80
26      11/13/1994,12 13 15 18 20 23 25 27 31 33 34 43 48 50 52 70 71 72 75 77
```

```
11/12/1994,02 05 10 11 17 19 23 24 25 38 43 45 53 54 63 67 69 75 76 78
11/11/1994,03 08 10 15 16 17 18 26 29 30 31 40 42 45 51 52 53 54 72 80
11/10/1994,03 08 28 30 34 35 39 43 47 48 53 59 60 65 68 75 76 77 78 79
11/09/1994,01 04 07 10 13 19 21 25 34 40 45 51 52 56 57 65 73 74 79 80
11/08/1994,08 12 15 17 18 21 22 33 34 35 40 49 50 54 57 60 62 71 72 77
11/07/1994,04 12 13 15 16 18 24 35 38 39 41 42 44 53 57 58 60 72 78 79
11/06/1994,01 02 05 06 13 15 17 20 25 29 33 35 40 41 49 52 67 72 74 75
11/05/1994,03 08 12 15 16 17 18 28 31 35 37 45 51 52 57 60 64 75 78 80
11/04/1994,01 03 09 10 11 12 13 16 17 23 26 34 35 37 47 53 63 69 70 79
11/03/1994,10 13 22 24 25 37 45 46 52 54 57 58 67 70 71 72 73 74 76 78
11/02/1994,03 13 18 25 26 27 40 41 45 52 53 56 57 62 65 69 73 75 76 79
11/01/1994,03 08 10 14 21 24 31 35 40 43 50 51 58 61 68 70 74 77 78 80
10/31/1994,06 07 09 14 21 29 32 35 38 48 50 51 55 58 59 68 69 70 76 77
10/30/1994,09 11 12 16 19 21 26 27 31 32 50 60 61 62 65 67 71 74 77 78
10/29/1994,01 03 04 05 24 36 38 39 47 52 53 57 58 59 62 67 68 69 75 78
10/28/1994,06 12 14 16 17 22 23 30 46 48 52 53 57 62 70 71 75 76 79 80
10/27/1994,02 03 04 07 09 11 23 29 31 32 34 38 42 49 50 59 61 62 70 79
10/26/1994,10 12 13 15 16 18 26 28 31 33 38 42 44 47 52 55 57 60 76 77
10/25/1994,01 03 06 11 12 17 20 26 34 35 40 43 62 64 69 72 73 74 75 77
10/24/1994,01 09 10 12 22 23 31 34 36 37 51 59 61 63 64 66 70 75 78 79
10/23/1994,10 12 20 23 26 27 29 30 34 37 38 53 59 60 63 66 69 71 73 78
10/22/1994,02 06 08 15 19 20 30 31 36 39 42 52 55 56 57 61 63 64 77 78
10/21/1994,03 14 22 26 37 40 41 42 46 47 53 59 61 62 64 65 69 71 76 80
10/20/1994,03 08 11 15 27 31 34 35 39 40 47 50 52 58 61 62 66 68 76 77
10/19/1994,03 04 09 19 23 24 26 29 31 37 49 50 51 53 58 61 64 68 73 75
10/18/1994,03 05 08 09 16 23 28 35 37 39 41 45 52 53 58 64 66 71 74 75
10/17/1994,03 09 25 26 30 36 49 53 55 56 57 60 61 62 66 69 73 74 75 77
10/16/1994,16 18 19 23 24 27 28 32 39 40 47 52 53 58 60 61 67 69 71 74
10/15/1994,05 11 16 18 19 23 25 26 31 32 43 49 53 56 58 66 67 70 71 80
10/14/1994,06 11 13 16 18 24 27 31 39 40 43 44 49 53 55 56 60 74 75 78
10/13/1994,01 03 13 18 19 27 29 31 33 34 36 42 48 52 57 58 59 60 64 65
10/12/1994,05 06 07 12 18 20 31 34 36 40 42 45 46 50 59 62 66 71 72 80
10/11/1994,04 12 15 20 23 31 33 34 35 43 46 50 53 54 61 62 69 71 75 76
10/10/1994,07 08 11 15 19 22 24 27 30 31 33 47 48 51 54 63 66 68 70 74
10/09/1994,05 10 14 16 19 22 24 26 28 31 35 36 39 42 49 53 54 60 62 80
10/08/1994,06 07 11 20 26 27 30 31 38 39 44 47 69 71 72 73 75 76 77 79
10/07/1994,03 06 09 14 18 19 21 22 31 36 38 39 40 48 51 52 56 59 63 64
10/06/1994,05 06 14 19 24 25 28 29 34 36 37 38 53 54 55 60 61 63 72 79
10/05/1994,06 08 15 18 20 21 24 37 39 40 42 43 47 48 50 55 58 63 65 66
10/04/1994,01 06 08 10 16 26 27 28 30 42 43 52 57 59 64 65 66 77 79 80
10/03/1994,06 10 11 14 16 35 40 41 43 46 49 51 58 65 69 73 74 75 76 80
10/02/1994,03 06 08 13 14 19 22 23 26 33 35 36 42 43 45 49 54 64 65 70
10/01/1994,03 08 12 20 25 34 35 43 46 53 57 60 61 62 66 67 68 69 72 77
09/30/1994,01 04 05 09 17 20 26 27 34 39 42 45 46 49 53 60 65 73 79 80
09/29/1994,06 07 12 13 14 22 34 35 41 47 48 49 52 58 60 65 69 70 71 78
09/28/1994,04 05 08 16 17 23 28 30 31 32 33 35 41 43 49 57 62 66 71 75
09/27/1994,01 03 04 07 10 14 17 18 19 25 32 35 36 38 43 48 59 63 67 76
09/26/1994,01 06 10 13 16 17 21 22 26 27 29 37 48 51 52 56 63 76 77 78
09/25/1994,05 06 11 15 19 22 27 30 42 44 49 51 52 57 58 64 72 74 77 80
09/24/1994,04 07 11 17 18 37 39 43 44 46 48 51 56 59 61 67 73 74 76 80
09/23/1994,02 04 10 16 18 29 30 31 32 36 42 44 48 55 61 62 67 68 72 78
09/22/1994,09 10 13 14 15 18 33 41 43 44 50 51 55 57 63 67 71 74 75 76
09/21/1994,04 05 06 08 16 17 21 30 33 37 44 45 46 52 59 61 62 70 75 78
09/20/1994,10 14 18 21 23 25 30 31 35 40 47 49 53 58 64 67 70 71 76 77
```

```
81      09/19/1994,08 09 13 23 26 30 33 39 42 45 52 54 56 57 58 66 71 72 74 76
82      09/18/1994,11 17 20 21 37 38 39 41 44 48 49 52 54 58 60 64 66 69 70 80
83      09/17/1994,03 04 12 15 18 26 27 31 37 38 39 48 49 51 52 57 58 61 63 71
84      09/16/1994,03 11 13 14 21 23 24 29 31 32 41 42 45 52 61 69 70 72 73 78
85      09/15/1994,04 06 10 12 21 24 31 32 35 36 37 42 56 63 68 69 73 76 77 78
86      09/14/1994,04 12 15 19 29 32 37 42 43 44 45 46 48 50 64 67 71 73 74 75
87      09/13/1994,12 17 20 23 25 29 31 32 35 37 42 43 45 48 55 56 74 75 78 79
88      09/12/1994,11 12 14 24 33 34 36 41 42 49 50 51 53 56 59 67 68 72 74 80
89      09/11/1994,10 11 14 18 19 25 27 28 31 33 41 42 51 53 55 59 63 65 67 75
90      09/10/1994,02 12 14 15 16 17 19 20 21 24 45 49 51 53 55 64 66 77 79 80
91      09/09/1994,04 17 26 27 28 36 41 44 48 49 52 53 55 60 61 65 66 69 78 79
92      09/08/1994,03 11 12 13 15 17 20 24 42 46 56 59 60 64 65 68 71 72 74 79
93      09/07/1994,02 07 08 19 21 26 28 29 32 33 38 46 49 51 52 58 62 71 79 80
94      09/06/1994,02 08 13 16 19 20 22 24 30 32 33 39 43 44 51 57 66 67 75 76
95      09/05/1994,06 09 10 13 28 33 34 46 47 48 51 53 54 60 62 65 68 71 72 78
96      09/04/1994,01 09 16 19 25 27 29 35 37 40 44 49 50 58 60 65 66 69 72 77
97      09/03/1994,01 12 17 28 29 30 34 37 44 45 52 53 56 58 60 64 66 70 76 78
98      09/02/1994,14 17 22 26 29 32 33 37 38 41 58 61 62 66 67 71 72 75 77 78
99      09/01/1994,04 07 08 13 16 17 18 20 22 26 29 35 36 38 50 68 72 76 77 78
00      08/31/1994,02 03 05 08 11 18 21 28 29 36 39 41 45 52 63 64 66 74 76 78
01      08/30/1994,04 12 16 17 18 24 26 28 30 32 38 42 49 55 59 70 71 74 75 80
02      08/29/1994,01 03 06 08 09 12 16 26 27 39 42 52 53 55 64 71 73 75 76 77
03      08/28/1994,06 07 11 13 14 16 26 28 30 32 37 39 43 47 57 58 66 68 79 80
04      08/27/1994,01 02 06 13 26 27 29 36 38 45 46 51 52 53 56 57 67 68 76 80
05      08/26/1994,05 07 12 17 22 24 26 29 32 33 34 46 51 60 62 65 70 71 79 80
06      08/25/1994,01 08 13 14 17 20 23 29 32 41 42 43 45 48 57 58 63 69 72 79
07      08/24/1994,01 04 12 13 14 16 26 31 35 37 45 57 58 59 63 67 73 74 77 80
08      08/23/1994,03 04 06 10 13 14 16 19 26 30 33 35 39 45 48 54 55 57 68 76
09      08/22/1994,01 03 05 30 31 32 40 41 42 49 52 53 56 57 59 65 67 71 77 78
10      08/21/1994,05 06 08 12 15 20 21 27 31 35 41 54 55 58 63 69 70 72 73 80
11      08/20/1994,04 07 09 10 13 15 20 23 28 30 34 37 39 41 42 54 55 56 64 66
12      08/19/1994,01 03 08 10 19 23 27 32 33 36 37 39 43 45 50 56 69 70 79 80
13      08/18/1994,01 05 08 13 19 23 24 29 30 37 41 51 52 57 59 60 65 71 72 79
14      08/17/1994,05 08 09 12 20 21 29 31 36 38 46 51 58 59 60 63 64 67 71 75
15      08/16/1994,06 11 19 22 24 25 27 29 33 34 36 40 47 51 53 56 63 70 71 79
16      08/15/1994,12 16 18 20 24 29 31 34 35 40 41 42 43 54 55 59 62 64 78 80
17      08/14/1994,04 05 06 09 14 16 25 26 31 48 51 53 54 58 60 62 64 68 73 74
18      08/13/1994,03 06 09 15 19 21 22 24 30 43 52 53 54 57 60 62 63 68 77 79
19      08/12/1994,05 07 10 14 15 23 24 28 30 38 39 40 42 56 57 59 62 71 79 80
20      08/11/1994,01 02 04 07 14 17 22 33 36 45 46 49 58 59 60 67 68 71 73 74
21      08/10/1994,09 12 17 22 23 24 34 36 39 45 46 47 50 54 66 67 69 74 77 80
22      08/09/1994,02 07 08 15 17 21 26 33 42 44 45 49 52 56 59 64 69 72 77 78
23      08/08/1994,01 02 04 05 07 11 19 25 27 39 49 61 64 66 67 69 72 74 75 76
24      08/07/1994,05 08 09 10 11 12 25 26 27 29 30 32 36 37 40 42 49 52 74 77
25      08/06/1994,04 07 09 17 24 39 40 42 47 53 57 58 60 64 65 66 67 72 73 79
26      08/05/1994,01 02 03 08 09 11 12 17 21 24 25 26 27 38 52 54 59 64 73 75
27      08/04/1994,01 04 05 18 19 26 28 30 38 46 50 53 55 56 58 63 69 74 76 77
28      08/03/1994,13 15 19 22 23 29 32 37 39 40 45 48 49 53 56 58 59 72 77 79
29      08/02/1994,01 04 11 15 19 22 23 27 34 36 42 44 45 49 54 55 57 64 65 78
30      08/01/1994,03 05 07 10 16 17 23 29 33 34 35 37 38 48 51 59 67 75 79 80
31      07/31/1994,03 04 07 12 14 20 28 29 32 37 38 42 43 49 52 59 65 69 70 72
32      07/30/1994,05 12 13 20 21 23 25 27 28 37 46 52 53 57 66 71 73 77 78 79
33      07/29/1994,05 06 08 19 25 26 29 32 34 37 42 44 45 50 53 57 60 66 69 75
34      07/28/1994,03 08 11 13 26 29 33 34 35 36 43 48 55 62 64 66 71 72 75 76
```

```
07/27/1994,02 04 05 07 16 26 31 36 39 40 45 51 58 59 62 63 70 71 72 76
07/26/1994,01 14 17 20 23 28 35 36 37 38 48 50 60 61 62 64 71 74 76 77
07/25/1994,07 10 16 17 18 21 30 34 42 43 50 51 54 61 62 63 67 69 70 73
07/24/1994,04 08 10 14 15 18 25 27 29 38 39 40 43 49 63 64 69 72 74 79
07/23/1994,04 07 09 13 14 18 21 27 28 30 35 39 43 47 54 58 62 68 72 78
07/22/1994,04 07 08 11 15 16 17 23 34 35 37 43 47 48 53 54 57 60 70 76
07/21/1994,03 06 09 10 15 21 23 27 28 29 41 47 48 54 55 59 68 71 77 80
07/20/1994,03 07 08 11 23 32 39 42 43 45 49 51 52 54 56 61 67 70 73 78
07/19/1994,04 08 09 10 12 15 24 28 37 44 45 50 51 52 56 58 60 66 75 76
07/18/1994,02 06 11 12 18 19 25 33 37 39 44 45 46 51 53 62 63 69 72 79
07/17/1994,01 03 06 11 13 14 18 24 25 26 31 35 36 46 50 52 57 60 63 75
07/16/1994,04 12 16 17 19 20 27 30 33 38 42 56 60 61 66 67 71 73 74 77
07/15/1994,02 05 07 09 12 16 27 31 38 47 48 55 59 60 61 62 63 66 67 69
07/14/1994,03 05 14 17 21 25 30 31 32 38 39 43 45 52 53 67 69 71 73 74
07/13/1994,03 05 06 30 33 35 43 45 48 49 51 55 60 62 66 69 75 76 77 80
07/12/1994,09 11 12 18 19 25 31 32 38 39 41 47 54 57 70 71 74 75 76 80
07/11/1994,03 06 08 09 23 25 28 32 35 36 43 45 46 47 53 56 64 67 73 76
07/10/1994,03 05 10 11 16 17 23 28 30 34 44 46 57 59 61 67 70 71 74 78
07/09/1994,07 08 09 10 19 20 22 28 34 41 51 55 56 58 59 62 63 65 67 79
07/08/1994,02 09 11 14 15 17 19 20 23 25 28 33 34 37 45 46 50 64 76 80
07/07/1994,04 13 16 17 19 29 30 32 34 35 40 56 63 65 66 68 74 76 77 78
07/06/1994,02 04 07 17 18 27 34 37 38 41 53 54 57 62 64 66 72 77 79 80
07/05/1994,12 24 26 28 38 41 42 45 46 47 48 53 58 59 68 69 75 77 79 80
07/04/1994,02 04 06 09 13 17 19 22 27 29 35 38 42 43 48 54 55 63 72 74
07/03/1994,01 03 06 18 20 21 22 24 27 28 30 32 42 50 56 58 62 75 78 79
07/02/1994,03 04 17 20 21 24 25 26 27 34 35 38 44 56 60 64 67 69 74 78
07/01/1994,12 13 14 23 27 33 35 38 45 46 47 59 60 61 62 63 69 73 74 78
06/30/1994,05 15 16 25 27 29 30 34 36 39 40 43 46 47 51 52 73 74 79 80
06/29/1994,07 14 19 20 23 25 26 28 34 44 47 50 51 53 59 60 63 64 73 78
06/28/1994,03 07 08 09 20 25 26 28 32 36 39 42 45 48 57 61 67 75 78 80
06/27/1994,03 07 16 27 29 30 40 45 47 49 56 62 63 67 68 71 74 75 77 79
06/26/1994,03 05 08 09 11 13 25 26 29 32 36 47 51 54 61 62 67 71 75 78
06/25/1994,01 02 03 11 17 20 21 22 27 28 29 33 42 44 51 53 54 68 69 78
06/24/1994,03 10 24 28 32 37 38 39 40 41 42 48 49 50 59 61 62 68 73 74
06/23/1994,07 10 11 22 23 28 38 40 41 42 48 62 63 65 66 68 71 74 79 80
06/22/1994,04 06 08 09 13 14 17 18 20 34 36 41 42 43 50 57 64 69 73 78
06/21/1994,10 26 32 37 38 39 40 45 46 48 51 53 55 62 64 65 66 69 70 80
06/20/1994,06 14 15 22 28 29 40 42 46 49 50 52 58 62 63 64 66 70 76 80
06/19/1994,05 09 14 18 21 24 26 27 29 32 33 37 41 43 49 50 53 57 74 76
06/18/1994,02 06 09 11 17 20 24 26 29 33 37 47 48 54 65 69 70 71 72 80
06/17/1994,01 08 12 14 15 17 20 24 25 35 37 40 41 47 49 56 63 68 76 80
06/16/1994,05 11 12 14 20 22 27 29 31 35 41 43 50 61 66 70 71 74 77 80
06/15/1994,01 16 17 18 23 31 36 37 39 44 53 55 57 58 59 68 70 75 76 78
06/14/1994,14 18 25 28 29 31 32 36 43 49 52 53 55 58 61 64 65 69 75 76
06/13/1994,01 08 09 11 13 23 25 29 38 39 41 50 52 53 55 60 62 66 72 74
06/12/1994,05 07 11 15 17 19 26 29 30 31 37 42 47 54 65 68 71 74 78 80
06/11/1994,05 10 14 18 24 25 26 29 36 37 42 45 47 48 50 52 54 63 64 75
06/10/1994,04 11 13 14 15 26 27 30 32 38 46 48 49 51 60 61 64 65 73 80
06/09/1994,09 10 20 21 23 24 32 33 38 39 40 45 48 54 65 66 69 73 76 80
06/08/1994,06 09 16 25 26 27 31 35 36 42 45 47 52 54 64 66 67 68 75 78
06/07/1994,06 12 13 21 22 24 31 35 36 40 43 52 55 56 59 61 63 68 70 80
06/06/1994,01 03 08 10 11 14 17 20 21 22 30 36 40 48 55 62 66 73 74 80
06/05/1994,05 10 28 30 32 38 39 43 46 53 54 59 60 61 63 66 67 68 69 78
06/04/1994,02 07 14 15 20 22 23 31 34 38 40 44 50 52 55 58 60 61 62 76
```

```
89      06/03/1994,02 05 06 08 12 13 23 26 27 28 31 35 41 44 49 50 54 58 63 64
90      06/02/1994,14 20 22 23 24 28 33 34 35 36 43 44 48 51 52 55 57 60 62 69
91      06/01/1994,02 10 11 18 26 31 34 41 51 53 56 59 60 64 68 70 73 74 75 77
92      05/31/1994,02 04 09 11 13 20 27 28 34 38 40 42 59 60 64 66 71 73 74 76
93      05/30/1994,02 05 11 21 30 34 41 43 48 55 56 57 58 59 62 65 67 72 74 77
94      05/29/1994,02 06 11 18 19 24 31 36 37 46 47 51 58 60 61 63 65 67 74 79
95      05/28/1994,05 07 08 11 22 34 40 41 45 46 47 49 51 54 56 64 65 69 70 74
96      05/27/1994,02 05 10 19 21 29 34 36 39 42 43 44 51 56 60 66 68 69 78 80
97      05/26/1994,02 08 10 11 12 15 18 21 26 28 30 33 39 40 52 55 57 61 74 79
98      05/25/1994,05 15 16 19 20 26 29 31 33 36 42 48 54 56 58 59 60 76 78 79
99      05/24/1994,05 10 15 17 18 21 25 31 32 36 40 42 45 49 52 53 63 67 69 72
00      05/23/1994,02 04 06 10 15 21 23 29 33 40 46 54 56 61 62 64 66 75 79 80
01      05/22/1994,06 18 19 26 27 30 31 39 40 43 45 46 52 58 60 63 67 69 79 80
02      05/21/1994,09 12 14 16 23 24 26 29 31 34 49 51 53 61 62 63 66 68 72 79
03      05/20/1994,02 03 05 08 12 15 17 27 34 41 43 46 51 55 61 66 72 73 75 77
04      05/19/1994,05 06 07 08 11 16 19 28 37 45 47 49 52 53 54 60 66 67 71 79
05      05/18/1994,07 10 16 17 23 24 27 34 35 38 44 47 52 53 64 65 68 70 77 79
06      05/17/1994,06 09 11 12 15 21 32 35 39 48 49 52 53 64 67 73 76 78 79 80
07      05/16/1994,02 06 17 18 19 20 22 26 28 38 45 46 47 49 54 60 63 66 72 73
08      05/15/1994,01 02 04 08 14 20 31 33 36 37 45 48 59 60 65 71 75 77 79 80
09      05/14/1994,02 03 08 11 14 19 20 22 28 31 43 47 52 53 54 58 63 66 70 78
10      05/13/1994,07 11 13 16 18 19 23 31 34 35 37 46 54 60 61 65 66 69 77 79
11      05/12/1994,01 02 04 06 24 30 33 41 44 50 53 54 55 56 57 58 59 66 70 80
12      05/11/1994,06 09 17 29 30 35 37 38 42 44 47 51 53 54 56 57 59 65 71 72
13      05/10/1994,06 07 13 18 19 20 22 23 32 33 35 41 48 52 53 63 74 77 79 80
14      05/09/1994,01 05 07 11 23 30 33 39 40 49 50 51 54 59 64 68 71 74 75 79
15      05/08/1994,01 02 07 14 18 28 32 33 38 43 44 46 51 56 57 60 65 67 73 80
16      05/07/1994,04 07 09 11 17 22 29 36 43 47 49 50 52 59 60 69 71 74 79 80
17      05/06/1994,04 05 06 07 08 11 16 25 29 43 46 50 54 59 60 66 68 69 72 75
18      05/05/1994,09 10 14 20 21 22 25 35 37 43 45 46 50 58 62 69 73 78 79 80
19      05/04/1994,04 05 09 10 11 17 19 24 35 39 44 49 50 51 61 65 69 74 77 79
20      05/03/1994,06 23 26 33 34 38 39 41 42 44 46 47 51 54 58 60 69 75 77 78
21      05/02/1994,04 06 09 13 17 28 30 39 41 42 43 45 47 50 65 66 68 71 73 79
22      05/01/1994,01 05 09 11 14 15 17 18 19 20 24 25 27 29 36 38 54 56 66 67
23      04/30/1994,01 11 14 16 18 21 26 27 34 43 48 52 54 55 64 65 67 70 73 74
24      04/29/1994,06 08 11 12 13 14 20 28 31 42 43 48 50 56 58 59 63 67 68 78
25      04/28/1994,05 06 08 11 15 16 26 29 34 35 38 39 44 54 56 62 67 73 76 80
26      04/27/1994,03 08 17 19 22 31 33 36 38 42 44 45 47 52 56 61 62 72 73 77
27      04/26/1994,02 04 15 17 21 24 27 32 33 37 38 40 42 46 49 50 54 55 78 80
28      04/25/1994,01 03 07 12 16 18 20 21 23 24 38 39 43 44 45 50 68 69 71 76
29      04/24/1994,01 10 14 17 20 31 36 37 38 40 45 48 53 54 56 61 63 64 71 78
30      04/23/1994,04 06 09 11 12 16 17 25 29 30 31 34 35 36 42 43 60 71 78 80
31      04/22/1994,02 03 05 13 14 16 17 21 28 30 34 47 49 50 52 63 66 72 74 80
32      04/21/1994,10 12 13 14 16 23 25 28 30 31 32 33 40 44 48 49 53 54 67 69
33      04/20/1994,05 07 09 11 18 22 29 30 39 41 46 52 54 56 59 61 70 72 75 79
34      04/19/1994,06 08 11 14 15 17 26 27 28 29 34 39 44 46 50 53 56 58 61 64
35      04/18/1994,01 08 12 15 16 23 33 34 39 41 43 51 54 56 58 61 69 73 74 75
36      04/17/1994,03 09 13 21 22 27 38 40 42 46 52 54 56 61 63 67 69 70 76 77
37      04/16/1994,05 10 16 17 18 24 34 35 36 43 46 48 58 59 63 64 66 69 73 76
38      04/15/1994,08 09 14 16 23 25 31 33 35 37 40 45 46 48 53 57 59 62 64 67
39      04/14/1994,02 08 10 12 18 26 27 31 33 34 44 45 48 49 50 55 62 64 68 70
40      04/13/1994,02 08 09 10 13 20 21 25 28 39 41 43 44 58 59 62 71 72 73 76
41      04/12/1994,02 17 27 30 33 35 36 38 42 47 49 51 55 56 57 61 63 72 77 78
42      04/11/1994,01 04 05 12 14 19 22 27 28 35 41 42 45 55 56 59 64 66 74 75
```

```
43      04/10/1994,02 09 10 11 12 16 17 22 23 26 29 35 38 44 47 56 58 73 79 80
44      04/09/1994,06 07 08 09 12 13 14 21 25 28 29 35 37 38 41 54 69 72 74 78
45      04/08/1994,01 03 05 11 13 15 17 19 24 26 27 30 31 47 57 58 59 64 68 80
46      04/07/1994,07 09 12 14 15 17 18 29 36 40 43 47 54 58 62 66 68 74 75 78
47      04/06/1994,05 08 10 22 28 33 38 39 40 44 49 52 62 64 65 69 70 73 77 79
48      04/05/1994,02 06 10 14 15 19 22 25 28 34 44 53 55 57 59 64 65 76 78 80
49      04/04/1994,02 12 20 23 31 37 41 42 45 46 49 55 56 58 68 70 71 72 75 77
50      04/03/1994,01 02 17 18 20 28 35 37 41 47 48 49 56 59 63 65 71 73 77 80
51      04/02/1994,02 04 07 11 23 28 33 42 43 47 49 54 55 58 62 63 64 66 71 72
52      04/01/1994,03 05 06 12 19 23 28 34 37 39 40 44 49 52 57 60 65 73 74 78
53      03/31/1994,07 08 10 14 15 22 31 34 39 41 47 51 52 53 54 58 61 69 74 77
54      03/30/1994,01 09 14 17 18 19 20 21 22 24 31 34 35 57 63 70 71 75 78 79
55      03/29/1994,01 07 14 17 22 25 32 36 37 38 40 51 52 53 62 63 64 65 71 78
56      03/28/1994,01 05 07 12 14 19 22 29 35 38 41 49 50 63 66 67 68 72 74 75
57      03/27/1994,06 08 09 10 13 14 24 34 36 40 45 46 54 55 62 68 74 75 76 77
58      03/26/1994,04 05 08 10 12 16 17 18 20 30 33 34 35 39 65 66 70 71 73 80
59      03/25/1994,01 14 19 20 26 27 28 38 41 45 47 52 53 55 57 59 65 68 74 80
60      03/24/1994,03 04 08 21 22 29 30 32 37 38 39 42 43 45 46 58 67 68 73 78
61      03/23/1994,02 06 08 23 31 33 34 37 40 45 47 48 60 61 63 69 70 71 73 75
62      03/22/1994,01 18 19 23 26 30 36 47 50 51 58 61 65 67 69 72 74 75 76 79
63      03/21/1994,07 15 20 25 28 29 34 39 47 51 52 57 58 62 65 66 69 72 76 77
64      03/20/1994,02 04 08 09 12 28 34 35 38 43 44 47 50 51 54 65 69 72 76 78
65      03/19/1994,02 04 12 16 17 19 29 32 33 35 43 44 47 48 49 52 55 62 68 78
66      03/18/1994,04 05 06 07 13 19 23 26 37 46 48 59 65 66 67 68 69 70 71 77
67      03/17/1994,02 13 16 17 22 24 25 28 34 41 45 47 48 53 56 58 65 75 76 79
68      03/16/1994,01 04 08 09 11 23 26 30 33 35 36 38 39 46 47 50 69 74 77 78
69      03/15/1994,01 02 04 13 14 26 28 30 43 45 46 47 51 52 56 58 60 66 68 74
70      03/14/1994,06 19 22 23 26 27 28 30 32 37 38 42 43 44 46 52 53 62 69 80
71      03/13/1994,02 03 07 13 19 20 21 28 31 32 38 39 48 49 51 52 53 57 61 78
72      03/12/1994,01 02 04 19 20 31 35 38 43 44 49 52 56 60 64 65 69 74 75 79
73      03/11/1994,04 05 06 07 13 15 16 28 29 31 32 33 34 50 57 62 65 66 67 76
74      03/10/1994,04 06 07 08 15 27 39 42 49 52 56 57 60 62 63 68 71 78 79 80
75      03/09/1994,07 11 17 19 21 23 30 36 37 39 41 43 47 49 53 65 67 71 76 77
76      03/08/1994,01 05 11 20 22 23 24 28 30 31 32 38 43 55 60 61 65 67 72 79
77      03/07/1994,03 07 09 10 15 19 20 24 27 28 30 31 37 38 40 48 53 55 58 71
78      03/06/1994,02 10 12 19 22 28 29 31 32 38 41 42 50 54 56 65 68 70 74 79
79      03/05/1994,02 04 05 06 20 31 32 36 37 43 48 50 53 55 58 62 64 66 71 78
80      03/04/1994,08 12 13 17 18 28 30 31 35 36 39 41 45 48 49 50 60 65 77 78
81      03/03/1994,03 06 07 12 13 14 15 16 20 25 35 42 53 55 60 61 62 63 64 73
82      03/02/1994,01 04 09 14 27 37 47 51 52 56 57 58 59 62 64 68 74 75 77 78
83      03/01/1994,02 08 11 12 14 17 28 39 41 43 51 57 59 64 65 68 69 72 73 78
84      02/28/1994,06 12 13 15 18 21 23 35 40 41 45 49 62 63 64 68 69 71 79 80
85      02/27/1994,06 17 21 26 29 30 31 35 40 43 45 47 52 55 57 58 62 63 68 75
86      02/26/1994,09 12 21 24 25 26 32 33 39 45 52 53 54 58 62 65 66 70 72 73
87      02/25/1994,04 06 15 16 20 21 26 27 28 29 48 53 57 58 64 66 73 74 75 79
88      02/24/1994,05 11 18 20 34 40 44 48 50 51 53 58 60 61 64 66 68 72 73 77
89      02/23/1994,06 09 10 15 16 17 20 27 34 36 43 48 50 58 59 60 62 69 74 75
90      02/22/1994,08 15 20 22 25 30 32 34 35 41 42 44 51 57 63 67 72 74 78 80
91      02/21/1994,03 08 13 14 15 18 21 45 47 48 49 53 55 57 64 66 68 69 71 72
92      02/20/1994,03 12 16 21 25 29 30 31 33 34 35 40 41 44 56 59 61 64 67 79
93      02/19/1994,08 10 11 15 17 20 21 26 29 32 33 46 48 53 55 58 68 75 77 79
94      02/18/1994,01 09 11 16 28 29 33 34 39 47 48 53 58 59 64 66 71 76 77 80
95      02/17/1994,05 10 13 16 23 27 30 38 40 42 46 47 48 51 56 60 64 69 72 80
96      02/16/1994,05 07 09 11 17 19 23 29 30 32 35 36 40 44 58 61 63 65 70 78
```

```
97      02/15/1994,03 07 13 25 28 30 31 41 44 51 54 55 59 60 62 70 71 72 73 76
98      02/14/1994,02 04 07 11 13 14 16 18 26 31 33 36 43 50 51 55 58 59 63 66
99      02/13/1994,06 12 15 18 24 28 33 39 41 47 54 58 59 62 65 70 73 74 77 78
00      02/12/1994,04 09 10 11 20 21 31 36 39 40 47 57 58 63 68 75 76 78 79 80
01      02/11/1994,08 31 33 34 35 37 39 40 44 47 48 49 59 60 61 64 66 74 75 80
02      02/10/1994,02 04 05 11 17 23 27 31 35 37 41 48 49 57 59 60 64 70 76 80
03      02/09/1994,02 03 04 09 11 19 27 29 31 32 39 56 59 61 63 65 74 78 79 80
04      02/08/1994,03 07 10 12 23 24 26 27 28 29 31 33 34 45 55 57 58 59 66 70
05      02/07/1994,01 02 07 12 17 25 26 30 33 36 41 43 45 46 47 48 52 53 73 79
06      02/06/1994,08 09 11 15 18 23 24 26 30 33 34 48 55 56 67 69 75 77 79 80
07      02/05/1994,05 13 14 17 18 22 23 30 35 40 42 43 57 66 68 70 71 77 78 80
08      02/04/1994,01 12 14 20 22 24 25 27 30 31 38 39 47 53 55 60 70 72 77 79
09      02/03/1994,07 10 12 15 27 28 43 45 46 48 55 56 61 62 68 69 71 74 75 79
10      02/02/1994,04 05 07 08 11 12 23 28 30 32 34 38 42 55 66 69 72 77 79 80
11      02/01/1994,04 17 20 24 30 40 41 42 46 47 49 56 59 60 64 65 68 71 72 80
12      01/31/1994,05 06 08 09 12 16 18 26 29 30 33 40 52 54 57 59 61 70 71 72
13      01/30/1994,02 08 10 22 25 27 33 36 39 43 48 50 52 58 59 69 70 71 72 80
14      01/29/1994,04 09 11 12 20 27 29 40 44 45 48 50 51 60 62 64 66 70 73 77
15      01/28/1994,03 06 09 11 14 15 21 22 36 50 54 55 56 58 61 69 70 71 75 80
16      01/27/1994,04 08 10 12 23 24 25 42 46 49 52 55 58 61 64 67 71 72 75 76
17      01/26/1994,02 05 06 11 12 15 17 18 22 32 42 44 47 49 53 56 65 71 78 80
18      01/25/1994,04 05 07 11 12 14 15 20 26 30 34 38 40 41 57 60 61 62 64 71
19      01/24/1994,01 02 04 06 11 12 17 18 19 21 24 31 36 49 53 54 55 57 58 76
20      01/23/1994,04 09 12 13 14 16 21 23 26 32 33 34 38 44 47 49 57 63 68 79
21      01/22/1994,01 03 10 11 16 22 24 28 30 33 46 48 50 55 59 62 64 66 70 75
22      01/21/1994,05 17 19 22 23 26 28 32 36 45 50 52 54 59 60 61 68 69 70 76
23      01/20/1994,02 05 11 17 18 25 27 30 34 37 42 45 49 62 64 66 68 69 71 77
24      01/19/1994,04 12 14 15 21 28 33 35 37 40 47 48 53 56 63 64 66 67 74 76
25      01/18/1994,05 06 15 17 20 27 28 39 41 43 47 49 50 52 55 61 62 67 70 77
26      01/17/1994,02 06 09 10 14 16 18 23 28 33 37 40 41 45 50 57 63 67 69 79
27      01/16/1994,03 04 09 15 17 20 23 26 34 40 42 47 49 55 61 64 66 72 75 78
28      01/15/1994,06 13 14 19 22 25 26 28 31 35 38 53 54 56 63 69 71 73 74 77
29      01/14/1994,05 10 15 20 23 27 32 33 44 45 46 49 50 52 55 59 63 69 72 77
30      01/13/1994,07 08 09 10 11 12 23 25 30 33 37 48 49 54 58 62 65 66 69 80
31      01/12/1994,01 08 10 11 12 16 26 32 37 46 47 51 57 60 61 63 65 69 75 77
32      01/11/1994,07 14 16 21 23 28 37 38 41 48 49 56 57 60 69 71 72 75 76 79
33      01/10/1994,03 06 21 29 30 38 42 45 46 48 55 56 66 68 72 73 74 76 78 80
34      01/09/1994,02 08 10 15 18 23 29 32 36 37 40 41 46 51 54 61 70 73 78 79
35      01/08/1994,01 03 13 14 16 20 21 28 29 36 44 51 55 56 59 62 64 66 69 76
36      01/07/1994,01 02 03 15 22 23 31 41 43 50 52 53 55 63 64 66 68 75 76 80
37      01/06/1994,04 07 12 16 17 19 23 32 44 47 49 51 55 57 60 64 65 71 73 79
38      01/05/1994,04 08 09 11 12 15 18 25 26 27 39 44 45 58 62 67 68 70 75 79
39      01/04/1994,03 10 20 21 22 26 30 35 38 40 45 47 48 55 59 60 63 70 71 76
40      01/03/1994,01 05 21 22 23 24 26 30 35 37 41 55 63 64 65 66 68 70 74 76
41      01/02/1994,01 05 08 13 16 26 28 33 35 42 43 46 47 50 54 57 59 61 64 72
42      01/01/1994,01 02 11 12 18 25 26 30 32 42 48 58 60 65 67 68 69 70 72 74
43      12/31/1993,04 08 20 22 26 30 31 37 38 40 46 55 56 59 60 68 70 75 77 80
44      12/30/1993,06 13 14 15 19 25 28 29 32 42 44 46 54 56 58 62 67 68 71 72
45      12/29/1993,02 06 09 13 19 21 24 26 28 32 35 39 40 42 43 51 57 63 71 73
46      12/28/1993,03 04 07 08 12 13 15 19 25 35 36 37 38 40 48 49 65 72 79 80
47      12/27/1993,04 12 19 20 21 24 25 27 28 35 44 47 53 57 59 60 64 66 68 77
48      12/26/1993,04 11 16 17 27 29 34 35 37 43 47 49 50 51 62 63 67 70 71 72
49      12/24/1993,01 17 18 25 31 32 36 39 45 47 49 55 62 63 66 72 73 74 79 80
50      12/23/1993,01 10 12 14 15 21 22 23 27 28 29 31 35 37 45 56 61 62 65 77
```

```
51      12/22/1993,01 02 05 08 12 29 31 37 41 50 55 56 57 58 61 62 63 65 67 74
52      12/21/1993,05 06 09 16 28 30 36 37 38 39 40 51 56 58 66 68 71 73 75 80
53      12/20/1993,03 12 15 17 28 30 33 41 43 44 45 50 55 60 64 65 71 75 77 78
54      12/19/1993,02 04 08 12 18 20 29 32 33 37 38 42 49 50 51 55 57 69 77 78
55      12/18/1993,02 08 11 14 20 22 26 31 44 47 49 51 56 58 66 68 71 73 75 78
56      12/17/1993,03 05 06 10 11 12 19 27 32 33 35 37 38 40 45 48 61 67 71 76
57      12/16/1993,10 11 13 18 21 22 26 34 35 40 44 52 58 59 60 64 72 74 76 80
58      12/15/1993,01 07 14 16 20 23 24 26 27 30 36 37 38 46 47 51 56 59 75 77
59      12/14/1993,03 04 08 09 18 20 22 26 31 44 48 49 53 54 58 63 65 68 69 78
60      12/13/1993,02 05 06 07 12 19 21 22 28 33 34 36 38 40 41 43 48 57 69 73
61      12/12/1993,04 08 10 12 18 25 34 38 39 49 52 56 59 61 66 67 74 76 77 80
62      12/11/1993,04 10 11 18 19 23 26 27 32 33 36 54 58 60 65 67 72 75 77 78
63      12/10/1993,01 03 14 17 18 19 21 22 24 27 28 32 37 47 52 53 62 64 69 73
64      12/09/1993,06 09 11 12 17 18 23 26 28 32 33 35 40 51 53 57 66 67 69 74
65      12/08/1993,01 02 04 08 16 17 18 25 27 33 42 49 52 53 59 60 61 62 75 80
66      12/07/1993,01 05 06 07 17 19 21 22 28 32 45 49 53 55 57 61 63 69 70 74
67      12/06/1993,02 07 13 17 18 19 21 27 29 32 36 39 44 55 58 64 65 66 74 75
68      12/05/1993,08 12 15 18 19 20 28 34 40 41 44 57 63 66 67 69 70 74 79 80
69      12/04/1993,07 12 14 17 18 24 31 38 40 44 46 49 57 64 68 69 70 71 74 76
70      12/03/1993,05 07 08 10 12 17 19 24 25 30 31 34 36 42 47 51 62 65 66 73
71      12/02/1993,08 10 18 21 22 27 29 43 45 46 51 52 53 55 62 63 69 70 78 80
72      12/01/1993,07 10 23 24 27 33 40 52 53 54 55 58 61 66 67 70 73 75 77 79
73      11/30/1993,01 03 04 06 07 13 14 22 33 37 45 46 57 61 63 66 74 75 76 77
74      11/29/1993,02 07 11 14 16 17 19 23 33 35 39 54 62 63 65 66 67 68 70 75
75      11/28/1993,01 06 09 19 21 27 33 34 37 39 40 44 45 47 51 52 55 66 70 76
76      11/27/1993,03 09 11 14 17 21 26 32 38 39 42 43 50 51 58 61 65 67 69 76
77      11/26/1993,03 07 08 16 22 25 34 35 36 42 50 54 56 61 64 70 71 72 73 75
78      11/25/1993,01 02 10 15 16 19 24 29 38 47 51 53 57 58 64 67 69 70 74 77
79      11/24/1993,01 06 07 17 25 29 36 42 43 44 50 57 65 66 69 71 72 73 75 77
80      11/23/1993,03 06 07 12 16 23 25 30 31 32 33 34 43 44 48 52 56 59 63 67
81      11/22/1993,02 03 04 10 15 22 23 26 42 43 45 46 47 49 54 56 62 67 74 77
82      11/21/1993,06 07 12 23 26 31 34 37 40 41 42 43 51 55 56 58 59 66 67 79
83      11/20/1993,06 08 09 14 15 21 24 28 31 36 40 41 42 44 50 59 65 70 73 79
84      11/19/1993,06 09 11 15 20 24 30 32 36 40 42 43 49 50 53 54 60 68 71 80
85      11/18/1993,11 13 17 24 25 27 31 32 44 45 46 49 52 56 60 63 70 72 76 80
86      11/17/1993,07 11 15 19 21 23 29 31 39 40 41 48 53 56 67 70 72 75 78 79
87      11/16/1993,01 08 14 20 21 24 28 33 36 38 45 46 50 56 62 64 66 67 69 73
88      11/15/1993,07 11 16 17 21 24 25 27 39 55 56 58 59 60 62 65 69 70 73 77
89      11/14/1993,02 03 05 07 12 15 17 26 34 36 37 46 52 59 60 61 73 74 76 77
90      11/13/1993,09 16 17 19 21 24 25 26 28 31 32 33 35 40 41 56 69 70 72 75
91      11/12/1993,01 02 05 07 14 22 27 28 31 33 34 35 46 54 56 61 68 71 74 75
92      11/11/1993,02 08 11 19 23 25 29 38 46 49 54 56 59 60 62 64 68 71 77 78
93      11/10/1993,03 07 08 09 10 12 13 17 19 24 32 36 38 43 50 57 64 70 73 76
94      11/09/1993,01 05 06 07 08 10 13 19 25 34 36 39 45 49 51 56 62 74 75 77
95      11/08/1993,02 08 09 12 14 18 19 25 30 32 39 42 43 51 53 54 58 61 66 68
96      11/07/1993,07 12 14 19 27 29 31 32 34 37 38 40 46 50 51 58 63 64 67 77
97      11/06/1993,10 13 25 26 28 29 30 31 37 41 50 52 56 58 62 71 72 76 78 79
98      11/05/1993,02 06 07 14 15 17 20 28 38 41 45 48 55 59 62 64 68 73 76 77
99      11/04/1993,01 03 06 09 17 19 21 23 26 28 29 48 51 52 64 66 73 74 75 78
00      11/03/1993,02 08 15 16 20 28 29 39 42 43 44 45 49 53 54 57 61 70 71 76
01      11/02/1993,02 03 12 14 15 21 22 25 27 31 32 52 54 55 59 64 66 69 71 73
02      11/01/1993,06 13 15 18 29 30 34 35 37 42 46 49 56 57 62 64 67 72 73 78
03      10/31/1993,02 09 11 15 19 21 26 27 34 39 49 52 53 54 55 65 66 69 73 75
04      10/30/1993,01 07 08 09 17 20 21 22 25 27 28 30 32 40 54 61 63 70 78 80
```

05 10/29/1993,03 08 10 15 23 25 26 28 31 43 44 46 50 51 53 57 58 64 66 67
06 10/28/1993,19 20 26 29 31 32 35 36 37 42 43 47 48 59 60 69 70 71 78 80
07 10/27/1993,01 02 07 08 15 18 22 23 28 30 34 37 42 49 64 66 68 70 76 78
08 10/26/1993,06 11 26 27 29 30 38 40 41 44 46 60 61 62 63 66 71 74 78 80
09 10/25/1993,02 05 10 11 13 16 17 28 32 34 35 40 45 47 49 55 62 63 65 74
10 10/24/1993,01 06 12 14 15 18 22 23 25 28 30 40 42 45 53 54 59 62 64 79
11 10/23/1993,03 04 07 15 20 21 34 36 37 41 53 54 56 58 64 72 74 78 79 80
12 10/22/1993,02 03 04 13 14 15 19 22 29 40 45 46 50 51 56 58 60 66 70 75
13 10/21/1993,06 10 11 13 18 22 25 28 43 46 48 53 56 57 62 66 67 71 79 80
14 10/20/1993,06 07 08 10 12 15 16 17 19 25 29 30 34 37 38 53 72 74 75 78
15 10/19/1993,04 06 09 18 19 22 23 32 33 35 40 42 43 44 52 56 63 64 68 74
16 10/18/1993,06 11 17 18 22 25 31 33 37 39 48 50 52 54 60 68 69 71 75 78
17 10/17/1993,03 06 15 24 25 27 28 29 32 33 34 39 42 45 48 49 53 63 74 80
18 10/16/1993,04 05 13 18 20 24 27 33 35 36 41 43 46 54 55 58 59 73 74 79
19 10/15/1993,03 07 08 18 21 22 27 32 34 37 50 52 55 56 64 67 70 76 79 80
20 10/14/1993,03 07 18 22 24 38 39 40 41 46 47 51 52 59 63 65 70 72 74 75
21 10/13/1993,05 08 14 19 24 30 31 33 39 44 52 54 55 56 58 63 64 67 71 78
22 10/12/1993,07 12 15 25 29 30 35 37 39 47 51 54 55 59 64 65 68 70 77 80
23 10/11/1993,09 11 16 19 20 21 22 28 32 41 46 50 58 68 69 72 74 76 78 80
24 10/10/1993,01 02 05 06 11 22 32 38 40 42 44 51 52 55 57 58 70 71 77 78
25 10/09/1993,03 06 08 09 17 21 22 26 30 42 47 55 57 63 64 65 67 68 73 79
26 10/08/1993,04 10 13 17 23 24 26 28 31 32 39 41 46 50 51 53 60 62 63 71
27 10/07/1993,03 07 23 26 29 32 33 40 42 45 47 52 55 60 63 67 68 70 71 77
28 10/06/1993,01 07 09 16 21 35 38 42 44 47 48 51 55 57 66 67 68 69 72 79
29 10/05/1993,02 03 05 07 11 14 19 21 24 30 35 40 41 47 51 54 56 70 71 78
30 10/04/1993,01 11 21 25 28 29 32 39 43 44 49 50 51 59 65 66 72 75 77 80
31 10/03/1993,07 17 19 22 24 25 38 39 42 44 45 47 48 52 53 55 58 59 62 71
32 10/02/1993,01 08 11 15 17 31 35 39 40 48 52 53 55 58 59 66 77 78 79 80
33 10/01/1993,02 03 04 13 19 22 26 27 33 34 38 53 56 65 66 71 72 75 76 77
34 09/30/1993,04 12 18 19 23 24 30 39 42 46 49 50 63 65 66 70 72 75 76 79
35 09/29/1993,01 04 07 14 27 28 29 33 36 41 43 47 52 55 59 62 70 74 76 79
36 09/28/1993,12 16 17 26 27 29 31 35 36 43 46 48 50 59 61 62 64 67 70 74
37 09/27/1993,01 05 07 16 26 33 36 40 45 48 54 55 58 59 63 64 68 75 77 79
38 09/26/1993,05 06 10 17 19 27 29 38 39 43 44 45 50 51 59 67 70 73 79 80
39 09/25/1993,02 07 09 13 20 21 28 41 47 48 51 53 55 56 59 60 62 63 66 70
40 09/24/1993,01 13 14 15 18 22 23 30 31 32 35 36 43 48 51 69 73 74 76 78
41 09/23/1993,05 08 09 10 12 13 14 19 20 28 30 31 33 41 45 57 62 73 77 80
42 09/22/1993,01 04 11 13 18 20 22 23 35 37 38 45 51 54 57 58 63 64 66 72
43 09/21/1993,08 10 11 16 17 23 24 27 28 30 33 35 50 52 63 69 70 74 76 79
44 09/20/1993,06 12 19 20 23 35 37 40 41 46 47 49 51 52 61 64 68 69 76 78
45 09/19/1993,07 09 15 17 18 29 36 39 44 45 52 57 58 61 66 67 69 77 79 80
46 09/18/1993,02 05 07 09 10 13 18 19 23 27 33 37 47 49 51 52 60 76 77 79
47 09/17/1993,04 06 14 15 17 18 20 27 28 29 30 41 46 49 50 59 61 64 70 77
48 09/16/1993,01 02 16 17 21 23 25 27 28 29 30 33 46 51 52 58 60 64 72 77
49 09/15/1993,01 03 04 07 08 15 18 21 24 36 37 39 44 45 50 54 60 61 64 79
50 09/14/1993,10 11 12 15 16 18 19 26 31 36 39 41 53 55 61 66 68 69 72 73
51 09/13/1993,01 03 07 11 15 19 20 23 32 33 34 37 40 49 61 63 65 67 68 76
52 09/12/1993,05 06 11 19 20 21 32 35 37 39 42 43 46 47 58 59 61 64 74 77
53 09/11/1993,01 06 11 14 20 21 22 25 26 33 35 38 39 41 43 44 47 58 59 60
54 09/10/1993,03 10 11 18 19 25 26 32 36 40 41 44 55 56 59 66 68 69 76 79
55 09/09/1993,05 06 08 10 15 24 25 26 27 28 35 38 40 42 44 59 62 69 74 80
56 09/08/1993,04 08 16 20 24 27 29 31 33 36 40 49 56 60 62 64 68 70 75 80
57 09/07/1993,02 03 04 06 10 20 21 24 35 39 41 42 55 59 61 62 63 66 75 77
58 09/06/1993,01 02 06 08 12 14 16 23 29 30 32 42 46 47 51 53 58 62 74 77

```
59      09/05/1993,03 08 09 10 13 17 32 35 39 44 46 49 50 55 61 64 67 69 71 80
60      09/04/1993,08 13 16 17 21 22 28 29 32 33 35 46 55 56 57 72 77 78 79 80
61      09/03/1993,02 04 11 15 20 33 39 40 48 51 52 57 59 61 65 67 71 73 76 77
62      09/02/1993,02 13 16 22 33 35 40 41 44 49 51 52 54 63 65 70 74 77 78 79
63      09/01/1993,04 06 11 18 23 31 33 34 36 37 42 46 54 56 58 60 66 71 73 78
64      08/31/1993,02 03 05 06 13 18 21 24 27 31 34 37 38 43 52 55 63 65 78 80
65      08/30/1993,03 07 10 13 16 17 18 19 23 34 40 43 47 50 56 60 65 71 72 75
66      08/29/1993,01 06 08 14 15 16 18 19 23 30 35 37 42 43 57 64 71 73 76 78
67      08/28/1993,09 16 17 21 25 39 41 42 50 51 58 59 60 62 69 70 74 75 78 80
68      08/27/1993,02 06 12 20 23 24 27 39 40 41 45 47 52 54 55 57 67 68 71 80
69      08/26/1993,01 06 12 17 22 29 33 40 43 48 50 52 54 56 57 59 62 68 74 76
70      08/25/1993,03 11 12 16 18 19 20 22 25 26 38 40 45 50 52 60 64 74 75 78
71      08/24/1993,01 09 13 16 25 26 41 43 47 54 56 57 62 64 69 71 73 76 78 79
72      08/23/1993,07 09 12 14 15 18 21 24 28 29 36 37 48 53 56 69 75 76 79 80
73      08/22/1993,01 10 19 22 23 24 25 27 28 33 34 42 43 44 47 54 64 65 73 76
74      08/21/1993,13 18 21 24 26 28 33 34 36 37 38 44 45 46 57 65 66 69 70 78
75      08/20/1993,10 11 14 17 19 25 27 29 31 35 43 45 48 49 50 51 52 54 59 64
76      08/19/1993,04 23 26 28 31 33 35 36 39 47 52 53 55 57 62 63 73 74 77 79
77      08/18/1993,02 03 05 06 08 17 19 21 23 29 34 44 53 54 62 65 67 71 74 76
78      08/17/1993,01 05 11 15 34 36 39 41 43 47 49 51 53 55 57 58 59 65 73 77
79      08/16/1993,06 11 15 24 27 31 34 43 46 48 49 52 56 59 60 67 74 75 76 77
80      08/15/1993,04 09 10 12 22 23 27 30 31 34 39 43 44 49 52 54 55 62 71 79
81      08/14/1993,04 05 09 10 12 14 18 22 23 25 32 47 48 52 54 59 60 76 77 79
82      08/13/1993,03 07 12 16 18 22 25 28 35 40 43 52 53 61 62 67 69 71 74 80
83      08/12/1993,01 09 16 18 19 21 23 27 28 35 36 39 41 42 43 51 60 65 72 77
84      08/11/1993,02 09 17 19 20 22 31 33 40 42 43 44 53 55 57 64 67 72 78 80
85      08/10/1993,04 05 09 11 16 21 23 27 28 30 31 34 39 42 56 57 60 66 72 73
86      08/09/1993,01 13 17 19 20 22 23 28 32 33 34 36 42 47 52 54 66 68 73 75
87      08/08/1993,02 06 10 14 16 22 36 39 46 48 53 54 57 59 62 64 69 74 76 79
88      08/07/1993,01 05 11 20 27 29 30 32 42 47 49 50 52 53 59 63 67 69 73 74
89      08/06/1993,03 09 12 14 20 24 27 32 34 39 51 52 55 56 58 61 71 75 77 79
90      08/05/1993,04 09 18 19 23 24 28 32 35 40 42 48 51 52 53 54 57 58 68 78
91      08/04/1993,04 05 10 13 25 28 30 33 35 41 43 45 46 57 58 62 66 67 68 75
92      08/03/1993,07 10 11 17 22 23 25 30 32 33 39 42 46 47 49 57 58 65 70 74
93      08/02/1993,09 20 29 30 31 39 40 41 47 48 51 58 64 65 68 70 73 75 77 78
94      08/01/1993,01 04 10 23 30 33 36 41 45 49 51 58 62 64 65 68 70 72 76 80
95      07/31/1993,03 06 07 09 10 15 28 29 33 37 44 45 47 48 49 52 57 63 64 78
96      07/30/1993,03 07 09 11 12 16 18 24 31 32 38 40 42 47 48 50 55 63 79 80
97      07/29/1993,01 09 11 13 17 18 21 28 31 33 34 35 40 43 53 57 63 65 67 74
98      07/28/1993,08 09 10 12 15 17 25 27 29 38 39 40 45 48 50 56 57 58 71 75
99      07/27/1993,01 08 12 18 22 24 28 34 41 42 50 52 55 63 68 71 75 76 78 80
00      07/26/1993,06 09 10 12 15 21 26 32 33 36 37 46 47 51 54 58 60 72 74 76
01      07/25/1993,01 04 05 10 14 20 25 37 40 44 45 49 51 52 61 68 69 70 76 78
02      07/24/1993,03 11 12 14 18 19 21 33 36 41 49 50 53 56 57 60 64 73 76 80
03      07/23/1993,03 16 22 24 30 31 32 33 35 36 37 42 55 60 63 65 67 76 77 78
04      07/22/1993,01 03 05 07 14 16 20 23 27 31 32 35 41 42 45 46 63 64 65 73
05      07/21/1993,03 05 16 18 24 29 31 40 42 48 49 54 55 56 64 65 70 73 79 80
06      07/20/1993,02 09 13 19 20 24 27 29 31 34 40 47 52 57 65 66 68 70 72 75
07      07/19/1993,07 10 11 15 20 25 37 39 42 43 45 46 57 58 59 64 68 73 74 77
08      07/18/1993,03 04 05 06 07 21 22 24 26 35 39 52 57 61 64 66 70 74 75 79
09      07/17/1993,09 10 13 18 20 24 25 30 31 37 42 51 58 61 62 64 70 71 74 78
10      07/16/1993,01 03 04 11 13 20 22 26 29 34 46 50 51 58 60 67 72 73 77 80
11      07/15/1993,04 05 10 12 15 16 21 22 23 33 35 42 53 57 59 60 69 70 77 78
12      07/14/1993,11 12 16 17 24 26 34 43 45 46 47 58 59 64 65 68 69 71 75 77
```

```
07/13/1993,11 19 20 23 31 37 39 41 51 52 53 54 56 57 61 63 64 67 72 73
07/12/1993,09 14 15 17 19 24 30 32 35 40 43 45 46 49 50 54 59 64 67 79
07/11/1993,15 16 21 25 26 34 38 40 41 45 47 49 50 52 53 55 59 60 70 72
07/10/1993,02 05 11 14 16 22 27 34 38 42 49 55 61 62 65 71 72 76 79 80
07/09/1993,04 06 14 15 16 17 18 22 24 34 36 40 43 44 52 54 57 63 64 67
07/08/1993,03 09 10 13 19 20 26 27 33 45 48 49 55 57 64 65 69 71 73 77
07/07/1993,02 04 07 08 18 22 25 28 31 34 35 37 38 42 55 59 60 67 68 75
07/06/1993,01 06 08 09 11 12 14 15 22 23 30 37 38 39 52 63 66 67 68 76
07/05/1993,04 09 11 13 15 22 24 29 31 40 42 45 54 55 60 62 64 71 73 78
07/04/1993,01 02 04 05 07 13 17 18 19 21 28 32 34 39 42 46 65 76 77 80
07/03/1993,05 08 12 16 21 28 29 31 37 39 41 42 45 50 59 60 63 69 72 79
07/02/1993,11 23 27 28 29 30 37 39 42 49 52 53 55 59 62 63 64 72 73 77
07/01/1993,05 10 15 17 22 26 27 28 29 30 35 39 42 43 48 49 54 67 70 76
06/30/1993,01 02 09 10 13 14 21 22 25 27 31 36 38 42 54 57 61 62 66 72
06/29/1993,05 06 08 14 25 27 30 44 45 47 48 57 60 65 68 74 75 76 77 80
06/28/1993,07 13 14 22 24 26 29 32 33 35 36 44 48 50 57 66 67 68 74 76
06/27/1993,04 12 15 19 25 26 27 30 32 34 36 40 42 50 57 59 62 69 72 76
06/26/1993,02 04 05 07 10 13 22 23 25 37 38 39 43 49 52 56 61 64 68 75
06/25/1993,08 09 11 14 19 21 22 23 25 31 32 36 37 38 46 50 59 66 71 73
06/24/1993,02 03 05 08 10 12 13 14 24 25 28 38 40 55 65 70 72 75 76 77
06/23/1993,02 03 05 08 18 20 24 25 30 38 45 50 52 55 65 67 69 72 76 78
06/22/1993,06 07 11 13 15 17 18 19 26 30 34 36 38 39 42 44 47 63 77 78
06/21/1993,03 05 07 18 19 22 24 28 29 30 39 41 50 53 55 62 67 71 72 77
06/20/1993,07 12 22 27 30 31 34 40 41 44 46 53 56 65 67 70 72 73 74 76
06/19/1993,05 06 07 14 15 16 30 33 36 39 40 47 55 56 58 59 63 65 66 70
06/18/1993,04 05 07 10 14 23 28 29 34 44 47 54 56 61 63 64 66 72 77 79
06/17/1993,01 03 05 18 30 40 42 47 51 52 54 58 59 63 64 67 69 71 77 78
06/16/1993,08 09 15 16 17 18 19 26 33 35 37 38 47 51 54 63 64 65 70 71
06/15/1993,02 03 07 16 19 22 23 31 37 39 41 42 44 45 47 48 51 69 76 77
06/14/1993,02 04 17 19 23 24 27 30 32 41 49 51 57 58 59 63 69 73 76 80
06/13/1993,01 03 04 06 09 15 16 19 21 28 37 38 40 46 61 64 71 76 77 78
06/12/1993,03 09 10 15 30 31 33 34 38 39 44 47 49 54 64 70 71 72 76 80
06/11/1993,02 06 12 21 23 24 28 30 32 34 36 42 53 58 60 64 66 67 74 76
06/10/1993,08 13 15 21 23 26 30 34 36 41 45 58 62 63 65 67 69 72 76 77
06/09/1993,04 12 16 23 28 32 37 45 49 50 51 52 61 64 65 68 70 72 74 75
06/08/1993,15 18 26 28 29 37 40 41 46 47 48 50 52 55 57 59 61 62 73 77
06/07/1993,01 09 10 14 15 16 18 21 27 32 34 40 44 51 58 65 73 76 77 79
06/06/1993,08 10 11 14 25 32 36 40 45 54 57 58 59 60 63 68 69 70 71 75
06/05/1993,05 07 15 18 20 22 24 29 32 34 38 39 49 54 56 62 68 75 76 78
06/04/1993,01 03 05 06 10 13 14 20 25 33 40 50 52 57 60 67 71 76 77 78
06/03/1993,12 13 15 17 25 27 28 30 34 44 49 53 54 56 63 66 67 69 78 79
06/02/1993,05 06 07 14 17 20 22 30 37 40 45 46 59 60 62 68 69 70 76 80
06/01/1993,01 02 04 06 07 13 21 23 26 31 46 51 55 59 60 63 74 75 76 78
05/31/1993,06 13 15 21 24 27 29 34 35 43 44 48 52 61 63 65 69 74 76 80
05/30/1993,04 07 08 09 11 16 19 23 27 34 36 42 44 48 50 52 61 69 70 75
05/29/1993,01 02 11 13 16 21 23 24 27 33 35 40 41 45 46 54 64 74 79 80
05/28/1993,03 07 12 16 18 20 23 24 27 29 32 48 49 53 59 66 67 69 70 76
05/27/1993,03 05 12 14 15 17 18 28 31 36 40 44 47 55 59 62 66 67 70 78
05/26/1993,04 05 06 15 17 25 26 27 32 33 34 37 38 43 48 52 60 65 69 70
05/25/1993,02 09 14 15 18 20 24 29 30 39 46 58 59 60 64 66 68 74 75 76
05/24/1993,03 08 12 16 21 23 24 27 29 36 40 50 54 57 59 64 66 74 76 79
05/23/1993,04 05 07 08 15 22 23 26 30 33 34 43 53 55 63 69 74 75 78 79
05/22/1993,06 09 10 11 13 14 15 22 24 25 34 39 44 50 62 65 69 74 75 78
05/21/1993,04 07 08 10 20 24 31 35 42 43 51 53 54 57 60 62 67 68 73 75
```

```
67      05/20/1993,02 05 09 13 14 18 20 22 23 26 30 35 40 45 52 62 65 70 72 75
68      05/19/1993,06 07 08 11 17 20 24 30 33 36 38 42 43 44 50 59 68 73 77 79
69      05/18/1993,06 07 17 20 21 22 28 32 33 34 40 44 47 52 59 61 63 65 77 78
70      05/17/1993,05 10 19 21 27 28 29 40 42 48 49 55 59 67 68 69 70 72 76 80
71      05/16/1993,01 04 05 06 15 18 23 25 32 35 37 38 39 40 44 46 53 56 68 78
72      05/15/1993,01 03 08 09 13 14 17 21 39 43 45 48 49 54 57 61 63 68 72 74
73      05/14/1993,03 06 08 09 15 21 22 23 25 33 38 45 47 50 54 59 61 72 73 77
74      05/13/1993,09 10 18 20 21 22 28 29 31 33 35 49 51 58 64 68 72 73 74 77
75      05/12/1993,02 05 06 07 11 16 19 28 29 33 41 50 52 53 59 60 69 72 78 79
76      05/11/1993,01 02 15 19 24 25 35 39 46 49 55 59 60 62 65 67 69 71 77 79
77      05/10/1993,09 11 12 13 15 16 20 25 41 50 51 52 53 59 60 61 66 67 71 78
78      05/09/1993,01 02 03 05 06 08 09 12 19 26 27 28 40 43 46 56 58 68 73 75
79      05/08/1993,04 13 15 17 18 26 28 38 39 42 45 49 55 58 64 67 73 76 77 79
80      05/07/1993,14 15 21 23 28 37 40 47 49 54 55 56 59 60 62 67 68 71 74 78
81      05/06/1993,02 10 11 12 13 14 23 33 36 38 45 49 50 51 52 53 54 56 71 74
82      05/05/1993,10 11 14 22 28 31 32 41 42 44 51 53 54 62 66 67 68 73 77 79
83      05/04/1993,03 07 11 17 21 22 24 27 30 35 36 39 43 50 51 58 61 66 72 74
84      05/03/1993,07 10 14 16 20 25 26 30 31 32 34 44 45 48 56 60 62 67 75 80
85      05/02/1993,03 06 07 08 10 11 14 15 20 34 36 41 47 48 53 56 61 69 73 76
86      05/01/1993,02 03 06 08 12 15 19 24 29 36 38 45 46 53 55 57 59 60 62 67
87      04/30/1993,04 18 21 24 25 29 39 42 44 45 49 60 65 67 68 69 73 74 75 76
88      04/29/1993,01 03 14 15 19 20 21 30 41 47 48 49 62 63 64 65 67 71 73 75
89      04/28/1993,01 07 08 10 15 22 23 27 28 30 42 47 48 49 55 57 60 62 69 70
90      04/27/1993,01 02 10 11 16 17 18 22 25 27 35 45 46 50 52 54 61 67 76 78
91      04/26/1993,04 09 10 18 20 21 26 27 29 30 32 37 39 50 52 65 66 69 71 74
92      04/25/1993,06 07 25 26 27 29 31 32 35 38 41 46 56 60 64 67 68 73 74 75
93      04/24/1993,05 09 13 14 20 22 25 28 31 36 42 48 50 53 68 72 74 75 76 78
94      04/23/1993,02 03 05 06 12 13 16 18 20 35 38 46 50 53 68 69 75 77 79 80
95      04/22/1993,04 09 19 21 29 32 40 41 49 51 53 55 63 69 71 75 76 77 78 79
96      04/21/1993,01 03 11 16 18 23 25 29 30 34 41 46 49 52 60 65 66 68 69 72
97      04/20/1993,03 06 12 14 20 21 27 28 33 36 38 41 44 47 50 56 65 72 73 76
98      04/19/1993,02 11 16 20 21 22 31 38 44 47 51 53 57 61 66 68 69 71 72 79
99      04/18/1993,05 06 07 10 16 25 28 29 39 40 42 43 45 48 54 55 57 62 70 75
00      04/17/1993,01 02 12 20 29 34 35 37 38 42 45 54 56 57 58 63 64 70 71 78
01      04/16/1993,04 13 16 20 25 37 38 41 44 51 52 55 57 61 63 64 65 74 76 78
02      04/15/1993,05 13 17 18 19 28 31 32 34 36 38 39 40 45 49 52 56 63 77 80
03      04/14/1993,01 03 09 13 14 19 25 26 29 39 46 48 51 54 56 57 60 61 70 75
04      04/13/1993,03 21 25 28 32 34 39 41 45 50 52 54 58 61 63 65 66 69 74 77
05      04/12/1993,05 10 15 20 23 25 27 30 32 50 51 52 53 58 66 68 69 75 78 80
06      04/11/1993,03 06 07 16 18 21 24 25 26 29 33 40 44 48 50 54 56 58 67 78
07      04/10/1993,02 09 12 14 15 19 26 29 30 34 39 46 47 51 53 54 65 68 72 80
08      04/09/1993,01 07 11 20 22 23 24 25 31 32 38 44 48 56 63 64 68 72 75 77
09      04/08/1993,06 07 08 09 14 18 29 30 33 36 39 40 42 48 58 66 71 72 73 79
10      04/07/1993,03 04 05 06 11 13 22 25 28 35 42 44 51 55 58 61 62 74 78 80
11      04/06/1993,03 08 11 21 23 24 28 30 32 38 43 45 46 55 58 62 67 68 72 78
12      04/05/1993,09 10 16 20 22 24 29 31 38 40 46 51 52 54 60 65 68 77 78 80
13      04/04/1993,01 03 04 11 12 19 29 31 32 37 44 51 54 56 63 64 65 74 77 79
14      04/03/1993,01 04 08 10 12 17 31 32 44 46 47 52 55 56 59 67 70 73 77 78
15      04/02/1993,02 08 09 11 13 15 36 40 42 47 48 49 51 53 57 61 63 64 71 73
16      04/01/1993,04 14 17 19 20 23 30 31 32 40 54 55 57 59 62 63 65 66 70 78
17      03/31/1993,03 05 06 14 16 32 33 34 42 46 50 54 55 58 59 60 62 67 68 74
18      03/30/1993,01 02 13 16 17 19 24 31 33 38 41 45 48 53 56 62 68 73 76 79
19      03/29/1993,01 03 08 10 15 22 23 28 32 38 43 46 48 54 55 57 59 63 66 73
20      03/28/1993,02 05 08 11 12 16 19 26 29 30 39 46 48 51 58 59 67 69 72 78
```

```
03/27/1993,05 06 16 22 27 29 31 37 40 41 47 48 49 52 54 60 62 68 70 79
03/26/1993,03 12 16 18 19 23 37 38 43 53 55 56 57 61 64 65 69 74 77 78
03/25/1993,01 06 08 09 12 13 14 20 24 28 31 32 37 39 45 63 66 72 78 80
03/24/1993,04 07 11 12 15 17 19 22 24 26 34 39 47 48 53 54 64 70 72 73
03/23/1993,03 07 08 12 14 16 17 18 26 28 34 36 53 56 59 60 71 73 74 75
03/22/1993,04 08 09 15 21 24 25 27 31 32 37 38 40 41 43 51 53 55 62 79
03/21/1993,09 12 13 22 24 29 30 36 37 38 40 47 48 49 53 62 64 67 77 79
03/20/1993,05 07 08 11 12 13 23 26 27 29 30 31 38 39 40 43 45 55 60 61
03/19/1993,01 02 20 24 29 32 38 46 47 48 50 51 52 53 62 63 65 66 68 77
03/18/1993,02 07 11 12 14 23 25 34 35 40 41 42 45 46 50 53 60 62 69 76
03/17/1993,03 05 06 09 13 21 22 24 30 31 38 41 42 44 50 53 62 69 75 80
03/16/1993,01 24 25 28 34 38 41 42 44 46 47 49 51 53 67 71 72 73 76 79
03/15/1993,01 04 07 08 14 18 23 25 32 39 40 41 42 48 50 58 61 65 72 77
03/14/1993,03 06 16 17 21 23 25 26 32 34 36 46 48 50 59 61 62 64 75 78
03/13/1993,02 04 07 08 17 19 25 26 27 29 31 32 34 37 40 51 53 65 67 74
03/12/1993,07 08 14 15 21 24 27 36 37 38 41 46 53 55 60 63 66 67 71 79
03/11/1993,09 15 16 17 22 24 25 35 39 50 51 53 58 62 63 66 70 73 74 79
03/10/1993,03 09 13 17 24 29 33 36 49 55 57 60 61 62 64 69 70 73 78 79
03/09/1993,02 07 10 11 13 14 17 19 20 26 32 35 37 48 52 55 65 66 70 71
03/08/1993,05 06 11 15 22 26 38 40 42 46 48 49 50 51 56 67 69 73 74 78
03/07/1993,03 07 09 10 11 14 15 16 17 18 22 35 38 43 48 50 54 65 78 80
03/06/1993,11 13 14 15 16 17 18 22 23 24 32 34 35 37 39 43 48 60 62 68
03/05/1993,06 07 09 12 19 26 34 42 44 45 46 47 49 51 55 59 60 65 66 78
03/04/1993,01 02 03 06 11 14 24 28 31 32 35 36 39 41 60 61 63 64 76 80
03/03/1993,10 18 19 26 29 32 38 42 45 48 53 54 56 60 65 67 72 76 77 78
03/02/1993,04 05 06 07 10 24 27 37 41 42 43 48 56 60 62 69 73 74 77 79
03/01/1993,09 10 11 14 23 27 29 30 37 39 51 53 55 59 71 72 73 74 75 79
02/28/1993,06 07 08 11 16 26 28 32 39 42 52 54 56 60 66 69 70 71 74 78
02/27/1993,03 14 15 18 20 23 24 28 29 30 33 34 37 42 66 69 70 73 76 80
02/26/1993,01 03 05 11 12 15 16 20 28 31 34 38 39 43 48 51 60 68 69 80
02/25/1993,06 08 11 15 18 23 28 31 34 36 39 41 49 62 63 65 68 71 72 77
02/24/1993,01 03 05 08 09 12 15 19 21 23 29 30 34 38 40 42 48 50 60 78
02/23/1993,01 03 07 10 16 26 28 30 34 35 39 43 45 46 53 61 63 64 68 76
02/22/1993,04 08 09 15 20 21 31 33 34 43 46 47 49 51 54 56 60 62 66 71
02/21/1993,06 07 11 19 25 28 30 33 35 37 45 46 47 48 54 55 56 64 67 76
02/20/1993,04 07 13 16 21 25 27 28 30 36 38 46 47 49 54 59 61 70 71 77
02/19/1993,03 08 09 16 18 23 31 33 34 38 45 46 50 51 63 64 73 75 76 77
02/18/1993,05 06 13 15 26 31 33 40 41 45 46 54 55 56 59 69 73 76 78 80
02/17/1993,04 10 11 18 19 21 34 35 36 41 48 53 56 57 63 65 68 71 73 79
02/16/1993,01 04 05 10 12 14 21 25 27 38 50 52 59 60 62 73 74 75 78 80
02/15/1993,01 04 06 08 09 10 18 19 29 34 36 38 41 44 45 48 68 69 71 77
02/14/1993,03 06 12 16 24 31 35 36 40 42 46 52 54 55 57 60 62 65 74 78
02/13/1993,02 08 11 13 14 19 27 29 30 31 32 48 50 53 56 59 65 70 72 80
02/12/1993,05 20 26 29 31 32 42 45 46 47 48 50 56 65 66 67 71 73 75 79
02/11/1993,04 05 06 09 18 20 27 32 36 39 41 43 47 49 51 52 63 75 77 78
02/10/1993,02 03 09 11 12 17 19 20 21 24 26 32 35 41 45 52 59 70 72 77
02/09/1993,05 06 10 16 18 28 33 34 35 36 37 42 48 50 52 54 59 65 70 76
02/08/1993,01 03 09 14 15 17 32 34 36 42 51 54 56 58 63 66 70 71 78 80
02/07/1993,04 06 07 09 10 20 25 27 33 38 47 50 58 59 64 65 66 67 73 77
02/06/1993,01 03 05 09 10 13 25 31 34 35 39 40 44 45 57 62 64 68 73 80
02/05/1993,08 09 12 14 18 23 27 29 33 37 38 41 43 44 50 52 56 58 67 70
02/04/1993,07 09 18 20 22 25 27 31 32 36 41 42 43 52 53 56 72 73 78 80
02/03/1993,04 05 12 30 32 33 37 42 44 45 46 47 50 52 55 58 59 62 68 72
02/02/1993,02 08 12 15 16 18 22 23 28 30 37 41 44 47 53 56 64 65 70 78
```

```
02/01/1993,02 04 08 12 13 15 27 30 32 37 39 46 47 57 58 60 62 63 68 70
01/31/1993,05 06 07 11 16 18 21 28 33 35 40 41 44 48 52 53 56 59 60 62
01/30/1993,02 03 04 05 12 14 19 21 30 36 39 51 54 58 61 62 66 73 76 78
01/29/1993,01 10 15 17 40 41 42 51 52 56 57 60 61 66 70 72 74 75 77 80
01/28/1993,08 11 17 23 31 37 41 42 43 45 55 57 62 64 67 69 74 75 76 80
01/27/1993,02 13 17 20 22 28 29 30 34 36 38 41 42 44 45 46 50 54 62 69
01/26/1993,04 08 11 12 17 25 27 31 36 37 38 41 43 45 47 55 59 60 63 75
01/25/1993,01 05 06 07 11 13 22 25 27 31 35 41 43 45 48 53 55 71 72 74
01/24/1993,01 05 09 18 19 20 24 28 30 34 38 39 46 47 52 63 66 71 77 79
01/23/1993,02 06 10 14 21 25 28 37 38 41 42 44 46 48 51 52 60 61 66 76
01/22/1993,01 02 08 10 11 14 15 21 23 24 35 36 45 50 52 63 65 72 73 80
01/21/1993,06 12 23 25 27 31 35 37 38 39 41 43 48 51 56 64 69 71 72 74
01/20/1993,10 13 16 22 29 30 36 39 46 47 51 54 56 58 65 68 75 76 77 79
01/19/1993,02 12 13 15 21 24 26 31 32 34 38 40 45 51 53 54 56 73 79 80
01/18/1993,02 11 12 22 33 34 36 38 42 48 49 51 55 57 60 63 65 69 76 77
01/17/1993,07 09 10 17 19 22 23 25 28 39 43 58 60 61 63 64 65 66 67 74
01/16/1993,03 04 05 06 08 09 11 13 24 28 31 35 39 41 43 46 51 61 64 69
01/15/1993,03 04 18 19 21 22 26 32 35 37 38 40 41 47 53 58 64 66 71 74
01/14/1993,03 08 14 25 27 34 39 41 44 45 47 49 52 54 59 65 68 76 77 79
01/13/1993,01 03 04 06 11 13 34 41 43 48 51 58 60 66 67 71 72 74 75 80
01/12/1993,01 03 04 08 09 13 16 20 24 25 46 50 52 63 66 69 72 75 76 78
01/11/1993,03 04 11 13 18 29 38 39 42 43 46 47 51 53 59 62 66 69 73 75
01/10/1993,01 09 13 16 21 25 31 41 44 49 52 53 57 63 65 66 67 70 77 78
01/09/1993,09 12 13 18 20 29 30 45 47 51 52 53 55 65 67 68 70 72 73 80
01/08/1993,01 11 14 17 19 28 32 36 47 49 50 52 53 55 58 62 64 69 77 79
01/07/1993,04 05 17 20 28 29 31 34 48 53 58 64 67 68 69 70 75 76 78 79
01/06/1993,06 09 16 22 23 24 33 37 41 47 50 54 57 59 60 61 64 66 72 78
01/05/1993,02 09 11 16 19 22 23 27 28 30 50 51 64 66 68 70 71 72 78 80
01/04/1993,01 08 13 15 21 23 27 35 41 42 44 47 48 51 56 57 60 68 72 75
01/03/1993,03 04 09 11 13 23 24 31 32 44 45 52 53 62 63 70 71 74 76 78
01/02/1993,01 08 09 17 19 25 29 33 36 38 41 42 45 49 54 60 61 64 70 80
01/01/1993,04 06 17 21 23 24 32 35 37 42 43 44 47 52 53 54 57 61 68 76
12/31/1992,01 02 06 10 13 15 21 24 30 34 41 43 59 65 68 73 75 78 79 80
12/30/1992,01 03 04 09 10 11 13 14 22 24 27 42 53 57 59 68 69 73 75 78
12/29/1992,01 02 15 24 27 30 35 36 37 40 43 44 53 57 60 61 73 74 75 76
12/28/1992,10 15 20 21 28 32 34 35 36 39 49 52 54 56 61 63 64 66 73 75
12/27/1992,02 08 19 22 25 26 27 32 35 42 43 44 56 60 61 65 69 72 73 79
12/26/1992,11 12 15 20 24 25 42 43 49 51 52 53 55 56 63 66 70 75 77 78
12/24/1992,06 16 18 21 25 37 39 41 42 44 50 53 54 57 59 61 62 69 75 80
12/23/1992,01 05 09 14 15 16 19 20 23 24 27 28 41 45 48 53 62 69 70 75
12/22/1992,05 10 13 19 20 24 35 37 38 42 46 48 51 56 61 63 65 69 70 76
12/21/1992,07 09 10 11 14 23 31 36 38 42 46 50 52 55 59 60 61 69 73 77
12/20/1992,02 05 09 10 13 14 18 21 23 24 36 51 52 53 58 60 64 68 71 80
12/19/1992,04 08 10 14 16 17 20 22 27 30 36 40 52 57 59 65 70 74 76 77
12/18/1992,07 10 12 14 18 19 23 48 57 58 59 60 63 65 73 74 75 77 78 80
12/17/1992,04 09 13 18 23 34 37 41 54 57 60 61 63 66 69 71 74 75 79 80
12/16/1992,01 07 08 09 13 15 16 20 21 25 27 29 32 43 47 53 57 60 63 78
12/15/1992,17 19 27 31 32 34 37 38 40 42 47 50 56 60 63 66 69 71 75 77
12/14/1992,03 18 23 24 25 26 28 29 33 34 37 44 53 57 59 61 68 70 71 74
12/13/1992,02 04 05 06 13 15 18 20 27 29 31 33 37 38 43 44 49 64 69 75
12/12/1992,02 03 04 07 08 15 16 19 26 38 41 42 43 45 56 60 61 65 68 70
12/11/1992,02 08 13 18 20 23 26 29 39 43 44 45 51 60 63 64 68 69 70 74
12/10/1992,04 07 11 18 19 21 22 24 26 27 31 33 38 40 41 50 53 58 64 67
12/09/1992,01 04 08 14 16 22 25 26 29 36 40 42 46 51 54 58 59 62 64 73
```

```
29      12/08/1992,09 11 15 16 19 22 24 31 32 35 40 43 44 45 48 51 52 56 60 78
30      12/07/1992,12 19 20 22 23 25 27 29 34 40 42 45 57 64 69 71 75 77 79 80
31      12/06/1992,05 08 10 16 21 26 29 37 39 45 46 52 53 56 60 61 69 73 75 77
32      12/05/1992,01 05 06 08 09 13 15 23 31 38 45 46 48 50 51 53 63 68 76 80
33      12/04/1992,20 23 26 28 31 32 33 38 39 41 43 46 49 53 54 58 65 74 77 80
34      12/03/1992,06 15 16 18 27 33 36 37 40 44 53 55 56 57 63 65 74 77 78 79
35      12/02/1992,04 05 08 10 15 18 23 29 38 41 43 44 52 53 57 60 68 70 72 80
36      12/01/1992,04 11 21 22 23 30 35 39 44 50 55 57 60 61 66 69 71 76 77 78
37      11/30/1992,04 14 15 16 21 24 25 34 41 56 57 58 59 62 64 65 67 68 78 79
38      11/29/1992,05 14 19 30 33 36 39 42 54 59 60 61 67 68 71 74 77 78 79 80
39      11/28/1992,17 22 23 26 31 32 39 40 41 42 44 47 49 51 57 63 64 66 72 77
40      11/27/1992,07 10 12 13 16 17 23 28 31 37 38 39 41 43 44 51 57 68 75 80
41      11/26/1992,10 13 15 17 20 27 32 36 40 41 43 51 60 67 68 70 71 73 74 77
42      11/25/1992,05 13 16 23 25 30 34 36 39 40 49 52 53 54 57 59 63 64 73 80
43      11/24/1992,04 08 09 19 21 35 41 43 44 46 49 59 60 64 69 71 72 73 76 80
44      11/23/1992,04 11 12 13 17 18 22 25 31 35 36 38 41 43 53 59 62 63 68 80
45      11/22/1992,02 07 08 10 16 17 25 29 31 33 38 39 41 54 55 56 57 61 63 69
46      11/21/1992,03 04 09 10 17 21 22 28 31 35 43 44 45 51 54 60 65 73 74 80
47      11/20/1992,01 02 03 10 11 21 22 23 30 34 39 43 57 58 62 64 65 71 77 80
48      11/19/1992,04 06 17 19 26 29 30 39 44 45 48 50 55 60 61 65 67 71 77 79
49      11/18/1992,07 20 25 28 31 34 42 43 48 52 61 62 63 64 65 66 68 71 72 80
50      11/17/1992,10 11 12 16 27 28 29 32 33 34 44 52 53 55 56 59 63 65 68 72
51      11/16/1992,01 10 11 12 15 18 24 27 32 38 39 53 56 64 65 67 68 71 72 75
52      11/15/1992,05 16 18 22 23 30 32 36 41 44 47 48 53 59 62 67 68 70 74 80
53      11/14/1992,01 06 13 15 19 27 29 39 40 48 49 51 52 54 58 59 62 69 72 77
54      11/13/1992,06 08 17 26 30 31 32 42 44 46 49 53 55 56 59 61 66 69 72 74
55      11/12/1992,08 11 14 16 21 27 31 36 37 39 40 42 45 55 67 70 73 74 75 77
56      11/11/1992,02 04 07 16 22 23 24 31 32 39 49 53 59 65 66 69 73 76 77 79
57      11/10/1992,01 06 07 09 15 17 20 21 29 36 37 45 50 56 60 61 71 75 76 79
58      11/09/1992,01 03 07 10 14 15 17 20 22 23 26 32 37 44 52 68 71 75 76 79
59      11/08/1992,08 21 23 26 30 34 37 41 43 50 51 52 60 62 65 66 74 75 77 79
60      11/07/1992,01 03 05 16 17 19 26 29 30 34 41 43 47 52 53 54 59 63 67 69
61      11/06/1992,03 04 14 16 23 24 27 30 33 34 36 40 42 48 50 51 63 65 70 71
62      11/05/1992,06 08 11 14 17 19 21 23 30 34 39 46 49 50 58 66 67 70 76 77
63      11/04/1992,04 06 11 14 23 26 27 28 29 32 37 47 50 58 59 61 66 67 76 78
64      11/03/1992,01 05 12 14 15 16 17 28 33 34 39 47 50 52 59 61 62 64 73 80
65      11/02/1992,04 12 15 20 21 23 35 36 37 42 44 48 56 59 60 69 74 76 77 79
66      11/01/1992,05 13 16 18 23 32 35 43 44 53 57 59 66 70 71 72 75 76 78 79
67      10/31/1992,01 10 16 19 22 27 29 32 36 37 38 53 56 57 60 67 68 69 72 73
68      10/30/1992,04 19 24 25 29 31 32 40 41 42 44 49 50 54 56 66 69 70 71 76
69      10/29/1992,01 02 12 13 22 26 29 33 36 37 44 47 48 57 58 62 69 70 73 79
70      10/28/1992,09 12 16 18 21 23 26 35 38 44 46 47 50 51 53 59 62 63 68 78
71      10/27/1992,06 13 17 21 24 31 32 36 37 45 46 51 55 57 60 64 69 72 74 77
72      10/26/1992,02 05 08 10 13 17 23 24 25 28 30 34 38 40 49 52 57 60 62 77
73      10/25/1992,03 04 15 18 21 25 26 27 28 29 39 50 55 58 59 62 67 70 73 77
74      10/24/1992,03 05 12 13 25 32 38 43 44 47 48 49 51 59 63 67 71 74 77 78
75      10/23/1992,05 12 21 22 26 31 35 45 50 51 61 63 65 66 70 72 73 76 77 79
76      10/22/1992,05 15 16 24 26 29 30 31 32 35 40 45 47 58 59 62 67 73 75 79
77      10/21/1992,04 11 15 24 29 32 33 36 38 42 44 49 55 63 64 66 70 73 77 79
78      10/20/1992,04 09 10 11 14 16 22 23 30 33 38 39 42 49 50 53 60 72 74 76
79      10/19/1992,09 16 27 29 31 34 36 40 41 42 49 50 56 57 61 65 66 73 77 80
80      10/18/1992,03 15 18 19 26 27 30 34 35 37 44 46 56 58 60 61 62 65 68 79
81      10/17/1992,04 07 12 15 17 22 23 26 27 31 41 42 45 49 53 54 70 71 73 79
82      10/16/1992,02 05 16 18 19 20 21 27 28 35 36 49 51 55 57 61 66 67 73 79
```

```
83      10/15/1992,01 06 07 11 12 16 23 25 29 37 49 50 55 57 61 63 68 69 76 78
84      10/14/1992,01 04 19 22 24 27 29 32 35 36 41 52 54 55 58 62 72 73 75 77
85      10/13/1992,03 07 08 13 20 22 28 30 36 44 49 50 52 55 66 69 70 74 77 78
86      10/12/1992,02 09 12 16 19 23 45 47 53 59 60 62 63 65 69 70 71 72 73 74
87      10/11/1992,04 06 09 14 21 24 26 35 36 43 44 52 62 63 67 72 74 75 77 78
88      10/10/1992,05 06 16 18 23 29 33 34 35 44 45 53 60 61 62 63 71 73 76 77
89      10/09/1992,03 04 06 07 14 16 17 21 26 27 30 34 35 41 43 46 55 64 74 76
90      10/08/1992,07 08 09 11 14 25 29 31 33 38 42 48 49 50 63 64 65 70 72 74
91      10/07/1992,01 04 08 16 19 23 25 29 37 39 40 42 43 54 58 62 70 72 75 77
92      10/06/1992,05 07 08 09 19 20 21 27 29 31 35 37 44 49 54 56 61 64 70 78
93      10/05/1992,02 03 04 07 08 14 23 25 30 34 40 42 47 52 54 60 63 70 74 80
94      10/04/1992,01 05 08 20 22 32 33 40 43 44 46 47 51 64 65 74 75 78 79 80
95      10/03/1992,08 13 19 25 26 28 31 34 42 44 45 46 52 53 62 63 65 70 71 77
96      10/02/1992,06 16 18 27 29 30 31 32 37 44 45 47 51 53 57 68 69 76 78 79
97      10/01/1992,01 14 20 21 24 28 31 33 38 41 45 51 52 55 71 73 77 78 79 80
98      09/30/1992,01 06 07 09 14 19 25 28 29 40 42 46 48 49 58 59 63 67 74 80
99      09/29/1992,13 19 22 25 27 32 33 34 38 45 46 49 54 57 60 61 66 70 77 79
00      09/28/1992,06 11 17 33 34 35 36 44 47 49 55 57 58 59 60 64 69 70 74 75
01      09/27/1992,08 10 16 22 24 26 29 34 35 38 41 44 50 53 57 69 71 73 75 77
02      09/26/1992,05 09 12 14 15 17 29 33 36 38 40 45 47 52 54 55 62 64 68 78
03      09/25/1992,02 04 08 09 10 17 18 23 28 37 41 42 49 60 61 65 70 75 78 79
04      09/24/1992,02 03 10 12 13 16 18 19 20 24 25 27 29 41 45 47 53 54 59 68
05      09/23/1992,11 12 15 17 18 20 22 32 42 46 48 49 50 51 52 66 73 78 79 80
06      09/22/1992,02 07 08 12 14 15 17 22 23 25 29 34 37 41 42 49 59 67 78 80
07      09/21/1992,05 06 09 14 16 18 32 38 40 44 50 52 58 59 62 71 73 74 78 79
08      09/20/1992,05 07 10 17 19 20 22 23 29 36 38 41 47 51 57 61 64 67 74 77
09      09/19/1992,04 11 19 21 25 39 45 49 52 54 55 58 62 65 66 68 70 73 76 77
10      09/18/1992,06 07 11 14 26 28 33 35 36 45 51 58 59 64 66 68 70 73 75 80
11      09/17/1992,05 06 09 12 22 24 25 27 30 38 41 45 51 56 60 65 68 69 73 79
12      09/16/1992,15 17 25 26 28 29 34 35 37 42 47 51 54 57 60 61 66 68 70 75
13      09/15/1992,01 04 08 16 19 20 21 23 26 28 30 37 38 40 47 51 61 62 64 79
14      09/14/1992,03 10 11 15 18 21 25 27 32 44 46 50 52 53 54 59 62 67 77 79
15      09/13/1992,03 07 12 15 17 20 27 31 33 36 40 42 51 54 63 64 68 73 74 79
16      09/12/1992,02 06 07 15 16 17 18 23 26 33 34 44 49 55 56 58 64 67 68 76
17      09/11/1992,04 05 08 09 11 21 22 37 38 40 43 44 51 52 59 63 64 68 74 79
18      09/10/1992,02 05 08 09 15 22 25 27 31 38 41 49 56 61 62 67 70 78 79 80
19      09/09/1992,01 02 08 10 14 15 19 20 28 29 39 40 41 50 52 57 63 67 70 80
20      09/08/1992,05 07 08 12 13 15 16 29 39 42 44 52 56 57 64 65 67 71 72 73
21      09/07/1992,06 08 17 19 20 22 37 38 42 46 49 60 63 64 71 72 74 76 78 79
22      09/06/1992,05 10 13 14 25 26 32 37 39 42 44 49 50 51 56 57 62 66 72 80
23      09/05/1992,07 10 12 16 17 18 26 28 31 37 40 42 43 48 60 64 66 76 79 80
24      09/04/1992,02 11 13 20 21 23 31 35 41 44 45 47 49 51 55 61 62 63 70 71
25      09/03/1992,03 08 14 15 22 28 32 35 38 39 45 48 49 53 58 68 74 75 76 79
26      09/02/1992,08 15 22 24 25 26 29 30 32 35 40 41 51 56 57 67 69 71 73 75
27      09/01/1992,08 10 19 21 24 29 37 41 44 46 49 56 57 62 64 66 68 72 73 74
28      08/31/1992,12 13 21 22 23 28 32 39 45 49 51 53 55 56 61 66 72 78 79 80
29      08/30/1992,06 12 21 22 25 26 29 32 36 38 43 47 60 61 66 67 68 72 73 77
30      08/29/1992,04 07 09 10 11 14 16 19 21 26 27 32 40 54 63 64 71 73 74 78
31      08/28/1992,05 09 11 19 31 34 35 36 37 40 42 45 46 48 55 57 60 66 72 73
32      08/27/1992,01 02 04 06 10 14 19 22 25 28 29 32 33 39 48 61 65 71 72 78
33      08/26/1992,02 03 05 14 17 21 22 23 27 29 31 35 36 59 63 68 69 71 75 78
34      08/25/1992,13 24 28 30 32 34 37 40 41 42 43 44 46 51 53 58 67 69 73 77
35      08/24/1992,01 09 15 19 21 24 26 28 29 38 39 42 43 48 52 61 63 68 74 79
36      08/23/1992,02 05 06 08 16 22 29 34 37 48 49 52 54 58 63 67 72 74 77 80
```

```
08/22/1992,01 04 05 06 07 09 17 22 26 28 29 40 44 46 50 63 67 70 73 75
08/21/1992,01 07 10 21 22 24 25 31 32 33 37 38 42 48 49 54 64 66 71 75
08/20/1992,03 09 14 18 24 25 28 30 34 35 36 38 42 45 52 57 63 67 72 79
08/19/1992,01 02 04 05 06 17 27 29 31 34 37 39 46 48 51 58 60 72 73 77
08/18/1992,05 07 08 10 14 15 23 24 28 33 49 56 58 60 62 65 69 73 76 79
08/17/1992,03 08 12 16 18 20 22 25 28 29 31 35 37 44 46 56 61 65 74 75
08/16/1992,03 04 05 08 10 13 15 18 21 23 38 41 43 53 55 67 73 75 76 79
08/15/1992,01 09 12 13 14 15 23 24 26 27 28 32 39 41 43 47 48 54 70 79
08/14/1992,05 08 12 13 20 22 24 27 34 35 39 41 48 50 56 57 61 63 67 70
08/13/1992,01 06 20 21 30 31 32 34 35 37 42 43 45 53 56 60 63 64 68 75
08/12/1992,01 13 14 15 21 25 26 30 31 33 40 43 50 53 61 63 64 66 76 77
08/11/1992,05 11 12 13 24 32 34 44 45 46 49 50 51 52 53 60 63 66 67 80
08/10/1992,04 06 14 15 24 25 26 29 32 42 43 45 49 50 52 62 73 75 76 77
08/09/1992,04 09 13 16 18 20 28 32 33 35 37 41 45 46 52 56 63 65 66 69
08/08/1992,01 07 08 13 15 16 17 25 26 34 35 38 49 55 58 61 62 67 73 78
08/07/1992,02 03 04 08 12 15 25 31 34 35 41 42 45 47 52 60 62 69 71 80
08/06/1992,01 10 21 28 29 32 38 45 46 47 53 55 57 58 59 60 61 69 71 73
08/05/1992,04 07 10 12 13 16 21 29 37 38 45 47 48 51 56 66 70 72 74 77
08/04/1992,12 13 16 24 27 29 30 32 33 36 37 39 48 54 58 59 66 68 74 79
08/03/1992,02 03 07 08 10 17 23 27 32 34 37 38 42 45 46 51 63 65 70 80
08/02/1992,05 22 27 28 31 34 35 38 39 42 53 55 60 62 63 68 71 72 74 79
08/01/1992,02 03 06 11 13 16 18 25 30 35 39 44 46 48 50 55 60 66 72 75
07/31/1992,01 04 10 11 13 19 20 25 30 35 40 47 54 56 58 62 64 68 74 75
07/30/1992,08 13 18 19 24 30 33 35 41 42 44 49 50 53 58 64 67 69 76 78
07/29/1992,03 08 12 15 23 32 42 48 49 53 56 57 58 59 60 61 68 71 72 77
07/28/1992,05 09 10 18 19 26 30 34 39 45 49 50 54 58 59 60 61 63 64 77
07/27/1992,01 03 08 11 12 14 21 25 27 29 33 34 41 48 49 52 67 73 74 75
07/26/1992,07 09 11 13 14 18 21 24 26 31 35 36 45 54 58 64 66 69 77 80
07/25/1992,02 10 12 13 15 22 24 35 40 44 46 48 52 53 55 58 59 62 63 68
07/24/1992,04 06 15 34 37 41 42 47 48 56 57 60 61 62 64 65 74 76 77 80
07/23/1992,01 02 05 13 18 20 22 25 27 29 31 39 43 61 63 64 68 70 71 79
07/22/1992,05 06 08 20 24 25 27 28 30 32 39 49 50 60 69 70 72 74 77 78
07/21/1992,03 04 05 06 11 14 18 19 33 34 43 47 49 54 55 57 60 63 64 72
07/20/1992,02 03 04 11 12 13 21 27 28 37 38 41 47 52 55 57 74 75 76 77
07/19/1992,01 03 08 11 17 27 34 39 40 42 49 50 55 56 69 72 73 74 77 78
07/18/1992,01 11 15 24 28 34 37 38 39 40 49 51 53 56 59 62 70 73 78 79
07/17/1992,07 09 12 14 15 22 27 30 36 37 38 42 45 47 49 50 57 61 66 78
07/16/1992,01 03 07 15 24 27 29 34 38 40 44 50 54 58 62 69 72 74 76 80
07/15/1992,01 05 08 13 21 22 25 27 29 31 33 36 46 50 51 61 69 76 77 78
07/14/1992,06 08 14 17 21 22 28 33 35 37 39 41 43 45 51 60 62 68 70 72
07/13/1992,03 05 06 12 20 25 35 36 40 42 43 50 55 57 58 66 68 70 77 80
07/12/1992,10 12 14 18 22 23 29 30 39 40 41 42 44 48 49 53 55 58 65 74
07/11/1992,06 20 21 24 25 31 34 36 37 38 39 41 43 45 46 57 58 63 66 78
07/10/1992,02 08 09 12 13 15 18 19 20 29 32 33 35 36 47 55 59 62 66 74
07/09/1992,07 08 10 15 17 24 38 48 52 53 56 57 59 60 64 68 71 75 79 80
07/08/1992,01 02 04 05 06 12 20 22 37 39 41 49 53 56 57 59 64 73 75 79
07/07/1992,04 09 10 11 12 15 17 21 22 23 26 32 33 52 54 55 57 65 69 78
07/06/1992,03 05 07 14 18 21 23 28 30 42 43 49 52 54 57 65 66 69 70 75
07/05/1992,01 06 07 08 09 12 14 15 37 42 56 58 60 61 62 70 71 74 76 77
07/04/1992,04 06 08 10 17 20 22 29 32 33 36 42 45 48 51 53 56 68 72 77
07/03/1992,05 07 10 12 18 24 26 29 30 33 41 45 50 53 59 61 62 64 68 80
07/02/1992,06 12 15 17 21 23 29 31 32 34 41 42 49 59 65 69 72 77 78 79
07/01/1992,02 04 08 10 14 17 18 19 21 26 27 36 38 39 40 51 55 59 61 71
06/30/1992,16 23 25 36 37 40 42 47 48 49 51 55 56 58 60 61 64 67 70 74
```

```
91      06/29/1992,04 05 06 10 14 29 30 33 37 38 44 46 50 55 57 59 61 64 68 69
92      06/28/1992,02 09 10 11 14 17 21 24 28 31 36 37 39 49 60 62 64 66 68 70
93      06/27/1992,06 07 09 12 18 20 22 24 29 31 32 33 37 41 43 48 53 56 67 76
94      06/26/1992,03 07 11 17 21 28 29 30 35 36 43 46 55 57 59 64 65 71 73 74
95      06/25/1992,02 03 04 08 09 19 23 24 25 33 35 37 42 49 55 56 59 60 63 78
96      06/24/1992,01 02 04 06 15 19 21 25 26 32 33 36 37 43 49 55 62 63 68 72
97      06/23/1992,02 03 26 29 36 38 40 41 46 48 50 53 57 60 62 63 67 72 76 79
98      06/22/1992,07 10 12 13 14 18 19 23 25 27 32 36 37 39 43 45 47 49 57 68
99      06/21/1992,11 18 21 24 26 28 29 31 32 34 38 39 46 48 53 60 61 65 67 71
00      06/20/1992,03 05 09 33 37 40 49 50 51 53 54 61 64 65 66 69 70 76 77 78
01      06/19/1992,17 26 27 33 34 36 37 40 41 45 48 53 54 55 59 64 69 72 73 80
02      06/18/1992,02 04 07 10 14 15 24 27 28 36 37 45 48 49 53 61 64 65 71 78
03      06/17/1992,03 07 08 13 14 18 20 24 26 30 31 35 40 52 66 67 70 71 72 74
04      06/16/1992,03 04 07 12 14 16 17 23 38 41 45 46 50 56 62 65 66 69 74 78
05      06/15/1992,01 02 17 25 27 33 40 45 46 49 50 51 57 62 65 70 71 72 77 78
06      06/14/1992,02 05 10 14 18 26 38 39 40 44 47 48 49 52 53 62 66 71 72 76
07      06/13/1992,04 08 11 17 26 29 31 37 38 42 49 52 54 58 61 65 71 75 79 80
08      06/12/1992,02 06 09 10 15 17 18 20 24 26 29 38 41 51 59 60 61 66 72 79
09      06/11/1992,03 04 15 16 17 18 27 28 31 37 40 42 54 61 64 65 68 69 78 79
10      06/10/1992,03 07 09 12 22 31 35 38 39 47 49 50 51 52 53 58 62 63 66 68
11      06/09/1992,07 15 18 19 28 30 31 32 38 40 42 44 46 57 58 64 66 70 73 74
12      06/08/1992,03 05 19 20 21 24 26 29 31 37 39 43 45 46 59 60 67 72 76 79
13      06/07/1992,01 03 07 08 14 16 18 28 29 39 40 41 45 50 59 63 65 69 70 75
14      06/06/1992,02 03 11 14 22 27 32 37 39 41 49 57 60 61 62 64 67 71 72 79
15      06/05/1992,02 07 08 11 23 27 28 32 33 36 39 44 45 50 59 62 65 70 77 80
16      06/04/1992,08 12 17 19 20 22 29 31 32 33 36 42 46 52 53 55 58 68 70 78
17      06/03/1992,02 05 08 11 17 18 23 43 44 50 54 55 56 60 61 70 75 78 79 80
18      06/02/1992,01 06 08 10 30 33 40 43 46 47 49 51 56 60 61 62 63 64 77 78
19      06/01/1992,04 07 15 18 23 33 39 40 42 46 52 55 56 58 60 69 72 75 78 79
20      05/31/1992,01 04 05 08 13 14 20 25 26 30 42 43 44 45 47 54 58 59 67 78
21      05/30/1992,09 10 14 19 24 25 30 31 35 38 44 46 48 50 51 55 58 66 75 80
22      05/29/1992,14 17 20 27 28 30 31 32 42 44 49 56 60 67 70 72 74 75 76 77
23      05/28/1992,01 05 10 21 23 29 31 36 41 42 43 46 56 59 61 62 75 76 79 80
24      05/27/1992,01 07 10 16 24 25 29 30 34 38 39 43 48 55 56 61 67 69 73 80
25      05/26/1992,04 14 21 25 28 31 37 39 41 42 43 47 48 51 54 57 61 69 73 76
26      05/25/1992,01 04 11 14 21 26 32 40 41 43 44 45 48 51 53 59 60 70 72 73
27      05/24/1992,07 13 18 20 21 33 42 43 46 47 49 53 55 56 62 65 69 75 77 79
28      05/23/1992,08 09 13 15 16 20 22 27 32 34 35 36 38 45 60 62 66 70 77 80
29      05/22/1992,05 06 07 13 14 18 24 31 37 39 49 52 53 55 58 59 67 74 78 80
30      05/21/1992,10 12 14 15 20 23 26 27 30 31 37 44 49 51 55 59 66 68 71 80
31      05/20/1992,02 06 18 20 22 28 37 39 44 45 53 54 61 63 64 68 75 76 77 80
32      05/19/1992,11 13 14 15 23 25 26 27 28 31 36 49 51 54 55 58 61 69 77 80
33      05/18/1992,02 04 06 17 18 24 29 30 36 44 45 50 53 54 59 66 71 73 75 79
34      05/17/1992,01 07 21 23 33 36 39 40 43 44 49 51 63 64 67 68 69 72 74 77
35      05/16/1992,02 04 05 10 11 12 18 19 29 33 35 48 52 53 56 59 60 61 67 71
36      05/15/1992,04 05 08 11 14 16 20 26 31 32 37 42 43 51 54 58 60 68 73 77
37      05/14/1992,04 06 11 13 16 20 25 27 28 31 32 34 47 49 52 53 60 62 64 78
38      05/13/1992,01 09 15 17 31 35 39 46 54 56 60 66 69 70 71 72 73 76 79 80
39      05/12/1992,02 05 06 10 15 20 22 28 30 34 43 46 48 57 62 69 73 75 79 80
40      05/11/1992,04 14 18 19 30 38 39 42 49 53 55 56 58 59 61 67 69 72 78 79
41      05/10/1992,04 07 08 15 18 20 27 36 39 44 45 52 55 66 67 71 73 75 78 79
42      05/09/1992,02 04 13 16 18 19 35 38 48 49 53 54 55 57 61 63 66 68 71 80
43      05/08/1992,05 12 18 23 24 27 29 31 34 38 44 45 48 50 55 56 62 70 75 76
44      05/07/1992,07 08 09 11 20 26 31 38 41 49 52 53 61 62 63 64 66 68 69 73
```

```
05/06/1992,02 03 08 13 14 20 22 23 25 32 34 44 51 55 62 65 67 70 71 72
05/05/1992,15 18 20 24 27 31 37 39 43 45 48 54 55 56 65 67 70 73 76 77
05/04/1992,02 05 08 15 16 18 21 34 38 43 50 53 54 55 58 63 66 69 73 77
05/03/1992,02 06 12 14 19 25 32 34 37 43 45 46 50 55 59 60 66 70 74 76
05/02/1992,02 11 12 13 16 20 22 25 28 29 31 48 50 53 62 70 72 77 79 80
05/01/1992,05 14 19 30 31 33 39 44 47 52 59 61 63 64 65 71 72 74 75 76
04/30/1992,02 06 07 09 11 24 26 27 29 33 35 45 49 55 62 63 65 66 72 75
04/29/1992,04 08 11 12 15 18 21 22 30 34 40 48 51 53 55 56 62 68 74 78
04/28/1992,01 02 03 09 22 23 31 33 34 38 42 48 51 57 62 67 70 73 78 79
04/27/1992,04 05 06 08 11 14 38 39 45 47 48 52 63 68 70 71 73 74 78 80
04/26/1992,04 07 09 10 12 14 34 41 44 47 50 52 54 60 62 66 67 73 74 76
04/25/1992,03 10 12 18 21 39 44 45 49 50 51 55 57 59 62 65 66 76 77 78
04/24/1992,01 02 08 09 16 17 18 28 32 35 36 38 40 45 57 61 62 71 78 79
04/23/1992,07 10 14 15 19 20 26 43 44 48 51 53 55 57 61 68 69 71 76 77
04/22/1992,02 08 09 10 16 20 32 35 38 43 44 45 52 55 59 68 69 70 75 77
04/21/1992,06 11 19 22 23 27 28 29 30 34 41 45 46 55 58 59 68 70 77 78
04/20/1992,02 04 05 12 20 25 30 31 38 41 42 51 57 65 66 67 71 73 78 80
04/19/1992,01 06 09 13 16 17 21 26 29 31 48 50 64 67 72 75 76 78 79 80
04/18/1992,05 16 17 18 23 30 38 39 41 46 52 53 57 61 62 65 68 74 76 78
04/17/1992,03 06 07 13 19 23 31 33 36 37 44 47 51 59 63 65 69 74 76 78
04/16/1992,02 03 13 14 22 24 30 31 35 40 41 46 48 54 58 59 60 71 79 80
04/15/1992,01 04 05 07 15 17 20 21 34 37 41 49 51 52 56 64 66 67 68 79
04/14/1992,21 25 28 32 33 36 37 40 43 45 48 50 54 62 63 64 66 67 68 72
04/13/1992,04 05 23 25 28 32 39 43 47 49 53 55 60 62 63 65 71 75 78 80
04/12/1992,02 06 07 08 10 11 13 14 17 21 22 23 26 31 36 37 42 55 64 72
04/11/1992,02 03 06 09 12 14 23 27 28 29 31 41 42 53 57 61 62 72 78 79
04/10/1992,03 06 09 11 12 16 18 25 30 38 42 43 46 55 63 64 69 71 75 79
04/09/1992,02 04 05 08 21 25 28 29 30 31 37 42 47 50 57 64 65 66 67 76
04/08/1992,03 10 11 14 16 17 31 32 34 36 38 40 41 51 56 59 61 69 75 80
04/07/1992,04 09 13 22 25 29 31 46 54 58 59 66 68 69 70 71 74 76 77 78
04/06/1992,01 05 08 19 20 25 28 29 30 31 33 35 37 40 44 46 48 70 73 78
04/05/1992,02 06 08 16 22 26 31 35 36 40 43 46 48 59 60 65 68 72 74 77
04/04/1992,03 08 15 18 19 20 24 28 35 42 44 46 51 52 60 62 65 67 78 80
04/03/1992,02 03 06 09 10 21 22 26 35 36 39 42 50 53 58 60 66 68 71 72
04/02/1992,02 06 11 13 17 18 21 22 25 31 34 36 39 40 47 57 62 63 73 80
04/01/1992,05 08 14 15 17 22 24 33 37 41 43 55 56 62 63 64 69 76 78 80
03/31/1992,02 08 09 12 14 28 30 35 37 39 40 42 44 45 55 56 60 61 74 75
03/30/1992,09 10 22 26 28 36 37 39 42 45 51 52 55 56 57 58 59 60 63 75
03/29/1992,01 03 05 08 13 21 24 27 33 34 36 40 46 52 54 66 72 73 75 76
03/28/1992,01 02 03 18 20 22 23 32 36 38 47 52 54 59 62 70 71 72 74 75
03/27/1992,02 03 06 17 19 28 34 35 38 44 46 47 50 57 61 62 64 67 70 77
03/26/1992,02 03 04 11 13 14 17 20 23 26 30 31 40 50 55 57 62 66 73 78
03/25/1992,03 04 07 10 15 24 25 31 32 34 35 41 43 48 49 56 69 71 73 77
03/24/1992,03 04 06 13 14 16 17 18 21 31 45 48 56 57 58 61 71 76 79 80
03/23/1992,04 06 07 09 14 16 23 31 34 36 42 44 45 46 63 64 66 67 74 75
03/22/1992,04 06 16 17 20 21 22 37 40 41 42 46 49 51 53 61 75 76 79 80
03/21/1992,03 08 09 18 23 27 33 39 40 44 46 52 56 58 62 63 68 69 75 79
03/20/1992,01 02 04 10 15 19 22 25 27 31 38 40 48 51 52 56 59 61 72 74
03/19/1992,01 07 10 11 17 18 21 22 24 26 30 31 33 44 45 48 52 56 69 80
03/18/1992,08 09 12 15 20 23 27 40 44 47 50 51 52 61 63 65 69 70 73 76
03/17/1992,01 04 05 08 10 11 12 14 18 20 26 36 44 49 52 57 62 66 67 72
03/16/1992,01 08 11 16 19 22 24 32 38 44 45 47 48 59 63 69 73 74 77 79
03/15/1992,01 02 04 10 17 20 21 23 35 46 48 49 54 63 71 72 75 77 78 80
03/14/1992,02 04 11 17 19 22 27 28 37 44 48 49 57 58 60 61 63 67 71 80
```

99 03/13/1992,04 09 10 14 25 33 35 36 37 39 41 44 45 50 55 57 65 67 69 79
00 03/12/1992,01 02 11 13 15 17 20 29 37 44 46 48 55 57 59 61 63 65 67 71
01 03/11/1992,01 03 05 07 09 23 30 34 39 41 47 57 61 64 66 67 68 70 73 77
02 03/10/1992,07 09 12 17 20 21 23 24 28 39 49 57 58 62 64 72 73 76 78 80
03 03/09/1992,09 11 12 19 23 24 26 27 28 43 45 47 52 53 58 63 68 71 79 80
04 03/08/1992,06 07 10 15 16 19 26 27 31 39 42 44 52 54 55 60 65 67 72 75
05 03/07/1992,03 05 08 09 14 15 16 18 27 34 38 44 45 51 54 63 65 68 70 71
06 03/06/1992,03 05 06 12 18 36 40 41 43 48 51 55 56 64 66 71 72 73 79 80
07 03/05/1992,03 07 08 11 16 18 25 26 27 31 34 36 45 46 49 51 62 71 78 80
08 03/04/1992,03 05 08 12 15 19 20 23 26 32 36 37 45 47 51 52 64 66 76 78
09 03/03/1992,03 07 08 09 10 11 14 15 18 28 30 35 46 51 55 67 73 74 76 79
10 03/02/1992,04 07 10 12 14 16 20 22 23 24 38 40 47 48 51 59 63 76 77 80
11 03/01/1992,08 09 12 14 18 24 32 35 43 45 48 52 55 62 64 67 72 74 77 79
12 02/29/1992,02 04 05 08 12 19 24 27 29 34 35 36 39 40 43 47 51 52 58 80
13 02/28/1992,06 08 09 16 20 22 23 24 32 35 40 42 47 54 57 58 66 69 75 79
14 02/27/1992,01 09 10 22 25 29 36 40 43 44 50 53 54 59 69 70 75 76 78 79
15 02/26/1992,02 05 09 13 15 16 17 19 24 29 34 36 38 44 46 54 56 61 70 75
16 02/25/1992,11 12 17 20 21 22 23 24 26 28 31 41 43 45 47 50 66 74 75 79
17 02/24/1992,05 06 11 24 25 30 36 37 42 43 48 52 54 59 66 70 72 74 76 78
18 02/23/1992,01 06 07 12 14 17 22 23 27 29 36 39 43 52 59 61 73 74 76 80
19 02/22/1992,05 08 16 17 22 26 29 30 45 51 54 60 61 62 64 67 68 69 74 75
20 02/21/1992,02 06 08 13 20 25 32 33 36 44 49 51 58 66 67 73 74 75 78 79
21 02/20/1992,01 03 09 10 15 17 21 36 41 43 44 46 52 53 62 65 66 67 69 74
22 02/19/1992,02 06 07 08 25 28 35 38 41 45 56 57 58 62 64 65 67 68 69 72
23 02/18/1992,02 05 14 16 18 20 22 27 33 39 41 49 53 55 57 59 61 62 72 77
24 02/17/1992,03 07 09 10 14 15 16 23 25 27 30 31 36 44 47 48 55 59 68 77
25 02/16/1992,01 09 12 14 15 23 25 26 30 31 32 34 35 37 39 52 65 66 70 76
26 02/15/1992,06 07 16 17 25 35 38 40 45 47 48 60 65 69 70 71 74 78 79 80
27 02/14/1992,14 17 19 23 24 27 32 34 39 40 44 50 51 56 59 61 64 69 75 78
28 02/13/1992,03 06 07 13 14 32 37 39 40 45 47 55 65 66 68 74 76 77 78 79
29 02/12/1992,02 06 07 10 13 16 17 19 20 23 24 26 37 40 47 50 52 53 76 78
30 02/11/1992,02 04 14 19 22 24 25 28 33 45 46 49 50 52 54 60 68 70 71 72
31 02/10/1992,03 15 28 29 31 34 36 40 42 45 53 64 68 69 70 72 73 74 78 79
32 02/09/1992,13 14 15 16 17 22 23 34 45 46 48 51 58 62 63 64 66 67 72 76
33 02/08/1992,06 10 11 12 19 24 26 30 31 34 38 44 50 55 59 64 70 73 74 80
34 02/07/1992,01 02 09 14 15 17 22 33 37 38 39 41 43 50 52 71 75 77 79 80
35 02/06/1992,04 06 08 16 19 20 21 23 28 39 41 47 49 52 58 62 64 66 75 80
36 02/05/1992,02 03 04 07 08 11 13 27 33 35 36 40 42 43 44 46 47 68 73 76
37 02/04/1992,02 12 14 18 19 24 26 27 28 31 32 37 40 48 53 55 62 68 75 78
38 02/03/1992,05 06 09 21 26 27 35 38 43 47 49 51 56 58 60 62 64 66 73 74
39 02/02/1992,09 11 21 24 26 36 37 40 42 46 53 54 59 60 68 72 73 78 79 80
40 02/01/1992,01 05 10 12 19 26 28 33 35 38 42 48 49 52 54 59 61 65 68 80
41 01/31/1992,05 06 15 20 21 24 25 29 30 32 44 48 54 60 66 69 75 77 78 79
42 01/30/1992,05 07 09 24 25 27 29 30 37 41 42 45 48 62 63 67 69 70 71 75
43 01/29/1992,03 05 08 12 13 21 25 30 32 38 41 49 55 58 61 68 71 74 75 80
44 01/28/1992,01 04 05 06 14 19 21 23 27 28 37 44 45 50 52 53 67 73 75 76
45 01/27/1992,07 16 18 21 25 31 32 38 42 44 45 50 53 54 59 63 66 71 75 80
46 01/26/1992,02 05 07 14 20 21 29 31 32 33 43 45 50 51 53 56 58 65 67 69
47 01/25/1992,02 05 11 13 18 23 25 27 29 30 35 40 44 49 51 53 60 65 72 77
48 01/24/1992,06 07 09 13 17 26 28 29 30 33 41 44 46 47 52 57 58 69 70 79
49 01/23/1992,01 03 17 19 21 22 27 30 34 39 40 47 48 51 59 64 65 66 69 70
50 01/22/1992,01 05 14 25 32 43 47 48 49 54 55 59 61 64 65 67 71 72 74 77
51 01/21/1992,02 11 12 14 25 28 33 35 42 46 47 50 52 53 57 58 60 70 71 73
52 01/20/1992,02 05 08 21 22 24 25 27 28 29 31 34 43 44 46 52 53 73 76 80

```
53      01/19/1992,01 03 14 19 26 36 38 39 40 45 46 50 51 53 55 56 62 63 67 69
54      01/18/1992,03 08 14 15 17 30 33 34 37 38 40 44 49 50 54 61 66 71 77 78
55      01/17/1992,02 08 10 14 17 22 23 36 40 44 50 53 55 58 63 70 72 74 76 77
56      01/16/1992,09 11 14 17 21 24 26 30 35 42 45 51 52 55 59 60 62 65 68 77
57      01/15/1992,02 08 11 13 14 15 16 18 27 30 34 39 45 47 49 50 51 55 63 70
58      01/14/1992,05 11 15 16 20 26 31 41 44 47 50 53 58 63 67 69 71 74 75 76
59      01/13/1992,03 07 13 14 24 30 39 44 45 47 48 54 57 63 67 70 74 76 77 80
60      01/12/1992,02 06 10 11 12 15 17 21 28 29 31 36 42 51 56 68 71 72 75 77
61      01/11/1992,03 08 13 17 18 24 26 28 29 35 38 43 50 59 61 66 67 72 74 78
62      01/10/1992,01 02 04 09 12 13 16 19 26 27 35 36 38 39 47 53 55 61 63 66
63      01/09/1992,02 14 23 24 29 33 36 39 45 47 48 49 50 60 64 66 67 74 76 77
64      01/08/1992,02 10 11 14 18 22 23 26 28 32 38 43 47 48 56 66 69 73 75 78
65      01/07/1992,01 09 15 18 19 20 22 27 28 31 33 40 47 52 55 60 62 63 72 77
66      01/06/1992,03 04 08 16 17 20 31 37 41 47 51 54 55 57 66 68 71 75 77 78
67      01/05/1992,01 02 04 08 13 19 23 34 40 52 54 55 56 58 60 68 72 73 76 78
68      01/04/1992,01 02 07 11 21 23 25 29 31 33 37 42 48 59 61 66 67 68 69 73
69      01/03/1992,02 15 22 23 24 36 37 39 44 46 47 49 50 51 55 60 61 65 73 76
70      01/02/1992,06 16 24 25 26 36 38 40 45 47 48 50 52 56 58 60 69 72 78 80
71      01/01/1992,04 07 10 11 14 17 19 35 39 41 43 45 51 55 56 64 71 72 74 79
72      12/31/1991,01 03 08 20 21 23 27 29 30 38 39 47 56 58 59 63 67 76 78 79
73      12/30/1991,06 08 09 11 14 20 25 28 30 36 40 44 49 51 53 56 73 74 76 80
74      12/29/1991,10 12 13 15 28 29 33 37 38 41 46 47 53 58 61 63 66 71 72 76
75      12/28/1991,08 17 24 25 26 28 33 39 51 55 56 62 65 67 72 73 75 76 77 79
76      12/27/1991,02 09 10 15 18 32 34 36 37 38 43 48 50 52 56 65 66 70 78 80
77      12/26/1991,02 03 07 13 16 18 19 24 27 41 46 49 56 58 62 67 70 72 75 79
78      12/24/1991,02 09 15 18 21 24 31 32 35 39 41 42 43 44 45 53 63 67 74 76
79      12/23/1991,02 03 05 07 08 12 15 18 19 23 34 35 39 41 46 47 51 54 72 74
80      12/22/1991,02 04 06 10 15 16 19 29 30 33 34 35 39 47 52 57 66 72 73 75
81      12/21/1991,08 12 14 19 24 26 27 30 31 37 44 46 49 59 60 62 63 66 75 80
82      12/20/1991,06 15 19 21 22 23 31 33 34 35 42 43 45 49 52 55 61 65 67 69
83      12/19/1991,11 16 18 29 30 34 36 38 39 45 59 62 67 68 70 71 72 75 76 79
84      12/18/1991,06 08 10 21 30 31 35 43 46 53 59 60 62 63 68 69 72 74 76 77
85      12/17/1991,02 06 07 09 16 19 21 25 31 35 39 49 52 64 67 70 72 75 79 80
86      12/16/1991,04 07 15 16 17 22 29 33 42 44 49 50 57 58 59 65 68 71 75 78
87      12/15/1991,18 19 20 25 37 38 40 42 43 45 47 48 50 56 58 68 70 75 77 79
88      12/14/1991,01 06 07 11 13 15 16 23 27 43 47 48 50 54 60 65 68 69 71 77
89      12/13/1991,03 06 08 14 16 20 21 24 26 30 43 56 58 67 71 73 76 77 78 80
90      12/12/1991,06 11 14 17 20 26 28 34 43 54 57 61 62 63 66 69 73 74 75 77
91      12/11/1991,04 09 21 22 25 26 37 38 43 44 46 47 48 55 70 71 76 77 79 80
92      12/10/1991,03 07 13 14 18 19 27 31 32 33 42 46 48 57 61 62 64 70 71 80
93      12/09/1991,08 11 15 18 21 24 26 27 33 34 38 40 42 58 64 67 69 70 78 80
94      12/08/1991,01 02 09 14 23 26 27 32 35 36 39 44 46 48 51 55 57 65 69 80
95      12/07/1991,04 11 13 16 18 19 30 34 42 43 50 52 55 59 65 66 71 74 75 76
96      12/06/1991,03 07 08 09 12 15 16 19 25 30 31 50 58 62 63 65 72 73 74 76
97      12/05/1991,10 13 15 16 18 19 23 35 38 41 45 47 51 59 61 64 65 74 78 80
98      12/04/1991,05 07 12 14 27 30 31 32 36 37 45 48 51 62 64 68 71 75 77 78
99      12/03/1991,03 05 06 12 13 15 18 26 38 43 45 46 59 61 65 67 72 74 76 78
00      12/02/1991,03 08 10 15 19 23 27 28 31 39 46 47 54 60 66 67 68 69 75 78
01      12/01/1991,08 15 22 25 27 28 32 38 40 41 42 50 56 57 60 64 65 71 73 78
02      11/30/1991,01 13 23 24 27 31 34 37 39 44 45 53 54 63 64 65 67 68 75 78
03      11/29/1991,07 10 12 15 19 20 22 23 26 29 41 44 45 50 55 61 66 68 72 80
04      11/28/1991,06 07 08 10 13 14 17 21 23 26 27 29 35 47 48 49 56 59 67 73
05      11/27/1991,07 16 20 23 30 35 36 38 39 43 48 56 58 59 60 63 76 77 78 80
06      11/26/1991,05 06 08 24 37 40 43 47 50 51 53 55 56 58 61 62 66 73 76 77
```

```
07      11/25/1991,05 13 15 19 20 27 29 33 34 38 39 50 52 53 60 61 65 69 77 79
08      11/24/1991,01 02 04 11 15 20 24 30 33 38 42 43 44 50 56 68 70 78 79 80
09      11/23/1991,07 10 14 18 20 27 28 29 30 31 35 41 46 50 54 56 60 65 66 78
10      11/22/1991,02 03 07 17 19 24 25 32 33 38 43 48 51 60 62 63 65 66 68 73
11      11/21/1991,15 20 23 27 28 30 31 41 44 47 49 53 54 58 63 67 68 69 77 78
12      11/20/1991,02 08 10 12 13 16 23 30 32 39 40 42 45 48 54 61 62 63 64 76
13      11/19/1991,01 02 09 10 12 14 15 24 26 30 34 38 44 45 58 60 63 64 66 80
14      11/18/1991,01 03 04 11 12 14 18 29 30 37 39 48 57 61 66 68 70 71 77 78
15      11/17/1991,01 06 12 13 22 25 26 27 38 44 47 49 50 53 56 57 60 63 65 80
16      11/16/1991,04 05 15 22 29 32 38 41 44 45 58 60 66 67 68 70 73 75 79 80
17      11/15/1991,01 07 09 18 25 26 28 44 46 49 50 52 55 58 62 63 67 69 71 72
18      11/14/1991,04 05 06 07 17 18 25 28 31 33 34 41 48 51 53 58 63 70 76 80
19      11/13/1991,03 04 05 06 10 12 15 24 28 33 36 40 41 47 48 62 65 69 74 76
20      11/12/1991,05 08 13 17 23 30 31 34 35 36 38 39 45 47 51 55 58 68 72 75
21      11/11/1991,03 10 18 25 31 33 35 36 37 43 45 46 47 53 55 66 69 71 73 79
22      11/10/1991,09 10 19 22 27 29 30 40 43 45 46 48 49 51 55 56 65 71 77 78
23      11/09/1991,14 15 20 21 22 27 29 35 37 41 43 48 50 53 55 57 67 68 70 71
24      11/08/1991,05 09 12 13 17 18 23 25 26 27 28 29 34 39 51 54 55 61 70 71
25      11/07/1991,05 07 12 14 18 25 27 29 38 40 44 46 47 49 54 61 67 71 75 78
26      11/06/1991,03 07 16 20 21 22 29 31 36 38 47 53 54 58 60 63 66 67 70 76
27      11/05/1991,01 04 08 17 18 20 36 46 49 51 52 55 58 59 62 67 68 70 71 79
28      11/04/1991,06 07 11 12 17 18 22 23 29 32 37 46 47 51 52 54 55 56 77 80
29      11/03/1991,01 04 06 07 10 13 17 21 23 31 42 44 47 53 55 60 61 67 77 79
30      11/02/1991,04 06 08 09 12 14 21 22 23 26 28 34 39 47 49 50 52 56 60 74
31      11/01/1991,05 07 12 15 18 22 25 34 44 45 52 53 54 60 61 62 68 71 73 78
32      10/31/1991,04 08 13 15 20 27 28 29 31 32 38 39 47 51 57 69 71 73 77 78
33      10/30/1991,04 08 09 11 16 17 23 32 33 38 41 43 54 60 61 62 72 73 74 79
34      10/29/1991,06 21 26 28 30 32 36 40 46 53 54 59 61 66 67 70 74 76 77 78
35      10/28/1991,05 08 10 13 16 17 20 24 26 34 38 45 46 58 59 61 62 70 79 80
36      10/27/1991,04 07 10 21 22 25 28 30 31 33 37 47 50 51 54 63 64 69 78 79
37      10/26/1991,02 04 05 09 10 11 13 25 36 39 44 53 57 65 68 69 70 74 76 77
38      10/25/1991,02 04 05 10 12 15 20 28 29 36 37 44 48 49 53 55 58 69 70 74
39      10/24/1991,01 02 08 12 13 15 21 25 32 39 43 45 47 48 59 67 70 72 74 78
40      10/23/1991,02 05 06 09 12 18 20 22 23 27 54 57 62 65 66 67 69 73 76 77
41      10/22/1991,01 04 07 16 17 18 19 35 38 40 46 49 50 58 60 62 64 73 77 78
42      10/21/1991,04 07 08 09 19 20 21 22 23 25 30 36 41 53 54 55 61 63 69 75
43      10/20/1991,06 10 12 16 25 32 35 36 38 42 48 51 53 57 61 63 68 69 72 74
44      10/19/1991,02 04 06 07 17 23 26 30 36 37 39 40 45 53 57 67 68 74 76 80
45      10/18/1991,04 09 10 16 19 20 27 28 32 37 48 49 52 58 59 67 69 70 71 78
46      10/17/1991,02 03 13 14 28 29 32 40 43 48 50 51 53 56 60 63 64 66 68 80
47      10/16/1991,13 18 21 27 29 31 35 36 37 41 44 48 53 54 58 62 67 69 73 80
48      10/15/1991,02 04 05 08 12 16 28 29 35 37 39 46 50 60 61 68 72 75 77 78
49      10/14/1991,13 15 17 23 27 35 36 42 45 46 47 52 54 55 57 61 71 72 75 79
50      10/13/1991,08 20 21 24 28 33 38 44 45 51 52 59 63 65 66 71 72 74 79 80
51      10/12/1991,13 16 24 25 28 31 32 40 46 51 54 66 68 69 70 72 74 76 78 80
52      10/11/1991,02 03 05 15 18 23 28 41 45 50 51 53 54 57 58 59 63 68 70 74
53      10/10/1991,01 04 05 14 23 28 40 41 43 44 49 50 52 56 57 58 59 63 74 80
54      10/09/1991,04 06 07 08 21 22 24 36 39 40 41 52 54 55 58 61 63 64 70 72
55      10/08/1991,02 04 05 08 10 18 20 22 32 37 40 45 46 47 52 59 61 62 70 79
56      10/07/1991,01 08 09 12 16 19 22 26 27 28 30 42 44 57 62 65 66 69 74 75
57      10/06/1991,01 04 05 16 18 19 20 22 23 26 32 41 44 53 54 57 58 63 65 74
58      10/05/1991,07 16 24 25 30 36 41 46 54 57 59 60 63 65 66 70 73 76 79 80
59      10/04/1991,01 06 11 14 15 18 21 23 27 31 33 37 46 58 62 65 67 73 75 77
60      10/03/1991,07 08 12 14 16 18 25 31 35 39 42 50 51 56 57 58 63 65 73 75
```

```
61      10/02/1991,01 11 17 23 25 26 28 32 36 40 43 48 50 54 57 62 71 78 79 80
62      10/01/1991,03 05 10 12 13 15 19 23 28 32 37 38 43 46 50 61 64 65 66 75
63      09/30/1991,06 07 09 18 20 21 28 30 32 35 43 44 45 47 53 61 65 66 76 80
64      09/29/1991,01 04 11 14 17 20 25 33 39 40 44 51 52 53 57 61 63 66 72 74
65      09/28/1991,02 09 19 24 25 31 34 36 39 44 49 51 53 57 59 60 66 68 74 77
66      09/27/1991,11 15 16 17 21 22 24 27 37 38 39 47 49 50 56 64 66 67 68 76
67      09/26/1991,02 05 06 18 20 21 22 23 27 30 33 48 51 53 56 60 61 63 73 75
68      09/25/1991,02 04 06 08 14 17 22 28 32 33 39 40 44 52 57 60 71 73 74 77
69      09/24/1991,02 08 10 11 18 19 23 26 33 37 39 49 58 60 63 64 67 68 73 75
70      09/23/1991,01 02 03 10 19 25 33 34 36 40 41 52 53 54 60 62 63 66 67 69
71      09/22/1991,03 06 11 14 15 26 27 30 31 34 42 43 44 56 58 73 74 75 76 80
72      09/21/1991,03 04 05 07 12 16 24 26 44 48 51 55 61 66 68 69 74 75 79 80
73      09/20/1991,04 05 06 18 22 23 24 25 26 31 34 37 45 49 51 61 62 69 76 80
74      09/19/1991,01 07 08 09 13 22 28 30 31 33 37 40 47 52 65 68 69 76 77 78
75      09/18/1991,02 04 10 11 17 20 26 29 30 33 43 45 49 51 53 59 63 69 74 75
76      09/17/1991,01 02 04 05 09 10 11 15 26 29 36 37 41 46 49 59 64 76 77 79
77      09/16/1991,11 15 17 24 29 32 34 36 41 42 45 46 47 51 54 63 66 72 73 80
78      09/15/1991,01 03 08 09 12 13 19 21 26 30 37 39 48 50 52 56 58 63 65 74
79      09/14/1991,12 13 17 19 28 29 33 37 38 39 43 56 57 58 62 64 65 68 75 77
80      09/13/1991,06 08 16 29 31 33 34 40 41 43 52 55 58 59 64 69 73 75 77 78
81      09/12/1991,13 19 24 30 31 38 44 46 47 49 53 54 60 61 65 67 73 77 78 79
82      09/11/1991,04 07 13 18 22 23 26 29 34 35 41 50 52 53 62 65 69 76 79 80
83      09/10/1991,01 03 04 06 10 21 22 27 30 32 35 42 44 54 55 57 58 64 67 73
84      09/09/1991,03 04 06 19 22 23 25 28 29 30 34 39 43 44 61 64 67 72 77 78
85      09/08/1991,06 07 10 13 19 21 26 27 30 43 45 46 52 57 59 61 66 67 77 79
86      09/07/1991,07 11 13 15 28 30 35 36 50 51 53 54 56 58 62 63 66 74 77 78
87      09/06/1991,05 14 18 22 25 26 30 33 35 39 44 45 47 63 69 71 74 76 78 79
88      09/05/1991,03 11 12 13 20 30 33 36 39 40 43 44 49 51 60 66 67 76 77 78
89      09/04/1991,03 09 11 13 18 19 20 26 30 42 44 45 58 59 62 63 65 66 76 79
90      09/03/1991,04 06 09 10 11 18 25 26 28 32 35 41 46 48 53 61 68 71 72 79
91      09/02/1991,04 05 12 16 17 21 24 28 29 39 45 46 48 51 55 57 62 66 68 71
92      09/01/1991,01 15 16 17 18 20 21 24 25 34 40 43 51 52 54 59 62 66 74 80
93      08/31/1991,03 07 12 16 20 22 28 29 31 40 42 47 50 54 59 62 64 67 72 80
94      08/30/1991,02 03 09 14 19 22 28 29 30 33 37 44 49 51 52 64 66 70 79 80
95      08/29/1991,03 04 11 20 24 27 29 33 38 39 40 46 49 50 55 62 64 65 73 78
96      08/28/1991,01 07 13 14 23 28 32 34 37 38 41 43 46 57 58 65 70 71 73 74
97      08/27/1991,03 10 14 20 26 28 33 36 38 42 43 45 46 53 55 57 59 71 72 80
98      08/26/1991,02 09 10 11 16 17 18 23 24 26 29 31 33 35 36 39 48 51 54 79
99      08/25/1991,04 05 08 10 14 15 16 21 24 25 26 32 41 56 57 58 61 63 66 72
00      08/24/1991,01 05 09 20 23 36 41 45 50 51 54 61 62 71 72 73 74 75 77 79
01      08/23/1991,03 05 06 09 13 21 23 25 31 32 34 35 41 42 51 52 59 63 65 80
02      08/22/1991,03 09 11 16 18 31 35 37 38 40 42 44 47 49 57 61 64 70 74 80
03      08/21/1991,05 06 08 12 15 26 31 38 40 44 47 49 56 62 63 64 69 74 76 79
04      08/20/1991,04 10 11 14 18 26 31 32 41 42 45 46 47 59 60 61 67 77 79 80
05      08/19/1991,01 02 16 18 20 24 27 32 33 41 43 45 46 48 54 62 63 71 75 80
06      08/18/1991,11 14 15 17 25 29 32 34 38 39 43 45 53 54 56 57 61 63 67 69
07      08/17/1991,01 02 07 12 14 17 20 21 22 28 43 44 47 52 56 58 67 69 71 72
08      08/16/1991,04 05 06 10 12 13 16 30 37 38 41 43 49 54 60 70 72 74 75 79
09      08/15/1991,03 12 13 18 23 30 33 37 41 44 47 51 54 55 58 61 63 64 69 70
10      08/14/1991,02 03 07 15 17 18 20 23 27 28 33 37 38 39 43 49 54 74 77 79
11      08/13/1991,04 05 11 15 16 18 20 23 27 30 34 39 45 50 56 62 65 68 71 76
12      08/12/1991,05 06 12 15 18 21 22 23 29 30 32 33 38 48 49 52 54 57 66 72
13      08/11/1991,02 03 05 13 14 17 24 28 29 37 40 45 49 56 68 70 71 76 78 80
14      08/10/1991,02 07 08 11 12 15 27 35 36 37 41 43 50 54 56 57 62 64 69 70
```

```
15      08/09/1991,11 12 25 27 29 32 41 46 47 48 58 63 64 65 67 71 73 75 76 78
16      08/08/1991,04 06 10 11 16 19 21 23 32 46 51 52 53 60 65 68 70 71 73 79
17      08/07/1991,02 04 07 08 10 15 18 22 24 27 30 37 38 44 61 63 65 66 68 78
18      08/06/1991,04 06 16 17 18 21 32 33 35 41 43 49 55 56 64 65 73 74 78 80
19      08/05/1991,02 11 12 20 22 26 29 39 43 45 47 51 57 66 67 68 71 76 78 79
20      08/04/1991,02 03 11 16 18 20 24 31 32 39 43 45 49 52 59 61 71 77 79 80
21      08/03/1991,07 09 10 12 17 18 24 29 34 35 40 42 49 57 63 64 67 70 78 80
22      08/02/1991,02 03 06 07 12 13 27 35 43 47 49 53 54 57 58 65 68 69 77 78
23      08/01/1991,08 09 10 14 16 25 28 31 32 33 35 40 43 44 48 60 71 75 78 79
24      07/31/1991,06 09 11 23 27 28 29 31 33 38 39 42 45 51 52 56 61 65 68 72
25      07/30/1991,01 04 08 19 22 28 31 34 37 38 45 46 48 50 51 57 58 65 75 80
26      07/29/1991,04 05 19 23 34 36 42 45 47 49 53 54 59 60 61 65 69 76 77 78
27      07/28/1991,01 05 06 14 18 26 29 33 37 41 43 45 47 49 53 56 61 68 77 78
28      07/27/1991,06 14 15 17 20 25 26 29 32 36 37 41 42 58 61 65 66 72 75 80
29      07/26/1991,01 02 11 19 20 22 23 28 30 33 34 36 39 40 42 43 45 47 49 74
30      07/25/1991,02 10 13 15 21 32 34 37 38 45 47 52 53 54 61 66 68 73 74 76
31      07/24/1991,02 07 14 18 20 23 27 31 35 41 44 45 47 51 56 59 61 64 71 76
32      07/23/1991,10 14 16 17 29 31 36 38 39 42 44 47 49 50 58 62 66 76 77 80
33      07/22/1991,01 09 21 22 25 26 27 28 31 33 40 49 55 62 63 65 68 75 76 77
34      07/21/1991,02 08 20 23 24 25 29 32 35 40 45 51 52 55 58 73 74 76 78 80
35      07/20/1991,03 15 16 17 19 20 26 30 36 39 43 46 47 52 54 60 62 64 65 68
36      07/19/1991,01 06 17 18 19 20 28 34 37 38 40 42 44 48 50 54 56 66 77 79
37      07/18/1991,08 09 10 11 12 14 16 17 19 20 26 40 43 50 54 55 57 60 76 77
38      07/17/1991,01 02 04 05 06 10 19 26 33 34 45 47 48 51 62 64 67 68 73 76
39      07/16/1991,03 11 19 26 29 34 37 41 44 52 54 56 61 62 63 65 69 72 73 79
40      07/15/1991,01 02 04 06 07 11 17 18 35 36 37 41 46 51 53 55 64 65 67 69
41      07/14/1991,04 14 18 21 22 27 29 33 34 36 50 56 57 61 64 68 69 71 72 79
42      07/13/1991,02 05 10 13 18 20 23 26 33 39 46 47 51 56 60 65 67 69 76 78
43      07/12/1991,07 15 19 24 25 27 32 38 39 41 43 47 60 63 66 67 71 75 77 78
44      07/11/1991,03 05 24 26 27 30 33 37 38 39 41 44 47 49 57 66 68 69 75 77
45      07/10/1991,01 07 12 19 23 26 29 33 34 42 46 48 50 53 62 65 70 73 77 80
46      07/09/1991,08 10 15 17 18 26 27 35 38 47 49 55 57 58 60 62 64 65 73 78
47      07/08/1991,14 22 29 30 32 33 35 37 43 53 61 64 68 70 71 74 76 78 79 80
48      07/07/1991,01 03 06 10 11 12 16 17 21 25 30 34 35 37 39 41 43 64 65 76
49      07/06/1991,03 05 06 09 11 15 18 23 25 26 28 39 40 46 49 50 55 60 64 74
50      07/05/1991,03 04 11 14 17 18 20 21 22 39 55 57 58 59 66 68 69 76 77 80
51      07/04/1991,02 03 04 07 09 14 20 22 30 34 35 38 40 55 63 64 72 73 75 78
52      07/03/1991,02 07 09 16 24 26 31 32 37 43 44 45 59 63 65 68 73 76 77 78
53      07/02/1991,01 02 06 07 09 11 12 14 15 20 22 36 37 38 41 54 66 72 74 75
54      07/01/1991,03 05 11 15 16 20 22 25 29 42 44 45 54 56 59 62 63 70 77 80
55      06/30/1991,04 05 12 23 24 26 27 41 42 43 44 49 60 63 66 69 71 76 77 79
56      06/29/1991,15 16 21 23 24 27 29 31 32 40 45 48 49 52 53 55 67 71 72 75
57      06/28/1991,03 07 08 16 18 23 27 29 32 34 37 43 44 45 48 52 55 59 64 76
58      06/27/1991,01 04 05 06 16 21 26 36 42 46 49 53 54 55 60 62 63 64 72 74
59      06/26/1991,05 07 10 11 14 26 31 35 37 42 49 51 55 56 59 63 66 67 68 73
60      06/25/1991,05 12 13 14 18 20 23 27 34 35 47 52 56 58 61 62 65 70 74 80
61      06/24/1991,04 06 10 11 16 24 26 29 30 38 44 45 46 48 54 58 60 62 69 71
62      06/23/1991,01 02 05 11 20 21 34 36 41 46 47 53 55 56 61 70 72 76 79 80
63      06/22/1991,09 15 25 29 32 33 35 41 42 43 52 53 54 57 61 62 64 69 70 76
64      06/21/1991,10 11 12 13 15 17 20 22 24 31 35 41 56 60 65 69 73 74 78 79
65      06/20/1991,08 11 14 24 27 28 29 30 33 42 55 58 62 65 66 67 69 72 79 80
66      06/19/1991,01 06 11 14 16 25 26 28 32 40 41 46 51 54 62 63 66 70 73 80
67      06/18/1991,10 14 15 23 30 31 32 33 35 39 41 42 43 45 48 51 54 64 65 78
68      06/17/1991,02 03 06 07 08 15 18 34 37 38 40 41 45 49 57 62 67 74 76 80
```

```
69      06/16/1991,04 07 10 12 23 25 30 33 37 46 47 54 55 56 60 62 64 73 75 76
70      06/15/1991,01 04 05 06 13 17 20 23 26 28 40 41 44 47 51 52 59 76 78 80
71      06/14/1991,03 09 13 19 22 23 24 26 27 32 37 43 48 54 55 58 62 69 71 73
72      06/13/1991,01 16 18 23 25 28 30 32 33 35 36 39 42 50 51 55 63 66 68 80
73      06/12/1991,04 08 09 10 11 12 14 19 20 29 38 40 46 53 58 59 71 73 78 80
74      06/11/1991,05 09 10 13 21 26 27 29 34 37 41 50 52 56 58 60 62 69 77 80
75      06/10/1991,03 05 10 16 19 20 21 30 35 39 40 43 45 46 49 70 72 76 77 79
76      06/09/1991,03 12 16 17 20 32 34 36 41 44 46 47 49 56 61 64 69 70 72 79
77      06/08/1991,02 04 10 15 17 20 27 28 34 39 45 46 49 50 55 60 61 66 76 78
78      06/07/1991,03 06 08 14 19 34 35 39 40 42 45 47 50 51 57 59 62 65 72 75
79      06/06/1991,05 08 09 10 21 22 25 26 29 38 50 61 66 67 68 72 75 76 78 80
80      06/05/1991,01 04 05 06 13 16 19 25 36 47 52 54 55 61 62 63 66 69 73 79
81      06/04/1991,02 03 14 19 27 30 35 36 40 44 45 47 57 59 62 64 66 68 71 74
82      06/03/1991,05 15 16 19 20 21 23 26 27 28 32 33 36 54 58 60 62 73 76 79
83      06/02/1991,02 17 18 22 26 36 38 45 49 51 52 54 60 61 64 69 71 77 78 79
84      06/01/1991,01 06 07 17 19 20 27 29 34 36 37 47 49 50 60 64 65 67 71 73
85      05/31/1991,01 02 07 08 09 11 18 20 22 23 36 51 52 53 56 63 66 68 72 74
86      05/30/1991,01 18 21 32 33 34 41 47 49 50 51 58 59 63 65 70 74 77 78 79
87      05/29/1991,04 10 15 18 19 26 28 29 40 42 45 49 57 58 60 61 69 70 73 80
88      05/28/1991,05 08 10 12 13 14 15 21 26 30 37 40 43 46 58 62 63 69 71 72
89      05/27/1991,01 03 04 09 18 22 28 30 44 45 50 52 57 58 61 62 63 65 69 71
90      05/26/1991,02 04 05 10 15 19 21 27 30 35 38 39 40 41 46 47 50 62 66 69
91      05/25/1991,02 06 08 09 11 17 20 23 29 30 33 34 35 37 43 47 50 53 62 65
92      05/24/1991,01 07 08 17 18 19 25 26 32 35 43 46 48 49 50 56 60 65 76 79
93      05/23/1991,01 04 05 20 28 30 31 37 39 40 43 44 47 51 58 61 67 73 75 79
94      05/22/1991,06 07 13 14 19 22 30 31 34 41 42 51 57 58 62 65 71 74 77 78
95      05/21/1991,03 08 10 13 25 37 38 40 41 46 52 57 58 59 67 68 70 77 79 80
96      05/20/1991,02 03 04 13 15 16 20 24 27 30 40 41 44 48 52 57 65 69 77 80
97      05/19/1991,02 09 10 15 16 21 24 32 34 38 43 44 47 51 52 54 62 71 75 76
98      05/18/1991,11 17 18 19 21 28 30 34 37 48 49 50 60 62 66 70 73 76 78 79
99      05/17/1991,18 22 26 28 30 34 37 38 40 42 45 46 49 50 52 53 57 59 61 73
00      05/16/1991,02 10 12 15 18 27 29 31 44 46 47 50 52 56 65 67 70 72 75 80
01      05/15/1991,03 04 06 08 14 15 16 17 24 35 40 49 53 55 61 65 72 73 75 76
02      05/14/1991,05 06 08 12 16 18 19 20 26 27 31 34 44 50 60 65 69 70 71 79
03      05/13/1991,02 03 11 15 19 23 24 26 31 34 38 44 49 50 53 55 63 65 67 77
04      05/12/1991,01 03 09 15 20 21 24 37 39 41 43 47 48 49 51 54 60 66 74 76
05      05/11/1991,08 14 22 25 32 33 37 42 43 45 54 55 57 58 61 64 71 72 74 76
06      05/10/1991,01 03 06 09 12 17 21 26 33 39 40 50 55 64 65 69 70 72 74 80
07      05/09/1991,01 04 06 10 15 30 33 43 44 45 50 54 56 57 58 66 67 75 78 79
08      05/08/1991,02 07 15 18 19 20 22 27 29 31 33 36 41 49 51 52 57 62 69 80
09      05/07/1991,12 13 14 20 27 31 36 37 38 39 40 42 45 50 55 58 59 60 61 66
10      05/06/1991,02 11 12 13 16 29 32 35 38 46 49 51 52 54 57 64 73 76 77 78
11      05/05/1991,01 03 08 10 21 27 28 30 36 39 41 43 47 50 51 68 71 72 74 80
12      05/04/1991,21 22 25 29 32 34 37 40 44 45 51 57 59 64 65 69 71 73 76 79
13      05/03/1991,01 07 08 12 15 20 22 24 25 29 35 38 44 46 53 60 63 68 73 76
14      05/02/1991,09 10 13 16 18 20 21 27 28 30 33 34 35 39 60 63 65 68 69 76
15      05/01/1991,04 06 09 10 17 23 26 34 35 37 38 40 49 67 69 71 72 73 74 75
16      04/30/1991,01 03 07 18 21 22 32 34 40 41 45 52 53 54 57 61 62 65 66 74
17      04/29/1991,04 08 09 11 13 20 21 30 31 38 44 45 55 61 62 68 72 77 79 80
18      04/28/1991,01 08 21 23 24 25 26 28 34 37 41 42 43 44 48 50 52 53 60 79
19      04/27/1991,06 08 11 14 17 18 32 33 41 43 50 53 54 60 64 70 72 73 75 79
20      04/26/1991,06 12 15 29 32 34 36 38 39 46 48 49 50 52 53 68 74 77 78 79
21      04/25/1991,06 08 15 21 28 32 33 36 40 45 46 47 48 50 54 58 60 64 68 73
22      04/24/1991,06 09 10 13 16 19 26 31 37 41 42 44 45 52 55 60 67 68 72 78
```

```
04/23/1991,01 03 04 09 21 25 28 29 36 48 54 58 59 61 62 65 68 69 70 72
04/22/1991,01 07 08 15 16 21 23 26 32 40 43 61 63 64 65 68 73 76 78 80
04/21/1991,02 10 13 15 17 18 27 29 34 36 37 38 45 47 48 63 67 71 75 77
04/20/1991,02 10 15 17 25 26 29 30 40 46 47 50 53 55 59 68 72 75 78 79
04/19/1991,02 03 05 07 14 16 18 20 27 32 36 38 41 51 54 55 69 77 78 79
04/18/1991,03 06 14 18 19 21 23 36 42 43 46 51 58 59 63 66 68 72 79 80
04/17/1991,03 08 10 13 14 19 21 23 28 29 30 42 48 49 51 52 61 66 77 80
04/16/1991,01 03 10 14 15 16 20 24 26 32 39 45 54 59 62 63 66 69 71 77
04/15/1991,02 11 14 20 22 23 24 25 28 30 31 36 39 48 50 67 70 75 78 79
04/14/1991,03 04 07 11 21 27 31 36 38 40 47 50 60 63 64 65 66 70 72 77
04/13/1991,01 04 05 17 18 24 27 34 41 43 45 52 53 60 62 63 72 75 77 79
04/12/1991,02 04 05 23 28 29 30 35 37 39 41 43 47 48 50 62 66 68 69 74
04/11/1991,04 08 15 18 20 25 32 34 39 43 51 57 60 70 71 72 74 76 77 78
04/10/1991,02 04 08 10 14 18 23 24 30 33 34 40 42 45 46 60 63 65 75 78
04/09/1991,01 09 12 15 16 17 20 23 30 31 38 39 44 45 50 57 66 67 68 69
04/08/1991,06 15 17 18 21 24 30 33 35 36 37 49 50 52 57 59 60 62 69 76
04/07/1991,03 13 14 17 22 24 31 38 48 49 50 55 60 66 67 68 73 74 78 80
04/06/1991,10 11 12 13 14 16 19 20 23 26 28 30 36 65 71 73 74 75 76 79
04/05/1991,05 06 07 12 19 29 39 40 41 49 51 61 63 64 66 67 74 76 77 80
04/04/1991,11 14 21 25 35 37 39 42 48 49 51 52 53 56 57 62 70 72 76 78
04/03/1991,01 09 11 16 22 24 27 33 41 51 57 59 60 63 66 67 70 72 74 76
04/02/1991,01 03 05 14 18 19 22 25 29 30 35 38 47 50 55 60 71 73 78 80
04/01/1991,04 09 11 13 14 16 19 20 21 31 37 41 45 47 53 60 61 72 73 77
03/31/1991,02 13 14 20 23 26 30 32 34 38 41 45 47 48 53 55 62 68 72 75
03/30/1991,01 06 09 10 17 21 26 27 28 33 34 37 53 54 57 62 65 69 75 77
03/29/1991,02 04 05 12 15 16 17 20 31 33 44 51 56 59 61 63 64 65 70 75
03/28/1991,04 11 16 18 21 23 24 31 39 44 45 50 53 58 67 68 73 75 76 79
03/27/1991,03 12 19 27 28 29 33 34 38 40 46 47 53 57 59 65 71 76 78 79
03/26/1991,03 06 08 20 25 32 36 39 43 44 45 48 56 57 58 60 62 65 70 77
03/25/1991,08 10 14 16 17 18 19 21 28 36 43 48 49 52 54 58 60 68 76 77
03/24/1991,07 09 16 18 22 24 27 37 38 39 41 45 52 54 55 56 59 60 71 75
03/23/1991,04 06 08 14 18 23 24 25 27 33 34 38 43 44 60 66 72 73 77 78
03/22/1991,04 06 13 16 20 28 31 41 44 45 54 55 56 57 64 65 67 73 77 80
03/21/1991,01 08 09 13 20 21 25 29 31 34 38 42 56 61 62 64 72 73 74 79
03/20/1991,02 03 04 18 21 29 30 32 34 41 42 56 59 60 61 62 65 72 77 80
03/19/1991,06 07 08 12 17 18 19 25 28 36 39 45 46 47 53 55 57 72 74 76
03/18/1991,01 02 03 04 06 13 15 17 21 23 34 35 38 45 47 50 53 57 63 79
03/17/1991,05 13 15 20 26 28 31 34 37 44 46 51 52 53 57 64 70 71 72 76
03/16/1991,02 04 08 11 20 28 29 33 34 41 47 49 50 51 56 57 64 72 74 78
03/15/1991,04 06 08 12 15 17 31 34 36 38 40 45 48 50 57 66 67 68 78 80
03/14/1991,05 24 25 28 32 37 42 43 44 47 48 52 58 60 62 63 67 69 74 79
03/13/1991,01 03 20 21 27 29 33 36 47 51 60 62 64 67 70 71 74 75 77 79
03/12/1991,02 05 07 09 12 16 19 23 24 32 34 36 44 48 50 54 67 74 77 78
03/11/1991,01 12 18 22 23 31 32 33 35 36 37 40 44 47 59 62 69 71 78 79
03/10/1991,01 04 05 18 22 24 28 31 41 43 45 59 61 63 64 68 71 72 77 79
03/09/1991,01 06 12 15 17 24 28 30 32 37 41 42 45 47 56 60 66 70 74 76
03/08/1991,06 07 08 11 17 23 24 27 31 35 36 37 43 45 51 56 57 72 74 75
03/07/1991,01 04 07 08 09 11 14 15 18 27 29 41 47 49 51 66 68 71 72 75
03/06/1991,02 03 07 13 20 21 22 25 27 39 46 48 52 55 57 66 69 72 76 80
03/05/1991,01 03 11 22 23 24 26 31 33 36 38 41 43 46 47 58 59 69 70 74
03/04/1991,19 24 26 27 30 35 36 37 40 43 45 48 51 52 53 60 62 66 69 74
03/03/1991,01 04 12 17 22 23 26 30 34 59 66 70 71 72 73 74 75 77 78 80
03/02/1991,02 07 09 12 15 21 25 30 31 34 36 40 41 47 49 52 62 65 66 67
03/01/1991,06 07 08 13 14 16 20 24 25 27 32 33 37 38 53 58 63 65 68 70
```

```
77      02/28/1991,08 10 16 24 27 33 34 36 38 54 56 58 60 62 64 65 67 70 72 77
78      02/27/1991,06 11 19 20 31 33 36 41 45 48 52 58 61 62 71 73 74 77 79 80
79      02/26/1991,06 14 17 18 21 23 31 32 36 39 44 46 49 54 55 56 59 66 73 77
80      02/25/1991,26 28 29 33 34 36 38 42 44 50 56 59 61 64 69 70 72 73 76 77
81      02/24/1991,02 03 10 11 12 19 25 27 41 43 45 48 49 56 59 69 70 74 78 80
82      02/23/1991,06 09 11 22 24 30 32 34 37 49 51 52 54 66 70 72 74 77 79 80
83      02/22/1991,02 05 06 14 18 24 25 32 35 37 39 47 50 52 56 57 72 73 77 80
84      02/21/1991,05 06 08 11 21 23 26 32 41 44 48 55 57 58 62 63 70 72 74 79
85      02/20/1991,07 12 16 17 27 30 31 39 45 48 52 53 56 58 63 69 70 73 77 78
86      02/19/1991,04 05 06 08 09 11 13 14 15 20 26 34 35 49 50 56 58 64 68 74
87      02/18/1991,01 02 03 08 11 14 15 19 23 24 25 32 44 49 51 60 63 65 68 80
88      02/17/1991,05 07 08 11 23 24 26 31 33 39 43 47 54 55 59 65 70 72 75 78
89      02/16/1991,04 06 08 12 14 20 23 26 30 34 37 41 43 55 58 64 65 66 72 74
90      02/15/1991,01 03 05 15 18 20 21 34 37 40 43 44 45 53 57 60 61 65 70 75
91      02/14/1991,03 04 05 07 08 09 13 30 31 33 37 38 50 55 58 60 61 71 72 77
92      02/13/1991,01 09 10 11 17 24 29 30 38 40 43 45 58 63 65 71 74 76 77 80
93      02/12/1991,13 15 18 23 29 30 34 39 41 45 49 50 55 60 61 64 71 73 74 79
94      02/11/1991,03 13 14 15 18 19 29 35 38 45 47 57 61 62 68 69 73 75 76 80
95      02/10/1991,04 07 14 15 16 18 22 24 27 29 33 36 45 50 62 65 67 69 70 79
96      02/09/1991,02 04 06 07 11 14 15 19 20 26 30 42 55 56 57 59 60 66 72 77
97      02/08/1991,02 04 09 13 16 18 31 33 35 40 41 46 47 52 55 62 69 70 74 76
98      02/07/1991,02 04 13 17 18 23 24 28 42 43 48 56 57 60 68 70 71 73 75 79
99      02/06/1991,07 12 13 18 19 26 38 39 40 41 43 48 49 62 64 74 75 76 77 79
00      02/05/1991,03 04 06 07 08 09 13 17 25 32 37 46 48 52 56 62 64 70 71 74
01      02/04/1991,04 07 10 14 16 17 19 26 27 38 39 47 51 52 66 67 68 77 79 80
02      02/03/1991,11 13 16 17 19 22 27 30 33 47 50 52 54 57 60 65 66 69 72 73
03      02/02/1991,05 11 12 15 18 19 23 27 38 46 48 52 53 58 61 62 66 73 74 78
04      02/01/1991,05 12 15 21 22 23 32 36 37 41 44 59 60 61 62 63 65 67 68 75
05      01/31/1991,02 06 10 16 26 33 38 40 45 47 49 52 56 57 68 71 73 76 77 78
06      01/30/1991,01 05 22 34 38 42 44 50 54 58 59 60 61 64 65 69 73 74 78 79
07      01/29/1991,08 11 13 14 16 23 28 31 42 44 48 51 52 55 56 68 72 74 75 79
08      01/28/1991,11 16 20 21 30 31 39 40 43 48 49 51 54 55 70 74 77 78 79 80
09      01/27/1991,09 12 16 22 24 28 29 32 37 39 50 53 57 61 63 69 70 75 76 78
10      01/26/1991,05 07 09 11 16 25 29 32 38 41 45 46 48 56 59 66 73 74 78 80
11      01/25/1991,04 05 11 12 17 22 23 24 26 30 31 32 37 38 50 64 69 73 74 76
12      01/24/1991,01 13 15 18 19 22 24 31 34 38 39 40 51 56 57 61 64 68 71 73
13      01/23/1991,07 09 18 19 24 25 28 29 34 37 38 40 41 42 43 51 52 59 71 78
14      01/22/1991,01 04 06 09 21 23 32 35 38 52 53 56 57 61 68 70 72 74 75 76
15      01/21/1991,04 08 11 16 17 22 28 30 31 35 39 40 43 46 47 56 61 66 69 77
16      01/20/1991,02 09 15 18 20 21 22 24 26 28 42 51 59 61 62 65 69 70 78 80
17      01/19/1991,01 06 12 15 16 29 31 32 35 39 43 46 48 52 54 55 63 66 68 71
18      01/18/1991,07 09 25 28 35 36 39 41 44 46 54 57 60 62 64 66 71 76 77 78
19      01/17/1991,02 03 17 20 29 33 34 39 44 45 52 54 55 56 58 62 66 67 69 79
20      01/16/1991,03 04 05 08 09 10 16 17 21 26 28 29 30 45 48 56 57 70 72 73
21      01/15/1991,04 13 17 22 26 27 34 41 43 45 53 56 63 64 65 73 76 78 79 80
22      01/14/1991,01 07 12 14 18 21 24 26 29 33 37 39 40 41 44 45 56 61 65 70
23      01/13/1991,04 07 08 14 15 21 23 27 28 31 37 39 49 53 54 55 59 68 69 77
24      01/12/1991,02 04 06 07 09 24 29 31 39 42 44 47 50 54 56 58 60 65 78 80
25      01/11/1991,03 11 14 18 20 21 22 24 30 31 32 42 49 56 61 62 71 73 75 80
26      01/10/1991,02 03 09 21 22 24 26 27 28 29 34 35 41 42 51 54 65 66 75 80
27      01/09/1991,03 07 10 12 13 15 16 32 36 42 44 45 47 55 59 60 62 65 73 80
28      01/08/1991,02 11 14 19 20 24 26 30 32 34 36 37 39 46 48 51 55 58 65 79
29      01/07/1991,02 04 07 09 12 14 16 26 27 30 32 37 47 48 49 53 56 61 63 77
30      01/06/1991,03 05 10 19 32 34 35 38 48 53 57 59 60 66 67 70 71 72 76 78
```

```
01/05/1991,06 10 17 21 23 28 31 34 36 38 39 47 51 52 54 60 62 63 65 70
01/04/1991,01 02 03 04 08 16 23 24 29 30 31 33 42 43 44 52 57 62 67 72
01/03/1991,02 08 15 18 26 28 29 33 35 41 49 51 60 63 65 73 76 78 79 80
01/02/1991,07 08 11 12 13 15 25 28 36 37 38 44 47 49 55 57 62 64 65 80
01/01/1991,04 05 14 15 17 24 28 31 32 43 44 49 54 56 57 60 61 69 71 78
12/31/1990,01 06 07 13 15 16 22 23 30 33 35 43 49 52 57 63 64 69 78 80
12/30/1990,04 06 09 10 12 13 14 17 21 24 27 28 36 39 40 41 56 62 63 75
12/29/1990,03 08 12 16 23 42 43 44 51 54 56 58 62 64 66 67 68 70 76 80
12/28/1990,01 03 06 11 14 18 21 28 31 38 41 45 54 55 58 59 65 67 70 80
12/27/1990,03 14 15 16 18 24 26 35 38 40 42 48 51 59 62 66 67 73 74 76
12/26/1990,02 12 14 16 17 18 20 21 33 34 46 58 59 61 67 68 69 74 76 80
12/24/1990,01 05 09 15 17 19 21 24 27 29 33 34 38 41 42 46 49 57 62 78
12/23/1990,02 04 06 17 18 21 23 39 43 45 47 48 53 61 63 66 68 72 74 78
12/22/1990,01 06 08 10 19 20 29 30 36 44 47 51 56 57 58 62 72 74 75 79
12/21/1990,02 05 09 12 26 31 32 33 34 51 52 56 59 63 66 67 68 69 73 74
12/20/1990,05 07 08 09 10 13 20 22 24 25 31 40 45 51 53 57 65 68 70 76
12/19/1990,02 21 22 23 31 33 36 42 43 44 48 51 52 58 60 63 73 76 77 79
12/18/1990,01 06 07 10 16 18 24 39 41 47 51 55 56 57 59 68 70 73 74 80
12/17/1990,06 07 11 13 17 19 22 35 40 41 42 43 45 60 62 64 68 73 75 78
12/16/1990,01 04 10 19 22 23 29 30 36 41 43 49 50 56 60 64 68 78 79 80
12/15/1990,07 08 09 10 15 21 26 35 45 47 48 50 55 62 64 65 68 69 73 78
12/14/1990,04 10 12 22 23 27 31 32 37 38 39 43 54 56 60 66 67 73 79 80
12/13/1990,02 03 06 08 14 23 33 40 41 46 47 50 53 54 58 59 67 72 75 76
12/12/1990,01 09 16 19 21 27 36 44 49 50 54 59 63 64 65 66 68 69 70 71
12/11/1990,02 10 12 13 21 26 27 28 46 48 50 57 60 62 63 65 68 74 75 79
12/10/1990,01 08 10 11 15 23 27 32 33 35 42 43 44 46 55 58 63 68 69 75
12/09/1990,09 11 12 14 18 22 30 31 32 37 38 49 52 54 57 58 67 72 77 79
12/08/1990,01 02 04 06 10 11 12 18 19 34 42 44 50 53 59 65 66 75 77 80
12/07/1990,03 06 08 10 15 16 19 22 23 25 33 36 38 48 54 60 68 73 75 77
12/06/1990,06 19 20 26 27 28 31 40 46 48 50 52 57 59 65 71 72 74 78 80
12/05/1990,02 10 13 15 17 22 25 30 32 37 39 41 51 56 58 65 66 68 70 78
12/04/1990,09 12 22 23 26 28 31 32 37 44 45 49 51 56 67 68 69 70 72 80
12/03/1990,12 17 19 21 22 28 30 34 35 37 57 58 65 68 70 72 73 74 78 79
12/02/1990,01 19 22 23 24 28 31 36 39 40 48 49 56 57 63 68 73 75 78 80
12/01/1990,03 06 08 14 19 22 26 30 33 37 39 41 46 48 55 61 63 78 79 80
11/30/1990,03 10 13 16 17 18 21 24 27 28 41 47 58 59 61 69 71 72 73 78
11/29/1990,01 08 13 16 17 24 25 28 34 39 44 46 51 55 60 74 75 77 78 80
11/28/1990,03 06 09 11 15 17 19 27 30 32 39 41 42 45 48 49 61 62 70 72
11/27/1990,02 04 06 15 21 23 35 40 41 46 48 49 51 55 57 58 61 64 71 77
11/26/1990,07 09 10 15 19 30 32 34 42 44 48 49 51 53 54 58 63 68 73 77
11/25/1990,05 12 13 19 23 26 30 31 33 36 37 41 43 54 57 59 66 70 71 75
11/24/1990,02 04 06 08 09 16 19 20 23 25 27 33 44 46 49 52 57 62 64 69
11/23/1990,01 06 07 10 12 14 15 16 21 22 44 50 51 53 61 63 73 75 76 79
11/22/1990,06 07 09 10 12 15 18 31 33 51 59 61 62 67 68 69 71 75 77 79
11/21/1990,01 03 07 11 13 16 19 21 24 34 43 53 55 63 65 72 73 75 77 78
11/20/1990,07 10 12 20 23 27 28 29 31 47 50 51 55 57 62 63 69 72 74 77
11/19/1990,03 11 12 18 20 27 31 34 36 47 48 49 53 59 64 67 68 69 77 79
11/18/1990,06 07 12 13 15 16 21 23 28 35 38 41 42 56 57 60 67 73 76 79
11/17/1990,04 14 17 21 25 27 34 36 41 46 51 54 57 58 60 62 65 67 71 77
11/16/1990,03 04 06 10 16 17 25 30 33 47 48 49 57 58 59 61 67 69 71 74
11/15/1990,02 03 04 05 17 18 25 27 31 33 40 47 48 51 54 55 63 66 76 80
11/14/1990,05 08 17 20 23 37 40 41 43 46 50 52 53 55 56 67 68 73 77 78
11/13/1990,04 06 10 15 18 23 25 31 44 45 47 53 63 65 66 67 68 73 78 79
11/12/1990,06 07 10 13 16 23 24 25 29 32 38 40 50 54 55 61 70 72 75 77
```

```
11/11/1990,01 13 18 19 28 30 31 33 36 38 40 44 46 51 52 58 62 65 71 73
11/10/1990,01 11 16 27 32 37 50 54 56 57 59 61 63 64 65 70 74 76 77 78
11/09/1990,04 08 10 11 16 20 22 25 29 36 38 40 41 50 55 56 58 71 73 79
11/08/1990,03 04 05 09 10 11 14 15 32 42 43 45 46 54 55 56 57 69 70 71
11/07/1990,04 12 13 14 18 25 26 30 31 35 41 42 44 47 49 59 61 67 72 79
11/06/1990,04 18 21 23 28 29 34 36 44 49 50 54 55 63 64 66 68 69 71 74
11/05/1990,03 07 09 12 15 17 23 25 27 29 44 53 59 62 66 68 71 73 75 78
11/04/1990,04 05 06 08 09 13 16 18 26 35 36 44 46 51 55 60 61 65 75 80
11/03/1990,05 06 07 09 10 19 20 23 31 35 38 40 49 58 59 60 63 69 73 80
11/02/1990,03 04 11 21 23 26 31 32 42 45 48 54 56 60 62 66 68 72 74 80
11/01/1990,01 04 19 20 22 26 35 37 39 40 41 45 46 47 48 58 63 66 69 71
10/31/1990,07 13 17 22 23 26 31 32 35 36 39 44 45 54 56 68 72 73 77 79
10/30/1990,02 03 04 09 11 14 17 20 28 37 38 40 42 53 54 56 67 72 77 80
10/29/1990,02 07 11 12 14 18 20 38 42 47 52 55 56 57 60 65 73 74 75 79
10/28/1990,08 10 15 19 21 28 34 37 40 41 42 47 48 49 53 54 57 61 68 75
10/27/1990,01 02 03 09 13 14 16 27 28 31 35 36 41 42 45 49 55 63 64 78
10/26/1990,02 03 08 15 27 30 32 44 47 54 61 67 69 70 71 74 76 77 78 79
10/25/1990,01 10 14 18 20 21 22 27 32 37 38 44 47 48 59 63 65 67 70 74
10/24/1990,01 02 10 12 18 26 29 32 36 44 45 47 49 52 53 64 66 67 70 73
10/23/1990,04 05 07 09 10 14 23 26 28 31 38 50 52 56 57 59 66 69 70 74
10/22/1990,04 08 09 10 11 16 17 21 23 26 27 34 38 50 57 59 65 67 75 79
10/21/1990,08 11 24 25 28 35 36 39 42 49 51 56 57 58 60 62 63 72 77 78
10/20/1990,05 16 17 21 23 35 40 41 46 47 58 61 65 67 69 70 71 74 78 80
10/19/1990,03 05 10 12 17 21 22 23 31 34 40 49 50 52 55 58 60 64 72 75
10/18/1990,08 09 12 16 21 30 34 37 38 41 42 43 48 56 59 62 67 73 78 80
10/17/1990,01 03 05 06 13 14 17 22 23 34 45 49 55 65 66 67 68 70 75 77
10/16/1990,03 10 16 17 18 27 28 34 39 48 49 52 59 65 66 69 71 72 74 76
10/15/1990,01 03 05 07 09 12 14 19 22 26 35 36 37 38 40 48 50 51 73 79
10/14/1990,02 11 17 27 30 34 37 38 39 40 53 56 58 59 63 65 68 74 76 79
10/13/1990,08 10 11 17 18 19 23 25 26 32 36 42 50 51 53 59 64 69 70 72
10/12/1990,05 14 17 20 21 23 39 42 45 46 47 50 51 57 59 65 71 72 75 76
10/11/1990,02 07 16 17 20 21 22 27 29 30 31 35 40 44 45 63 67 68 73 79
10/10/1990,01 02 03 06 09 10 17 18 21 26 29 36 43 48 61 66 69 73 76 80
10/09/1990,01 08 12 15 19 23 24 26 32 35 37 39 41 44 45 46 51 64 73 76
10/08/1990,03 06 08 09 28 35 38 40 41 43 44 47 49 50 51 53 58 65 71 77
10/07/1990,01 02 03 04 10 18 25 33 41 44 48 54 55 58 61 62 65 67 68 71
10/06/1990,05 09 11 16 19 25 26 27 34 37 38 42 43 52 60 63 66 67 72 76
10/05/1990,07 08 12 13 18 20 28 30 34 40 44 46 47 50 51 57 61 62 69 79
10/04/1990,02 03 04 06 09 11 14 17 19 21 28 43 45 49 50 58 59 62 68 71
10/03/1990,01 02 05 14 15 17 20 22 24 26 41 47 50 53 57 58 63 66 71 78
10/02/1990,02 07 10 11 12 13 22 23 30 31 33 34 36 39 58 63 66 75 76 79
10/01/1990,03 09 11 12 28 29 33 41 45 47 50 53 54 70 73 75 77 78 79 80
09/30/1990,02 14 15 17 18 19 21 30 31 44 45 54 59 60 61 63 64 72 76 78
09/29/1990,05 09 16 18 19 22 27 31 42 44 45 48 58 63 64 65 70 75 77 79
09/28/1990,06 07 09 29 30 32 35 37 48 49 50 53 55 57 67 68 70 72 74 80
09/27/1990,05 09 11 12 14 18 26 32 36 37 39 42 48 49 54 61 67 68 73 80
09/26/1990,07 08 09 12 13 22 23 24 38 39 40 42 51 55 56 59 60 64 66 72
09/25/1990,01 10 12 13 15 16 17 27 32 33 36 43 51 54 56 59 62 66 70 76
09/24/1990,01 09 21 22 23 25 31 32 44 46 48 54 55 57 58 62 66 70 74 79
09/23/1990,15 18 20 22 25 27 28 31 43 46 50 52 56 58 60 61 68 73 76 78
09/22/1990,07 14 18 19 21 30 31 32 36 46 52 56 58 66 68 71 74 75 76 78
09/21/1990,06 07 10 11 13 22 25 28 31 38 39 40 41 49 56 58 60 69 70 75
09/20/1990,08 10 15 24 25 26 34 38 39 40 43 44 45 50 56 59 60 65 78 80
09/19/1990,05 12 17 21 23 32 33 40 44 46 52 54 55 56 57 62 66 67 69 72
```

```
09/18/1990,06 07 16 19 20 26 29 31 32 33 34 35 43 48 56 64 65 67 73 76
09/17/1990,02 05 06 20 23 25 28 30 31 49 50 51 58 61 62 68 72 73 76 79
09/16/1990,01 03 04 16 19 21 27 38 41 46 47 49 52 59 60 61 72 74 76 77
09/15/1990,06 07 08 11 21 23 24 29 31 32 37 43 44 52 58 59 61 66 77 79
09/14/1990,04 06 12 19 22 24 28 34 38 39 42 46 48 52 53 54 64 65 66 77
09/13/1990,01 02 06 17 26 27 30 31 39 42 49 51 55 59 60 61 63 68 72 77
09/12/1990,02 07 11 14 15 17 24 26 28 33 36 41 43 47 50 53 64 65 67 74
09/11/1990,01 02 04 05 07 10 20 22 28 35 39 42 50 55 56 58 68 71 74 78
09/10/1990,01 06 08 14 17 21 32 42 49 57 58 61 64 69 73 75 76 77 79 80
09/09/1990,08 10 13 15 16 19 26 40 42 46 47 48 50 55 57 58 61 62 64 76
09/08/1990,01 02 10 11 15 17 27 33 35 37 38 41 42 45 47 50 58 65 68 69
09/07/1990,05 06 10 11 14 17 22 29 32 39 50 51 52 55 60 61 62 74 78 79
09/06/1990,01 03 05 12 19 23 25 26 29 42 44 46 48 59 60 61 76 77 78 79
09/05/1990,09 22 24 26 33 34 38 40 43 46 49 53 54 58 62 66 67 74 75 77
09/04/1990,04 09 13 18 19 20 22 29 34 48 54 58 64 68 71 73 75 77 79 80
09/03/1990,03 15 19 22 24 27 35 38 43 50 52 57 58 62 65 66 67 71 72 78
09/02/1990,01 06 07 13 14 16 18 25 41 45 47 54 56 58 61 64 66 69 70 78
09/01/1990,06 07 11 14 16 25 29 31 42 44 45 52 57 59 66 70 71 75 77 78
08/31/1990,05 23 25 28 29 30 32 38 41 44 46 48 49 54 58 68 72 74 75 77
08/30/1990,01 03 04 11 12 16 18 23 33 34 36 44 53 55 61 65 68 72 74 79
08/29/1990,09 10 15 17 19 26 29 31 33 37 38 41 44 48 49 62 64 66 67 78
08/28/1990,07 08 17 18 20 25 27 28 29 30 34 39 40 41 48 50 57 66 75 77
08/27/1990,01 15 17 18 19 20 30 32 34 35 37 39 43 44 48 54 65 66 68 75
08/26/1990,06 08 13 15 18 21 24 32 34 35 36 41 44 46 53 59 62 64 75 76
08/25/1990,03 05 06 10 13 14 15 21 24 31 34 36 39 45 48 63 67 75 78 80
08/24/1990,04 09 10 19 26 27 31 33 34 35 43 45 52 54 57 58 67 68 69 77
08/23/1990,03 06 11 14 17 33 36 38 41 45 54 58 68 69 71 73 77 78 79 80
08/22/1990,09 12 19 25 26 38 39 40 46 47 48 54 56 60 63 66 71 74 75 79
08/21/1990,02 04 06 08 09 21 23 26 28 29 30 35 41 43 47 61 62 66 68 76
08/20/1990,02 16 21 23 25 33 34 37 44 46 51 53 57 62 63 64 71 74 78 79
08/19/1990,01 05 11 13 20 21 22 24 33 36 39 43 46 54 55 57 58 62 66 71
08/18/1990,01 04 07 08 32 33 35 37 38 39 40 41 45 47 48 56 64 71 72 76
08/17/1990,02 04 05 13 17 27 34 38 39 40 43 51 53 58 60 64 66 73 76 80
08/16/1990,01 07 15 19 24 26 31 34 35 36 37 43 44 45 49 54 62 72 73 74
08/15/1990,10 12 13 19 21 22 27 30 31 40 42 45 50 59 61 63 66 68 74 80
08/14/1990,03 05 16 18 25 26 29 31 34 44 49 50 51 54 55 65 67 70 78 80
08/13/1990,01 02 04 09 13 16 17 36 37 54 57 60 61 62 64 73 74 75 78 80
08/12/1990,02 03 06 08 09 10 13 18 22 28 29 39 40 43 47 54 60 65 78 80
08/11/1990,02 09 11 18 20 22 24 26 28 31 36 43 45 52 53 63 65 68 72 77
08/10/1990,01 03 04 13 19 20 23 24 32 37 39 41 42 45 46 49 55 60 70 71
08/09/1990,01 11 16 20 25 26 30 32 39 53 55 57 58 60 63 66 69 71 77 79
08/08/1990,02 04 10 12 15 19 26 38 45 48 49 53 55 57 59 64 68 69 71 74
08/07/1990,01 02 11 12 15 18 35 36 39 41 46 49 50 54 58 66 68 71 72 79
08/06/1990,02 07 08 09 11 12 18 19 21 24 27 29 36 37 39 51 53 55 62 67
08/05/1990,03 07 08 09 10 11 23 32 36 37 39 43 46 49 52 57 70 73 75 79
08/04/1990,03 14 17 20 21 24 25 30 31 33 39 43 44 46 49 50 51 53 78 80
08/03/1990,02 15 19 20 21 23 24 36 43 44 51 53 58 65 66 68 72 75 76 79
08/02/1990,02 03 07 12 13 16 19 26 27 28 33 41 59 61 67 69 71 78 79 80
08/01/1990,01 02 09 11 24 31 33 34 37 41 44 45 53 55 56 63 66 68 79 80
07/31/1990,01 03 05 08 18 21 23 27 32 37 45 47 48 54 58 59 63 67 74 76
07/30/1990,01 09 11 18 21 23 31 32 42 44 52 56 62 64 67 68 69 73 78 79
07/29/1990,05 10 14 15 20 21 31 36 40 43 45 57 59 61 65 69 71 73 79 80
07/28/1990,02 04 05 08 09 14 15 16 21 28 30 31 33 39 41 43 67 73 75 78
07/27/1990,02 03 12 14 18 19 28 29 30 32 35 36 38 42 53 55 56 59 77 80
```

```
93      07/26/1990,04 07 08 09 10 14 19 23 25 33 34 36 39 46 47 48 50 64 72 79
94      07/25/1990,01 11 17 22 23 29 30 37 40 43 50 51 54 59 62 66 71 76 79 80
95      07/24/1990,02 03 08 20 30 32 39 40 41 44 46 49 52 54 58 64 65 68 72 80
96      07/23/1990,05 06 09 15 17 19 21 23 25 31 32 33 37 42 43 48 55 63 65 71
97      07/22/1990,05 07 11 14 19 24 30 44 46 48 49 51 52 55 58 67 73 74 76 77
98      07/21/1990,09 10 12 19 20 25 26 30 34 36 47 52 64 68 69 71 73 74 75 79
99      07/20/1990,02 03 04 18 19 20 27 37 43 48 51 53 54 58 59 64 68 74 76 78
00      07/19/1990,01 02 11 12 18 21 23 24 25 28 39 52 53 62 66 69 71 72 74 76
01      07/18/1990,01 05 06 17 23 31 35 45 46 53 54 55 57 58 61 62 66 67 73 77
02      07/17/1990,03 11 12 13 15 17 29 32 38 39 44 47 48 50 58 63 65 70 71 76
03      07/16/1990,01 08 09 16 17 19 23 34 36 39 43 46 50 55 59 60 63 64 66 71
04      07/15/1990,09 15 24 26 27 32 34 40 41 46 51 54 55 60 64 65 66 70 75 80
05      07/14/1990,04 09 12 16 22 31 33 37 38 41 43 53 55 58 62 66 71 72 79 80
06      07/13/1990,04 07 08 15 23 24 39 40 42 45 47 48 51 53 54 62 66 71 74 78
07      07/12/1990,02 05 08 12 14 17 29 31 36 37 42 45 50 53 59 60 68 72 73 76
08      07/11/1990,12 13 14 15 19 20 23 25 26 27 29 37 38 44 47 48 59 65 67 71
09      07/10/1990,04 07 08 09 11 19 20 23 26 31 34 35 51 52 59 63 66 67 68 71
10      07/09/1990,02 11 13 24 29 31 34 42 44 46 47 58 60 62 63 66 70 74 75 76
11      07/08/1990,01 03 04 05 11 16 17 19 27 31 36 38 43 46 51 52 58 63 74 79
12      07/07/1990,03 08 14 15 18 24 31 33 34 38 41 46 47 50 53 58 61 62 65 67
13      07/06/1990,02 03 08 14 15 21 28 33 35 36 38 39 40 47 48 54 55 61 69 73
14      07/05/1990,02 03 11 16 20 23 26 33 34 41 42 47 50 55 56 59 70 71 73 79
15      07/04/1990,02 05 06 09 13 16 17 20 29 32 40 44 49 52 59 61 62 69 74 78
16      07/03/1990,01 08 15 24 27 35 36 37 38 39 50 51 54 55 56 57 61 63 69 77
17      07/02/1990,01 04 06 09 11 14 16 19 21 22 24 28 30 31 37 45 55 56 59 60
18      07/01/1990,05 09 11 24 25 33 34 38 41 47 49 54 57 60 65 66 73 75 78 79
19      06/30/1990,07 19 24 34 35 38 39 40 42 43 44 45 47 48 58 65 68 69 71 73
20      06/29/1990,06 10 12 17 18 19 20 21 27 30 32 39 40 48 50 55 56 59 68 74
21      06/28/1990,06 07 09 12 15 19 26 31 33 34 37 44 45 60 62 64 65 67 69 74
22      06/27/1990,02 03 07 21 25 26 27 28 30 31 32 33 35 39 40 58 60 61 74 80
23      06/26/1990,01 08 14 16 17 19 28 33 34 41 48 49 50 54 55 59 62 70 72 79
24      06/25/1990,05 07 10 12 13 14 24 28 29 30 31 33 36 37 38 41 55 64 65 69
25      06/24/1990,02 04 10 19 20 21 22 27 29 38 54 58 66 68 71 72 74 75 76 79
26      06/23/1990,01 03 12 14 15 17 18 24 25 32 36 42 43 44 49 58 61 74 76 80
27      06/22/1990,06 07 16 18 20 24 31 35 45 47 48 51 55 60 64 67 75 76 77 79
28      06/21/1990,01 16 17 18 22 30 31 38 47 49 51 59 62 63 64 65 69 70 72 77
29      06/20/1990,06 18 29 32 34 37 39 42 44 45 52 53 54 58 61 65 67 69 72 73
30      06/19/1990,07 08 11 12 15 17 18 19 24 25 28 32 38 49 57 58 65 70 73 77
31      06/18/1990,04 06 09 12 19 24 25 26 28 30 37 38 42 45 46 49 57 67 74 76
32      06/17/1990,08 13 16 22 23 26 34 38 40 49 54 61 65 67 68 69 70 72 73 76
33      06/16/1990,02 05 08 12 16 22 23 41 42 43 44 47 50 52 60 62 63 71 78 79
34      06/15/1990,06 09 12 13 18 20 24 30 31 45 49 53 54 55 56 58 61 76 78 80
35      06/14/1990,02 04 13 15 22 27 32 36 43 51 54 55 56 58 63 64 68 69 76 79
36      06/13/1990,06 10 12 13 16 21 33 40 41 43 58 59 61 62 63 68 70 71 75 78
37      06/12/1990,02 07 14 18 19 26 32 34 39 47 50 51 52 54 55 56 57 63 65 70
38      06/11/1990,01 02 03 07 09 10 14 16 24 32 36 43 46 47 56 65 69 71 74 78
39      06/10/1990,05 09 12 15 18 20 29 30 34 35 39 41 43 53 59 62 67 69 72 79
40      06/09/1990,05 22 25 29 31 33 34 37 38 39 41 43 44 46 50 54 56 57 66 78
41      06/08/1990,01 11 15 20 29 30 33 34 37 39 47 49 54 65 70 72 74 75 78 79
42      06/07/1990,08 11 16 27 28 29 33 37 40 42 43 44 45 60 62 65 68 75 78 80
43      06/06/1990,02 03 05 07 08 15 21 23 38 42 46 49 61 64 67 68 69 76 79 80
44      06/05/1990,03 06 10 21 24 28 29 30 33 43 46 55 58 60 69 72 73 74 76 77
45      06/04/1990,05 10 13 15 17 23 24 28 29 31 33 37 41 49 53 60 68 76 77 78
46      06/03/1990,07 15 20 22 27 30 43 46 51 54 55 63 68 71 73 74 76 77 79 80
```

```
47      06/02/1990,01 04 05 22 25 28 31 34 37 38 45 49 52 54 56 57 60 65 68 69
48      06/01/1990,01 05 10 21 35 36 38 39 43 55 57 59 60 61 64 65 67 68 70 75
49      05/31/1990,02 07 08 11 12 15 23 37 41 44 49 50 55 58 69 71 75 76 77 80
50      05/30/1990,03 08 11 13 18 24 25 33 34 36 40 48 53 63 64 67 68 71 75 78
51      05/29/1990,01 02 05 08 09 15 23 25 32 34 35 37 44 45 49 55 56 60 72 77
52      05/28/1990,04 09 10 13 15 24 32 34 36 37 43 51 54 55 57 59 63 71 75 79
53      05/27/1990,03 06 09 15 18 22 24 28 33 37 42 48 58 62 63 64 70 75 77 80
54      05/26/1990,01 03 06 15 17 19 25 31 33 38 41 42 45 53 56 66 67 68 75 76
55      05/25/1990,04 05 06 07 12 15 18 25 30 34 39 41 49 52 55 56 57 60 62 77
56      05/24/1990,01 07 17 20 23 32 33 36 37 41 48 52 53 55 58 67 68 73 74 78
57      05/23/1990,02 03 08 21 23 24 26 33 34 35 37 45 51 54 61 62 65 69 72 75
58      05/22/1990,02 03 04 14 18 24 25 27 32 34 36 39 52 55 59 67 70 74 79 80
59      05/21/1990,06 07 22 29 30 33 36 37 38 47 50 51 52 58 61 63 71 72 74 80
60      05/20/1990,01 06 14 15 18 25 26 33 34 37 38 40 47 50 53 54 66 68 71 80
61      05/19/1990,02 05 07 20 25 26 31 35 36 39 46 48 51 54 55 62 64 72 73 77
62      05/18/1990,07 16 18 21 22 24 33 35 41 45 47 52 59 61 63 70 72 73 75 79
63      05/17/1990,15 17 18 19 29 30 32 40 42 43 46 50 55 61 62 69 70 71 76 79
64      05/16/1990,03 05 19 22 25 27 28 30 31 32 37 46 53 54 55 60 61 63 65 79
65      05/15/1990,07 08 15 17 18 21 22 23 26 27 28 41 42 43 50 52 70 72 76 78
66      05/14/1990,03 04 11 13 36 42 45 46 49 50 51 53 55 59 65 68 71 72 73 80
67      05/13/1990,02 07 10 12 13 23 25 31 36 42 43 52 54 55 64 65 71 72 77 79
68      05/12/1990,04 05 13 16 18 19 27 29 32 35 39 44 46 50 52 57 70 71 76 79
69      05/11/1990,01 04 09 12 16 20 25 29 30 33 34 35 40 42 44 58 64 65 66 71
70      05/10/1990,02 05 14 15 19 27 37 38 40 41 42 50 51 53 54 55 66 68 70 75
71      05/09/1990,01 08 09 15 16 25 38 41 44 45 52 53 56 59 60 61 69 70 73 76
72      05/08/1990,01 04 08 10 12 19 21 24 31 34 38 43 44 45 48 57 60 72 78 80
73      05/07/1990,02 04 05 11 15 16 18 27 36 41 44 50 53 56 57 61 72 74 76 78
74      05/06/1990,04 07 12 18 22 23 26 30 33 34 37 38 40 42 53 55 64 65 74 76
75      05/05/1990,01 09 10 13 16 18 24 27 31 33 41 48 56 57 59 61 69 74 76 78
76      05/04/1990,03 07 15 16 18 21 22 24 25 29 35 38 39 43 61 64 65 67 70 80
77      05/03/1990,08 09 10 11 12 14 15 26 27 28 29 30 31 34 39 48 53 57 70 74
78      05/02/1990,01 03 15 18 22 23 26 28 31 39 42 45 46 52 59 63 67 70 71 79
79      05/01/1990,02 10 12 21 25 26 28 31 35 36 37 39 40 51 55 61 64 68 70 74
80      04/30/1990,08 14 20 21 24 29 30 32 34 41 42 50 53 56 63 64 68 70 78 80
81      04/29/1990,06 09 13 23 26 29 31 40 51 54 55 57 60 64 65 67 70 75 76 80
82      04/28/1990,16 17 21 22 24 29 37 40 41 42 46 52 53 57 62 63 66 72 79 80
83      04/27/1990,03 19 21 22 27 30 32 33 38 39 41 46 47 48 53 59 66 70 71 79
84      04/26/1990,02 11 12 27 28 35 40 41 51 53 54 60 61 66 67 71 72 75 76 77
85      04/25/1990,04 05 10 14 18 19 20 24 26 27 44 50 59 60 62 64 67 72 75 78
86      04/24/1990,04 07 12 14 25 28 30 34 35 37 38 44 46 50 51 56 61 74 77 78
87      04/23/1990,02 03 04 13 14 16 18 30 31 36 37 41 42 48 50 51 54 55 59 80
88      04/22/1990,01 08 09 11 14 20 35 37 40 42 47 49 56 57 59 61 66 67 69 71
89      04/21/1990,01 08 09 11 18 25 26 36 38 39 41 42 46 50 51 64 67 72 78 80
90      04/20/1990,13 16 18 21 23 24 25 27 29 33 39 42 44 51 53 55 62 66 67 78
91      04/19/1990,13 18 21 24 34 39 40 41 42 43 46 47 49 50 52 53 57 60 68 74
92      04/18/1990,02 03 04 06 09 17 19 21 25 35 36 44 49 53 62 63 65 68 76 78
93      04/17/1990,01 05 07 08 12 14 21 25 29 38 45 49 50 58 61 62 66 68 77 80
94      04/16/1990,04 06 07 09 14 15 18 19 23 31 34 36 47 53 54 58 59 63 65 67
95      04/15/1990,08 13 17 19 20 29 34 42 44 45 48 50 52 53 58 60 63 69 77 80
96      04/14/1990,02 03 04 06 07 11 16 18 20 21 28 30 39 49 56 61 63 68 70 80
97      04/13/1990,04 16 18 20 23 26 27 30 33 39 45 51 58 61 62 63 75 76 78 79
98      04/12/1990,03 07 17 19 21 22 26 28 29 30 33 39 45 50 52 53 60 65 67 69
99      04/11/1990,04 09 15 19 21 23 24 25 31 32 34 44 49 53 55 57 64 65 68 75
00      04/10/1990,02 03 07 08 09 15 19 25 27 33 35 36 49 52 61 64 68 69 75 80
```

01 04/09/1990,03 04 08 11 13 16 19 25 33 34 40 47 55 64 65 68 70 72 76 79
02 04/08/1990,06 09 13 18 22 24 27 29 32 33 37 43 44 46 53 54 55 64 65 66
03 04/07/1990,07 12 14 18 19 20 21 29 30 37 38 44 45 50 52 58 65 68 69 73
04 04/06/1990,01 02 03 06 15 20 23 31 37 38 40 45 46 55 62 67 68 75 76 80
05 04/05/1990,03 08 11 12 19 21 23 24 27 32 38 44 54 58 62 64 67 69 75 77
06 04/04/1990,03 05 08 09 10 16 17 20 29 32 34 40 46 48 51 55 62 66 72 79
07 04/03/1990,02 06 10 13 14 16 17 25 40 47 48 51 52 54 58 61 67 72 74 79
08 04/02/1990,04 06 14 15 17 26 27 29 31 34 35 49 56 58 63 67 71 74 75 80
09 04/01/1990,05 08 14 16 19 20 27 30 33 34 36 40 49 50 52 54 55 56 75 76
10 03/31/1990,04 05 09 13 14 17 23 38 40 41 45 46 56 57 59 61 73 74 76 79
11 03/30/1990,03 04 05 06 19 23 35 42 44 45 53 56 60 64 65 67 68 69 72 76
12 03/29/1990,03 09 18 19 24 26 37 41 47 48 50 54 55 56 64 65 66 71 74 75
13 03/28/1990,12 15 17 23 27 28 33 41 43 44 47 48 50 55 57 61 62 63 67 68
14 03/27/1990,06 09 10 16 25 29 32 34 36 38 40 41 43 44 51 54 59 62 78 79
15 03/26/1990,03 14 23 29 31 35 36 37 38 43 46 51 52 57 59 64 65 69 70 75
16 03/25/1990,07 13 19 20 26 27 36 37 41 44 45 47 51 55 57 60 62 71 75 79
17 03/24/1990,07 08 09 15 17 18 21 34 41 44 50 53 55 65 69 70 76 77 78 80
18 03/23/1990,07 10 16 17 23 24 36 37 41 43 45 47 49 50 60 64 66 67 78 79
19 03/22/1990,07 08 09 21 22 25 26 38 43 45 46 48 50 53 57 60 62 63 66 70
20 03/21/1990,02 03 10 11 15 19 30 36 40 42 51 52 59 60 63 64 69 74 76 80
21 03/20/1990,06 08 12 13 28 30 32 33 48 52 53 60 61 62 66 69 72 74 76 80
22 03/19/1990,01 06 07 12 18 22 24 34 37 39 42 45 48 54 55 56 63 71 73 75
23 03/18/1990,01 02 04 06 17 19 22 28 29 36 37 39 41 49 52 59 62 64 65 66
24 03/17/1990,01 08 10 16 17 19 22 36 38 46 56 57 58 60 61 64 66 68 77 78
25 03/16/1990,04 10 11 22 23 25 35 38 39 42 43 44 45 47 55 56 61 63 73 76
26 03/15/1990,01 02 15 24 25 26 27 30 36 37 39 43 45 48 50 62 63 64 69 75
27 03/14/1990,01 13 14 17 20 30 32 39 40 41 43 49 56 57 61 63 70 72 74 80
28 03/13/1990,05 26 28 29 36 38 40 41 45 46 47 55 56 57 63 64 67 70 74 75
29 03/12/1990,17 20 21 26 28 35 40 41 47 50 53 54 61 63 65 71 73 74 77 80
30 03/11/1990,01 14 16 21 29 30 32 37 41 42 46 48 54 56 58 65 67 75 76 77
31 03/10/1990,13 20 21 22 23 25 32 33 37 39 40 42 46 48 57 59 61 76 77 79
32 03/09/1990,04 10 12 14 20 24 29 37 38 39 41 45 49 50 52 58 59 63 67 71
33 03/08/1990,02 04 06 16 29 35 37 39 42 46 53 57 61 62 65 68 69 71 73 80
34 03/07/1990,02 03 06 11 14 15 26 27 30 34 39 45 50 53 66 69 72 75 76 79
35 03/06/1990,05 06 17 20 24 30 35 36 38 40 42 45 49 54 57 58 59 60 73 76
36 03/05/1990,04 05 07 08 14 15 21 27 29 35 39 41 45 48 66 68 70 74 75 77
37 03/04/1990,01 10 14 19 22 24 26 34 36 39 48 52 55 57 66 70 71 75 76 80
38 03/03/1990,03 07 08 09 15 17 20 21 24 31 38 41 56 58 61 63 65 68 72 79
39 03/02/1990,03 08 11 12 13 17 20 26 27 43 45 46 58 66 69 71 72 73 75 78
40 03/01/1990,01 02 03 06 11 12 19 23 25 41 42 44 46 49 59 64 72 75 78 80
41 02/28/1990,02 05 11 14 21 24 25 27 36 38 41 47 60 61 63 74 75 76 77 79
42 02/27/1990,01 07 08 11 15 16 23 26 30 31 35 42 45 46 54 55 57 58 62 73
43 02/26/1990,11 14 15 16 20 25 27 29 35 38 39 45 46 48 63 64 69 74 75 76
44 02/25/1990,15 20 23 24 25 27 31 32 43 48 52 54 55 61 62 66 67 70 72 77
45 02/24/1990,01 05 08 13 14 17 18 20 21 22 25 46 51 54 55 61 63 66 69 76
46 02/23/1990,01 11 12 14 23 28 31 33 35 36 40 41 46 50 57 60 61 73 75 79
47 02/22/1990,10 16 23 24 29 31 33 34 40 49 51 56 64 70 71 73 74 76 77 79
48 02/21/1990,02 13 14 28 40 41 42 43 47 52 55 56 61 64 66 69 70 73 74 75
49 02/20/1990,01 04 08 09 14 16 24 25 26 29 34 42 63 69 70 71 74 77 79 80
50 02/19/1990,01 03 06 11 21 22 25 29 35 41 45 47 56 58 62 67 69 72 74 77
51 02/18/1990,05 06 08 11 13 15 24 25 29 33 40 48 51 57 59 60 71 76 77 79
52 02/17/1990,01 03 04 06 09 18 21 23 26 30 31 33 41 42 56 59 68 71 77 79
53 02/16/1990,03 04 09 10 11 22 24 31 38 46 47 50 52 53 54 55 60 62 65 78
54 02/15/1990,01 08 10 15 18 21 23 28 33 38 39 42 49 51 55 61 62 71 77 80

```
55      02/14/1990,06 11 13 20 21 27 30 31 39 41 43 45 50 51 54 62 67 72 73 74
56      02/13/1990,05 06 07 13 18 20 25 26 29 48 49 52 54 57 58 65 67 68 71 76
57      02/12/1990,02 06 08 15 19 20 23 28 30 33 35 44 47 48 50 51 52 62 74 77
58      02/11/1990,09 12 13 14 16 22 23 24 29 37 42 49 51 52 53 55 57 64 69 73
59      02/10/1990,04 05 06 20 21 25 26 28 32 33 36 42 43 46 48 53 66 73 75 76
60      02/09/1990,02 05 10 18 19 21 32 33 35 37 42 43 47 53 55 57 60 63 64 79
61      02/08/1990,03 07 09 10 12 17 19 20 25 29 30 33 34 36 42 56 60 62 69 77
62      02/07/1990,08 15 16 23 26 28 32 38 42 56 58 62 63 70 71 72 75 76 79 80
63      02/06/1990,02 06 08 11 16 19 21 23 26 27 29 31 37 43 50 62 70 74 77 79
64      02/05/1990,03 05 07 10 15 24 27 32 37 39 40 44 46 47 58 62 66 68 69 78
65      02/04/1990,02 04 06 12 14 20 21 25 29 33 39 47 56 61 66 67 72 73 78 79
66      02/03/1990,07 08 09 30 32 33 35 37 40 41 44 49 50 53 57 60 62 63 65 76
67      02/02/1990,01 04 06 08 10 15 21 27 30 32 34 39 45 48 49 53 62 64 73 80
68      02/01/1990,04 05 08 13 21 26 28 33 35 36 37 42 44 47 58 65 67 71 72 74
69      01/31/1990,07 09 14 16 20 22 24 28 32 34 38 39 43 44 58 61 63 67 68 74
70      01/30/1990,03 08 17 18 20 29 32 33 34 38 40 41 55 56 62 65 66 70 71 76
71      01/29/1990,04 13 14 15 18 32 34 39 41 43 45 50 53 64 66 70 71 75 78 80
72      01/28/1990,02 06 14 23 30 33 36 38 40 41 59 60 61 64 72 75 77 78 79 80
73      01/27/1990,04 06 24 27 31 35 36 38 43 49 56 62 63 65 67 68 71 73 76 77
74      01/26/1990,02 04 06 10 14 15 18 41 43 47 49 54 55 56 57 63 69 73 77 79
75      01/25/1990,02 06 12 15 23 36 37 38 41 43 47 55 57 60 61 71 74 76 77 79
76      01/24/1990,01 03 05 09 13 22 25 26 28 33 37 38 41 43 51 57 58 62 67 71
77      01/23/1990,02 04 07 12 13 15 16 20 21 24 30 39 40 44 45 46 51 61 64 73
78      01/22/1990,02 04 08 16 17 20 25 33 36 45 46 48 50 52 55 58 62 69 79 80
79      01/21/1990,01 02 03 04 08 15 30 31 35 43 45 47 48 51 59 64 71 76 79 80
80      01/20/1990,05 18 20 23 32 34 36 38 40 43 48 50 51 54 56 63 64 73 75 78
81      01/19/1990,06 10 11 12 14 25 39 40 41 42 43 53 58 63 64 65 66 67 74 76
82      01/18/1990,09 16 19 26 32 33 35 37 41 42 43 45 47 48 54 56 60 69 73 75
83      01/17/1990,03 05 14 26 28 29 34 36 44 45 46 47 56 58 60 61 67 70 71 75
84      01/16/1990,06 08 11 14 16 21 26 46 48 49 50 52 53 54 58 61 67 69 74 75
85      01/15/1990,05 12 13 14 15 25 30 32 34 40 42 58 60 62 63 65 70 72 74 80
86      01/14/1990,05 06 07 08 09 12 13 14 18 25 32 34 36 48 52 54 57 58 73 79
87      01/13/1990,07 14 21 22 24 25 26 29 43 45 53 54 55 56 58 60 62 65 68 75
88      01/12/1990,04 05 07 08 15 17 21 22 26 28 31 35 36 39 42 44 56 61 65 73
89      01/11/1990,05 14 15 16 18 19 21 22 26 35 42 49 54 55 58 59 63 66 76 79
90      01/10/1990,01 06 08 14 23 24 26 27 30 35 39 44 52 55 61 65 66 72 73 76
91      01/09/1990,01 03 07 09 14 18 20 28 29 40 41 48 56 60 65 66 72 76 78 80
92      01/08/1990,01 05 07 09 12 16 18 24 27 30 38 41 45 60 62 63 67 69 74 77
93      01/07/1990,03 10 19 25 28 29 35 50 54 56 58 61 63 69 73 74 76 77 79 80
94      01/06/1990,04 05 20 26 28 29 32 34 36 40 48 49 53 56 62 63 64 69 72 74
95      01/05/1990,05 06 09 12 18 20 21 36 43 47 50 53 54 57 64 65 68 72 75 79
96      01/04/1990,01 02 03 04 13 19 23 31 34 39 41 43 47 58 62 65 69 72 75 78
97      01/03/1990,01 05 10 11 20 21 22 23 25 27 29 30 34 39 46 54 60 65 74 79
98      01/02/1990,08 09 17 18 19 20 30 31 34 42 44 47 51 54 63 64 69 70 71 72
99      01/01/1990,06 09 17 21 31 35 36 40 42 45 50 52 59 67 68 69 74 77 78 79
00      12/31/1989,02 07 08 16 19 21 23 24 37 51 52 57 59 61 64 68 69 70 72 74
01      12/30/1989,01 08 13 14 20 26 33 34 39 45 46 49 57 65 66 70 71 72 74 78
02      12/29/1989,01 03 05 07 09 11 18 19 28 29 31 32 37 40 41 51 56 61 73 78
03      12/28/1989,05 12 16 18 19 26 39 42 46 50 51 56 58 59 63 66 68 72 75 80
04      12/27/1989,08 10 13 14 15 18 26 27 30 32 33 34 35 42 46 52 56 62 64 72
05      12/26/1989,02 20 23 24 28 29 30 36 39 41 45 47 58 59 60 70 73 75 77 78
06      12/24/1989,02 04 11 12 16 18 20 24 32 33 36 43 54 61 62 63 73 74 77 80
07      12/23/1989,01 05 06 19 21 22 26 27 28 32 35 40 45 46 50 51 57 59 61 63
08      12/22/1989,04 10 11 15 17 18 19 20 25 27 28 31 35 46 47 48 55 57 66 74
```

09 12/21/1989,01 09 14 25 26 33 38 39 41 46 47 51 52 57 66 71 72 75 77 78
10 12/20/1989,17 21 23 24 27 32 34 36 37 44 46 55 58 59 61 63 66 74 79 80
11 12/19/1989,01 10 14 15 16 21 27 31 38 40 43 51 56 57 60 64 67 68 70 78
12 12/18/1989,06 07 10 14 21 22 25 28 30 31 33 43 44 45 48 54 65 67 72 79
13 12/17/1989,09 10 13 17 19 22 31 34 36 37 44 53 57 62 63 64 66 69 73 77
14 12/16/1989,02 08 10 16 18 19 20 21 26 28 31 35 39 40 48 60 64 65 73 78
15 12/15/1989,01 06 07 09 16 28 30 31 34 38 43 48 49 52 53 54 66 67 72 78
16 12/14/1989,01 03 09 11 16 17 19 23 25 30 31 34 37 45 48 52 54 57 58 76
17 12/13/1989,07 09 12 15 18 22 35 39 41 44 46 48 51 52 55 57 59 67 78 80
18 12/12/1989,04 05 11 17 18 20 25 26 27 33 34 35 41 43 46 47 52 57 70 77
19 12/11/1989,03 07 08 09 10 11 14 23 26 28 35 36 42 46 48 55 56 68 74 78
20 12/10/1989,03 05 06 09 11 17 20 21 25 33 35 39 42 43 44 52 57 59 62 71
21 12/09/1989,08 12 19 24 26 29 32 43 49 52 54 55 57 59 60 65 68 69 72 76
22 12/08/1989,03 08 17 25 27 31 32 34 38 39 42 45 49 55 56 67 69 75 76 78
23 12/07/1989,07 08 10 14 15 24 25 29 30 33 34 35 37 40 60 64 66 70 73 75
24 12/06/1989,10 11 15 22 23 25 27 35 37 40 42 43 45 47 57 59 60 64 73 74
25 12/05/1989,04 05 09 16 23 28 30 36 38 41 42 45 48 54 59 65 68 69 71 72
26 12/04/1989,06 15 18 23 25 30 32 34 36 38 39 44 47 56 57 58 65 68 71 72
27 12/03/1989,01 04 15 25 30 35 40 43 44 48 49 52 57 58 60 63 64 67 71 80
28 12/02/1989,01 11 15 21 24 25 27 31 35 42 45 48 57 60 64 65 67 72 73 79
29 12/01/1989,01 03 05 06 15 17 19 32 37 38 39 41 42 48 52 54 56 64 66 76
30 11/30/1989,10 16 18 29 38 39 44 45 47 48 52 62 63 68 69 71 73 75 78 79
31 11/29/1989,06 12 14 18 24 26 27 28 48 49 53 55 57 61 69 71 73 75 77 78
32 11/28/1989,02 03 05 11 12 24 26 28 29 32 36 42 44 51 52 53 59 68 71 77
33 11/27/1989,02 04 09 16 17 20 21 24 25 35 36 39 44 58 64 65 67 68 69 76
34 11/26/1989,01 03 04 05 20 32 34 42 50 51 52 54 56 58 63 66 73 75 77 78
35 11/25/1989,02 03 06 08 09 15 20 21 25 34 38 41 50 51 59 60 63 67 70 79
36 11/24/1989,03 07 17 25 27 32 35 36 37 38 40 43 45 54 59 64 66 68 69 74
37 11/23/1989,02 05 10 11 13 15 20 21 26 30 31 32 58 59 62 67 69 71 76 79
38 11/22/1989,14 16 17 20 21 25 30 37 40 50 52 54 55 58 61 63 72 73 77 80
39 11/21/1989,01 02 08 11 12 13 17 21 24 27 35 38 43 52 53 56 62 64 66 79
40 11/20/1989,02 14 15 16 23 24 31 38 39 43 44 48 50 53 54 56 59 60 77 79
41 11/19/1989,01 02 04 06 11 13 16 19 33 36 37 39 42 46 54 60 67 68 70 76
42 11/18/1989,06 08 18 24 26 31 37 39 43 50 51 52 54 60 62 64 69 73 74 79
43 11/17/1989,03 06 09 11 14 16 17 27 28 40 44 50 51 52 53 60 61 71 73 79
44 11/16/1989,01 02 09 11 12 13 38 39 42 43 47 50 51 54 61 62 71 73 77 79
45 11/15/1989,04 05 09 10 23 24 26 30 31 38 40 53 58 59 61 62 67 74 76 77
46 11/14/1989,04 06 09 10 12 13 16 28 31 38 39 40 41 51 56 58 75 77 79 80
47 11/13/1989,01 04 11 12 13 16 19 20 21 26 30 46 47 50 52 54 58 68 74 77
48 11/12/1989,02 03 08 10 13 19 20 22 24 27 28 42 47 48 55 61 63 69 74 80
49 11/11/1989,06 10 15 17 18 27 41 45 48 52 54 55 57 58 63 66 71 73 76 77
50 11/10/1989,03 10 18 22 25 27 28 29 30 31 32 52 55 56 58 59 60 62 68 73
51 11/09/1989,04 06 15 18 19 20 22 24 25 30 38 40 42 43 48 50 64 71 72 79
52 11/08/1989,18 19 20 23 28 33 37 43 45 48 52 58 59 60 61 63 66 69 72 79
53 11/07/1989,09 16 18 29 30 31 35 41 44 47 50 51 58 60 64 65 71 74 79 80
54 11/06/1989,01 03 06 08 10 13 14 15 32 37 39 43 47 48 51 57 61 72 74 76
55 11/05/1989,03 06 09 12 16 25 30 36 41 48 49 55 59 66 70 71 73 75 77 78
56 11/04/1989,01 09 11 12 19 24 26 29 32 33 35 37 38 40 41 45 60 73 74 79
57 11/03/1989,02 09 10 14 16 19 21 27 32 41 45 47 53 54 57 62 63 64 77 79
58 11/02/1989,01 09 10 18 19 21 33 40 52 54 57 64 68 70 72 74 76 77 79 80
59 11/01/1989,03 04 06 07 09 12 14 16 18 24 29 31 33 38 39 43 47 65 72 73
60 10/31/1989,06 13 17 22 26 28 31 33 34 36 37 42 47 49 50 52 53 67 72 77
61 10/30/1989,02 03 14 25 34 35 38 39 40 44 48 49 53 58 64 65 67 73 74 79
62 10/29/1989,03 05 06 13 15 21 24 26 28 30 43 44 45 47 50 53 62 69 76 77

```
63      10/28/1989,03 05 08 10 16 17 19 20 26 33 34 36 39 44 49 57 60 71 75 78
64      10/27/1989,07 08 14 17 18 19 20 23 29 32 36 37 45 51 59 63 65 70 77 80
65      10/26/1989,02 12 16 22 24 26 39 40 44 46 47 48 55 60 66 67 72 78 79 80
66      10/25/1989,02 15 21 25 28 30 33 38 40 43 46 61 65 66 67 70 73 74 76 80
67      10/24/1989,02 03 08 21 27 29 30 36 39 48 49 51 59 61 62 64 66 67 68 70
68      10/23/1989,03 07 11 13 16 21 24 25 33 45 50 51 53 55 57 60 62 63 66 72
69      10/22/1989,06 07 10 15 29 30 31 35 40 46 49 56 58 61 73 75 76 77 78 80
70      10/21/1989,03 04 12 13 14 23 28 32 34 36 50 51 56 63 64 67 68 69 70 76
71      10/20/1989,07 08 11 19 28 29 34 44 46 48 50 57 63 65 67 68 73 76 77 80
72      10/19/1989,11 16 20 29 30 32 36 39 43 50 52 57 63 65 66 70 72 73 74 76
73      10/18/1989,09 12 13 16 19 20 23 24 33 36 45 52 63 64 67 68 70 77 78 79
74      10/17/1989,08 13 15 16 19 27 31 32 35 42 43 45 48 55 61 66 69 72 77 79
75      10/16/1989,03 06 07 09 13 15 21 23 35 44 46 49 50 53 55 58 60 65 72 79
76      10/15/1989,02 08 10 15 26 28 35 36 39 40 42 45 51 58 59 61 65 66 73 79
77      10/14/1989,02 04 08 15 17 23 26 33 35 36 37 38 43 46 48 65 70 73 78 79
78      10/13/1989,01 02 11 16 19 20 38 41 42 44 47 51 59 60 63 65 67 71 75 80
79      10/12/1989,03 05 11 14 17 19 25 29 30 34 36 37 38 41 43 67 71 72 78 79
80      10/11/1989,02 08 14 16 20 22 25 32 34 40 42 51 53 56 58 59 62 70 74 80
81      10/10/1989,06 15 17 21 24 27 32 37 41 44 46 52 54 59 61 65 66 69 77 78
82      10/09/1989,04 06 14 16 17 24 26 27 30 33 34 37 42 50 51 57 60 68 70 74
83      10/08/1989,02 03 05 24 32 35 37 51 52 54 56 59 60 68 69 71 73 76 78 80
84      10/07/1989,02 04 06 11 15 17 18 19 20 34 35 36 37 45 48 57 74 75 76 77
85      10/06/1989,05 13 17 19 21 24 25 36 39 42 44 49 50 51 53 65 73 74 76 78
86      10/05/1989,01 07 12 16 19 25 29 34 38 40 45 47 50 53 60 62 66 71 72 74
87      10/04/1989,01 02 04 10 12 18 19 20 24 30 37 42 44 51 58 61 63 65 69 79
88      10/03/1989,15 17 25 31 33 44 45 46 49 51 55 56 57 59 60 72 75 76 78 80
89      10/02/1989,02 03 05 06 11 13 14 15 16 19 25 37 43 50 54 55 56 64 65 80
90      10/01/1989,03 05 06 08 20 26 28 31 35 41 44 47 48 52 53 64 66 69 72 78
91      09/30/1989,06 13 17 20 26 30 39 42 43 48 51 53 54 57 58 61 63 72 73 77
92      09/29/1989,06 10 12 14 15 19 25 32 41 42 44 48 51 54 56 63 68 70 74 78
93      09/28/1989,04 05 06 08 11 21 22 27 37 43 47 49 51 53 54 60 61 69 72 74
94      09/27/1989,02 05 10 17 20 22 33 36 41 47 49 50 53 65 67 69 73 74 77 80
95      09/26/1989,04 06 10 11 23 29 32 35 38 43 46 47 51 53 54 55 64 66 70 77
96      09/25/1989,12 13 14 24 29 30 35 40 45 47 49 51 56 58 63 65 66 71 73 76
97      09/24/1989,02 14 15 17 18 20 24 27 29 35 40 42 43 45 51 52 54 57 60 63
98      09/23/1989,04 12 13 19 26 28 29 32 33 42 43 46 52 55 61 63 65 69 73 80
99      09/22/1989,02 10 12 19 20 21 22 24 25 27 34 36 38 44 46 52 53 65 67 77
00      09/21/1989,01 03 04 05 08 09 12 15 19 22 24 26 28 38 42 43 53 61 68 79
01      09/20/1989,03 04 05 10 18 19 20 25 42 46 47 50 51 60 68 70 71 72 74 75
02      09/19/1989,01 05 09 10 13 18 20 21 27 30 32 37 38 39 42 51 55 62 76 79
03      09/18/1989,01 10 17 30 33 37 38 40 44 46 54 56 57 59 63 64 67 68 78 80
04      09/17/1989,07 08 16 20 22 23 26 28 30 31 38 42 47 52 60 62 71 73 76 77
05      09/16/1989,02 04 09 11 13 14 20 26 29 33 34 39 46 48 51 55 56 59 61 73
06      09/15/1989,01 02 19 20 22 30 31 33 45 46 47 49 52 57 60 61 62 65 67 77
07      09/14/1989,08 09 11 13 15 19 25 31 35 36 38 45 47 48 61 64 67 70 71 78
08      09/13/1989,01 02 06 10 16 19 24 25 28 29 30 32 35 36 41 46 50 54 63 65
09      09/12/1989,03 05 09 14 20 36 40 44 47 50 51 57 58 59 61 63 64 71 72 80
10      09/11/1989,01 06 07 15 16 17 21 29 30 35 42 45 55 63 64 66 67 68 78 80
11      09/10/1989,01 06 09 11 17 23 27 30 31 35 37 41 56 62 64 67 70 74 75 76
12      09/09/1989,05 07 08 09 10 13 25 29 34 37 40 41 44 51 57 62 68 69 75 79
13      09/08/1989,01 02 03 06 12 14 20 22 28 40 45 46 55 58 59 60 62 70 79 80
14      09/07/1989,03 04 05 11 17 19 27 28 30 32 33 39 43 45 63 67 68 71 75 79
15      09/06/1989,09 11 17 18 19 21 22 25 28 36 37 42 45 48 51 54 64 65 74 80
16      09/05/1989,01 02 07 17 18 21 23 27 29 30 35 41 44 45 46 54 64 71 72 77
```

```
17      09/04/1989,02 22 24 29 34 38 42 43 45 46 48 49 51 58 61 65 66 68 69 71
18      09/03/1989,07 09 14 15 17 18 19 23 34 37 40 43 54 62 66 68 72 73 77 80
19      09/02/1989,02 06 10 12 13 14 15 16 17 20 35 48 51 54 59 71 75 77 78 80
20      09/01/1989,01 03 14 22 26 29 30 33 35 37 38 43 45 53 56 60 62 64 68 75
21      08/31/1989,02 05 06 07 11 20 27 38 39 40 41 44 51 61 66 67 68 72 77 78
22      08/30/1989,02 05 06 07 14 16 19 25 30 33 49 52 54 57 59 63 64 72 76 78
23      08/29/1989,04 05 06 07 08 13 15 17 21 37 38 41 49 64 65 70 71 74 76 80
24      08/28/1989,03 10 18 24 25 26 27 29 32 37 39 43 51 56 58 59 64 65 75 79
25      08/27/1989,04 10 13 14 17 19 22 27 29 32 38 40 50 51 57 60 63 68 77 79
26      08/26/1989,08 26 27 32 35 36 38 40 46 48 55 56 59 63 65 66 67 68 71 79
27      08/25/1989,05 06 07 08 10 13 24 30 32 34 40 42 48 51 61 64 71 72 78 80
28      08/24/1989,02 03 11 12 15 16 17 23 24 31 42 53 54 57 58 61 68 69 77 78
29      08/23/1989,02 03 04 07 08 10 15 24 32 43 46 50 51 52 53 54 66 69 70 78
30      08/22/1989,06 08 14 16 19 28 29 34 35 37 41 47 51 58 62 65 67 68 71 75
31      08/21/1989,07 14 28 30 39 41 44 47 49 51 53 56 58 59 62 63 65 69 77 78
32      08/20/1989,05 06 12 15 16 18 21 23 37 38 39 40 46 58 62 70 71 73 75 76
33      08/19/1989,02 03 04 11 19 20 25 29 41 42 45 48 49 53 57 58 65 71 72 79
34      08/18/1989,02 09 10 11 12 15 19 20 24 26 28 37 47 56 58 59 64 74 75 78
35      08/17/1989,01 06 07 10 12 20 24 27 31 32 35 37 42 59 60 62 63 64 68 69
36      08/16/1989,05 10 11 21 25 27 30 33 35 36 40 43 44 46 57 60 64 72 73 76
37      08/15/1989,01 04 15 21 26 28 32 42 43 45 51 54 56 57 60 61 64 74 77 79
38      08/14/1989,04 05 11 12 18 22 24 27 40 43 44 47 57 59 61 63 68 70 78 80
39      08/13/1989,01 05 09 13 19 25 26 27 32 36 37 40 41 52 53 64 71 76 78 79
40      08/12/1989,02 03 05 12 16 17 22 32 39 42 51 58 59 61 65 68 72 73 76 80
41      08/11/1989,01 10 15 16 17 20 21 23 33 34 36 38 41 47 56 58 61 63 75 80
42      08/10/1989,01 03 04 09 11 12 23 28 37 40 41 44 51 54 55 56 65 66 68 77
43      08/09/1989,01 02 08 18 25 29 31 33 34 39 40 45 48 54 62 63 65 73 75 78
44      08/08/1989,06 10 11 15 17 22 32 41 42 43 44 56 58 61 71 72 73 75 77 80
45      08/07/1989,03 04 05 14 17 22 24 27 28 32 37 45 48 54 56 57 70 74 78 80
46      08/06/1989,06 14 15 16 22 23 36 38 42 44 51 52 56 57 60 65 72 73 74 79
47      08/05/1989,02 05 07 10 11 14 15 16 17 18 20 26 31 35 37 59 62 65 69 72
48      08/04/1989,02 06 23 27 28 31 32 33 34 37 41 42 43 47 53 56 57 60 67 79
49      08/03/1989,02 03 05 09 10 13 15 18 30 37 45 47 54 55 58 65 69 70 72 76
50      08/02/1989,02 03 05 08 12 17 19 27 38 45 49 57 59 62 63 66 69 70 71 80
51      08/01/1989,01 04 06 12 19 23 29 34 36 37 45 49 55 59 63 66 75 76 77 79
52      07/31/1989,06 17 18 19 22 34 35 37 39 41 50 54 63 65 66 72 75 78 79 80
53      07/30/1989,02 06 08 11 14 19 30 32 35 40 48 53 54 60 61 64 66 67 72 80
54      07/29/1989,01 06 12 14 15 24 30 35 38 46 47 50 54 61 62 66 67 69 73 77
55      07/28/1989,04 11 17 20 21 22 34 36 37 39 40 41 48 60 63 66 68 72 73 78
56      07/27/1989,01 03 04 13 15 23 24 28 29 34 36 38 43 47 60 61 62 67 72 77
57      07/26/1989,01 04 05 17 20 26 30 35 36 40 46 49 57 60 62 63 70 71 73 74
58      07/25/1989,01 04 07 10 16 25 34 41 42 43 44 53 56 65 66 67 73 76 78 79
59      07/24/1989,04 06 07 13 17 18 25 26 28 40 41 48 56 65 66 70 72 74 78 79
60      07/23/1989,01 04 16 19 24 26 28 32 37 41 53 54 56 59 61 63 64 71 76 80
61      07/22/1989,01 03 04 19 27 31 35 37 43 45 49 50 54 55 58 62 64 66 69 75
62      07/21/1989,01 03 11 14 16 19 22 26 31 41 45 54 55 59 63 73 75 77 78 79
63      07/20/1989,01 02 04 11 15 18 19 21 31 32 35 41 45 50 54 56 63 69 72 73
64      07/19/1989,01 03 05 07 08 10 14 21 31 32 35 39 43 47 56 59 62 64 65 69
65      07/18/1989,02 06 08 13 14 20 29 37 44 51 55 57 58 67 68 69 72 74 79 80
66      07/17/1989,07 09 11 15 16 18 20 24 25 26 27 29 31 32 56 61 62 68 71 72
67      07/16/1989,01 03 10 13 14 16 18 19 24 26 32 34 38 40 57 64 66 72 74 76
68      07/15/1989,02 09 10 14 15 20 26 30 32 38 49 52 53 57 60 61 62 67 68 71
69      07/14/1989,07 12 14 16 17 29 30 33 37 45 47 48 54 58 61 63 68 70 71 77
70      07/13/1989,01 02 03 07 11 13 17 21 26 28 30 33 42 45 55 64 65 75 78 79
```

```
71      07/12/1989,01 04 10 13 16 18 19 24 26 31 33 41 48 50 53 57 60 64 66 76
72      07/11/1989,02 03 04 09 22 25 26 27 29 36 37 43 48 49 50 53 56 63 65 68
73      07/10/1989,09 15 16 17 22 24 26 28 30 37 38 39 40 48 50 53 59 61 68 73
74      07/09/1989,01 04 05 11 14 22 24 25 26 27 35 42 43 44 47 49 55 68 73 78
75      07/08/1989,01 03 06 15 16 17 20 22 27 29 31 32 37 55 67 68 70 76 78 80
76      07/07/1989,01 04 09 16 20 22 26 33 46 52 54 56 61 63 64 66 71 73 74 76
77      07/06/1989,09 12 16 24 28 29 31 35 45 47 48 49 50 52 53 55 68 69 77 80
78      07/05/1989,02 07 11 12 13 14 16 18 19 22 32 37 39 44 46 47 52 53 54 59
79      07/04/1989,02 05 10 16 19 21 26 34 36 39 40 51 52 54 58 62 67 70 72 79
80      07/03/1989,04 06 10 13 15 22 24 25 32 35 43 44 45 46 47 48 53 55 66 67
81      07/02/1989,01 08 13 27 29 32 34 43 46 50 52 54 56 59 66 69 70 75 79 80
82      07/01/1989,07 08 12 13 14 16 19 21 26 37 41 47 48 53 55 58 63 67 73 78
83      06/30/1989,04 05 08 14 15 17 19 23 27 30 32 42 43 47 48 54 68 69 73 77
84      06/29/1989,17 18 20 35 36 38 40 42 43 44 48 53 62 64 66 67 68 75 77 80
85      06/28/1989,02 04 18 19 24 33 37 41 42 45 48 52 54 58 63 64 65 70 71 76
86      06/27/1989,01 08 09 13 17 18 24 27 28 31 32 37 38 51 53 54 64 71 72 77
87      06/26/1989,03 13 14 21 22 26 31 33 34 39 44 47 50 54 55 62 65 71 73 77
88      06/25/1989,13 26 28 29 32 33 34 37 38 43 44 47 51 59 65 67 72 73 74 76
89      06/24/1989,01 03 11 12 22 26 27 31 42 43 44 45 51 61 62 71 73 74 77 79
90      06/23/1989,03 05 06 07 09 16 28 38 39 40 41 51 56 59 61 65 67 69 70 72
91      06/22/1989,01 02 06 07 16 17 18 19 21 24 27 30 37 45 61 62 69 70 75 76
92      06/21/1989,06 25 26 27 35 36 40 43 44 46 53 54 58 61 63 70 71 76 78 79
93      06/20/1989,04 06 10 12 16 17 25 38 45 49 51 52 55 59 62 71 72 75 76 80
94      06/19/1989,05 12 14 18 21 29 30 31 32 45 53 56 63 68 71 73 74 76 78 79
95      06/18/1989,07 09 14 24 28 32 38 41 42 43 45 46 49 53 54 60 64 73 74 78
96      06/17/1989,01 07 10 14 15 20 21 22 31 36 47 51 57 59 60 64 66 70 72 75
97      06/16/1989,01 02 08 13 20 23 24 27 28 29 35 37 41 45 46 52 59 61 63 78
98      06/15/1989,03 05 07 09 17 24 29 32 42 51 55 58 59 60 61 62 64 67 76 80
99      06/14/1989,05 09 11 12 24 40 41 43 46 51 54 60 68 71 74 76 77 78 79 80
00      06/13/1989,02 03 04 05 07 09 10 12 13 25 36 49 51 52 64 66 72 78 79 80
01      06/12/1989,24 25 26 29 32 33 34 37 39 42 44 47 48 50 61 62 69 74 76 78
02      06/11/1989,03 14 19 33 40 43 45 46 49 50 52 57 58 61 68 70 75 76 79 80
03      06/10/1989,06 11 15 16 21 23 29 31 34 36 37 38 43 54 61 64 67 69 73 77
04      06/09/1989,01 03 04 09 16 25 27 30 32 33 35 36 42 43 47 57 59 60 75 76
05      06/08/1989,04 09 10 12 13 15 21 25 28 31 42 45 46 53 57 68 69 78 79 80
06      06/07/1989,01 02 03 09 10 12 16 20 23 30 34 36 39 42 54 67 68 73 76 78
07      06/06/1989,01 02 08 16 22 28 30 32 35 38 42 44 51 57 64 66 68 72 75 76
08      06/05/1989,01 02 04 18 21 27 33 35 37 38 39 41 43 46 49 54 66 67 73 78
09      06/04/1989,02 07 11 12 23 32 35 36 37 40 41 44 45 47 50 59 60 61 68 80
10      06/03/1989,06 11 12 18 21 22 32 37 38 41 42 44 51 57 58 64 65 70 71 72
11      06/02/1989,02 04 06 09 10 11 17 20 25 31 33 37 47 48 49 51 63 67 73 77
12      06/01/1989,02 06 10 14 21 23 29 31 34 36 37 39 45 52 54 60 61 66 69 74
13      05/31/1989,02 09 11 19 23 24 31 32 38 53 55 57 59 60 62 65 69 70 72 76
14      05/30/1989,01 02 03 04 06 12 14 21 30 31 32 35 40 41 43 50 68 69 78 80
15      05/29/1989,01 06 07 09 11 19 23 27 32 40 41 44 46 50 53 63 64 68 70 77
16      05/28/1989,08 12 17 18 23 25 27 29 32 39 42 53 55 59 64 67 71 74 76 79
17      05/27/1989,02 10 24 26 27 31 32 33 37 38 41 43 48 51 53 56 58 70 75 78
18      05/26/1989,02 04 08 14 23 26 38 43 44 46 47 49 52 54 63 66 68 69 75 79
19      05/25/1989,05 08 12 14 18 32 34 36 37 40 41 42 45 47 49 64 67 71 75 77
20      05/24/1989,01 02 05 06 09 11 14 16 17 25 32 33 38 52 53 61 64 69 76 80
21      05/23/1989,02 07 08 14 19 22 25 27 29 34 39 40 43 47 49 52 64 66 68 72
22      05/22/1989,03 11 14 15 21 23 24 25 28 29 37 40 45 46 61 62 63 66 68 80
23      05/21/1989,07 19 20 24 26 30 32 36 38 41 43 44 45 48 49 50 59 62 66 69
24      05/20/1989,18 20 21 24 30 36 40 41 44 45 46 49 56 58 64 65 67 71 73 76
```

```
05/19/1989,01 02 05 07 13 20 27 28 31 47 49 50 52 57 62 66 68 71 72 79
05/18/1989,03 05 09 10 11 13 15 19 40 43 47 51 59 63 69 72 75 76 77 79
05/17/1989,03 08 09 10 11 16 21 24 25 32 33 41 47 48 50 52 56 59 63 66
05/16/1989,03 07 10 20 23 24 33 36 49 56 57 59 65 68 70 74 75 76 77 78
05/15/1989,01 02 03 06 07 11 13 25 26 27 42 47 48 63 71 73 74 75 78 79
05/14/1989,01 08 10 14 18 20 28 30 36 38 39 41 43 45 50 51 65 67 68 80
05/13/1989,01 02 09 13 16 17 22 30 33 39 40 43 45 49 53 62 66 68 70 76
05/12/1989,05 07 13 14 18 22 24 31 32 33 35 38 44 49 50 53 59 61 65 79
05/11/1989,01 13 19 23 28 29 30 31 34 44 45 48 49 51 52 58 66 68 74 77
05/10/1989,02 23 31 33 35 36 45 50 51 52 57 60 63 65 69 71 73 77 79 80
05/09/1989,05 12 15 16 17 18 22 23 27 29 30 34 35 39 45 50 53 65 77 80
05/08/1989,02 11 16 18 24 34 35 38 41 45 48 49 54 61 63 65 71 75 76 78
05/07/1989,06 08 09 11 12 19 23 24 28 29 44 50 54 55 62 69 70 74 76 80
05/06/1989,02 08 12 17 29 34 38 41 46 48 49 50 58 59 60 65 66 73 78 79
05/05/1989,03 08 12 13 14 16 23 29 33 34 35 39 43 52 53 57 67 74 75 79
05/04/1989,06 16 20 22 30 35 37 42 45 47 51 52 53 55 57 62 63 66 72 74
05/03/1989,02 03 08 09 12 13 19 34 36 46 50 53 67 68 70 71 76 77 79 80
05/02/1989,01 03 10 12 14 16 22 24 27 38 41 43 50 57 62 63 64 69 72 76
05/01/1989,04 07 16 17 26 27 29 33 45 50 52 54 60 61 72 73 74 77 78 79
04/30/1989,01 02 03 10 17 18 23 29 31 32 38 49 52 53 60 62 63 67 71 80
04/29/1989,01 03 05 07 12 13 19 24 27 35 37 41 42 43 51 55 58 62 78 80
04/28/1989,09 10 13 14 16 18 20 21 30 32 37 40 42 43 54 61 63 70 72 76
04/27/1989,02 04 08 13 15 16 19 22 23 31 41 46 52 60 64 67 68 72 74 78
04/26/1989,06 07 13 15 18 20 21 24 33 41 47 48 49 55 58 62 64 66 74 79
04/25/1989,01 03 07 11 18 26 37 48 49 50 52 53 55 59 65 70 75 76 77 79
04/24/1989,03 06 09 16 17 18 20 35 39 41 42 45 49 51 53 57 63 64 75 77
04/23/1989,01 02 03 05 06 08 13 24 29 31 37 40 44 52 53 58 60 63 74 79
04/22/1989,04 06 08 10 13 14 15 29 36 37 40 46 49 56 59 62 71 73 77 80
04/21/1989,02 05 14 16 17 18 20 22 30 37 39 46 47 56 62 66 67 69 75 79
04/20/1989,04 05 10 12 18 23 24 26 27 31 32 34 36 46 50 52 59 60 73 78
04/19/1989,03 04 05 07 08 09 16 20 25 26 28 29 31 40 45 69 71 73 74 77
04/18/1989,07 17 20 25 26 28 31 38 44 45 51 56 58 60 62 65 66 67 76 80
04/17/1989,06 09 10 11 15 17 25 26 29 33 38 43 46 54 58 59 63 69 71 72
04/16/1989,01 03 08 14 18 22 25 27 29 33 38 43 46 50 53 59 67 70 71 75
04/15/1989,04 06 07 10 13 14 17 18 20 21 22 23 26 27 53 54 57 61 64 65
04/14/1989,03 06 16 17 18 20 24 27 30 43 44 46 52 54 56 57 59 61 62 71
04/13/1989,01 04 06 07 09 21 22 36 40 45 51 58 59 63 64 65 66 68 69 76
04/12/1989,05 11 13 14 17 20 21 32 34 35 39 45 47 54 57 58 59 63 75 77
04/11/1989,01 13 17 20 22 23 27 28 34 37 46 49 55 58 62 65 73 74 75 77
04/10/1989,09 16 17 21 25 28 31 34 35 45 54 59 61 63 65 71 72 75 78 80
04/09/1989,01 04 08 09 15 16 26 29 32 35 36 43 54 60 61 65 66 71 73 79
04/08/1989,02 09 21 23 24 25 38 39 41 43 50 53 54 55 60 66 68 74 76 78
04/07/1989,02 08 09 10 12 14 18 27 37 42 43 44 49 52 53 57 68 70 74 76
04/06/1989,15 17 22 30 31 32 33 36 40 42 50 53 55 56 61 71 72 74 75 77
04/05/1989,07 09 17 22 23 25 30 32 35 36 42 48 54 58 60 64 67 68 69 75
04/04/1989,02 07 08 11 13 18 21 23 31 33 36 37 41 44 46 47 48 57 58 78
04/03/1989,02 03 05 07 08 09 16 18 21 23 25 27 29 31 46 52 58 70 77 78
04/02/1989,08 21 26 28 32 33 34 35 38 45 46 47 49 50 53 54 68 74 77 79
04/01/1989,09 18 21 31 32 37 39 42 43 44 54 57 58 63 65 69 75 77 78 79
03/31/1989,07 10 15 20 30 31 36 40 44 46 47 58 59 60 61 62 72 77 79 80
03/30/1989,14 16 20 21 24 26 28 29 39 40 41 48 50 54 65 68 69 72 76 78
03/29/1989,03 12 13 14 28 36 39 48 55 57 59 64 65 68 70 72 73 75 78 79
03/28/1989,04 13 17 23 24 33 35 36 38 42 44 45 52 54 55 60 66 70 73 80
03/27/1989,04 10 15 16 17 23 24 26 31 39 44 53 54 58 61 67 70 73 76 77
```

```
79      03/26/1989,04 05 19 23 25 28 31 38 42 45 50 53 58 60 61 62 64 72 73 74
80      03/25/1989,04 11 15 16 18 35 39 41 44 52 54 55 57 58 62 66 68 71 72 73
81      03/24/1989,04 06 09 11 13 14 29 31 32 43 44 46 60 62 63 64 67 70 71 72
82      03/23/1989,07 09 12 20 25 26 34 38 40 41 42 43 50 53 56 61 68 70 76 79
83      03/22/1989,05 08 11 14 17 19 27 36 38 40 42 45 48 55 59 67 70 75 77 79
84      03/21/1989,05 06 11 15 20 28 31 40 43 44 46 56 61 64 65 69 71 74 75 78
85      03/20/1989,11 16 17 18 19 20 23 24 27 30 31 36 38 41 56 57 60 65 72 78
86      03/19/1989,01 03 06 10 14 19 24 26 27 28 33 41 44 55 62 63 65 69 73 78
87      03/18/1989,02 05 08 10 13 19 20 23 26 39 43 44 47 55 67 71 72 74 75 80
88      03/17/1989,05 08 09 15 20 21 34 36 42 43 50 51 53 56 58 64 68 72 73 78
89      03/16/1989,02 13 15 16 23 37 38 39 40 42 44 45 56 60 63 66 70 71 77 79
90      03/15/1989,02 06 09 16 18 24 25 28 30 32 40 41 42 44 52 54 58 61 68 71
91      03/14/1989,02 08 11 12 13 16 22 24 38 43 44 46 50 52 55 60 61 63 71 72
92      03/13/1989,03 04 11 15 18 25 31 32 35 37 39 46 49 51 53 61 62 76 77 79
93      03/12/1989,02 03 04 07 11 15 18 25 30 34 38 40 41 45 54 57 62 73 78 79
94      03/11/1989,14 15 16 18 27 28 32 41 48 49 51 55 56 62 63 65 68 78 79 80
95      03/10/1989,02 03 10 17 18 28 40 46 48 50 53 54 59 61 69 70 73 75 78 80
96      03/09/1989,06 09 11 16 17 20 22 25 33 34 38 39 44 51 56 60 61 65 73 76
97      03/08/1989,02 03 04 11 12 13 18 19 25 26 29 37 38 43 46 48 51 59 64 70
98      03/07/1989,02 03 07 14 18 22 27 40 47 48 50 51 57 59 62 66 67 68 77 80
99      03/06/1989,11 18 23 31 33 39 43 47 49 50 51 52 54 55 56 66 67 68 70 77
00      03/05/1989,04 05 11 12 13 15 16 17 29 32 41 42 49 59 63 64 68 72 73 77
01      03/04/1989,05 09 11 13 19 21 29 31 35 38 43 44 47 49 53 56 74 77 78 80
02      03/03/1989,07 09 10 19 23 27 35 37 38 40 41 44 49 52 54 57 60 63 66 72
03      03/02/1989,05 14 15 27 28 31 38 39 49 56 60 61 62 63 65 70 71 72 74 75
04      03/01/1989,03 05 15 18 22 23 24 31 36 38 40 42 43 45 50 52 59 62 76 77
05      02/28/1989,01 08 10 13 14 25 31 34 37 38 40 43 45 46 58 68 70 74 75 79
06      02/27/1989,02 03 06 08 09 13 14 21 25 28 30 34 38 40 45 55 60 72 73 77
07      02/26/1989,02 16 18 20 22 25 27 32 35 39 45 46 56 61 68 73 76 77 79 80
08      02/25/1989,02 06 12 20 27 28 29 31 36 41 44 47 52 53 55 61 63 66 69 73
09      02/24/1989,02 03 04 15 18 20 23 26 37 41 42 46 49 50 52 53 65 67 68 71
10      02/23/1989,01 04 09 10 17 22 23 24 28 31 34 38 39 50 53 60 70 74 76 79
11      02/22/1989,04 06 07 08 10 13 15 21 30 32 34 36 40 42 53 56 58 60 65 66
12      02/21/1989,04 06 07 10 17 18 22 30 37 39 50 53 60 62 64 65 72 73 74 78
13      02/20/1989,06 12 15 19 20 24 27 35 43 46 50 51 53 56 57 59 68 76 78 79
14      02/19/1989,02 03 11 13 14 18 20 21 32 34 39 40 46 49 53 57 62 68 69 75
15      02/18/1989,02 04 14 19 20 21 22 27 31 49 51 52 57 58 63 64 65 66 78 79
16      02/17/1989,02 06 08 17 22 28 31 34 36 37 39 41 44 45 48 60 66 68 69 73
17      02/16/1989,01 05 08 10 13 14 16 18 22 26 27 28 29 48 59 71 72 73 76 78
18      02/15/1989,01 05 08 13 17 29 34 39 42 43 44 46 47 52 59 60 64 68 74 77
19      02/14/1989,02 13 16 17 18 19 27 37 38 42 43 47 49 50 57 62 65 70 72 80
20      02/13/1989,03 05 06 14 16 17 37 46 53 56 61 64 67 68 70 71 72 73 77 79
21      02/12/1989,04 09 16 18 21 23 27 32 34 44 50 52 55 59 61 63 70 73 79 80
22      02/11/1989,03 04 09 12 17 19 20 22 29 31 41 42 46 52 53 57 61 62 68 78
23      02/10/1989,02 04 06 11 13 16 21 29 32 44 47 51 52 58 62 63 71 72 75 77
24      02/09/1989,06 07 10 12 14 15 23 25 28 34 43 46 47 49 53 56 59 64 71 74
25      02/08/1989,14 15 16 18 24 26 35 41 42 48 53 59 62 65 68 71 76 77 78 80
26      02/07/1989,02 07 08 10 12 22 24 26 29 32 36 37 40 41 42 43 44 49 50 59
27      02/06/1989,03 06 07 10 18 20 30 33 36 39 46 49 50 52 54 56 62 69 73 77
28      02/05/1989,06 09 10 13 16 20 22 23 30 34 37 38 47 52 56 57 59 62 70 73
29      02/04/1989,04 05 06 11 12 13 14 21 26 28 30 34 40 46 56 58 65 69 70 76
30      02/03/1989,02 09 15 18 20 21 23 24 26 30 31 36 39 41 45 55 57 73 74 78
31      02/02/1989,02 06 10 11 17 18 20 32 34 40 49 50 51 54 55 56 62 69 75 77
32      02/01/1989,03 04 05 10 11 12 13 14 23 30 31 33 40 46 49 53 62 64 68 75
```

```
01/31/1989,10 17 19 20 24 34 35 36 37 40 42 45 48 50 57 61 65 70 75 78
01/30/1989,03 05 06 07 08 09 11 21 25 30 32 33 34 41 47 48 61 66 68 73
01/29/1989,04 10 13 14 17 24 28 31 39 45 46 54 57 58 60 62 63 65 77 78
01/28/1989,02 05 06 07 13 15 17 22 27 28 30 43 48 49 61 62 64 67 68 72
01/27/1989,10 15 19 24 28 29 30 34 39 45 51 55 56 60 61 67 72 74 78 79
01/26/1989,15 18 20 21 26 29 32 37 39 42 46 57 58 61 62 67 72 73 75 76
01/25/1989,03 07 10 12 14 18 22 24 28 34 36 37 39 44 56 60 61 64 71 75
01/24/1989,02 03 10 15 17 21 26 31 38 42 44 49 50 51 59 67 68 70 74 77
01/23/1989,02 05 06 20 22 25 27 33 41 44 47 52 55 57 66 69 71 72 76 79
01/22/1989,02 03 05 07 08 18 20 21 24 40 42 45 53 57 62 63 68 74 78 80
01/21/1989,01 03 14 21 30 33 38 39 44 47 48 50 52 62 65 69 72 73 74 75
01/20/1989,03 05 07 08 09 13 17 22 26 30 34 39 41 45 47 53 57 61 67 68
01/19/1989,12 20 22 26 32 41 46 49 55 57 58 61 62 65 66 69 70 77 78 79
01/18/1989,04 19 24 29 30 31 32 36 38 41 42 44 51 55 56 62 63 71 73 77
01/17/1989,08 11 22 29 30 32 33 35 38 40 42 45 53 54 70 71 74 75 76 80
01/16/1989,04 09 11 19 23 24 29 30 31 33 36 46 49 66 67 71 73 77 79 80
01/15/1989,02 04 08 10 21 22 24 28 31 32 35 37 42 51 52 54 58 62 66 67
01/14/1989,04 06 08 09 15 17 21 25 35 36 43 46 47 50 52 57 63 65 73 75
01/13/1989,01 05 14 17 21 24 28 34 46 54 58 59 61 62 64 66 67 68 75 78
01/12/1989,01 04 08 11 23 25 27 32 35 37 38 39 47 52 55 57 61 66 78 80
01/11/1989,03 11 12 18 19 24 33 34 38 41 42 50 51 53 57 58 62 64 70 72
01/10/1989,08 12 13 19 20 21 22 25 26 38 46 49 58 59 60 63 66 72 73 75
01/09/1989,03 07 12 14 17 18 23 24 26 33 41 43 50 51 53 56 60 68 74 79
01/08/1989,10 23 25 27 32 33 35 36 37 39 41 47 53 62 63 65 72 73 78 79
01/07/1989,03 04 14 19 20 28 32 36 37 40 41 45 46 47 49 53 61 62 65 66
01/06/1989,03 19 20 21 29 31 34 38 46 50 51 52 58 60 61 66 68 69 78 80
01/05/1989,01 02 03 04 08 14 16 19 20 28 38 39 42 47 51 54 71 74 76 79
01/04/1989,10 14 17 18 20 21 29 33 42 44 45 46 47 61 63 65 66 75 78 80
01/03/1989,01 06 07 09 13 17 20 22 23 27 37 39 45 50 53 54 57 61 65 69
01/02/1989,01 07 10 13 22 23 25 31 32 47 49 50 57 58 66 67 73 74 75 77
01/01/1989,03 11 16 17 24 26 29 31 32 35 45 47 49 57 58 59 63 73 74 78
12/31/1988,06 09 16 17 18 24 29 36 37 45 47 55 57 60 64 67 72 74 76 80
12/30/1988,14 15 18 23 25 27 32 36 40 41 49 53 57 60 66 67 70 75 79 80
12/29/1988,06 16 18 19 26 29 37 38 41 53 61 64 65 66 69 73 74 75 76 77
12/28/1988,03 08 10 14 20 24 25 27 29 30 33 35 37 40 41 43 48 53 57 68
12/27/1988,04 08 21 24 33 34 36 37 46 47 49 53 58 61 64 65 66 69 71 78
12/26/1988,02 09 12 18 37 38 39 42 46 47 49 53 58 62 67 72 75 77 78 79
12/24/1988,02 06 08 11 12 17 22 30 37 44 45 47 49 51 58 61 63 66 74 76
12/23/1988,01 03 06 11 13 24 25 30 32 34 35 43 45 52 56 62 64 65 69 80
12/22/1988,12 17 18 19 22 26 27 30 34 38 40 46 48 49 58 71 73 76 78 79
12/21/1988,04 05 07 08 09 19 27 34 37 41 44 51 53 64 68 73 76 77 78 79
12/20/1988,03 04 06 07 11 12 15 19 21 28 30 37 46 47 51 54 61 62 64 76
12/19/1988,01 07 12 14 18 22 26 30 37 47 49 55 58 59 60 61 68 69 72 79
12/18/1988,01 04 09 13 16 17 21 36 37 39 40 47 49 50 54 56 73 75 77 78
12/17/1988,03 05 13 23 28 35 37 38 43 47 49 53 55 56 58 61 63 64 65 71
12/16/1988,02 04 08 10 16 20 27 42 44 46 51 55 59 61 64 70 72 73 78 79
12/15/1988,04 05 07 09 12 14 16 20 25 30 32 36 37 42 50 53 55 68 74 78
12/14/1988,02 11 17 23 28 32 33 34 36 37 39 41 42 47 62 69 72 74 78 80
12/13/1988,09 18 19 21 22 24 27 28 29 39 40 41 45 61 64 71 72 75 76 79
12/12/1988,01 03 06 14 18 20 25 28 29 41 50 51 53 55 57 61 64 71 72 80
12/11/1988,04 12 13 18 20 22 24 26 29 30 32 38 42 45 47 49 52 53 55 64
12/10/1988,14 16 22 27 28 31 37 38 46 50 51 52 62 63 68 74 76 77 78 79
12/09/1988,01 04 05 07 08 10 13 14 15 20 26 27 37 41 48 53 55 71 73 79
12/08/1988,01 10 11 16 19 22 26 27 39 45 47 53 55 56 60 65 67 68 77 80
```

```
87      12/07/1988,04 10 12 14 19 20 22 23 29 33 40 43 46 50 52 59 65 66 68 75
88      12/06/1988,08 11 13 14 16 25 27 34 35 37 39 40 50 53 58 60 70 72 75 76
89      12/05/1988,02 09 22 24 26 29 34 37 41 42 52 53 58 59 62 63 69 71 77 80
90      12/04/1988,02 09 14 16 17 23 24 29 32 34 39 47 48 50 51 53 58 70 71 74
91      12/03/1988,08 10 11 21 22 24 26 35 40 45 46 48 55 56 61 65 66 71 77 80
92      12/02/1988,05 07 08 12 14 19 23 25 27 29 33 35 40 52 53 60 64 70 74 78
93      12/01/1988,01 08 19 23 25 27 28 29 30 36 38 46 49 50 52 64 71 73 77 80
94      11/30/1988,04 10 14 15 16 22 27 31 37 41 46 53 54 58 63 67 69 72 77 78
95      11/29/1988,07 09 15 19 21 22 26 30 33 43 50 53 54 57 61 62 63 64 72 78
96      11/28/1988,02 03 08 13 14 22 24 33 39 44 50 54 56 58 59 61 63 65 79 80
97      11/27/1988,06 08 11 20 21 23 25 28 31 36 41 45 47 52 56 64 68 70 72 78
98      11/26/1988,12 19 20 21 22 30 33 37 42 43 48 50 55 59 60 63 65 69 70 80
99      11/25/1988,01 05 19 22 23 27 34 35 40 42 43 44 47 48 49 67 68 69 71 80
00      11/24/1988,13 18 20 23 25 28 30 33 34 47 53 54 59 62 65 66 71 73 74 75
01      11/23/1988,02 04 05 13 21 25 26 27 35 37 38 50 55 57 58 63 71 72 75 76
02      11/22/1988,04 08 11 15 30 41 42 44 48 49 50 52 57 61 62 66 67 68 73 79
03      11/21/1988,04 08 13 14 15 16 19 37 38 42 44 49 53 57 63 65 72 73 75 76
04      11/20/1988,01 02 17 28 30 31 34 38 43 47 50 54 58 62 63 64 73 76 78 79
05      11/19/1988,03 08 15 19 21 27 35 38 43 45 49 52 53 55 56 59 61 66 69 79
06      11/18/1988,05 11 14 21 37 38 39 46 50 51 54 55 57 60 63 64 65 70 71 73
07      11/17/1988,01 10 16 21 24 29 37 42 43 49 52 58 61 62 68 69 71 72 73 74
08      11/16/1988,01 06 19 21 29 31 32 37 46 53 57 62 64 67 69 70 73 74 77 80
09      11/15/1988,05 06 14 17 21 22 23 30 36 46 47 54 57 66 67 72 75 77 79 80
10      11/14/1988,10 12 14 26 27 29 33 42 43 44 51 54 57 59 60 63 64 69 74 76
11      11/13/1988,04 05 06 11 13 19 24 25 28 29 33 50 58 60 63 68 71 72 76 79
12      11/12/1988,01 07 10 19 25 28 32 33 37 39 41 45 46 53 63 64 68 71 73 80
13      11/11/1988,01 04 05 06 07 10 16 17 18 23 32 38 44 45 64 67 69 73 76 79
14      11/10/1988,01 05 09 10 11 12 23 24 29 32 37 48 50 53 56 58 60 67 71 77
15      11/09/1988,01 14 15 16 19 22 24 25 31 40 43 47 57 64 65 67 69 70 77 80
16      11/08/1988,06 13 21 22 25 26 33 36 38 40 41 43 44 49 52 59 61 64 72 76
17      11/07/1988,18 21 25 33 34 38 41 44 45 49 54 56 62 68 72 74 75 76 77 80
18      11/06/1988,04 05 08 17 19 21 24 37 41 43 44 46 55 56 62 64 65 67 74 79
19      11/05/1988,05 06 13 15 26 27 28 33 34 36 39 45 46 47 51 54 59 62 66 74
20      11/04/1988,02 04 06 07 14 16 22 24 29 33 46 47 53 55 61 65 70 77 79 80
21      11/03/1988,05 06 07 10 15 20 23 27 28 36 38 45 59 66 67 69 74 75 77 78
22      11/02/1988,03 05 06 10 11 12 22 26 31 37 38 51 53 59 63 70 71 72 77 78
23      11/01/1988,03 10 12 13 16 17 18 19 25 29 31 32 41 44 66 68 71 73 74 76
24      10/31/1988,03 09 14 18 22 27 31 32 33 34 41 46 48 56 58 59 63 66 76 77
25      10/30/1988,03 04 09 10 13 16 23 27 35 39 41 43 47 49 52 55 68 70 76 77
26      10/29/1988,03 15 17 22 28 29 35 36 37 39 43 44 47 57 66 68 69 76 77 79
27      10/28/1988,04 08 09 13 14 16 19 23 25 31 39 57 58 60 66 67 68 70 74 76
28      10/27/1988,09 10 16 20 21 23 25 28 38 42 43 47 49 51 52 53 56 64 75 78
29      10/26/1988,03 09 16 18 23 24 27 31 32 36 38 39 42 48 53 54 56 60 63 68
30      10/25/1988,07 10 12 13 15 22 26 28 33 41 43 49 53 55 56 62 65 71 77 79
31      10/24/1988,02 08 15 21 24 25 26 28 35 39 40 41 43 44 46 48 57 64 77 78
32      10/23/1988,10 11 12 16 17 24 25 26 37 39 42 45 46 48 52 59 63 68 71 73
33      10/22/1988,01 10 13 17 19 21 22 23 25 27 29 30 41 48 50 64 66 68 71 77
34      10/21/1988,01 06 15 16 21 22 26 27 31 32 38 44 47 54 55 58 70 76 79 80
35      10/20/1988,03 04 05 09 14 15 16 18 21 25 27 30 43 45 48 49 59 65 70 71
36      10/19/1988,01 03 07 09 11 13 22 30 31 32 43 45 54 55 58 70 71 73 74 80
37      10/18/1988,13 14 26 30 32 35 41 42 44 50 52 53 59 60 62 63 64 71 75 80
38      10/17/1988,01 04 05 09 14 16 18 21 25 29 31 36 56 57 58 64 69 76 77 79
39      10/16/1988,02 08 09 10 17 22 26 28 30 38 44 50 54 57 58 64 66 69 72 77
40      10/15/1988,06 08 11 12 14 16 22 27 28 32 37 45 48 53 61 62 65 67 70 76
```

```
10/14/1988,02 06 08 15 18 20 21 24 27 37 40 42 43 44 45 61 64 71 74 78
10/13/1988,01 02 05 07 09 11 15 18 23 24 25 32 38 48 54 63 68 74 76 77
10/12/1988,07 08 11 19 27 31 32 33 36 37 39 43 49 65 66 68 69 72 74 78
10/11/1988,01 02 05 07 10 14 18 28 31 34 45 47 49 64 65 67 68 75 77 79
10/10/1988,01 03 06 09 12 15 24 28 36 38 46 48 51 52 63 64 69 73 76 78
10/09/1988,01 03 14 19 22 23 27 29 31 32 33 39 41 47 56 57 60 66 67 77
10/08/1988,01 02 04 06 07 13 16 17 28 33 40 41 54 57 58 59 66 70 71 73
10/07/1988,06 07 18 23 24 31 33 35 42 44 51 57 58 61 72 74 75 77 78 80
10/06/1988,02 08 15 21 34 37 38 40 42 48 50 52 53 54 58 61 63 64 72 77
10/05/1988,01 03 10 13 14 17 21 24 25 26 27 29 33 35 45 59 62 63 65 73
10/04/1988,01 04 10 18 19 24 28 29 33 38 40 44 47 51 56 58 62 65 69 79
10/03/1988,02 04 05 07 08 12 15 16 19 22 24 26 41 42 50 55 58 59 69 71
10/02/1988,01 05 12 15 17 30 36 37 43 44 45 48 55 57 58 64 66 69 71 80
10/01/1988,07 08 11 12 14 25 26 32 40 43 44 47 48 50 56 59 62 68 72 76
09/30/1988,09 11 12 14 15 16 17 34 39 41 42 45 55 56 62 64 65 67 72 80
09/29/1988,16 17 24 25 27 28 34 37 43 45 48 54 56 62 64 69 72 75 78 80
09/28/1988,16 20 21 27 32 33 43 44 59 62 63 64 66 67 69 71 74 77 78 79
09/27/1988,09 13 15 18 26 29 30 34 36 39 40 41 44 51 54 55 56 69 76 80
09/26/1988,05 25 28 40 43 45 47 51 52 53 54 56 62 64 66 72 73 74 76 80
09/25/1988,08 12 13 15 17 19 29 31 35 40 41 45 46 49 55 62 67 72 73 76
09/24/1988,11 16 19 22 24 25 28 32 39 42 45 50 51 53 56 68 73 74 76 77
09/23/1988,08 10 12 13 21 22 23 24 25 29 31 34 39 40 43 45 59 61 68 74
09/22/1988,02 06 08 09 13 18 20 29 30 31 34 36 39 48 55 57 60 61 64 68
09/21/1988,07 09 15 18 19 22 23 25 29 31 33 37 38 41 46 52 54 67 70 77
09/20/1988,02 05 06 24 27 30 31 37 39 41 45 48 53 55 61 64 69 71 77 78
09/19/1988,07 10 14 15 16 20 27 32 33 35 37 44 46 48 49 58 70 71 75 77
09/18/1988,13 19 25 26 28 40 47 49 50 52 54 56 58 59 62 65 68 71 72 78
09/17/1988,04 08 24 27 28 36 37 38 41 44 47 48 54 59 60 61 62 63 67 71
09/16/1988,02 10 14 17 18 23 25 34 35 41 47 53 56 61 62 63 66 67 69 80
09/15/1988,03 06 07 13 18 23 31 35 36 40 44 47 55 61 62 64 66 76 78 79
09/14/1988,01 10 11 12 21 25 27 28 34 37 39 54 55 61 66 67 70 72 78 79
09/13/1988,03 05 11 15 17 29 31 33 45 47 48 49 51 54 67 69 74 76 77 78
09/12/1988,09 13 15 26 31 34 39 53 56 60 61 62 67 68 69 72 74 76 79 80
09/11/1988,09 10 14 17 19 23 28 31 36 37 44 47 52 56 64 71 74 76 77 80
09/10/1988,07 09 10 15 24 25 27 39 42 46 47 55 64 67 68 71 74 77 79 80
09/09/1988,02 10 19 21 23 26 32 38 39 42 43 51 55 57 69 70 71 73 74 80
09/08/1988,03 04 05 18 22 23 28 31 35 37 42 51 56 61 62 64 70 71 73 80
09/07/1988,02 04 06 11 16 25 30 32 35 37 43 49 53 58 61 62 63 69 70 77
09/06/1988,03 04 10 16 19 21 29 31 34 36 38 47 48 51 52 61 69 72 76 78
09/05/1988,02 03 08 13 19 20 23 25 38 42 47 48 55 60 63 64 67 74 75 80
09/04/1988,01 07 08 12 16 21 23 34 39 40 45 49 53 62 67 73 75 76 77 79
09/03/1988,02 03 04 07 09 15 18 21 25 26 37 39 47 49 50 51 57 59 73 80
09/02/1988,03 04 23 24 25 33 34 38 42 44 45 46 48 51 57 58 59 63 70 76
09/01/1988,01 02 04 05 08 11 13 28 38 39 40 42 43 48 53 57 59 64 69 74
08/31/1988,01 03 07 14 19 22 25 26 28 39 47 52 55 57 60 64 71 73 74 79
08/30/1988,01 06 08 09 13 14 21 26 31 39 44 59 63 64 65 68 73 75 76 80
08/29/1988,03 04 10 16 18 23 25 27 34 36 38 39 40 47 48 53 55 59 71 79
08/28/1988,01 09 14 19 27 28 37 47 49 53 54 55 61 66 67 69 73 75 77 80
08/27/1988,08 10 15 18 22 26 33 35 36 37 40 46 53 55 57 61 67 71 78 80
08/26/1988,04 06 08 12 15 21 23 24 25 32 39 40 41 43 52 54 56 66 67 71
08/25/1988,02 04 08 09 18 25 28 34 41 48 54 55 60 62 63 67 68 69 71 73
08/24/1988,05 07 24 27 31 32 34 36 37 48 50 52 53 54 55 59 60 61 64 77
08/23/1988,01 05 11 22 28 30 35 37 39 44 46 48 51 57 58 69 75 77 78 80
08/22/1988,03 04 09 10 11 19 21 27 29 30 34 50 51 56 61 68 73 74 77 80
```

95 08/21/1988,03 09 16 19 22 25 26 28 29 43 53 57 59 60 64 66 68 69 72 79
96 08/20/1988,05 06 10 13 14 24 25 28 29 33 36 41 47 48 54 55 58 61 63 69
97 08/19/1988,04 06 08 16 17 19 26 29 39 40 42 44 46 48 49 51 52 54 62 68
98 08/18/1988,04 13 14 22 28 30 36 37 39 40 43 44 47 51 54 57 64 68 70 76
99 08/17/1988,10 12 15 21 22 24 27 28 30 33 34 36 39 45 53 54 57 69 72 74
00 08/16/1988,02 04 13 15 16 29 33 36 43 44 50 51 52 54 56 61 65 72 76 79
01 08/15/1988,01 07 16 17 19 30 40 46 47 48 49 52 53 60 61 64 71 72 76 77
02 08/14/1988,01 03 10 14 19 20 22 30 32 34 42 43 48 51 56 58 61 65 67 75
03 08/13/1988,02 04 06 13 17 18 19 21 27 44 46 48 53 55 61 65 68 71 72 78
04 08/12/1988,02 06 10 20 21 24 31 32 44 52 57 58 59 67 68 70 72 75 79 80
05 08/11/1988,04 05 11 20 25 28 33 37 43 46 48 53 56 59 65 66 67 74 77 80
06 08/10/1988,02 04 05 06 09 13 17 18 24 25 31 32 34 35 38 44 63 68 72 77
07 08/09/1988,02 11 15 16 19 22 24 25 28 35 48 52 53 56 57 59 62 65 74 77
08 08/08/1988,02 05 06 07 08 09 11 18 24 29 32 33 37 46 47 49 52 67 75 80
09 08/07/1988,01 04 07 21 23 27 30 43 45 47 50 56 59 66 67 70 71 73 74 80
10 08/06/1988,09 11 17 20 23 27 30 31 37 42 43 47 51 54 58 59 71 73 79 80
11 08/05/1988,01 04 06 10 14 21 22 27 30 33 40 43 47 50 53 54 55 57 58 80
12 08/04/1988,01 08 11 18 20 23 24 25 32 36 37 39 40 41 42 46 69 71 72 79
13 08/03/1988,08 12 13 16 18 23 25 28 30 36 39 40 44 50 61 64 69 72 75 80
14 08/02/1988,04 07 09 13 15 17 18 25 40 42 45 46 49 50 55 59 66 69 70 75
15 08/01/1988,02 06 08 09 10 16 27 28 30 33 34 36 41 47 51 58 70 73 75 79
16 07/31/1988,07 10 16 22 23 26 27 28 30 32 33 43 44 48 54 58 68 75 79 80
17 07/30/1988,01 03 07 16 20 22 26 32 41 42 44 46 47 54 55 56 59 62 65 78
18 07/29/1988,02 09 10 11 14 15 20 34 36 37 41 42 43 47 58 62 67 69 76 78
19 07/28/1988,01 07 08 09 12 14 16 45 46 47 49 52 54 55 59 62 64 68 69 73
20 07/27/1988,01 11 17 18 19 31 34 35 36 37 38 43 44 50 58 60 67 69 70 78
21 07/26/1988,03 15 24 30 31 32 33 35 37 41 48 52 53 56 58 71 72 74 77 80
22 07/25/1988,01 04 09 13 14 16 22 23 25 33 35 41 45 48 50 59 65 69 71 75
23 07/24/1988,01 02 06 07 09 10 11 18 20 21 23 28 43 48 55 56 64 67 74 76
24 07/23/1988,07 09 10 19 22 32 33 35 36 39 43 54 58 59 60 71 74 75 77 80
25 07/22/1988,06 07 11 15 16 21 24 31 35 38 39 51 53 56 63 64 65 66 72 79
26 07/21/1988,03 09 12 16 17 34 36 38 42 44 45 47 50 52 55 57 58 70 74 80
27 07/20/1988,03 04 12 17 20 22 23 29 35 36 43 46 47 49 57 58 60 74 76 79
28 07/19/1988,05 09 12 14 17 19 23 26 39 45 50 53 56 57 58 63 68 69 71 75
29 07/18/1988,06 08 12 18 26 30 32 33 40 41 43 44 52 53 55 58 59 66 68 79
30 07/17/1988,01 03 04 14 26 33 37 38 41 42 45 46 48 49 65 66 68 72 74 76
31 07/16/1988,03 04 12 17 28 30 32 36 37 38 42 47 49 50 57 67 69 74 78 80
32 07/15/1988,05 07 10 15 27 35 36 37 39 40 47 53 58 62 66 70 72 74 79 80
33 07/14/1988,07 12 17 21 22 23 28 29 33 35 37 42 45 50 51 61 63 69 71 80
34 07/13/1988,04 11 12 20 26 27 34 37 38 39 40 44 46 51 61 65 69 70 73 78
35 07/12/1988,01 03 04 08 15 19 22 23 25 46 51 52 56 63 64 65 72 74 75 80
36 07/11/1988,05 07 13 18 20 28 29 31 45 46 48 57 60 64 69 71 74 76 77 78
37 07/10/1988,08 11 13 15 16 22 31 32 37 40 41 44 45 49 50 58 61 65 71 75
38 07/09/1988,04 08 14 16 19 20 22 27 32 33 38 42 49 52 54 55 61 72 76 79
39 07/08/1988,01 02 16 17 20 21 23 27 35 37 41 46 48 49 55 57 69 73 77 79
40 07/07/1988,03 04 05 06 07 16 21 26 27 28 31 35 36 40 44 48 53 67 74 78
41 07/06/1988,03 08 13 15 16 17 22 41 44 45 49 50 51 53 55 57 60 68 77 78
42 07/05/1988,02 03 09 11 13 17 18 25 29 39 42 43 45 49 54 60 65 66 67 78
43 07/04/1988,02 04 08 09 11 14 20 21 24 26 30 36 44 49 56 60 67 68 74 80
44 07/03/1988,01 02 06 09 14 16 18 25 29 34 35 37 38 51 55 56 68 71 73 79
45 07/02/1988,05 06 15 22 29 31 38 40 45 48 49 56 58 63 67 69 71 77 78 79
46 07/01/1988,01 04 06 07 15 16 30 38 42 48 51 52 54 55 59 64 67 71 75 78
47 06/30/1988,03 06 07 09 16 17 19 26 30 33 47 48 52 62 63 66 68 75 78 79
48 06/29/1988,01 02 04 09 19 23 24 27 33 34 35 38 48 57 59 64 65 71 77 78

```
49      06/28/1988,06 08 22 23 24 25 30 39 40 41 43 52 56 65 66 71 75 76 77 79
50      06/27/1988,02 07 12 16 17 20 23 25 28 30 32 35 37 38 48 49 65 72 75 78
51      06/26/1988,03 04 08 12 17 22 24 26 29 40 42 46 47 60 65 66 69 72 75 78
52      06/25/1988,04 09 14 18 22 26 32 40 42 47 48 51 52 54 55 63 65 67 68 70
53      06/24/1988,07 08 13 19 21 27 29 30 36 42 46 49 50 51 55 59 64 66 74 79
54      06/23/1988,02 25 28 29 36 37 39 40 43 55 56 57 63 64 65 68 70 76 77 78
55      06/22/1988,01 02 10 11 12 14 29 30 31 39 43 51 54 60 64 65 70 73 77 78
56      06/21/1988,05 07 10 11 13 28 30 33 37 38 42 43 60 61 69 72 73 75 78 79
57      06/20/1988,01 02 03 04 06 08 20 21 23 24 28 30 36 49 51 54 57 65 74 77
58      06/19/1988,07 11 14 17 19 21 24 29 34 39 41 48 56 60 63 68 69 73 75 76
59      06/18/1988,02 06 11 12 13 19 27 28 31 35 36 39 41 48 50 51 65 70 71 78
60      06/17/1988,05 17 19 22 25 27 29 43 47 54 55 56 62 65 66 68 73 74 75 76
61      06/16/1988,01 02 05 06 15 20 25 26 46 48 49 51 52 53 54 64 65 66 71 80
62      06/15/1988,09 11 14 16 21 24 26 30 37 39 42 43 47 48 55 58 60 73 74 76
63      06/14/1988,08 12 14 15 18 21 29 38 39 40 41 43 44 49 53 60 63 66 75 77
64      06/13/1988,01 04 06 10 12 14 19 23 29 39 51 53 54 60 63 65 68 72 76 79
65      06/12/1988,07 17 19 20 25 35 40 41 47 49 52 53 54 55 56 57 61 76 77 78
66      06/11/1988,01 03 08 09 16 27 28 41 43 46 48 53 58 60 66 71 73 74 77 80
67      06/10/1988,02 04 09 15 16 21 22 25 26 28 30 34 45 48 50 59 64 65 68 80
68      06/09/1988,05 06 10 16 18 20 21 28 29 31 35 36 41 43 47 55 60 62 70 76
69      06/08/1988,10 12 19 30 32 34 38 43 45 48 50 51 54 57 58 59 60 62 69 75
70      06/07/1988,05 07 08 15 16 21 31 36 37 44 45 48 49 51 52 58 63 78 79 80
71      06/06/1988,04 05 06 09 13 14 17 20 28 44 46 48 51 52 58 59 62 67 69 78
72      06/05/1988,11 16 18 21 26 28 29 34 37 40 48 51 52 56 62 65 71 73 77 79
73      06/04/1988,03 04 10 12 13 15 20 24 29 30 39 42 47 56 60 64 68 77 78 79
74      06/03/1988,04 05 08 12 16 21 22 26 29 41 49 51 54 55 61 66 68 71 78 80
75      06/02/1988,03 07 15 16 17 27 32 33 34 36 37 46 48 51 53 54 62 63 73 74
76      06/01/1988,05 07 11 29 32 36 44 45 46 52 54 55 56 58 60 72 74 77 78 79
77      05/31/1988,06 16 18 23 24 29 32 34 35 39 46 49 50 57 58 59 60 61 69 76
78      05/30/1988,05 09 13 18 23 24 32 36 38 40 48 49 50 52 57 67 71 73 77 78
79      05/29/1988,02 03 05 23 29 32 39 41 45 47 49 50 51 52 56 58 62 69 70 73
80      05/28/1988,03 05 06 09 17 21 22 23 29 30 43 44 48 52 56 58 65 72 75 77
81      05/27/1988,01 10 11 15 29 36 37 47 51 61 62 66 67 68 69 71 74 78 79 80
82      05/26/1988,12 14 20 32 33 35 36 37 43 50 54 55 56 61 68 70 74 75 78 79
83      05/25/1988,02 05 07 09 10 11 21 25 44 50 51 53 56 59 60 66 71 72 73 80
84      05/24/1988,07 08 12 19 21 22 24 26 29 35 38 39 44 45 46 48 50 53 56 77
85      05/23/1988,02 03 04 05 08 09 11 16 19 20 23 25 33 34 56 64 68 70 72 79
86      05/22/1988,09 14 19 22 29 30 33 35 40 41 44 49 50 56 58 59 64 66 67 79
87      05/21/1988,02 05 09 17 18 23 25 32 35 37 40 41 43 51 52 57 59 60 78 79
88      05/20/1988,02 03 06 08 09 13 24 30 36 37 42 44 47 51 52 64 72 74 76 80
89      05/19/1988,05 08 23 25 26 30 33 36 37 41 43 46 47 49 51 60 63 66 68 80
90      05/18/1988,03 10 12 15 18 21 22 23 24 31 34 40 45 50 56 60 64 68 75 76
91      05/17/1988,02 05 09 11 28 29 31 33 40 48 50 54 55 63 65 68 73 77 78 80
92      05/16/1988,01 09 15 16 22 24 25 28 31 33 37 44 45 49 53 54 56 58 63 74
93      05/15/1988,02 05 06 07 14 16 17 20 21 24 25 30 35 37 44 62 64 66 67 68
94      05/14/1988,01 02 04 11 15 24 34 36 39 46 49 54 57 59 62 68 72 78 79 80
95      05/13/1988,04 08 09 11 24 27 28 30 40 42 46 47 54 56 67 68 74 75 77 79
96      05/12/1988,01 02 07 08 17 23 26 30 33 36 39 50 51 55 57 60 61 62 70 78
97      05/11/1988,01 03 05 11 12 13 17 23 24 27 28 34 36 46 51 55 56 70 71 75
98      05/10/1988,03 04 12 13 25 26 28 29 41 45 46 51 52 54 63 64 73 75 78 79
99      05/09/1988,01 03 13 14 19 24 27 29 34 35 36 41 42 46 52 59 62 64 74 80
00      05/08/1988,02 15 18 21 24 28 29 33 37 46 49 53 55 56 62 66 68 69 76 78
01      05/07/1988,04 06 07 18 19 20 22 24 31 32 42 44 46 54 55 61 66 74 77 80
02      05/06/1988,01 02 05 07 10 18 24 28 29 37 41 42 48 51 57 58 61 66 75 80
```

```
03      05/05/1988,05 09 13 26 33 37 38 39 41 44 45 48 50 51 53 55 57 61 64 75
04      05/04/1988,04 08 10 11 14 22 23 28 29 38 39 48 50 51 56 57 58 63 66 72
05      05/03/1988,02 04 09 20 25 35 36 38 41 43 46 47 54 59 61 72 77 78 79 80
06      05/02/1988,01 10 11 15 16 30 35 37 38 40 42 45 50 52 58 66 75 76 79 80
07      05/01/1988,02 04 05 06 09 10 15 21 28 30 34 43 54 55 56 65 66 75 77 79
08      04/30/1988,04 11 19 25 27 33 34 42 48 50 52 56 57 59 65 66 67 69 75 76
09      04/29/1988,02 04 14 16 18 21 31 33 38 41 42 43 44 53 57 58 62 65 71 78
10      04/28/1988,02 03 09 11 25 26 28 29 30 35 40 51 55 58 62 63 71 72 75 80
11      04/27/1988,07 08 12 13 16 20 21 22 23 24 33 38 47 48 49 52 58 66 70 73
12      04/26/1988,02 09 11 17 24 25 27 29 33 36 41 45 46 48 49 50 58 78 79 80
13      04/25/1988,03 10 12 26 28 30 32 33 35 40 47 49 50 51 53 55 59 62 66 80
14      04/24/1988,02 03 06 07 08 18 23 26 27 30 39 44 51 57 66 69 75 76 78 80
15      04/23/1988,02 09 11 15 19 20 21 22 25 26 33 39 42 45 50 55 68 69 75 80
16      04/22/1988,01 02 03 04 05 08 10 11 17 19 20 21 36 37 41 49 51 56 59 78
17      04/21/1988,01 13 17 18 21 23 25 27 31 37 40 53 54 56 61 65 68 74 77 79
18      04/20/1988,04 12 13 17 20 25 26 27 35 38 45 48 49 54 59 61 65 70 76 77
19      04/19/1988,09 11 12 13 14 15 19 27 28 29 32 35 37 47 54 56 68 70 78 79
20      04/18/1988,01 04 08 13 14 22 37 48 49 54 56 62 63 65 67 68 69 70 71 76
21      04/17/1988,10 17 18 21 23 30 31 33 39 40 43 45 47 48 51 58 65 72 79 80
22      04/16/1988,11 13 15 22 29 31 32 35 37 38 42 45 46 50 56 57 58 67 70 72
23      04/15/1988,02 04 09 20 21 23 24 28 30 37 38 40 50 57 58 63 65 68 70 73
24      04/14/1988,02 04 05 07 08 10 16 23 24 31 33 35 38 40 43 53 61 65 69 79
25      04/13/1988,03 05 08 13 14 19 20 23 26 32 39 42 50 51 52 54 57 62 66 78
26      04/12/1988,02 11 14 15 24 25 26 28 41 42 44 45 52 53 59 60 64 69 70 78
27      04/11/1988,02 04 10 11 15 16 17 21 25 28 32 38 45 47 49 53 55 58 62 73
28      04/10/1988,12 19 20 25 27 33 35 45 50 53 54 56 57 58 61 64 69 71 74 76
29      04/09/1988,02 06 16 17 22 31 35 37 48 49 53 55 57 64 68 70 73 74 78 80
30      04/08/1988,01 06 07 12 16 22 31 35 37 40 45 48 54 59 67 71 72 75 77 80
31      04/07/1988,01 02 10 12 14 21 22 23 32 41 43 45 47 48 51 64 68 69 73 78
32      04/06/1988,03 05 11 16 22 23 27 32 37 39 42 44 47 50 51 56 61 64 76 79
33      04/05/1988,05 09 12 13 17 28 36 39 43 47 49 50 56 58 61 62 66 72 75 80
34      04/04/1988,01 05 08 19 21 25 29 30 31 35 37 43 51 53 58 61 66 68 78 80
35      04/03/1988,01 04 11 12 15 18 19 23 25 31 34 41 44 47 51 53 58 63 70 72
36      04/02/1988,05 12 13 14 15 17 18 19 24 29 30 31 35 37 48 52 56 58 72 74
37      04/01/1988,01 02 06 09 26 33 34 37 46 48 50 52 54 55 56 61 63 66 77 79
38      03/31/1988,06 09 11 13 21 22 24 26 36 40 43 46 47 50 53 55 62 67 70 73
39      03/30/1988,05 08 09 10 14 16 17 27 37 43 48 58 59 64 65 66 68 71 72 76
40      03/29/1988,01 10 16 19 21 24 25 28 30 33 38 40 41 46 55 58 62 67 68 71
41      03/28/1988,03 11 17 27 29 31 38 42 44 46 55 56 61 63 68 70 75 78 79 80
42      03/27/1988,04 06 08 11 16 18 19 29 33 40 44 47 52 60 65 71 73 75 76 78
43      03/26/1988,05 06 07 09 13 16 17 20 21 22 27 29 30 33 35 56 59 62 65 69
44      03/25/1988,08 21 22 26 27 28 29 30 31 34 35 52 54 62 63 65 69 71 78 79
45      03/24/1988,02 06 08 09 10 13 17 22 30 31 46 49 55 59 61 62 69 74 79 80
46      03/23/1988,01 04 09 10 15 17 19 20 21 22 26 55 58 59 64 65 71 77 79 80
47      03/22/1988,04 05 08 09 10 20 21 25 32 39 42 45 51 53 54 59 66 67 68 76
48      03/21/1988,02 10 11 13 18 21 22 26 34 37 42 51 59 60 62 63 64 73 79 80
49      03/20/1988,01 02 07 11 14 18 19 20 22 25 26 37 50 68 70 71 75 76 77 78
50      03/19/1988,03 06 08 10 14 19 22 31 35 42 47 50 51 52 57 58 62 73 74 75
51      03/18/1988,01 04 05 07 13 14 16 22 35 36 42 46 49 54 55 64 66 75 78 79
52      03/17/1988,05 06 09 12 13 19 20 27 39 42 44 49 50 55 59 60 63 66 67 68
53      03/16/1988,04 07 13 14 17 24 29 33 38 40 41 42 43 44 48 57 58 68 72 79
54      03/15/1988,04 05 09 15 25 33 35 36 42 44 46 49 51 59 61 62 63 64 65 68
55      03/14/1988,03 09 14 16 20 22 30 38 41 45 46 56 57 60 63 64 65 67 69 79
56      03/13/1988,03 05 06 07 13 15 17 18 27 33 35 36 38 40 46 49 51 52 57 58
```

```
57      03/12/1988,02 03 07 16 20 24 28 37 39 42 44 53 58 61 63 64 65 73 74 79
58      03/11/1988,04 11 14 15 18 24 26 28 32 37 44 48 54 68 71 75 76 77 78 79
59      03/10/1988,02 03 07 09 11 12 13 16 18 19 22 37 43 44 45 57 61 65 66 74
60      03/09/1988,04 08 15 19 24 28 30 34 41 44 45 50 52 54 63 70 72 73 74 75
61      03/08/1988,08 16 19 20 29 34 35 36 39 41 50 51 52 60 67 71 73 77 79 80
62      03/07/1988,02 05 10 11 15 18 20 22 23 25 36 43 45 47 48 49 58 64 76 77
63      03/06/1988,01 04 06 08 16 17 20 24 27 34 42 43 45 48 51 58 59 67 74 80
64      03/05/1988,08 09 21 25 28 31 36 39 40 42 48 51 57 62 69 74 75 76 79 80
65      03/04/1988,01 07 12 14 19 27 29 31 32 40 43 52 54 59 62 68 73 76 77 78
66      03/03/1988,06 12 14 18 22 30 31 34 43 49 50 55 61 65 66 70 72 75 76 77
67      03/02/1988,03 11 17 21 24 28 30 31 34 40 41 45 48 50 53 55 57 58 65 78
68      03/01/1988,02 05 07 09 17 20 22 23 32 37 57 58 59 64 66 68 69 71 72 76
69      02/29/1988,06 07 13 14 16 19 22 28 34 36 37 43 45 46 53 56 58 61 63 66
70      02/28/1988,05 10 11 12 15 16 17 18 22 24 30 33 35 43 51 52 53 62 71 77
71      02/27/1988,02 07 08 09 16 17 22 24 27 28 39 46 54 58 59 73 75 77 78 80
72      02/26/1988,01 03 04 06 10 11 16 17 19 24 26 41 52 56 58 64 68 69 74 76
73      02/25/1988,04 05 14 17 18 19 21 22 24 25 28 38 42 44 46 53 54 56 73 79
74      02/24/1988,03 06 09 10 15 17 30 32 34 36 47 49 50 53 56 58 59 62 65 67
75      02/23/1988,02 05 10 12 16 17 19 23 24 25 26 29 33 37 38 42 60 68 71 80
76      02/22/1988,04 06 08 16 23 28 32 38 39 41 42 44 45 47 51 52 53 60 74 80
77      02/21/1988,01 02 03 07 08 09 12 16 20 30 37 38 40 46 50 51 58 64 77 78
78      02/20/1988,01 03 05 10 14 17 18 25 34 37 40 41 43 44 49 56 69 73 77 80
79      02/19/1988,11 12 15 17 19 24 35 38 42 45 46 50 53 54 55 58 59 68 75 77
80      02/18/1988,03 10 16 18 21 28 35 37 44 46 47 49 50 52 54 61 62 68 69 79
81      02/17/1988,05 10 11 12 15 16 19 34 38 41 42 57 65 69 70 73 74 76 78 80
82      02/16/1988,02 04 05 13 15 19 20 27 29 31 38 44 50 56 60 68 71 74 75 79
83      02/15/1988,01 02 05 07 14 22 27 30 32 41 42 46 49 53 56 57 58 72 75 79
84      02/14/1988,03 06 08 09 25 26 27 29 36 38 42 48 57 59 60 63 67 68 78 79
85      02/13/1988,01 02 06 07 09 16 31 34 38 44 47 49 54 59 63 67 70 71 79 80
86      02/12/1988,08 12 15 17 18 20 21 23 36 45 47 51 54 60 61 66 74 75 78 80
87      02/11/1988,06 07 08 10 15 25 27 28 33 35 44 58 61 63 64 65 75 76 77 79
88      02/10/1988,10 13 14 16 18 25 26 27 29 30 31 33 44 47 49 52 54 55 57 64
89      02/09/1988,08 11 25 29 30 36 37 43 45 49 50 52 54 59 64 65 69 70 77 79
90      02/08/1988,04 05 16 17 18 19 23 32 45 46 47 50 54 55 58 62 64 65 69 70
91      02/07/1988,01 04 06 07 08 10 11 16 17 21 24 27 37 43 46 52 54 61 68 79
92      02/06/1988,04 09 14 19 20 22 26 29 32 35 36 38 41 49 52 54 56 58 60 70
93      02/05/1988,02 09 12 14 20 28 34 50 52 53 54 58 61 63 65 68 69 71 72 74
94      02/04/1988,11 12 13 20 21 25 28 29 30 33 35 37 40 51 52 64 69 76 77 78
95      02/03/1988,02 03 06 07 12 13 20 26 30 33 41 46 50 51 54 55 64 70 73 80
96      02/02/1988,01 07 13 16 17 20 28 33 41 46 47 51 52 57 71 73 74 77 79 80
97      02/01/1988,06 16 21 23 24 35 41 42 43 45 46 49 54 56 59 63 67 68 72 80
98      01/31/1988,03 05 14 22 25 28 30 33 35 36 38 43 54 64 65 68 69 70 74 78
99      01/30/1988,04 05 07 09 12 18 21 22 23 28 29 33 36 40 41 48 65 68 71 78
00      01/29/1988,02 03 04 08 10 19 30 31 38 41 49 51 53 56 57 60 61 63 70 79
01      01/28/1988,04 07 15 16 22 23 30 38 45 47 49 50 54 56 58 62 66 71 73 74
02      01/27/1988,04 06 09 18 23 26 27 29 33 36 43 45 48 50 51 53 62 63 78 79
03      01/26/1988,04 11 13 17 18 19 21 24 27 33 42 48 58 61 64 65 67 69 75 80
04      01/25/1988,01 03 12 13 14 15 16 24 26 31 33 37 38 41 57 61 69 74 77 78
05      01/24/1988,02 07 10 11 19 20 29 31 37 39 40 41 43 48 54 60 69 73 74 78
06      01/23/1988,01 03 08 10 16 21 23 26 30 39 40 50 56 58 61 62 75 76 78 80
07      01/22/1988,05 09 11 15 16 18 26 27 28 29 30 32 36 40 41 45 48 51 74 76
08      01/21/1988,03 04 07 08 15 23 24 27 31 38 42 45 47 50 53 60 65 70 76 80
09      01/20/1988,15 16 17 20 22 27 30 31 32 35 36 41 44 46 48 52 57 70 72 76
10      01/19/1988,02 03 15 16 30 31 35 37 40 43 44 46 47 54 55 56 58 60 62 63
```

```
01/18/1988,05 06 07 10 12 13 21 23 28 33 38 48 51 61 62 63 64 65 79 80
01/17/1988,02 05 06 07 08 17 18 21 28 31 33 43 44 48 53 55 59 66 73 76
01/16/1988,02 04 10 13 14 15 28 33 34 35 40 44 52 53 54 60 62 71 76 80
01/15/1988,01 05 12 27 29 40 41 43 44 45 47 49 50 57 62 66 67 68 71 80
01/14/1988,09 15 16 20 25 30 33 34 41 44 48 49 50 52 57 58 64 71 75 76
01/13/1988,04 17 19 21 25 26 27 30 31 32 37 44 49 53 56 62 65 75 77 79
01/12/1988,02 05 09 13 18 23 26 27 28 29 30 32 44 45 51 57 58 64 75 78
01/11/1988,04 11 16 20 38 40 43 47 49 50 51 54 61 62 63 68 70 71 72 75
01/10/1988,07 08 09 13 15 26 28 38 39 49 50 51 52 53 59 64 67 72 75 78
01/09/1988,03 04 05 16 18 19 20 21 25 36 39 52 54 56 60 62 66 70 74 77
01/08/1988,02 05 06 07 08 09 10 13 31 36 41 47 52 53 55 60 65 68 73 79
01/07/1988,05 08 11 18 22 32 33 35 36 38 43 46 47 52 55 56 60 64 68 70
01/06/1988,03 20 21 22 23 24 26 27 30 33 34 42 46 47 48 50 53 59 70 77
01/05/1988,04 14 16 20 31 34 48 49 54 56 60 64 67 68 70 71 74 77 79 80
01/04/1988,02 06 08 09 10 16 18 21 23 27 36 39 40 53 55 61 70 74 75 80
01/03/1988,03 07 12 16 22 23 28 30 34 37 41 43 46 48 58 63 67 73 75 78
01/02/1988,02 12 18 20 23 28 29 31 32 35 38 47 54 55 58 60 63 71 76 77
01/01/1988,01 02 07 19 20 21 22 23 27 30 36 38 42 45 47 58 67 68 70 72
12/31/1987,03 08 17 21 24 27 29 41 48 49 51 53 57 61 62 69 73 75 78 79
12/30/1987,03 04 06 07 11 16 18 25 27 29 35 38 41 44 54 61 62 63 75 77
12/29/1987,04 09 10 13 14 22 28 34 41 45 46 47 51 54 55 58 63 71 75 78
12/28/1987,01 03 05 14 21 25 30 37 42 43 44 46 50 58 59 61 68 71 73 79
12/27/1987,03 06 13 17 18 21 29 31 37 39 44 51 53 57 59 65 72 73 75 80
12/26/1987,04 05 06 09 10 11 13 15 28 30 34 42 44 45 66 70 72 78 79 80
12/24/1987,11 15 20 21 23 28 31 33 35 36 37 42 44 46 48 54 56 68 78 80
12/23/1987,06 20 26 29 33 34 36 37 45 49 52 55 60 61 63 64 65 67 74 75
12/22/1987,05 07 09 13 15 26 27 29 32 33 36 43 49 60 63 67 71 72 73 78
12/21/1987,01 02 03 04 08 12 17 19 23 25 30 44 50 56 58 64 66 71 74 77
12/20/1987,02 06 09 22 23 24 32 40 42 50 57 58 60 61 64 70 74 75 76 77
12/19/1987,01 04 07 08 10 15 20 21 23 37 39 47 48 54 58 66 67 74 76 80
12/18/1987,06 08 11 17 18 21 23 28 36 46 49 54 57 61 63 64 73 75 77 79
12/17/1987,03 04 05 11 15 22 35 40 42 45 46 47 59 61 63 70 71 74 78 80
12/16/1987,13 14 15 23 26 31 32 36 43 46 49 58 59 63 65 66 67 75 76 80
12/15/1987,05 07 15 17 21 23 29 32 48 53 57 59 60 61 66 67 75 76 77 79
12/14/1987,02 03 04 10 12 15 19 22 29 30 50 52 60 62 64 74 75 77 78 80
12/13/1987,01 02 03 09 11 12 16 17 18 24 26 28 36 52 54 61 66 72 73 75
12/12/1987,03 04 16 17 18 22 26 27 30 32 40 44 48 50 51 52 53 63 64 77
12/11/1987,01 10 15 20 21 23 24 25 29 32 33 35 37 39 48 49 53 61 67 78
12/10/1987,06 13 18 21 22 24 27 34 36 45 54 58 61 64 67 68 72 74 75 80
12/09/1987,02 04 07 11 13 15 27 40 41 51 52 55 57 58 61 66 68 69 74 80
12/08/1987,03 04 06 07 09 10 20 25 26 33 40 41 44 51 52 54 73 75 78 79
12/07/1987,02 04 05 14 17 20 29 36 46 47 50 56 57 61 63 65 69 74 75 77
12/06/1987,02 04 09 10 15 20 22 27 33 35 39 40 42 45 46 51 58 63 76 77
12/05/1987,06 07 08 12 13 19 21 26 42 45 50 53 56 57 63 64 70 72 73 75
12/04/1987,05 06 07 13 14 16 20 21 27 29 30 31 35 36 42 44 51 57 65 66
12/03/1987,03 04 05 09 20 26 32 42 47 51 52 55 57 63 65 68 71 72 73 79
12/02/1987,01 04 09 11 13 15 20 26 35 37 45 48 49 52 54 55 60 64 65 68
12/01/1987,07 08 09 12 13 15 19 27 29 32 38 42 43 44 47 61 71 72 74 79
11/30/1987,01 18 28 31 33 38 42 46 49 54 55 57 61 62 68 70 73 75 78 80
11/29/1987,01 08 11 15 16 20 21 22 24 25 26 33 42 47 51 58 66 68 70 76
11/28/1987,04 05 06 07 10 11 12 24 28 32 35 36 38 47 49 51 70 73 75 78
11/27/1987,08 11 13 14 19 20 21 28 32 38 48 49 53 58 66 69 70 75 76 79
11/26/1987,04 07 08 14 17 29 34 37 41 42 47 50 54 55 64 68 71 74 75 78
11/25/1987,03 09 16 18 19 30 31 35 38 41 46 47 48 51 53 59 65 67 70 77
```

```
65      11/24/1987,07 10 11 14 24 25 29 30 32 34 35 37 45 49 58 64 66 76 78 79
66      11/23/1987,10 11 12 18 29 32 35 36 42 43 47 48 59 62 65 69 72 74 77 79
67      11/22/1987,07 12 20 23 24 25 30 32 33 34 41 49 56 61 64 65 67 74 77 80
68      11/21/1987,01 03 04 07 09 16 19 26 29 32 33 40 43 49 54 56 57 62 67 74
69      11/20/1987,02 03 05 06 09 18 21 22 28 37 40 49 54 56 59 63 64 76 78 80
70      11/19/1987,01 03 06 16 19 22 27 31 33 34 43 44 54 57 60 63 64 66 67 73
71      11/18/1987,05 16 26 28 32 34 35 42 47 51 54 56 58 62 66 68 69 72 74 80
72      11/17/1987,08 10 14 19 20 23 30 32 34 40 42 44 47 50 59 63 69 73 76 78
73      11/16/1987,06 10 13 18 21 25 28 33 35 36 38 44 48 51 56 57 59 64 66 73
74      11/15/1987,04 07 14 15 29 33 37 41 42 43 46 55 61 62 65 67 68 74 76 80
75      11/14/1987,03 06 08 09 12 15 17 18 31 35 36 58 60 61 65 67 68 69 72 75
76      11/13/1987,01 12 13 15 16 19 21 28 30 48 50 51 53 54 61 68 75 76 78 79
77      11/12/1987,04 05 06 08 10 16 17 21 26 30 34 38 42 47 52 54 61 70 74 77
78      11/11/1987,01 04 07 08 12 24 35 39 42 43 45 46 48 53 54 58 63 70 74 76
79      11/10/1987,05 06 07 10 11 12 18 20 28 30 33 37 40 43 45 47 54 75 78 79
80      11/09/1987,03 06 08 13 14 22 24 27 34 37 52 56 58 59 60 66 68 75 79 80
81      11/08/1987,04 09 15 20 21 23 24 27 31 43 54 56 60 62 63 67 68 74 76 79
82      11/07/1987,04 10 12 15 28 30 34 35 38 43 48 51 52 57 63 70 71 72 75 76
83      11/06/1987,06 09 18 20 21 22 26 28 33 36 38 39 42 50 56 63 71 75 76 79
84      11/05/1987,04 10 13 15 21 23 28 30 32 34 35 37 38 42 51 62 72 74 76 78
85      11/04/1987,11 12 16 24 25 30 35 42 44 46 49 50 56 57 62 63 67 69 72 75
86      11/03/1987,04 09 10 16 20 23 24 31 36 42 49 52 55 58 64 68 70 71 73 77
87      11/02/1987,01 04 08 10 15 16 19 20 25 27 28 38 43 54 59 65 71 72 75 76
88      11/01/1987,02 05 14 20 22 23 27 30 32 35 40 43 45 57 60 63 68 70 74 76
89      10/31/1987,05 06 11 13 22 28 30 43 50 54 57 59 62 64 66 73 74 75 78 79
90      10/30/1987,05 17 19 20 22 29 34 36 38 48 55 63 65 67 68 73 76 77 78 80
91      10/29/1987,11 18 19 22 25 26 29 31 32 33 41 43 47 48 56 63 68 70 76 78
92      10/28/1987,02 06 15 16 17 22 24 29 35 37 44 49 50 53 57 64 72 75 79 80
93      10/27/1987,01 05 08 21 23 27 29 31 34 38 42 49 53 54 56 58 69 76 79 80
94      10/26/1987,05 06 07 12 19 30 32 34 38 39 40 43 48 50 55 67 72 75 79 80
95      10/25/1987,02 05 13 14 15 16 27 31 33 36 43 45 52 56 57 66 67 68 69 70
96      10/24/1987,09 10 13 16 19 23 24 28 30 31 32 35 44 45 50 54 56 59 62 75
97      10/23/1987,03 05 08 09 12 14 15 19 20 25 30 34 39 61 62 63 65 67 73 76
98      10/22/1987,05 07 13 14 16 23 27 29 30 31 36 41 42 48 53 57 58 60 71 80
99      10/21/1987,04 09 18 26 27 36 38 41 43 47 50 54 55 56 57 63 71 72 77 80
00      10/20/1987,01 05 06 13 15 23 26 34 38 43 44 46 50 52 57 68 70 73 77 80
01      10/19/1987,04 08 10 15 16 18 20 24 30 31 33 35 42 44 54 55 56 58 78 80
02      10/18/1987,02 06 10 14 17 18 28 31 32 34 39 41 42 45 53 54 57 69 75 79
03      10/17/1987,01 06 07 11 17 30 31 32 34 40 45 46 47 61 62 63 64 68 72 78
04      10/16/1987,08 10 30 33 35 36 41 43 53 54 55 58 60 61 63 67 74 75 76 80
05      10/15/1987,01 02 03 04 09 15 17 19 25 28 34 37 38 42 45 51 66 67 68 79
06      10/14/1987,03 05 13 16 22 28 29 30 35 36 40 41 45 51 52 53 60 62 77 78
07      10/13/1987,05 13 18 20 24 27 34 36 42 46 47 52 54 59 60 62 69 75 76 80
08      10/12/1987,01 07 08 11 13 18 23 24 33 39 42 51 57 65 66 70 72 74 77 80
09      10/11/1987,02 06 08 14 15 21 25 26 30 31 35 38 42 44 48 52 56 60 74 76
10      10/10/1987,02 03 18 23 24 26 27 29 30 32 41 42 51 55 56 57 61 63 66 69
11      10/09/1987,01 03 04 11 16 27 31 36 38 43 44 47 53 54 58 63 65 67 69 70
12      10/08/1987,03 04 06 07 09 11 18 22 24 31 37 52 55 60 67 71 76 77 78 80
13      10/07/1987,01 03 11 15 17 24 25 26 34 41 43 46 48 61 62 65 67 71 78 79
14      10/06/1987,04 09 16 17 20 34 36 37 38 47 50 51 59 61 63 65 74 77 78 79
15      10/05/1987,03 12 15 21 24 26 30 35 36 39 40 43 50 54 56 60 62 69 72 74
16      10/04/1987,01 03 06 07 09 12 14 18 22 32 41 44 54 56 61 63 64 70 74 75
17      10/03/1987,02 04 08 09 10 16 28 31 38 41 43 44 51 63 64 66 68 73 76 80
18      10/02/1987,03 11 16 17 21 22 23 25 31 32 34 46 51 52 54 55 57 69 76 79
```

```
19      10/01/1987,02 03 04 13 16 20 25 31 35 42 44 52 54 59 65 68 75 76 78 79
20      09/30/1987,04 06 08 21 23 31 32 38 40 42 43 48 50 52 53 59 61 66 73 76
21      09/29/1987,01 06 07 18 25 32 33 35 42 44 49 50 52 55 56 68 70 72 79 80
22      09/28/1987,07 09 11 15 16 21 23 30 34 38 39 44 46 47 54 63 66 67 68 70
23      09/27/1987,14 21 25 26 29 30 32 37 41 43 44 47 51 52 53 55 63 71 75 77
24      09/26/1987,02 06 08 17 22 25 27 35 44 46 47 48 52 53 54 55 64 65 66 75
25      09/25/1987,12 16 18 19 22 24 25 33 37 39 40 43 52 54 55 58 64 65 69 70
26      09/24/1987,01 11 12 18 19 21 30 31 32 36 38 53 58 60 63 67 70 71 73 74
27      09/23/1987,08 15 25 31 34 36 39 40 41 47 49 59 69 70 72 73 75 76 77 78
28      09/22/1987,03 04 11 16 33 35 38 40 42 45 46 48 53 55 59 61 63 66 69 70
29      09/21/1987,01 06 07 09 10 15 25 26 33 38 40 46 50 52 64 68 71 72 78 80
30      09/20/1987,05 07 26 33 38 44 47 55 59 61 62 65 66 67 69 70 71 77 78 79
31      09/19/1987,08 18 19 21 30 35 38 42 43 44 46 53 55 58 60 62 70 72 73 80
32      09/18/1987,01 05 09 17 23 26 27 31 34 36 37 38 39 40 42 47 49 50 69 76
33      09/17/1987,06 17 18 22 25 28 37 39 41 44 45 47 53 54 55 58 65 67 70 75
34      09/16/1987,15 21 27 29 35 36 40 44 46 52 55 57 59 61 63 65 68 73 74 80
35      09/15/1987,01 03 12 13 17 19 25 26 28 30 35 36 46 48 52 64 67 69 70 72
36      09/14/1987,01 05 06 08 11 12 25 28 37 38 41 45 48 59 61 65 71 72 74 76
37      09/13/1987,01 07 14 22 24 26 27 37 41 42 43 45 46 50 52 62 63 65 68 73
38      09/12/1987,04 12 16 17 21 22 29 30 32 37 48 50 51 58 60 62 63 72 74 75
39      09/11/1987,03 05 16 22 32 41 44 45 48 49 51 56 59 64 65 67 68 70 73 74
40      09/10/1987,07 11 12 13 19 23 25 29 30 33 34 39 46 50 56 61 62 66 70 78
41      09/09/1987,01 02 09 13 16 19 26 34 36 38 39 42 48 54 57 58 68 70 74 79
42      09/08/1987,01 04 05 08 12 18 19 34 35 37 38 39 42 49 54 64 65 68 74 75
43      09/07/1987,01 06 14 19 22 26 27 33 34 40 46 47 50 51 62 65 72 76 79 80
44      09/06/1987,01 03 11 12 13 14 15 20 22 27 38 39 45 46 54 63 65 68 74 76
45      09/05/1987,08 10 18 21 23 37 41 43 44 45 47 48 50 52 54 55 56 66 73 75
46      09/04/1987,06 14 15 16 17 19 25 28 31 38 41 42 43 44 48 55 59 64 75 78
47      09/03/1987,02 04 09 22 27 29 30 31 33 37 41 43 50 53 55 56 61 64 70 76
48      09/02/1987,07 13 15 21 23 24 26 27 29 37 45 46 49 53 55 57 60 65 73 78
49      09/01/1987,10 12 19 23 24 25 28 33 37 41 42 45 64 65 67 71 74 75 76 78
50      08/31/1987,01 02 08 15 16 17 18 28 36 37 39 48 55 62 64 65 66 68 70 74
51      08/30/1987,13 14 16 17 19 27 30 31 33 37 41 42 46 52 54 56 57 61 72 79
52      08/29/1987,03 06 08 09 10 12 13 15 18 21 25 36 45 50 51 53 56 62 77 80
53      08/28/1987,03 06 08 09 14 20 21 26 29 32 36 38 40 54 55 68 69 75 78 80
54      08/27/1987,02 04 05 11 19 20 27 30 42 43 46 50 51 54 66 70 74 75 76 77
55      08/26/1987,02 04 07 12 14 15 19 23 25 29 32 33 39 44 48 62 64 68 74 76
56      08/25/1987,06 08 11 12 13 20 26 34 38 39 41 42 47 49 53 56 66 73 78 80
57      08/24/1987,06 08 13 15 17 19 24 30 34 37 41 47 50 51 54 57 62 69 71 77
58      08/23/1987,03 06 10 12 14 18 20 22 24 27 30 35 42 53 55 64 65 70 71 72
59      08/22/1987,01 03 06 09 10 16 19 22 24 25 29 31 32 46 50 53 58 65 75 78
60      08/21/1987,04 05 06 09 11 12 22 24 46 48 50 55 58 60 65 66 69 72 73 76
61      08/20/1987,01 13 19 22 26 29 37 40 41 42 44 54 57 60 61 67 68 69 79 80
62      08/19/1987,02 03 04 09 13 24 25 28 29 32 35 39 47 48 57 60 67 71 73 76
63      08/18/1987,02 04 07 13 15 16 18 32 36 42 43 45 50 53 56 64 65 67 74 78
64      08/17/1987,02 07 12 14 16 18 27 28 31 33 34 36 42 45 47 66 69 72 73 78
65      08/16/1987,02 08 10 13 18 19 20 26 28 29 32 36 38 45 48 49 53 70 78 79
66      08/15/1987,01 09 10 13 14 25 35 42 44 49 50 51 56 58 59 64 72 73 76 78
67      08/14/1987,04 07 11 18 29 35 45 49 50 51 54 55 57 61 65 75 76 77 78 79
68      08/13/1987,02 05 09 10 11 12 20 21 24 31 34 36 41 42 55 57 61 65 68 77
69      08/12/1987,03 05 06 08 09 22 27 29 34 37 47 49 54 61 62 63 68 69 71 72
70      08/11/1987,04 15 17 25 28 32 33 48 49 50 51 56 57 58 60 62 71 74 75 80
71      08/10/1987,01 12 15 20 22 26 27 30 33 49 50 59 63 66 67 70 72 76 79 80
72      08/09/1987,02 06 11 19 24 26 34 36 37 42 44 46 50 56 62 64 70 71 78 80
```

```
73      08/08/1987,11 16 18 21 22 23 25 30 38 40 43 47 50 54 59 60 65 67 72 73
74      08/07/1987,06 11 16 17 19 24 25 29 35 37 41 42 46 48 49 59 62 70 76 79
75      08/06/1987,03 09 14 20 25 33 35 38 39 43 46 53 56 58 59 67 74 75 77 80
76      08/05/1987,02 06 08 10 11 13 16 19 30 32 33 38 41 42 49 50 51 55 63 75
77      08/04/1987,02 08 16 18 20 21 23 24 26 28 31 38 50 55 57 62 70 71 77 79
78      08/03/1987,06 09 17 20 22 23 25 32 34 37 38 45 55 58 64 68 72 73 74 77
79      08/02/1987,04 05 10 11 12 13 25 28 30 42 45 46 49 56 59 61 63 65 72 74
80      08/01/1987,03 09 19 20 21 24 27 31 35 39 44 45 47 53 57 60 61 62 65 74
81      07/31/1987,03 05 06 13 20 24 30 31 34 41 46 47 50 58 63 67 68 70 71 72
82      07/30/1987,03 04 05 09 10 12 17 20 23 27 30 34 41 43 48 53 66 67 76 78
83      07/29/1987,01 04 07 14 15 20 27 30 33 37 47 48 50 53 58 64 65 67 70 75
84      07/28/1987,02 11 21 26 35 38 41 45 46 48 50 52 55 57 61 64 68 69 72 73
85      07/27/1987,02 06 10 11 12 13 21 36 37 39 41 46 50 58 59 69 70 71 74 75
86      07/26/1987,05 07 21 22 23 27 30 33 35 36 37 41 42 45 48 50 57 66 69 77
87      07/25/1987,04 07 08 10 11 12 21 36 42 44 45 48 49 50 52 53 55 68 76 80
88      07/24/1987,02 03 13 21 25 31 34 35 36 37 49 50 54 58 59 61 64 70 71 79
89      07/23/1987,01 04 07 10 13 20 26 35 41 45 51 52 54 57 58 60 63 71 72 73
90      07/22/1987,09 11 16 19 22 24 26 27 38 39 40 46 48 51 54 58 61 69 73 75
91      07/21/1987,01 02 03 09 11 16 17 23 24 27 30 31 36 40 41 45 53 73 74 80
92      07/20/1987,03 06 09 11 13 15 20 21 25 31 35 38 39 41 44 46 50 57 61 73
93      07/19/1987,05 07 11 13 15 16 22 30 34 37 38 47 48 62 67 70 71 73 77 78
94      07/18/1987,04 07 09 14 25 26 29 32 35 38 42 48 49 53 56 65 68 69 70 76
95      07/17/1987,01 02 04 07 08 27 31 37 45 49 51 53 55 59 65 66 67 68 69 70
96      07/16/1987,01 06 10 17 19 21 27 31 33 36 43 46 48 51 52 57 59 62 77 80
97      07/15/1987,01 05 06 15 16 22 24 30 39 41 43 45 49 58 60 62 64 68 76 80
98      07/14/1987,01 02 03 06 17 19 31 32 38 44 47 49 52 62 65 68 70 71 76 78
99      07/13/1987,01 04 05 10 14 18 20 21 23 28 29 34 40 41 53 54 58 72 75 77
00      07/12/1987,02 05 06 08 09 11 12 14 17 19 32 33 34 38 46 66 74 77 78 79
01      07/11/1987,03 07 10 14 15 17 19 22 31 32 35 39 51 52 57 61 65 74 75 77
02      07/10/1987,02 09 11 16 17 19 21 22 23 24 26 28 33 35 51 57 60 64 66 70
03      07/09/1987,01 03 29 32 41 42 52 53 54 56 59 61 64 65 68 69 70 73 77 79
04      07/08/1987,01 04 11 17 20 25 29 32 34 36 39 48 49 50 52 53 68 69 77 80
05      07/07/1987,06 09 11 21 24 25 30 34 35 41 45 53 54 58 61 63 71 75 77 80
06      07/06/1987,04 22 29 30 31 32 46 49 52 53 56 58 62 63 64 66 71 72 77 80
07      07/05/1987,01 04 16 17 18 19 27 32 42 45 46 48 50 52 56 59 60 67 68 75
08      07/04/1987,08 09 11 14 17 25 26 29 32 41 53 54 58 62 63 67 73 75 78 79
09      07/03/1987,02 07 10 13 14 27 29 31 33 36 37 58 62 63 64 66 69 71 72 74
10      07/02/1987,04 10 14 15 21 25 26 40 42 43 46 47 50 52 55 56 57 64 72 73
11      07/01/1987,01 02 04 08 14 16 17 20 24 26 31 40 52 57 59 67 71 73 74 79
12      06/30/1987,07 09 14 15 16 17 18 23 25 29 38 50 52 54 55 56 58 64 78 80
13      06/29/1987,05 08 09 18 19 25 35 37 45 47 51 52 58 65 66 73 74 76 78 80
14      06/28/1987,03 04 09 23 25 27 28 29 34 41 47 51 53 54 61 62 64 66 67 79
15      06/27/1987,02 08 11 14 16 26 29 30 42 47 50 54 56 61 62 71 75 76 78 80
16      06/26/1987,01 09 14 15 16 17 23 24 27 41 43 46 47 49 51 53 66 76 77 79
17      06/25/1987,05 06 10 19 21 22 35 37 38 39 43 46 49 50 53 56 58 59 69 79
18      06/24/1987,06 07 08 11 26 29 37 40 44 47 49 55 57 64 65 66 69 72 73 76
19      06/23/1987,02 03 08 09 10 12 31 32 34 38 41 47 49 52 53 55 65 67 68 73
20      06/22/1987,02 04 08 12 18 21 27 28 29 32 33 44 48 52 57 63 66 67 69 74
21      06/21/1987,01 10 15 16 21 24 27 30 36 45 49 50 52 54 55 56 60 62 64 75
22      06/20/1987,02 03 12 13 15 17 18 19 36 37 42 48 52 55 59 61 64 69 78 79
23      06/19/1987,04 05 07 08 10 18 29 33 39 41 42 44 56 57 61 71 72 73 79 80
24      06/18/1987,11 15 23 29 31 32 35 37 39 42 45 50 51 54 56 59 60 65 66 67
25      06/17/1987,02 03 12 18 21 27 29 36 55 59 62 66 67 68 69 70 73 74 79 80
26      06/16/1987,16 22 24 25 28 29 30 31 35 39 51 52 53 54 56 60 62 74 78 79
```

```
27      06/15/1987,02 06 09 10 14 16 17 19 20 24 28 32 45 46 48 55 61 65 70 72
28      06/14/1987,01 02 11 15 16 23 25 28 34 37 44 46 50 51 55 56 60 65 72 73
29      06/13/1987,01 06 10 14 16 17 25 26 35 40 51 55 58 60 64 66 67 70 72 76
30      06/12/1987,04 08 14 16 17 19 20 22 28 34 38 45 49 50 52 60 64 67 73 79
31      06/11/1987,10 12 18 19 20 21 22 23 24 26 38 40 42 51 53 58 69 75 77 78
32      06/10/1987,03 10 14 21 26 29 32 33 38 40 41 53 56 58 62 65 67 69 73 75
33      06/09/1987,04 05 12 13 19 27 28 30 32 37 44 45 60 64 65 66 69 73 74 76
34      06/08/1987,01 02 03 06 07 13 24 29 34 35 40 42 43 45 46 61 70 72 77 80
35      06/07/1987,01 07 08 09 11 14 20 21 25 32 33 35 40 41 44 45 52 56 57 75
36      06/06/1987,04 07 12 14 21 30 37 39 44 50 52 54 56 61 66 72 74 77 79 80
37      06/05/1987,02 04 17 20 22 37 40 48 49 51 53 55 57 60 66 69 71 72 73 74
38      06/04/1987,05 06 10 11 12 13 17 32 35 37 38 39 42 51 54 56 72 75 76 80
39      06/03/1987,01 03 10 12 22 23 28 33 47 50 51 52 57 62 63 64 66 72 75 78
40      06/02/1987,03 09 10 20 28 31 32 34 35 39 41 44 46 48 49 50 55 58 61 73
41      06/01/1987,07 09 11 12 13 16 18 19 22 24 27 34 40 46 47 53 57 59 69 78
42      05/31/1987,09 10 12 19 22 23 27 31 32 37 39 50 51 55 63 69 71 73 78 79
43      05/30/1987,01 07 08 09 13 17 18 20 24 26 28 31 34 36 39 47 50 58 73 78
44      05/29/1987,02 04 05 07 15 24 28 30 32 34 45 47 48 49 51 56 62 66 71 80
45      05/28/1987,04 12 13 14 22 25 29 30 31 38 39 41 42 45 47 53 63 68 70 76
46      05/27/1987,02 10 17 20 26 28 33 35 36 39 40 45 47 58 60 67 69 70 73 76
47      05/26/1987,01 09 14 16 21 23 25 27 37 41 46 49 55 58 59 64 66 71 73 80
48      05/25/1987,08 10 11 20 27 30 31 35 36 41 47 48 55 60 62 63 72 75 76 77
49      05/24/1987,08 15 18 19 23 26 34 39 46 48 52 55 56 61 62 64 69 74 78 79
50      05/23/1987,02 07 09 13 14 17 19 24 35 37 50 54 55 57 59 60 63 65 76 78
51      05/22/1987,05 11 14 15 16 17 20 29 33 35 36 37 40 42 48 66 70 72 73 77
52      05/21/1987,02 06 07 09 13 23 25 29 30 33 37 42 44 46 51 55 62 64 72 75
53      05/20/1987,03 07 15 17 22 23 32 33 34 35 38 44 45 52 64 65 67 68 70 76
54      05/19/1987,02 03 07 09 14 17 22 27 34 38 41 47 49 52 57 58 65 71 73 79
55      05/18/1987,03 04 06 10 11 14 18 20 29 31 32 37 40 44 61 68 70 73 74 75
56      05/17/1987,02 05 11 14 18 20 31 33 34 36 42 43 56 57 60 66 72 77 78 79
57      05/16/1987,06 08 11 12 20 22 30 41 43 45 47 50 52 54 57 61 67 70 78 80
58      05/15/1987,01 02 08 13 14 16 17 19 20 30 31 34 37 38 46 56 61 66 70 71
59      05/14/1987,05 08 15 17 21 23 27 29 32 45 46 48 59 61 62 64 74 76 77 79
60      05/13/1987,11 12 17 22 23 25 30 31 38 44 47 52 53 54 58 59 62 64 68 70
61      05/12/1987,08 09 16 18 23 26 33 34 42 45 46 48 49 51 53 54 59 67 76 80
62      05/11/1987,06 07 08 12 15 21 24 25 34 35 39 40 41 44 49 50 52 55 58 79
63      05/10/1987,12 21 22 23 25 34 37 38 39 40 46 53 56 59 60 63 67 75 77 79
64      05/09/1987,02 05 06 08 12 17 20 23 24 29 31 38 41 46 52 53 58 59 70 72
65      05/08/1987,05 12 14 17 18 22 25 26 27 29 34 42 45 47 50 58 61 66 69 79
66      05/07/1987,03 04 05 08 13 14 18 20 26 27 29 37 38 40 41 44 53 60 74 77
67      05/06/1987,01 03 05 07 09 12 20 23 24 31 43 48 51 52 62 64 66 67 70 79
68      05/05/1987,02 05 08 16 30 34 36 38 39 40 43 49 52 54 55 61 64 71 79 80
69      05/04/1987,12 18 23 24 25 34 36 39 48 55 56 60 61 65 66 67 69 71 76 79
70      05/03/1987,01 05 13 14 15 19 22 23 26 38 39 41 61 65 66 67 68 69 71 78
71      05/02/1987,01 04 11 16 18 20 28 29 38 46 47 53 62 64 70 72 74 76 78 80
72      05/01/1987,09 15 23 26 29 30 31 35 38 42 46 53 54 60 64 70 72 73 76 80
73      04/30/1987,12 14 15 20 22 26 27 34 37 42 43 44 51 52 54 55 58 60 66 76
74      04/29/1987,04 06 07 10 21 26 30 34 39 40 48 49 50 54 62 65 66 68 69 79
75      04/28/1987,01 05 06 13 18 23 25 32 33 35 37 42 45 47 57 60 70 72 78 80
76      04/27/1987,05 09 13 16 24 31 35 43 46 49 58 59 63 65 66 67 68 74 76 80
77      04/26/1987,07 09 11 12 14 28 31 32 34 43 48 51 55 58 59 67 69 71 75 77
78      04/25/1987,02 08 22 25 28 33 34 44 46 49 50 52 56 57 59 60 68 76 79 80
79      04/24/1987,06 07 08 11 18 20 21 25 31 34 45 47 49 51 54 60 62 64 70 77
80      04/23/1987,14 16 23 26 29 37 38 40 41 49 55 58 59 64 65 67 71 73 75 77
```

```
81      04/22/1987,04 06 12 20 24 27 28 29 30 36 43 51 60 62 63 66 70 71 73 74
82      04/21/1987,01 03 08 13 14 26 28 29 31 44 52 58 61 63 64 68 72 74 75 79
83      04/20/1987,04 09 11 14 16 18 23 31 34 36 39 44 52 55 64 65 68 69 73 76
84      04/19/1987,02 05 09 11 13 14 15 16 18 23 28 30 33 35 43 48 52 70 75 80
85      04/18/1987,02 03 05 09 11 13 14 18 19 24 34 40 43 49 51 52 62 69 71 72
86      04/17/1987,05 06 10 14 20 22 28 29 35 39 46 48 52 53 55 57 58 61 69 79
87      04/16/1987,06 11 17 24 28 32 33 36 39 43 44 46 48 53 57 59 62 69 76 79
88      04/15/1987,01 02 06 10 18 19 23 27 29 37 39 44 46 47 56 64 68 75 77 79
89      04/14/1987,02 04 07 12 22 26 33 37 40 44 49 54 55 58 61 63 64 68 71 79
90      04/13/1987,01 04 10 13 24 27 29 35 40 41 47 53 57 63 65 66 67 70 74 76
91      04/12/1987,03 06 07 09 15 21 22 32 36 38 40 51 52 54 58 62 67 71 73 79
92      04/11/1987,03 10 17 18 26 33 38 40 43 44 45 46 55 61 63 65 70 71 75 78
93      04/10/1987,04 09 10 12 13 17 18 40 43 45 55 59 66 67 70 71 72 74 79 80
94      04/09/1987,03 05 08 11 15 31 34 37 41 42 47 51 54 60 62 67 69 76 78 80
95      04/08/1987,04 06 07 09 13 14 17 21 27 32 34 35 41 42 45 50 51 66 70 80
96      04/07/1987,07 09 19 27 31 32 33 36 38 39 46 47 51 52 54 56 58 63 70 73
97      04/06/1987,03 04 08 10 11 15 20 24 30 37 48 55 56 60 61 62 66 67 72 80
98      04/05/1987,04 07 08 11 16 26 31 35 38 40 43 45 47 53 56 57 59 63 65 78
99      04/04/1987,02 04 07 10 14 17 23 28 32 36 39 46 50 53 65 70 73 76 77 80
00      04/03/1987,07 11 12 14 15 16 19 20 22 35 46 47 55 58 59 66 67 68 72 80
01      04/02/1987,02 03 04 05 11 13 17 22 29 34 35 37 39 49 60 69 73 76 77 78
02      04/01/1987,03 06 11 15 21 23 24 30 40 45 46 47 50 57 69 72 73 74 76 79
03      03/31/1987,03 06 08 16 19 23 26 33 35 40 43 45 59 62 64 69 70 72 73 80
04      03/30/1987,04 08 10 13 17 22 31 35 42 44 47 48 53 61 62 66 67 68 69 71
05      03/29/1987,01 02 06 13 16 17 18 21 22 28 39 45 47 51 52 53 61 62 71 76
06      03/28/1987,02 04 23 29 30 33 38 39 41 43 46 49 56 63 69 70 72 73 74 79
07      03/27/1987,02 06 11 17 18 22 26 27 28 30 33 35 38 52 54 56 60 61 65 79
08      03/26/1987,12 14 15 16 17 21 31 32 46 47 48 55 56 57 58 61 64 68 74 77
09      03/25/1987,05 08 09 11 14 17 22 24 26 27 29 30 32 36 44 46 56 65 69 74
10      03/24/1987,05 12 17 18 19 20 21 23 36 39 42 43 52 59 63 65 66 75 78 80
11      03/23/1987,03 04 08 14 15 17 20 22 27 35 42 43 44 47 55 57 59 63 68 71
12      03/22/1987,03 20 27 28 29 31 38 41 43 45 49 50 54 69 70 71 72 77 79 80
13      03/21/1987,07 08 25 30 33 34 35 38 40 43 47 50 54 55 60 61 64 70 79 80
14      03/20/1987,03 07 08 18 24 26 27 28 37 38 46 48 50 51 54 59 63 73 75 76
15      03/19/1987,01 04 06 07 11 12 14 16 21 22 28 36 37 41 46 50 56 69 77 78
16      03/18/1987,02 06 13 24 25 26 33 36 37 39 43 48 49 51 55 57 60 62 68 79
17      03/17/1987,03 10 11 12 13 18 23 25 28 31 34 40 44 46 60 63 64 74 76 77
18      03/16/1987,03 06 11 27 40 42 46 47 50 56 58 65 66 69 70 73 75 76 79 80
19      03/15/1987,01 02 04 08 11 13 14 17 25 33 35 39 41 53 55 57 62 64 71 80
20      03/14/1987,01 02 08 13 22 34 38 39 40 42 43 49 51 54 58 65 73 74 75 76
21      03/13/1987,01 08 17 22 24 31 32 37 44 47 49 55 56 57 58 60 62 75 76 80
22      03/12/1987,03 05 07 08 11 12 20 26 27 31 32 34 37 38 42 60 63 69 74 76
23      03/11/1987,03 05 06 07 19 21 24 28 35 39 42 44 50 56 57 58 73 74 78 79
24      03/10/1987,09 10 12 14 15 16 28 29 30 33 37 39 40 44 52 59 70 72 73 74
25      03/09/1987,02 06 07 11 15 29 32 33 34 35 36 40 45 50 51 56 62 67 69 72
26      03/08/1987,02 03 04 06 11 15 21 23 26 28 33 39 44 54 58 61 64 67 71 80
27      03/07/1987,09 10 11 13 14 16 18 19 21 26 30 38 40 44 45 49 59 67 69 76
28      03/06/1987,01 02 06 10 14 27 28 29 31 35 36 41 49 61 66 68 71 76 78 79
29      03/05/1987,01 02 04 06 14 15 20 24 25 30 33 38 42 48 52 59 62 66 71 79
30      03/04/1987,01 03 05 12 19 20 24 27 31 36 38 39 43 44 47 66 67 71 72 79
31      03/03/1987,06 10 14 16 20 22 27 30 39 43 46 47 52 62 63 69 70 74 79 80
32      03/02/1987,02 05 21 29 31 33 34 37 41 46 48 52 54 57 59 62 66 70 72 78
33      03/01/1987,02 03 14 18 19 27 32 34 35 39 40 47 50 52 54 61 62 69 73 80
34      02/28/1987,04 06 09 19 32 37 38 39 40 47 50 57 58 59 62 69 73 74 75 77
```

```
35      02/27/1987,02 04 07 11 12 14 22 27 29 32 33 34 41 50 52 60 65 72 78 80
36      02/26/1987,02 23 25 28 31 32 35 45 46 50 51 52 54 55 61 68 72 74 76 78
37      02/25/1987,01 05 10 17 25 30 31 33 37 40 41 45 47 49 51 54 59 64 70 77
38      02/24/1987,03 07 19 33 34 37 39 44 48 50 51 52 58 65 68 70 72 73 75 77
39      02/23/1987,01 03 07 10 23 25 31 33 41 46 47 52 57 59 60 62 67 71 76 78
40      02/22/1987,01 15 24 25 31 41 42 46 48 50 51 52 53 54 58 61 69 71 75 80
41      02/21/1987,02 08 11 15 16 22 25 26 31 33 39 45 48 49 50 52 54 62 74 78
42      02/20/1987,02 15 17 18 21 24 30 31 37 41 46 48 52 54 62 67 69 70 71 76
43      02/19/1987,03 07 08 14 18 22 23 32 33 35 36 37 38 42 60 63 69 71 73 75
44      02/18/1987,08 14 15 19 20 22 28 33 35 38 40 44 46 49 57 64 65 66 75 76
45      02/17/1987,01 03 04 15 16 18 22 25 31 49 53 55 59 65 67 68 69 70 76 78
46      02/16/1987,02 08 16 17 20 22 24 29 32 36 43 46 47 50 51 58 64 68 71 77
47      02/15/1987,05 11 12 15 22 25 31 32 35 43 45 47 54 57 59 64 65 69 72 78
48      02/14/1987,04 09 12 13 16 22 23 25 30 31 33 37 39 43 47 53 60 65 70 77
49      02/13/1987,03 05 06 18 21 24 28 36 42 44 45 50 52 53 54 59 61 73 75 79
50      02/12/1987,02 05 08 16 18 19 28 30 34 40 42 48 55 56 68 69 70 71 76 80
51      02/11/1987,01 08 09 10 17 20 21 23 25 26 28 33 39 53 54 58 59 61 63 79
52      02/10/1987,04 06 10 13 18 20 24 29 33 34 36 40 47 48 57 59 62 69 71 73
53      02/09/1987,01 04 12 13 20 21 29 36 38 44 49 53 58 61 68 73 74 76 77 79
54      02/08/1987,01 04 08 12 13 15 18 25 29 33 35 42 43 44 53 54 57 69 72 75
55      02/07/1987,03 05 06 13 15 21 25 28 29 30 35 41 44 53 57 58 66 68 70 74
56      02/06/1987,02 12 13 14 20 24 33 35 36 37 39 46 50 53 57 62 63 68 70 72
57      02/05/1987,08 14 21 22 25 43 44 45 46 47 50 53 57 58 65 66 67 77 78 79
58      02/04/1987,04 06 07 14 19 20 23 28 32 33 44 49 51 63 65 69 70 73 74 78
59      02/03/1987,01 03 05 18 20 23 28 30 37 42 43 56 61 64 66 67 69 72 75 78
60      02/02/1987,12 13 15 30 36 40 49 52 59 60 62 67 70 71 72 73 74 77 78 79
61      02/01/1987,01 05 07 10 11 16 27 36 45 46 48 52 54 58 63 65 67 71 74 79
62      01/31/1987,04 09 12 13 17 25 31 34 35 40 44 46 48 57 60 62 67 71 72 77
63      01/30/1987,03 10 11 12 13 17 20 26 29 33 34 36 52 58 61 71 74 75 78 79
64      01/29/1987,06 08 09 18 25 30 35 36 39 40 51 56 57 60 62 68 73 74 78 79
65      01/28/1987,08 09 13 15 18 26 29 31 32 36 41 45 53 55 60 62 63 73 75 78
66      01/27/1987,13 28 29 31 34 37 41 49 51 56 57 58 63 64 67 69 70 71 77 78
67      01/26/1987,08 12 17 20 26 31 37 38 39 42 47 50 51 57 61 62 67 69 73 77
68      01/25/1987,19 21 24 27 30 32 44 47 48 53 61 63 65 69 72 73 74 76 77 78
69      01/24/1987,04 05 08 14 17 19 22 32 36 38 39 44 48 50 51 56 66 67 78 79
70      01/23/1987,02 05 06 08 11 13 28 29 43 44 48 52 53 58 59 61 62 77 79 80
71      01/22/1987,02 07 14 18 19 23 26 29 32 35 40 41 43 46 52 62 68 70 76 80
72      01/21/1987,02 04 08 13 14 16 18 19 26 30 31 35 44 51 53 61 68 73 79 80
73      01/20/1987,01 14 16 22 23 27 29 30 34 36 39 40 43 44 62 64 67 71 73 76
74      01/19/1987,01 07 10 14 16 17 20 25 31 33 37 39 43 45 46 60 70 72 77 80
75      01/16/1987,02 07 10 12 24 27 30 32 36 37 41 43 47 60 61 62 64 69 72 78
76      01/15/1987,02 04 08 10 15 23 25 26 28 31 36 38 49 51 57 59 72 74 78 80
77      01/14/1987,05 07 08 25 27 28 32 36 38 47 51 52 53 55 58 59 60 72 74 77
78      01/13/1987,04 11 14 19 22 23 26 34 40 42 43 45 58 60 63 64 72 74 77 79
79      01/12/1987,05 12 15 18 20 25 26 33 34 35 48 49 52 54 57 62 65 69 71 76
```

Source Code: Src/28/Lottery_Pick_10_Winning_Numbers__Beginning_1987.csv

Design and implement a program which can determine the distribution of the numbers between 1 and 80 in this file. The input can be provided via stdin, gzipped, or as is.

Your program should be called like:

```
% cat Lottery_Pick_10_Winning_Numbers__Beginning_1987.csv | java NumberCounter
% java NumberCounter Lottery_Pick_10_Winning_Numbers__Beginning_1987.csv
% java NumberCounter Lottery_Pick_10_Winning_Numbers__Beginning_1987.csv.gz
```

My output looks like:

```
java NumberCounter Lottery_Pick_10_Winning_Numbers__Beginning_1987.csv
1:     3047   2:     3100   3:     3124   4:     3119
5:     3048   6:     3101   7:     2950   8:     3091
9:     3168   10:    3097   11:    3024   12:    3021
13:    3041   14:    3112   15:    3132   16:    3128
17:    3028   18:    3099   19:    2998   20:    3047
21:    3112   22:    3037   23:    3108   24:    3115
25:    2970   26:    3101   27:    3070   28:    3053
29:    3054   30:    3151   31:    3089   32:    3065
33:    3051   34:    3045   35:    3059   36:    3050
37:    3057   38:    3101   39:    3099   40:    3042
41:    3104   42:    3028   43:    3071   44:    3052
45:    3095   46:    3066   47:    3032   48:    2987
49:    3059   50:    3058   51:    3053   52:    3126
53:    3089   54:    3066   55:    2997   56:    3039
57:    3147   58:    3212   59:    3051   60:    3074
61:    3056   62:    3132   63:    3087   64:    3073
65:    3045   66:    3034   67:    2993   68:    3113
69:    3090   70:    2996   71:    3043   72:    3096
73:    3029   74:    3109   75:    2968   76:    3077
77:    3024   78:    3130   79:    3067   80:    3088
```

Your output does not have to be identical to mine, but similar.

Your Work:

Requirements:

• You can not pre-process the file.

• The input can be provided via stdin, gzipped, or as is. This is an example how your program should be called:

```
% java NumberCounter Lottery_Pick_10_Winning_Numbers__Beginning_1987.csv
1:     3047   2:     3100   3:     3124
 ...
% java NumberCounter Lottery_Pick_10_Winning_Numbers__Beginning_1987.csv.gz
1:     3047   2:     3100   3:     3124
 ...
% cat Lottery_Pick_10_Winning_Numbers__Beginning_1987.csv | java NumberCounter
1:     3047   2:     3100   3:     3124
 ...
```

• Your program must use one, and only one, try ressource statement.

• Your program may use more try statements.

• Your program should compile without warnings.

• You have to name the source code NumberCounter.java

Submission:

Submit your files via myCourses.

28.2. Homework 8.2 (10 Points)

Objective: Working with Files

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

The objective is to reading and processing a file.

Analyze the digits of pi based on even and odd in a visual representation. The provided program uses the colors

```
int red = Color.RED.getRGB();
int blue = Color.BLUE.getRGB();
```

You are free to change the colors to different values.

Your Work:

The data can be found: here (https://www.angio.net/pi/digits/pi1000000.txt)
Download the data from there.

You need to read, comprehend, and add to this program to complete this home work.

```
1       // original from: http://rosettacode.org/wiki/Pi_set#Java
2       // modified for color
3
4       import java.awt.Color;
5       import java.awt.Color;
6       import java.awt.Graphics;
7       import java.awt.image.BufferedImage;
8       import javax.swing.JFrame;
9       import java.util.Scanner;
10      import java.io.*;
11      import javax.imageio.ImageIO;
12      import java.util.zip.GZIPInputStream;
13
14      public class Visual extends JFrame {
15
16              private final int LENGTH_OF_SQUARE = 3;
17              private final int LENGTH                = 330;
18              private final int LENGTH_OF_WINDOW = LENGTH * LENGTH_OF_SQUARE
19              private BufferedImage theImage;
20              private String fileName = null;
21              Reader input;
22
23              public Visual() {
24                      super("Visual");
```

```
25
26                      setBounds(100, 100, LENGTH_OF_WINDOW, LENGTH_OF_WINDOW
27                      setResizable(false);
28                      setDefaultCloseOperation(EXIT_ON_CLOSE);
29              }
30              public Visual(String fileName) {
31                      this();
32                      this.fileName = fileName;
33              }
34
35              private char nextDigit(BufferedReader input)    {
36                      char buf[] = new char[1];
37                      /* add code here
38                         this method returns the next digit, which must be a
39                         Other characters must be discarded.
40                      */
41                      return buf[0];
42              }
43
44              private void saveImage(BufferedImage theImage)  {
45                      String suffix = "png";
46                      String outputFileName = fileName == null ? "output" :
47                      try {
48                              File outputfile = new File(outputFileName);
49                              ImageIO.write(theImage, suffix, outputfile);
50                      } catch (Exception e )  {
51                              e.printStackTrace();
52                      }
53
54              }
55              public void fillSquare(int xOrig, int yOrig, int color) {
56                      for (int x = 0; x < LENGTH_OF_SQUARE; x ++ )
57                              for (int y = 0; y < LENGTH_OF_SQUARE; y ++ )
58                                      theImage.setRGB(xOrig + x, yOrig + y ,
59              }
60              public void createImage()       {
61                      theImage = new BufferedImage(getWidth(), getHeight(),
62                                      BufferedImage.TYPE_INT_RGB);
63                      int red = Color.RED.getRGB();           // you might l
64                      int blue = Color.BLUE.getRGB();         // you might l
65                      int colorUsed;
66
67                      try (
68                              BufferedReader input
69                                      /* add code here
70                                              fileName is null or ending wit
71                                      */
72                          ) {
73                              for (int y = 0; y < getHeight(); y += LENGTH_O
74                                      for (int x = 0; x < getWidth(); x += L
75                                              char digit = nextDigit(input);
76                                              fillSquare(x, y,  digit % 2 ==
77                                      }
78
```

```
79                              }
80                      } catch ( Exception e ) {
81                              e.printStackTrace();
82                              System.exit(0);
83                      }
84              repaint();
85              saveImage(theImage);
86              System.exit(0);
87          }
88
89          @Override
90          public void paint(Graphics g) {
91              g.drawImage(theImage, 0, 0, this);
92          }
93
94          public static void main(String[] args) {
95              Visual aVisual = new Visual(args.length == 1 ? args[0] : null
96              aVisual.setVisible(true);
97              aVisual.createImage();
98          }
99      }
```

Source Code: Src/28/Visual.java

My program creates the following output:

Your output does not have to be identical to mine, but similar.

Requirements:

• You can not pre-process the file.

• The input can be provided via command-line argument or via standard in. The input can be provided via stdin, gzipped, or as is.

Your program should be called like:

```
% cat pi6.txt | java Visual
% java Visual pi6.txt
% java Visual pi6.txt.gz
```

• Your program must use one, and only one, try resource statement.

• Your program may use more try statements.

• Your program should compile without warnings.

• You have to name the source code NumberCounter.java

Submission:

Submit your files via myCourses.

28.3. Homework 8.3 (10 Points)

Objective: Working with Object I/O Streams

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your

solution during the grading session

Homework Description:

You have write a Password class which can be serialized, a Password write class and a Password read class. The password, and mybe more needs to be stored in a file which can be read by a program, and then displayed. The PasswordWrite.class write a file 1234.ser, which contains the serialized password object. The PasswordRead.class must be able to detect if the password in the servialized file has been modified after the file was written.

A normal execution would looks like this:

```
javac PasswordWrite.java
javac PasswordRead.java
java PasswordWrite 1234.ser
# now the file is copied on a usb stick, given to including PasswordRead.class
# you can now retrive the password with
java PasswordRead 1234.ser
The password is: abcdef
```

The PasswordRead.class must be able to detect if the password was modified in 1234.ser. If the password was modified this should be detected with a high probability. The execution should be like this:

```
javac PasswordWrite.java
javac PasswordRead.java
java PasswordWrite 1234.ser
# now the file is copied on a usb stick, given to and PasswordRead.class
# somebody modifies the password:
modifyIt
cmp modified.ser 1234.ser 2> /dev/null
modified.ser 1234.ser differ ...
cp modified.ser 1234.ser  # copies the file on the usb stick
# you can now not retrive the password with
java PasswordRead 1234.ser
The password is:
```

Explanation:

You have to design and implement the following classes:

```
Password.java       # The object
PasswordWrite.java  # this file is know only to the developer
PasswordRead.java       # this file is know only to the developer
PasswordRead.class      # this file is known to all
1234.ser                # this file is known to all
a program               # which can modify the password in 1234.ser
```

Your Work:

Your work is to implement the classes described above. The program which can modify 1234.ser can be a java program, or a shell script. Usefull for a shell script solution might be: od, dd, echo

Requirements:

• It is not required that you can can detect all possible modifications, but most.

• The modification must be done in such a way that the structure of the modified 1234.ser is intact, in other words PasswordRead.class must be able to read the the modified 1234.ser sucessfully.

• You have to submit all java files, the file which modifies 1234.ser, and a read me how to compile and execute the code so such the required behavior can be seen.

Example:

An example of a solution execution:

Submission:

Submit your files via myCourses.

Not Final

29. Homework 9

Posted: October/14/2020
Due: October/25/2020 24.00

The solutions for this homework are due October/25/2020 24.00. I recommend to submit at least one version of all homework solutions long before due date.

29.1. Homework 9.1 (10 Points)

Objective: Working with Threads

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Objective:

Grading:
Correctness: You can lose up to % if your solution is not correct
Quality: You can lose up to % if your solution is poorly designed
Testing: You can lose up to % if your solution is not well tested
Explanation: You can lose up to % if your solution if you can not explain your solution during the grading session

Homework Description:
The objective is to design and implement a multi threaded programs. This hw is a modification of hw 22.2. In hw 22.2 one single thread was used to determine if a number becomes palindromic within up to 3 iterations. For this hw you have to use one thread for each number your program is testing. The threads have to run in parallel.

Your Work:

You can start with your or my solution. I suggest you consider the modifications carefully before you start to write code.

The main loop in my original solution looks like:

```
for ( int numberToTest = START; numberToTest <= END; numberToTest ++ )  {
        int theNextnumber = 1;
        for ( int delayed = 1; (theNextnumber > 0 ) && ( delayed <= MAXIMUM_DE
                if ( delayed == 1 )
                        theNextnumber = delayedPalindrome(numberToTest, number
                else
                        theNextnumber = delayedPalindrome(numberToTest, theNex
        }
}
```

You have to create a new thread for each number (*numberToTest) you are testing.*

The execution of my program produces for these values

```
int START                = 78;
int END                  = 88;
int MAXIMUM_DELAYED      = 10;
```

the following output:

```
% java PalindromeUpdate
78:          delayed 4:   1353   +      3531   =      4884
79:          delayed 6:   14003  +      30041  =      44044
80:          delayed 1:   80     +      08     =      88
81:          delayed 1:   81     +      18     =      99
82:          delayed 2:   110    +      011    =      121
83:          delayed 1:   83     +      38     =      121
84:          delayed 2:   132    +      231    =      363
85:          delayed 2:   143    +      341    =      484
86:          delayed 3:   605    +      506    =      1111
87:          delayed 4:   1353   +      3531   =      4884
88:          delayed 6:   14003  +      30041  =      44044
```

Requirements:

• You have to create a new thread for each number you are testing.

• The output has to be sorted, based in the number which is analyzed. The variable name of this number based on the above code snippet is *numberToTest*.

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
 1      import java.lang.StringBuilder;
 2      import java.util.* ;
 3
 4      public class Palindrome extends Thread {
 5              static int GLOBAL_END                    = 88;
 6              static int MINIMUM_DELAYED      = 1;
 7              static int MAXIMUM_DELAYED      = 10;
 8              static String[] theResults;
 9              static List<Palindrome> theThreads = new ArrayList<Palindrome>
10              int END = 0;
11              int id;
12
13              int numberToTest;
14                      this.id    = id;
15                      this.END   = END;
16              }
17
18              public int  delayedPalindrome(int firstNumberofSequence, int t
19                      int rValue = 0;
20
21                      String        original       = Integer.toString(theNum
```

```
22                      StringBuilder aStringBuilder = new StringBuilder(origi
23                      String        lanigiro       = aStringBuilder.reverse
24                      int           rebmuNeht      = Integer.valueOf(lanigir
25
26                      int result = Integer.valueOf(new StringBuilder(origina
27                      String resultString  = Integer.toString(result);
28                      String resultStringReverse  = new StringBuilder(result
29                      rValue =  theNumber + rebmuNeht;                // mig
30
31                      String output = "";
32                      boolean notAlychrelNumber = ( ( MINIMUM_DELAYED  <= de
33                      boolean lastTry           = ( ( MINIMUM_DELAYED  <= de
34                      if  ( notAlychrelNumber || lastTry ) {
35                              if  ( notAlychrelNumber )       {
36                                      String reverseNumerFormat = "%0" + (""
37                                      String numberFormat = "%10d";
38                                      String stringFormat = "%4s";
39                                      String delayFormat = "%4d";
40                                      output += String.format(numberFormat,f
41                                      output += "delayed " + String.format(d
42                                      output += ":   ";
43                                      output += String.format(numberFormat,
44                                      output += String.format(stringFormat,
45                                      output += String.format(numberFormat,
46                                      output += String.format(stringFormat,
47                                      output += String.format(numberFormat,
48                                      theResults[id] = output;
49                                      rValue = 0;   //  meets requirement
50                              }
51                      }
52                      return rValue;
53              }
54
55              public void determineForOneNumber(int numberToTest)       {
56                      int theNextnumber = 1;
57                      int numberToTestMeetsRequirement = 0;           // mea
58                      int  delayed = 1;
59                      do {
60                              theNextnumber = delayedPalindrome(numberToTest
61                      } while  ( ( ! ( theNextnumber == numberToTestMeetsReq
62              }
63              public void run()       {
64                      while (  numberToTest <= END )  {
65                              determineForOneNumber(numberToTest);
66                              numberToTest = numberToTest + 1;
67                      }
68              }
69              public static void setUp(int soManyThreads) {
70                      int index = 0;
71                      int start = 0;
72                      int end =   0;
73                      int id = 0;
74                      while ( index < soManyThreads ) {
75                              Palindrome aPalindrome = new Palindrome(id ++,
```

```
76                              theThreads.add(aPalindrome);
77                              aPalindrome.start();
78                              index ++;
79                      }
80                      if (  end < GLOBAL_END )        {
81                              start = end + 1;
82                              end   = GLOBAL_END;
83                              Palindrome aPalindrome = new Palindrome(id, st
84                              theThreads.add(aPalindrome);
85                              aPalindrome.start();
86                      }
87              }
88              public static void waitForAll() {
89                      for ( int index = 0; index < theThreads.size(); index
90                              Palindrome aPalindrome = theThreads.get(index)
91                              try {
92                                      aPalindrome.join();
93                              } catch ( InterruptedException e )      {
94                                      e.printStackTrace();
95                              }
96                      }
97              }
98              public static void printResult()        {
99                              System.out.println(theResults[index]);
00                      }
01              }
02              public static void main( String args[] ) {
03                      waitForAll();
04                      printResult();
05              }
06      }
```

Source Code: Src/29_sol/Palindrome.java

29.2. Homework 9.2 (10 Points)

Objective: Working with Threads and File I/O

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Objective:

Grading:
Correctness: You can lose up to % if your solution is not correct
Quality: You can lose up to % if your solution is poorly designed
Testing: You can lose up to % if your solution is not well tested
Explanation: You can lose up to % if your solution if you can not explain your solution during the grading session

Homework Description:
The objective is to design and implement a multi threaded programs. This hw is a modification of hw 28.1. In hw 28.1 one single thread was used to determine the distribution of the numbers between 1 and 80. For this hw you have to divide the file in blocks and each block is analyzed by one thread. The threads have to run in parallel.

Your Work:

You can start with your or my solution. I suggest you consider the modifications carefully before you start to write code.

You can use the following information. The file consists of 12278 lines and each lines is 71 characters long. Assuming your program is using 3 threads, the following would be possible:

• the first thread determines the distribution of the numbers between 1 and 80 for the first n lines.

• the second thread determines the distribution of the numbers between 1 and 80 for the second n lines starting at line n + 1;

• the third thread determines the distribution of the numbers between 1 and 80 for the remaining lines

Each thread has to open and close its own IO communication endpoint. It must be possible to specify the filename, and how many threads will be used via command-line.

The execution of my program produces the following output:

```
% java NumberCounterMultiThreaded.java  10 Lottery_Pick_10_Winning_Numbers__Be
1:     3047   2:     3100   3:     3124   4:     3119
5:     3048   6:     3101   7:     2950   8:     3091
9:     3168   10:    3097   11:    3024   12:    3021
13:    3041   14:    3112   15:    3132   16:    3128
17:    3028   18:    3099   19:    2998   20:    3047
21:    3112   22:    3037   23:    3108   24:    3115
25:    2970   26:    3101   27:    3070   28:    3053
29:    3054   30:    3151   31:    3089   32:    3065
33:    3051   34:    3045   35:    3059   36:    3050
37:    3057   38:    3101   39:    3099   40:    3042
41:    3104   42:    3028   43:    3071   44:    3052
```

```
45:    3095   46:    3066   47:    3032   48:    2987
49:    3059   50:    3058   51:    3053   52:    3126
53:    3089   54:    3066   55:    2997   56:    3039
57:    3147   58:    3212   59:    3051   60:    3074
61:    3056   62:    3132   63:    3087   64:    3073
65:    3045   66:    3034   67:    2993   68:    3113
69:    3090   70:    2996   71:    3043   72:    3096
73:    3029   74:    3109   75:    2968   76:    3077
77:    3024   78:    3130   79:    3067   80:    3088
```

Your output has to be similar to mine and sorted based on the number 1 — 80.

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
 1      // https://data.ny.gov/Government-Finance/Lottery-Pick-10-Winning-Numb
 2      // Choose ten numbers from 1 to 80.
 3      // File sttructure
 4      // Draw Date,Winning Numbers
 5      // 09/30/2020,01 04 07 10 16 19 28 34 36 40 46 47 51 52 59 62 64 67 72
 6
 7      import java.util.zip.GZIPInputStream;
 8      import java.io.*;
 9
10      public class NumberCounterMultiThreaded extends Thread {
11
12
13              static final int    soManyNumbersAreUsed = 80;
14              static int[]  combinedAllNumbersOfAllThreads = new int[soManyN
15              static final int    soManyColumns = 2;
16              static int soManyThreads;
17              static int soManyLines = 12278;
18              static int soManyBytesPerLine = 71;
19              static NumberCounterMultiThreaded[] allThreads;
20              static int delta;
21              static int lastLineRead = -1;
22
23              static int id;
24              static String fileName = null;
25              Reader input;
26              int[]  theNumbers = new int[soManyNumbersAreUsed + 1 ];
27              int startPosition;
28              int endPosition;
29              int firstLine;
30              int lastLine;
31
32              public NumberCounterMultiThreaded(int id, String fileName) {
33                      this.id = id;
34                      this.fileName = fileName;
35                      firstLine = id * ( soManyLines / soManyThreads );
```

```
36                      if ( id == soManyThreads -1 )   {
37                              lastLine = soManyLines;
38                      } else {
39                              lastLine  = ( id + 1 ) * ( soManyLines / soMar
40                              lastLine  = lastLine > soManyLines ? soManyLir
41                      }
42
43                      startPosition = firstLine * soManyBytesPerLine;
44                      endPosition  = ( lastLine  *soManyBytesPerLine  ) - 1;
45
46              }
47              static private void printResult()       {
48                      for ( int index = 0; index < combinedAllNumbersOfAllTh
49                              for ( int column = 0; column <= 1 + soManyColu
50                                      int whichNumber = 1 + index * soManyCo
51                                      System.out.print(whichNumber + ":
52                              }
53                              System.out.println();
54                      }
55              }
56              private void processOneLine(String theLine)     {
57                      // System.out.println(theLine );
58                      theLine = theLine.substring(theLine.indexOf(",") + 1,
59                      String[] individualContent = theLine.split("\\s");
60                      for ( int index = 0; index < individualContent.length;
61                              theNumbers[Integer.valueOf(individualContent[i
62                      }
63
64              }
65              private void processInput()     {
66
67                      try (
68                              BufferedReader input = new BufferedReader(
69                                                      new InputStreamReac
70                                                          new FileInputSt
71                                                          )
72                                                  );
73                          ) {
74
75                              input.skip(startPosition);
76                              // String oneLine = input.readLine();
77                              for ( int index = firstLine; index < lastLine;
78                                      String oneLine = input.readLine();
79                                      processOneLine(oneLine);
80                              }
81                              // printResult();
82                      } catch ( Exception e ) {
83                              e.printStackTrace();
84                              System.exit(0);
85                      }
86              }
87              public void run()       {
88                      processInput();
89              }
```

```
90              public static void startAllTreads(String fileName)      {
91                      for ( int index = 0; index < soManyThreads; index ++ )
92                              allThreads[index] = new NumberCounterMultiThre
93                              (allThreads[index]).start();
94                      }
95              }
96              public static void waitForAllTreadsToEnd()      {
97                      for ( int index = 0; index < soManyThreads; index ++ )
98                              try     {
99                                      allThreads[index].join();
00                              } catch ( InterruptedException e )      {
01                                      e.printStackTrace();
02                                      System.exit(1);
03                              }
04                      }
05              }
06              public static void distributionCollection()     {
07                      for ( int index = 0; index < soManyThreads; index ++ )
08                              for ( int allUsedNumbers = 1; allUsedNumbers <
09                                      combinedAllNumbersOfAllThreads[allUsed
10                                                      allThreads[index].theN
11                      }
12              }
13              public static void setUp(String soManyThreadsRequested, String
14                      soManyThreads = Integer.valueOf(soManyThreadsRequested
15                      allThreads = new NumberCounterMultiThreaded[soManyThre
16                      startAllTreads(fileName);
17                      waitForAllTreadsToEnd();
18                      distributionCollection();
19                      printResult();
20
21              }
22              public static void main(String[] args) {
23                      setUp(args[0], args[1]);
24                      // aNumberCounterMultiThreaded.processInput();
25              }
26      }
```

Source Code: Src/29_sol/NumberCounterMultiThreaded.java

29.3. Homework 9.3 (10 Points)

Objective: Working with Threads and synchronized lists

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Objective:

Grading:
Correctness: You can lose up to % if your solution is not correct
Quality: You can lose up to % if your solution is poorly designed
Testing: You can lose up to % if your solution is not well tested
Explanation: You can lose up to % if your solution if you can not explain your solution during the grading session

Homework Description:

This hw is about a modification of your solution for hw 29.1. The objective is to design and implement multi threaded programs. In hw 29.1 one single thread was used to determine if a number becomes palindromic within up to 3 iterations. For this hw you have to use one thread for a block of numbers number. These threads have to run in parallel and you can only to print the numbers which meet the additional requirement: The number has to become palindromic between MINIMUM_DELAYED and MAXIMUM_DELAYED. The output below is shown for 5 and 9.

Your Work:

It must be possible to specify the numbers of threads on the command-line. The solution for hw 22.1 prints the results as soon as the restult is know. This strategy will not work for this hw, because the output has to be sorted. Therefore you have to store the results. You can not store the results in an array, because the results are not continues; see output below.

The following statement:

```
List<Result> theResults = Collections.synchronizedList(new ArrayList<Result>()
```

creates an list which allows multiple threads to safely modify the list. You have to use a list created with the above statement to store the results. The list needs to be sorted before the result can be printed.

This code snippet:

```
theResults.add( new Result(firstNumberofSequence, output) );
```

adds a new Result object for firstNumberofSequence to the list. The content of output is the string my solution for hw 22.1 prints.

The execution of my program produces for

```
int START                = 80;
int END                  = 90;
int MINIMUM_DELAYED      = 4;
int MAXIMUM_DELAYED      = 10;
```

the following output:

```
% java  Palindrome 4
        80:    delayed    8:        44044   +     44044   =     88088
        81:    delayed    8:        79497   +     79497   =    158994
        82:    delayed    4:          242   +       242   =       484
        83:    delayed    4:          484   +       484   =       968
        84:    delayed    6:         4884   +      4884   =      9768
        86:    delayed    4:         1111   +      1111   =      2222
        87:    delayed    5:         4884   +      4884   =      9768
        88:    delayed    7:        44044   +     44044   =     88088
        90:    delayed    8:        79497   +     79497   =    158994
```

Note: It is required that your output is sorted in the same was as my output is sorted.

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
 1      import java.lang.StringBuilder;
 2      import java.util.* ;
 3
 4      class   Result  implements Comparable<Result> {
 5              String output;
 6              int theNumber;
 7              Palindrome aPalindrome;
 8
 9              Result(int theNumber, String output)        {
10                      this.theNumber    = theNumber;
11                      this.output       = output;
12              }
13              public String toString()        {
14                      return  output;
15              }
16              public int compareTo(Result aResult)    {
17                      return theNumber - aResult.theNumber;
18              }
19      }
20      public class Palindrome extends Thread {
21              static int GLOBAL_END                   = 90;
22              static int MINIMUM_DELAYED      = 4;
23              static int MAXIMUM_DELAYED      = 10;
24              static List<Result> theResults = Collections.synchronizedList
25              static List<Palindrome> theThreads = new ArrayList<Palindrome>
26              int END = 0;
```

```
27
28              int numberToTest;
29              Result aResult;
30
31                       this.END = END;
32              }
33
34              public int  delayedPalindrome(int firstNumberofSequence, int t
35                      int rValue = 0;
36
37                      String        original       = Integer.toString(theNum
38                      StringBuilder aStringBuilder = new StringBuilder(origi
39                      String        lanigiro       = aStringBuilder.reverse(
40                      int           rebmuNeht      = Integer.valueOf(lanigir
41
42                      int result = Integer.valueOf(new StringBuilder(origina
43                      String resultString  = Integer.toString(result);
44                      String resultStringReverse  = new StringBuilder(result
45                      rValue =  theNumber + rebmuNeht;                // mig
46
47                      String output = "";
48                      boolean notAlychrelNumber = ( ( MINIMUM_DELAYED  <= de
49                      boolean lastTry           = ( ( MINIMUM_DELAYED  <= de
50                      if  ( notAlychrelNumber || lastTry ) {
51                              if  ( notAlychrelNumber )       {
52                                      String reverseNumerFormat = "%0" + ("'
53                                      String numberFormat = "%10d";
54                                      String stringFormat = "%4s";
55                                      String delayFormat = "%4d";
56                                      output += String.format(numberFormat,f
57                                      output += "delayed " + String.format(d
58                                      output += ":   ";
59                                      output += String.format(numberFormat,
60                                      output += String.format(stringFormat,
61                                      output += String.format(numberFormat,
62                                      output += String.format(stringFormat,
63                                      output += String.format(numberFormat,
64                                      theResults.add( new Result(firstNumber
65                                      rValue = 0;   //  meets requirement
66                              }
67                      }
68                      return rValue;
69              }
70
71              public void determineForOneNumber(int numberToTest)       {
72                      int theNextnumber = 1;
73                      int numberToTestMeetsRequirement = 0;           // mea
74                      int  delayed = 1;
75                      do {
76                              theNextnumber = delayedPalindrome(numberToTest
77                      } while  ( ( ! ( theNextnumber == numberToTestMeetsRec
78              }
79              public void run()       {
80                      while (  numberToTest <= END )  {
```

```
81                              determineForOneNumber(numberToTest);
82                              numberToTest = numberToTest + 1;
83                      }
84              }
85              public static void setUp(int soManyThreads) {
86                      int index = 0;
87                      int start = 0;
88                      int end =   0;
89                      while ( index < soManyThreads ) {
90                              Palindrome aPalindrome = new Palindrome(start,
91                              theThreads.add(aPalindrome);
92                              aPalindrome.start();
93                              index ++;
94                      }
95                      if (  end < GLOBAL_END )        {
96                              start = end + 1;
97                              end   = GLOBAL_END;
98                              Palindrome aPalindrome = new Palindrome(start,
99                              theThreads.add(aPalindrome);
00                              aPalindrome.start();
01                      }
02              }
03              public static void waitForAll() {
04                      for ( int index = 0; index < theThreads.size(); index
05                              Palindrome aPalindrome = theThreads.get(index)
06                              try {
07                                      aPalindrome.join();
08                              } catch ( InterruptedException e )      {
09                                      e.printStackTrace();
10                              }
11                      }
12              }
13              public static void printResult()        {
14                      Collections.sort(theResults);
15                      for ( int index = 0; index < theResults.size(); index
16                              Result aResult = theResults.get(index);
17                              System.out.println(aResult);
18                      }
19              }
20              public static void main( String args[] ) {
21                      waitForAll();
22                      printResult();
23              }
24      }
```

Source Code: Src/29_sol/Palindrome_3.java

Not Final

30. Homework 10

Posted: October/14/2020
Due: November/1/2020 24.00

The solutions for this homework are due November/1/2020 24.00. I recommend to submit at least one version of all homework solutions long before due date.

30.1. Homework 10.1 (10 Points)

Objective: Understanding multi thread programs

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Objective:

Grading:
Correctness: You can lose up to % if your solution is not correct
Quality: You can lose up to % if your solution is poorly designed
Testing: You can lose up to % if your solution is not well tested
Explanation: You can lose up to % if your solution if you can not explain your solution during the grading session

Homework Description:
The objective is to understand synchronized multi threaded programs.

Your Work:

Take a look at the following program:

```
 1      public class X extends Thread    {
 2          static Object o = new Object();
 3          static int counter = 0;
 4          int id;
 5          public X(int id)    {
 6              this.id = id;
 7              o       = new Object();
 8          }
 9          public void run () {
10              if ( id == 0 )  {
11                      new X(1).start();
12                      new X(2).start();
13                      return;
14              }
15              synchronized ( o ) {
16                      System.err.println(id + " --->");
17                      try {
18                              if ( counter == 0 )     {
19                                      counter = 1;
20                                      o.wait();
```

```
21                                  } else
22                                          o.notifyAll();
23                          }
24                          catch (  InterruptedException e ) {
25                          }
26                          System.err.println(id + " <---");
27                      }
28              }
29              public static void main (String args []) {
30                  new X(0).start();
31              }
32          }
33
```

Source Code: Src/30/X.java

This program can produce the following output:

```
1 --->
2 --->
2 <---
1 <---
```

Explain this output in a text file. Words like main thread, thread 1, thread 2, scheduler, etc. might be useful. You have to name your file 30_1a.txt.

Explain if or if not an other output(s) is/are possible. You have to name your file 30_1b.txt, 30_1c.txt, 30_1d.txt, etc. You have to modify the code so such this output is produced with high probability.

Requirements:

• You have to name the description files 30_1a.txt, and if needed 30_1b.txt.

• You have to submit a modified version of X.java which produces this output, if an other output is possible.

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

30.2. Homework 10.2 (10 Points)

Objective: Understanding multi thread programs

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

```
 1      public class OrganizedThreads extends Thread    {
 2              private int             id;
 3              private Object          first;
 4              private Object          second;
 5              static final int        MAX = 3;
 6              static Object           all[] = new Object[MAX];
 7              static int              createdSoFar = 1;
 8              static int              lastSeen = 0;
 9
10              static {
11                      for ( int index = 0; index < MAX; index ++ )
12                              all[index] = new Object();
13              }
14
15              public OrganizedThreads (int id, Object first, Object second)
16                      this.id         = id;
17                      this.first      = first;
18                      this.second     = second;
19              }
20              public void test () {
21                      if ( lastSeen + 1 != id )       {
22                              System.out.println("Something went wrong. Last
23                              System.exit(1);
24                      }
25                      lastSeen = ( lastSeen + 1 ) % MAX;
26
27              }
28              public void run () {
29                      while ( true )  {
30
31                              try { sleep(300); } catch (  InterruptedExcept
32                              synchronized ( first ) {
33                                      synchronized ( second ) {
34                                              second.notify();
35                                              test();
36                                              System.out.println(id);
37                                              try {
38                                                      if ( createdSoFar <= N
39                                                              createdSoFar++
40                                                              ( new Organize
41                                                      }
42                                              } catch ( Exception e ) { }
```

```
43                                          }
44                                          try {
45                                                  first.wait();
46                                          } catch ( Exception e ) { }
47                                  }
48                          }
49              }
50              public static void main (String args []) {
51                      new OrganizedThreads(1, all[0], all[1]).start();
52              }
53      }
```

Source Code: Src/30/OrganizedThreads.java

Your Work:

Explain if or if not an other output(s) is/are possible. You have to name your file 30_2b.txt, 30_2c.txt, 30_2d.txt, etc. You have to modify the code so such this output is produced with high probability.

Requirements:

• You have to name the description files 30_2.txt

Submission:

Submit your files via myCourses.

30.3. Homework 10.3 (10 Points)

Objective: Working with Threads and synchronized lists

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Objective:

Grading:
Correctness: You can lose up to % if your solution is not correct
Quality: You can lose up to % if your solution is poorly designed
Testing: You can lose up to % if your solution is not well tested
Explanation: You can lose up to % if your solution if you can not explain your solution during the grading session

Homework Description:

This hw is about a modification of your solution for hw 27.2. Your solution to hw 27.2 was a storage class. Modify this storage class so such you can use it instead of

```
List<Result> theResults = Collections.synchronizedList(new ArrayList<Result>()
```

for hw 29.3. You might need to add more methods to your storage class, because

```
public static void printResult()        {
        Collections.sort(theResults);
        for ( int index = 0; index < theResults.size(); index ++ ) {
                Result aResult = theResults.get(index);
                System.out.println(aResult);
        }
}
```

might not work, depending on your implementation.

Your Work:

You have to convert your Storage class to a thread safe storage environment. You have to synchronize the minimum amount of code.

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

Palindrome.java

```
 1      import java.lang.StringBuilder;
 2      import java.util.* ;
 3
 4
 5      class   Result  implements Comparable<Result> {
 6              String output;
 7              int theNumber;
 8              Palindrome aPalindrome;
 9
10              Result(int theNumber, String output)        {
11                      this.theNumber    = theNumber;
12                      this.output       = output;
13              }
14              public String toString()        {
15                      return  output;
16              }
17              public int compareTo(Result aResult)    {
18                      return theNumber - aResult.theNumber;
19              }
20      }
21      public class Palindrome extends Thread {
22              static int GLOBAL_END                   = 88;
23              static int MINIMUM_DELAYED      = 1;
24              static int MAXIMUM_DELAYED      = 10;
25              static SortedStorage<Result> theResults = new SortedStorage<Re
26              static StorageInterface<Result> theResult = new SortedStorage<
27              static List<Palindrome> theThreads = new ArrayList<Palindrome>
28              int END = 0;
29
30              int numberToTest;
31              Result aResult;
32
33                       this.END = END;
34              }
35
36              public int  delayedPalindrome(int firstNumberofSequence, int t
37                      int rValue = 0;
38
39                      String        original       = Integer.toString(theNum
40                      StringBuilder aStringBuilder = new StringBuilder(origi
41                      String        lanigiro       = aStringBuilder.reverse(
42                      int           rebmuNeht      = Integer.valueOf(lanigir
43
44                      int result = Integer.valueOf(new StringBuilder(origina
45                      String resultString  = Integer.toString(result);
46                      String resultStringReverse  = new StringBuilder(result
47                      rValue =  theNumber + rebmuNeht;                // mig
48
49                      String output = "";
50                      boolean notAlychrelNumber = ( ( MINIMUM_DELAYED  <= de
51                      boolean lastTry           = ( ( MINIMUM_DELAYED  <= de
52                      if  ( notAlychrelNumber || lastTry ) {
53                              if  ( notAlychrelNumber )       {
54                                      String reverseNumerFormat = "%0" + ("
```

```
55                                      String numberFormat = "%10d";
56                                      String stringFormat = "%4s";
57                                      String delayFormat = "%4d";
58                                      output += String.format(numberFormat,f
59                                      output += "delayed " + String.format(d
60                                      output += ":   ";
61                                      output += String.format(numberFormat,
62                                      output += String.format(stringFormat,
63                                      output += String.format(numberFormat,
64                                      output += String.format(stringFormat,
65                                      output += String.format(numberFormat,
66                                      theResults.add( new Result(firstNumber
67                                      rValue = 0;   //  meets requirement
68                              }
69                      }
70                      return rValue;
71              }
72
73              public void determineForOneNumber(int numberToTest)       {
74                      int theNextnumber = 1;
75                      int numberToTestMeetsRequirement = 0;           // mea
76                      int  delayed = 1;
77                      do {
78                              theNextnumber = delayedPalindrome(numberToTest
79                      } while  ( ( ! ( theNextnumber == numberToTestMeetsReq
80              }
81              public void run()       {
82                      while (  numberToTest <= END )  {
83                              determineForOneNumber(numberToTest);
84                              numberToTest = numberToTest + 1;
85                      }
86              }
87              public static void setUp(int soManyThreads) {
88                      int index = 0;
89                      int start = 0;
90                      int end =   0;
91                      while ( index < soManyThreads ) {
92                              Palindrome aPalindrome = new Palindrome(start,
93                              theThreads.add(aPalindrome);
94                              aPalindrome.start();
95                              index ++;
96                      }
97                      if (  end < GLOBAL_END )        {
98                              start = end + 1;
99                              end   = GLOBAL_END;
00                              Palindrome aPalindrome = new Palindrome(start,
01                              theThreads.add(aPalindrome);
02                              aPalindrome.start();
03                      }
04              }
05              public static void waitForAll() {
06                      for ( int index = 0; index < theThreads.size(); index
07                              Palindrome aPalindrome = theThreads.get(index)
08                              try {
```

```
09                                      aPalindrome.join();
10                              } catch ( InterruptedException e )      {
11                                      e.printStackTrace();
12                              }
13                      }
14              }
15              public static void printResult()        {
16                      // Collections.sort(theResults);
17      // System.out.println("printResult: " + theResults);
18                      List<Result> theResultsAsList = theResults.getList();
19                      while ( ! theResultsAsList.isEmpty() ) {
20                              Result aResult = theResultsAsList.get(0);
21                              theResultsAsList.remove(0);
22                      System.out.println(aResult);
23                      }
24              }
25              public static void main( String args[] ) {
26                      waitForAll();
27                      printResult();
28              }
29      }
```

Source Code: Src/30_sol/Palindrome.java

The interface:

```
1       public interface StorageInterface<E extends Comparable<E>> {
2          boolean add(E x);
3          boolean find(E x);
4          boolean includesNull();
5          boolean delete(E x);
6       }
```

Source Code: Src/30_sol/StorageInterface.java

The storage:

```
1       import java.util.* ;
2
3
4       public class SortedStorage<E extends Comparable<E>> implements Storage
5
6               private int size = 0;
7
8               private class Node<E>   {
9                       Node()  {
10                      }
11                      Node(E element) {
12                              payLoad = element;
13                              counter = 1;
14                              left = right = null;
15                      }
16                      private Node<E> left;
```

```
17                      private Node<E> right;
18                      E payLoad= null;
19                      int counter = 0;
20              }
21
22              Node<E> root;
23              int soManyNulls = 0;
24
25              public SortedStorage( ) {
26                      root = null;
27              }
28              synchronized public int howManyNulls()  {
29                      return soManyNulls;
30              }
31              public boolean includesNull()   {
32                      return soManyNulls > 0;
33              }
34              synchronized public boolean delete( E element)  {
35                      boolean rValue = false;
36                      if ( element == null )  {
37                              if ( soManyNulls > 0 )  {
38                                      soManyNulls --;
39                                      rValue = true;
40                              } else
41                                      rValue = false;
42                      } else {
43                              rValue = deleteElementInTree(element);
44                      }
45                      return rValue;
46              }
47              private boolean deleteElementInTree(E element)  {
48                      boolean rValue = false;
49                      Node<E> aNode = null;
50                      if ( root == null )
51                              rValue = false;
52                      else {
53                              aNode = findThisElementInTree(root, element);
54                              if ( aNode == null )
55                                      rValue = false;
56                              else {
57                                      aNode.counter--;
58                                      rValue = true;
59                                      if ( aNode.counter == 0 )       {
60                                              root = deleteThisElementInTree
61                                      }
62                              }
63                      }
64                      return ( rValue );
65              }
66              private Node<E> minimumElement(Node<E> thisNode) {
67                      if (thisNode.left == null)
68                              return thisNode;
69                      else {
70                              return minimumElement(thisNode.left);
```

```
71                      }
72              }
73
74              private Node<E> deleteThisElementInTree(Node<E> root, E payLoa
75                      if ( root == null )
76                              return null;
77                      if ( root.payLoad.compareTo(payLoad) > 0 ) {    // see
78                              root.left = deleteThisElementInTree(root.left,
79                      } else if ( root.payLoad.compareTo(payLoad) < 0 ) {
80                              root.right = deleteThisElementInTree(root.righ
81                      } else {
82                              if ( (root.left != null) && (root.right != nul
83                                      Node<E> tmp = root;
84                                      Node<E> minimumNodeOnRight = minimumEl
85                                      root.payLoad = minimumNodeOnRight.payL
86                                      root.counter = minimumNodeOnRight.coun
87                                      root.right = deleteThisElementInTree(r
88                              } else if (root.left != null) {
89                                      root = root.left;
90                              } else if (root.right != null) {
91                                      root = root.right;
92                              } else {
93                                      root = null;
94                              }
95                      }
96                      return root;
97              }
98              synchronized public boolean find(E element)     {
99      /*
00                      return ( ( ( element == null ) && ( howManyNulls() > 0
01                              findElementInTree(root, element) );
02      */
03                      if ( ( element == null ) && ( howManyNulls() > 0 ) )
04                              return true;
05                      else
06                              return findElementInTree(root, element);
07              }
08              private boolean findElementInTree(Node<E> node, E element)
09                      if ( element == null )
10                              return false;
11                      Node<E> resultNode =  findThisElementInTree(node, elem
12                      return  ( resultNode != null );
13
14              }
15              private Node<E> findThisElementInTree(Node<E> node, E element)
16                      if ( node == null )     {
17                              return null;
18                      } else if ( node.payLoad.compareTo(element) == 0)
19                              return node;
20                      } else if ( node.payLoad.compareTo(element) > 0 ) {
21                              return findThisElementInTree(node.left
22                      } else {
23                              return findThisElementInTree(node.righ
24                      }
```

```
25
26              }
27              synchronized public boolean add(E element)      {
28                      try { Thread.sleep(1000); } catch (  InterruptedExcept
29                      boolean rValue;
30                      if (  find(element) )   {
31                              rValue = false;
32                      } else {
33                              if ( element == null )
34                                      soManyNulls++;
35                              else
36                                      addElementToTree(element);
37                              rValue = true;
38                      }
39                      return rValue;
40              }
41              private void addElementToTree(E element)  {
42                      if ( root == null )     {
43                              root = new Node<E>(element);
44                      } else  {
45                              addThisElementToTree(root, element);
46                      }
47              }
48              private void addThisElementToTree(Node<E> node, E element)  {
49                      if (  node.payLoad.compareTo(element) == 0 )
50                              node.counter ++;
51                      else if ( node.payLoad.compareTo(element) > 0 ) {  //
52                              if ( node.left == null )        {
53                                      node.left = new Node<E>(element);
54                              } else
55                                      addThisElementToTree(node.left, elemer
56                      } else {
57                              if ( node.right == null )       {
58                                      node.right = new Node<E>(element);
59                              } else
60                                      addThisElementToTree(node.right, eleme
61                      }
62              }
63              synchronized public List<E>getList()  {
64                      List<E> theResultsAsList = new ArrayList<E>();
65
66                      fillList(root, theResultsAsList);
67                      return theResultsAsList;
68              }
69              private void fillList(Node<E> node, List<E> theResultsAsList)
70                      if ( node != null ) {
71                              if ( node.left != null )
72                                      fillList(node.left, theResultsAsList);
73
74                              theResultsAsList.add(node.payLoad);
75
76                              if ( node.right != null )
77                                      fillList(node.right, theResultsAsList)
78                      }
```

```
79              }
80              synchronized private String parse(Node<E> node)  {
81                      String rValue = "";
82                      if ( node != null ) {
83                              rValue = " ( ";
84                              if ( node.left == null )
85                                      rValue += "l: null ";
86                              else
87                                      rValue += parse(node.left) + " ";
88
89                              rValue += node.payLoad + "/" + node.counter +
90
91                              if ( node.right == null )
92                                      rValue += "r:  null ";
93                              else
94                                      rValue += "r: " + parse(node.right) +
95                              rValue = rValue + " ) ";
96                      }
97                      return rValue;
98      // Values stored: +l: +l: null 3/1 r:  null  5/1 r: +l: null 6/1 r:  r
99              }
00              synchronized public String toString()  {
01                      String rValue = "\n     includes so many null values =
02                      "\n     Values stored: " +  parse(root);
03                      return rValue;
04              }
05
06      }
07
```

Source Code: Src/30_sol/SortedStorage.java

Not Final

31. Homework 11

Posted: October/14/2020
Due: November/8/2020 24.00

The solutions for this homework are due November/8/2020 24.00. I recommend to submit at least one version of all homework solutions long before due date.

Snippet of the note from Provost Granberg to all faculty:

> Instead, during the final four weeks of instruction, I encourage faculty to identify opportunities to reduce your own workload and that of your students. Some faculty members are already taking this step by eliminating material they consider optional, reducing the number of assignments, or scheduling recharge days for their classes. Any of these options can provide both faculty and students with an opportunity to catch up and restore some energy.

In the spirit of her note you can choose which two of the three question you want to answer. All questions are worth 15 points. The maximum number of points for hw 11 is 30.

31.1. Homework 11.1 (15 Points)

Objective: Working with Threads

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Objective: The objective is to understand a synchronized multi threaded programs.

Grading:
Correctness: You can lose up to % if your solution is not correct
Quality: You can lose up to % if your solution is poorly designed
Testing: You can lose up to % if your solution is not well tested
Explanation: You can lose up to % if your solution if you can not explain your solution during the grading session

Homework Description:

You have to simulate and 4 station assembly line using one individual thread for each station.

Your Work:

You have to simulate the following assembly line:

Station 1 mounts tires on rims and send a tire at a time to station 2. Station 1 has an unlimited amount of tires and rims.

Station 2 takes one tire at a time and after it has four tires, it mounts the tires on a car and sends the car to station 3. Station 2 has only storage for 4 tires. Station 2 has an unlimited amount of tires and rims.

Station 3 takes one car at time and puts an engine in the car and sends the car to station 4. Station 2 has only storage for one car. Station 3 has an unlimited amount of engines.

Station 4 takes one car at time and puts 4 door at the car and sends the car out of the factory Station 2 has only storage for one car. Station 4 has an unlimited amount of doors.

Unfortunately every station is short one human and a station can only work if they have a complete crew, except Station 1. Station 1 has a complete crew.

In order to solve this problem, a team member from station 1, moves, after the task is completed, to station 2. The same human moves to station 3 after the stations 2 task is completed. The same human moves to station 4 afer the stations 3 task is completed. The same human moves back to station 1 afer the stations 4 task is completed, and station one can start their work.

The simulation ends after n cars have been produced. It is relevant that the assembly line will work in order. Meaning station i has to wait for station i - 1 to complete its production.

My program produces the following output for n == 2:

```
% java AssemblyLine
     Station 1: Mounts tires on rims and send 4 tires to station 2
          Station 2: Mounts tires on car and sends the car to station  3
               Station 3: Put engine in car and sends the car to station  4
                    Station 4: Add doors to car    and sends the car out the f
     Station 1: Mounts tires on rims and send 4 tires to station 2
          Station 2: Mounts tires on car and sends the car to station  3
               Station 3: Put engine in car   and sends the car to station  4
                    Station 4: Add doors to car    and sends the car out the f
%
```

Requirements:

• You have to name the file AssemblyLine.java.

• You have to show that the threads will execute in the deisred order.

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
 1      public class AssemblyLine extends Thread        {
 2
 3              private int             id;
 4              private Object          first;
 5              private Object          second;
 6              static final int        MAX = 4;
 7              static String[] stations  = { "Station 1: Mounts tires on rims
 8                                            "Station 2: Mounts tires on car
 9                                            "Station 3: Put engine in car ar
10                                            "Station 4: Add doors to car and
11              static Object           all[] = new Object[MAX];
12              static int              createdSoFar = 1;
13              static int              lastSeen = 0;
14              int                     soManyCreated = 2;
15              int                     soManyCreatedSofar;
16
17              static {
```

```
18                      for ( int index = 0; index < MAX; index ++ )
19                              all[index] = new Object();
20              }
21
22              public AssemblyLine (int id, Object first, Object second) {
23                      this.id         = id;
24                      this.first      = first;
25                      this.second     = second;
26                      if ( id > 1 )
27                              setDaemon(true);
28              }
29              public void test () {
30                      if ( lastSeen + 1 != id )       {
31                              System.out.println("Something went wrong. Last
32                              System.exit(1);
33                      }
34                      lastSeen = ( lastSeen + 1 ) % MAX;
35                      try { sleep(100); } catch ( Exception e ) { }
36
37              }
38
39              public static String padding(String theString, int soLongWillT
40                   return String.format("%1$" + soLongWillTheStringBe
41              }
42              public void run () {
43                      while ( true )  {
44                              synchronized ( first ) {
45                                      synchronized ( second ) {
46                                              second.notify();
47                                              test();
48                                              if ( ( id == 1 )
49                                                 ( ++soManyCreatedSofar > so
50                                                              return;
51                                              else
52                                                      System.out.println(pad
53                                              try {
54                                                      if ( createdSoFar <= M
55                                                              createdSoFar++
56                                                              ( new Assembly
57                                                      }
58                                              } catch ( Exception e ) { }
59                                      }
60                                      try {
61                                              first.wait();
62                                      } catch ( Exception e ) { }
63                              }
64                      }
65              }
66              public static void main (String args []) {
67                      new AssemblyLine(1, all[0], all[1]).start();
68              }
69      }
```

Source Code: Src/31_sol/AssemblyLine.java

31.2. Homework 11.2 (15 Points)

Objective: Working with synchronized Threads

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

You have to write a program which a modification of the producer consumer program. The storage is off fixed length. Your program has to create n producer and k consumer. Each producer is given an id, $1 \leq \text{id} \leq n$. Each consumer is given an id, $1 \leq \text{id} \leq k$. Each producer produces its id amount of object which need to be stored in a storage area. Each consumer consumer its id amount of object which need to be stored in a storage area. Your program has to check that only thread modifies the storage, and it has to monitor that the storage limitation are intact. The program has to terminate after the storage has been manipulated t times.

Your Work:

Each producer and consumer are represented as individual threads. You have to write code which self verifies:

• that only one thread, consumer or producer, manipulates the storage area at any given time.

• that the storage has after each operation no less than 0, and no more that capacity elements in it.

My storage can store up to 100 elements and supports the following methods:

void addItems(int id, int addTheseItems); void consumeItems(int id, int soManyToRemove);

The storage is using the following test method:

```
private void test()   {
              if ( soManThreadsAreExecutingInTheSynchronizedBlock != 1  )
                      System.out.println("soManThreadsAreExecutingInTheSynch
                      System.exit(0);
              }
              if ( soManyObjectsInStorage > N  )      {
                      System.out.println("overflow " + soManyObjectsInStorag
                      System.exit(0);
              }
              if ( soManyObjectsInStorage < 0 )       {
                      System.out.println("underflow " + soManyObjectsInStora
                      System.exit(0);
              }
              try {
                      Thread.sleep(1000);
              } catch ( Exception e ) { }
}
```

The execution of my program produces the following output:

```
% java ConsumerProducer -consumer 4 -producer 3 -storageSize 5 --maxNumberOfMa
1: AddItems: 1      1 <--- so many items in storage
       P: 1
1: AddItems: 1      2 <--- so many items in storage
       P: 1
1: AddItems: 1      3 <--- so many items in storage
       P: 1
1: AddItems: 1      4 <--- so many items in storage
       P: 1
1: AddItems: 1      5 <--- so many items in storage
       P: 1
1: AddItems: 1  waiting
4:     RemoveItems: 1      4 <--- so many items in storage
       C: 4
4:     RemoveItems: 1      3 <--- so many items in storage
       C: 4
4:     RemoveItems: 1      2 <--- so many items in storage
       C: 4
4:     RemoveItems: 1      1 <--- so many items in storage
       C: 4
4:     RemoveItems: 1      0 <--- so many items in storage
       C: 4
3:     RemoveItems: 1       waiting
2:     RemoveItems: 1       waiting
1:     RemoveItems: 1       waiting
3: AddItems: 1      1 <--- so many items in storage
       P: 3
3: AddItems: 1      2 <--- so many items in storage
       P: 3
3: AddItems: 1      3 <--- so many items in storage
       P: 3
3: AddItems: 1      4 <--- so many items in storage
       P: 3
3: AddItems: 1      5 <--- so many items in storage
       P: 3
3: AddItems: 1  waiting
2: AddItems: 1  waiting
 ...
```

Requirements:

• You have to name your program ConsumerProducer.java.

• It must be possible to specify on the command line the number of consumers, the number of producer, the storageSize, number of maxNumberOfManipulations.

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
 1
 2      import gnu.dtools.ritopt.IntOption;
 3      import gnu.dtools.ritopt.Options;
 4      import gnu.dtools.ritopt.Option;
 5      import java.util.ArrayList;
 6      import java.util.Vector;
 7      import java.util.Date;
 8      import java.util.List;
 9      import java.util.Collections;
10
11      public class ConsumerProducer extends Thread {
12
13
14       public static void main(String args[] )        {
15              IntOption soManyConsumerO             = new IntOption(5);
16              IntOption soManyProducerO             = new IntOption(5);
17              IntOption storageSizeO                = new IntOption(5);
18              IntOption maxNumberOfManipulationsO   = new IntOption(5);
19
20              Options options = new Options("ConsumerProducer");
21              options.register("consumer", soManyConsumerO);
22              options.register("producer", soManyProducerO);
23              options.register("storageSize", storageSizeO);
24              options.register("maxNumberOfManipulations", maxNumberOfManipu
25              options.process(args);
26              int soManyP = soManyProducerO.getValue();
27              int soManyC = soManyConsumerO.getValue();
28              int storageSize = storageSizeO.getValue();
29              int maxNumberOfManipulations = maxNumberOfManipulationsO.getVa
30
31              System.out.println("# producer = " + soManyP );
32              System.out.println("# consumer = " + soManyC );
33              System.out.println("# storageSize = " + storageSize );
34              System.out.println("# maxNumberOfManipulations = " + maxNumber
35              Storage theStorage = new Storage(storageSize, maxNumberOfManip
36
37              for (int id = 1 ; id <= soManyP ; id ++)        {
38                      new Producer(id, theStorage).start();
39              }
40              for (int id = 1 ; id <= soManyC ; id ++)        {
41                      new Consumer(id, theStorage).start();
42              }
43        }
44      }
45
46      class Storage {
47        int N;
48        static int soManThreadsAreExecutingInTheSynchronizedBlock = 0;
49        int soManyObjectsInStorage = 0;                       // counter, us
50        private Object sync = new Object();
51
52        private int soManyManipulations;
53        private int maxNumberOfManipulations;
54
```

```
55        Storage (int n, int maxNumberOfManipulations) {
56              this.N = n;
57              this.maxNumberOfManipulations = maxNumberOfManipulations;
58        }
59        void shallIterminate() {
60              soManyManipulations++;
61              if ( soManyManipulations > maxNumberOfManipulations )
62                      System.exit(0);
63        }
64        private void test()   {
65                      if ( soManThreadsAreExecutingInTheSynchronizedBlock !=
66                              System.out.println("soManThreadsAreExecutingIn
67                              System.exit(0);
68                      }
69                      if ( soManyObjectsInStorage > N  )      {
70                              System.out.println("overflow " + soManyObjects
71                              System.exit(0);
72                      }
73                      if ( soManyObjectsInStorage < 0 )       {
74                              System.out.println("underflow " + soManyObject
75                              System.exit(0);
76                      }
77                      try {
78                              Thread.sleep(1000);
79                      } catch ( Exception e ) { }
80        }
81        void addItems(int id, int addTheseItems)      {
82              shallIterminate();
83              synchronized ( sync ) {
84                      System.err.print(id + ": AddItems: " + addTheseItems )
85                      soManThreadsAreExecutingInTheSynchronizedBlock++;
86                      while  ( soManyObjectsInStorage + addTheseItems > N )
87                              try {
88                                      System.err.println("  waiting");
89                                      soManThreadsAreExecutingInTheSynchroni
90                                      sync.wait();
91                                      System.err.println("  woke up");
92                                      soManThreadsAreExecutingInTheSynchroni
93                              } catch ( InterruptedException e )      {
94                                      e.printStackTrace();
95                              }
96                      }
97
98                      soManyObjectsInStorage += addTheseItems;
99                      test();
00                      System.err.println("    " + soManyObjectsInStorage  +
01                      soManThreadsAreExecutingInTheSynchronizedBlock--;
02                      sync.notifyAll();
03                      System.err.println("    P: " + id );
04              }
05        }
06
07        int consumeItems(int id, int soManyToRemove)  {
08              shallIterminate();
```

```
09              synchronized ( sync ) {
10                      System.err.print( id + ":       RemoveItems: " + soMar
11                      soManThreadsAreExecutingInTheSynchronizedBlock++;
12                      while  ( soManyObjectsInStorage - soManyToRemove < 0 )
13                              try {
14                                      soManThreadsAreExecutingInTheSynchroni
15                                      System.err.println("     waiting ");
16                                      sync.wait();
17                                      System.err.println("     woke up" );
18                                      soManThreadsAreExecutingInTheSynchroni
19                              } catch ( InterruptedException e )      {
20                                      e.printStackTrace();
21                              }
22                      }
23                      soManyObjectsInStorage -= soManyToRemove;
24                      test();
25                      soManThreadsAreExecutingInTheSynchronizedBlock--;
26                      System.err.println("    " + soManyObjectsInStorage + "
27                      sync.notifyAll();
28                      System.err.println("    C: " + id );
29              }
30              return soManyObjectsInStorage;
31       }
32      }
33
34      class Consumer extends Thread {
35              int id;
36              Storage thisStorage;
37              Consumer(int id, Storage thisStorage)   {
38                      this.id = id;
39                      this.thisStorage = thisStorage;
40                      setName("Consumer: " + id );
41                      System.err.println("C: " + id );
42              }
43              public void run()       {
44                      while ( true ) {
45                              int consumed = thisStorage.consumeItems(id, 1)
46                      }
47              }
48      }
49
50
51      class Producer extends Thread {
52              int id;
53              Storage thisStorage;
54
55              Producer(int id, Storage thisStorage)   {
56                      this.id = id;
57                      this.thisStorage = thisStorage;
58                      setName("Producer: " + id );
59                      System.err.println("P: " + id );
60              }
61              public void run()       {
62                              // System.out.println("P id: " + id );
```

```
63                              // System.out.println("P SO_MANY: " + SO_MANY
64                      while ( true ) {
65                              thisStorage.addItems(id, 1);
66                      }
67              }
68      }
```

Source Code: Src/31_sol/ConsumerProducer.java

31.3. Homework 11.3 (15 Points)

Objective: The objective is to understand synchronized multi threaded programs.

Grading:
Correctness: You can lose up to % if your solution is not correct
Quality: You can lose up to % if your solution is poorly designed
Testing: You can lose up to % if your solution is not well tested
Explanation: You can lose up to % if your solution if you can not explain your solution during the grading session

Homework Description:

Your Work:

Take a look at the following program:

```
 1      public class XX extends Thread  {
 2              private String info;
 3              Object o_1;
 4              Object o_2;
 5              static boolean oneIsRunning = false;
 6
 7              public XX (String info, Object o_1, Object o_2, Object stop) {
 8                      this.info    = info;
 9                      this.o_1    = o_1;
10                      this.o_2    = o_2;
11                      this.stop    = stop;
12              }
13              public void run () {
14                      synchronized ( o_1 ) {
15                              System.out.println(info);
16                              try {
17                                      if ( ! oneIsRunning )   {
18                                              new XX("1", o_2, o_1, stop).st
19                                              oneIsRunning = true;
20                                              }
21                                      synchronized ( o_2 ) {
22                                              o_2.wait();
23                                              System.out.println("I will not
24                                      }
25                              } catch ( Exception e ) { }
26                      }
27              }
28              public static void main (String args []) {
29                      Object o_1 = new Object();
30                      Object o_2 = new Object();
31                      new XX("0", o_1, o_2, stop).start();
32              }
33      }
```

Source Code: Src/31/XX.java

This program can produce the following output:

Explain all possible executions of this program. You have to name your file 31.txt.

Requirements:

• You have to name the description files 31.txt.

• You have to submit a modified version of X.java which produces this output, if an other output is possible.

Submission:

Submit your files via myCourses.

Not Final

32. Homework 12

Posted: October/30/2020
Due: November/15/2020 24.00

The solutions for this homework are due November/15/2020 24.00. I recommend to submit at least one version of all homework solutions long before due date.

Snippet of the note from Provost Granberg to all faculty:

> Instead, during the final four weeks of instruction, I encourage faculty to identify opportunities to reduce your own workload and that of your students. Some faculty members are already taking this step by eliminating material they consider optional, reducing the number of assignments, or scheduling recharge days for their classes. Any of these options can provide both faculty and students with an opportunity to catch up and restore some energy.

In the spirit of her note you can choose which two of the three question you want to answer. All questions are worth 15 points. The maximum number of points for hw 11 is 30.

32.1. Homework 12.1 (15 Points)

Objective: Working with Threads

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Objective: The objective is to understand a synchronized multi threaded programs and explain why it does not work as expected.

Grading:
Correctness: You can lose up to % if your solution is not correct
Quality: You can lose up to % if your solution is poorly designed
Testing: You can lose up to % if your solution is not well tested
Explanation: You can lose up to % if your solution if you can not explain your solution during the grading session

Homework Description:

Take a look at the following program:

```
Source Code: Src/11/Philosopher.java
```

Your Work:

Is it guaranteed that every philosopher will be able to eat. Yes or no. Words like main thread, thread 1, thread 2, scheduler, etc. might be useful.

Requirements:

- You have to name your file 32.txt.

Submission:

Submit your files via myCourses.

32.2. Homework 12.2 (15 Points)

Objective: Working with synchronized Threads

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

The program in hw 12.1 night not work as it should.

Your Work:

If the program shown in hw 12.1 does not work, you need to fix it. Use the program from 12.1 as a basis and design and implement a program so such every philosopher is guaranteed to be able to eat.

• You have to name your program Philosopher.java

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
 1      public class Philospher extends Thread  {
 2
 3              private int             id;
 4              private Object          first;
 5              private Object          second;
 6              static final int        MAX = 4;        // number of philosphe
 7              static Object           all[] = new Object[MAX];
 8              static int              createdSoFar = 1;
 9              static int              lastSeen = 0;
10              int                     soOftenDoesOneEat = 2;
11              int                     soOftenHaveIEaten;
12
13              static {
14                      for ( int index = 0; index < MAX; index ++ )
15                              all[index] = new Object();
16              }
17
18              public Philospher (int id, Object first, Object second) {
19                      this.id         = id;
20                      this.first      = first;
21                      this.second     = second;
22                      if ( id > 1 )
23                              setDaemon(true);
24              }
25              public void test () {
26                      if ( lastSeen + 1 != id )       {
27                              System.out.println("Something went wrong. Last
28                              System.exit(1);
```

```
29                      }
30                      lastSeen = ( lastSeen + 1 ) % MAX;
31                      try { sleep(100); } catch ( Exception e ) { }
32
33              }
34
35              public static String padding(String theString, int soLongWillT
36                          return String.format("%1$" + soLongWillTheStringBe
37              }
38              public void run () {
39                      while ( true )  {
40                              synchronized ( first ) {
41                                      synchronized ( second ) {
42                                              second.notify();
43                                              test();
44                                              ++soOftenHaveIEaten;
45                                              if ( ( id == 1 )
46                                                 ( soOftenHaveIEaten > soOft
47                                                              return;
48                                              else {
49                                                      String comment = id +
50                                                      System.out.println( pa
51                                              } try {
52                                                      if ( createdSoFar <= N
53                                                              createdSoFar++
54                                                              ( new Philosph
55                                                      }
56                                              } catch ( Exception e ) { }
57                                      }
58                                      try {
59                                              first.wait();
60                                      } catch ( Exception e ) { }
61                              }
62                      }
63              }
64              public static void main (String args []) {
65                      new Philospher(1, all[0], all[1]).start();
66              }
67      }
```

Source Code: Src/32_sol/Philospher.java

32.3. Homework 12.3 (15 Points)

Objective: The objective is to design and implement a distribute application.

Grading:
Correctness: You can lose up to % if your solution is not correct
Quality: You can lose up to % if your solution is poorly designed
Testing: You can lose up to % if your solution is not well tested
Explanation: You can lose up to % if your solution if you can not explain your solution during the grading session

Homework Description:

Your Work:

For hw 4.1 you implemented a one person 3-dimensional battleship game. For this homework you have to modify hw 4.1 so such it can be played by two players on two different computers, or on the same computer in two different JVM's. In other words you and your opponent are using the same program developed by you.

You specify a scenario, ie. where are the boats in your ocean, and so does your opponent. Your program loads your ocean, your opponent reads their file. You try to sink your opponents boats and vice versa. In other words you modify the ocean in your opponents JVM. Your program has to show how your move modify your opponents ocean.

You have to use Sockets (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/net/Socket.html)
for your implementation.

A possible command line call could be:

```
# assuming your computer is spiegel.cs.rit.edu
# on your computer
% java BattleShip -hostname glados.cs.rit.edu -port 4567 -first -ocean northSe
# on glados.cs.rit.edu
% java BattleShip -hostname spiegel.cs.rit.edu -port 5678 -second -ocean eastS
```

An other possible command line call could be:

```
# assuming your computer is spiegel.cs.rit.edu
# on your computer
% java BattleShip  -first -ocean northSea.txt
Waiting on port: 5678
# on glados.cs.rit.edu
% java BattleShip -hostname spiegel.cs.rit.edu -port 5678 -second -ocean eastS
```

• You can assume that the input file is correct

• That you have a way way to communicate which port is used. In other words you do not need to send the port via your code.

• The name of the ocean must be specified on the command line.

• You have to test your solution

Submission:

Submit your files via myCourses.

Solution:

(This solution serves as the basis for the discussion in class. Sometimes there will be errors introduced to show common mistakes)

```
 1      import gnu.dtools.ritopt.IntOption;
 2      import gnu.dtools.ritopt.BooleanOption;
 3      import gnu.dtools.ritopt.StringOption;
 4      import gnu.dtools.ritopt.Options;
 5      import gnu.dtools.ritopt.Option;
 6
 7
 8      import java.io.File;
 9      import java.io.FileNotFoundException;
10      import java.util.Scanner;
11      import java.util.regex.Matcher;
12      import java.util.regex.Pattern;
13
14      import java.net.*;
15
16      import java.io.PrintWriter;
17      import java.io.BufferedWriter;
18      import java.io.BufferedReader;
19      import java.io.InputStreamReader;
20      import java.io.OutputStreamWriter;
21
22
23      public class BattleShip
24      {
25              BufferedReader readFrom;
26              PrintWriter writeTo;
27              ServerSocket   aServerSocket;
28
29              String ocean;
30              String hostName;
31              int port;
32              boolean first;
33              boolean second;
34
35              String WIDTH                    = "width";
36              String HEIGHT                   = "height";
37              String WATER                    = "w";
38              int WATER_VALUE                         = -1;
39              int WATER_HIT_VALUE             = -2;
40              int HIT_VALUE                   = -3;
41              String  lineDelimiter           = "#";
42
43              int[][] battleField             = null;         // This is the
44              int[][] originalBattleField     = null;         // will be ini
45              String  battleFieldString       = null;         // will be ini
46              int battleFieldWidth            = 0;
47              int battleFieldHeight           = 0;
48              Scanner battleFieldParser       = null;
49              char hit                        = 'x';
50              String fileName                         = null;
51
52
53              void parseArgs(String[] args )  {
```

```
54                      IntOption portO                 = new IntOption(0);
55                      StringOption oceanO             = new StringOption("")
56                      StringOption hostNameO          = new StringOption("")
57                      BooleanOption firstO            = new BooleanOption(fa
58                      BooleanOption secondO           = new BooleanOption(fa
59
60                      Options options                 = new Options("BattleS
61                      options.register("port", portO);
62                      options.register("ocean", oceanO);
63                      options.register("hostName", hostNameO);
64                      options.register("first", firstO);
65                      options.register("second", secondO);
66
67                      options.process(args);
68
69                      port = portO.getValue();
70                      ocean = oceanO.getValue();
71                      hostName = hostNameO.getValue();
72                      first = firstO.getValue();
73                      second = secondO.getValue();
74              }
75
76              private void removeAllPartsOfBoat(int column,int row)   {
77                      int boatId = originalBattleField[column][row];
78                      for ( int rows = 0; rows < battleFieldHeight; rows ++
79                              for ( int columns = 0; columns < battleFieldWi
80                                      if ( originalBattleField[columns][rows
81                                              battleField[columns][rows] = F
82                              }
83                      }
84              }
85              private void printBattleField(int[][] battleField) {
86                      System.out.println();
87                      for ( int rows = 0; rows < battleFieldHeight; rows ++
88                              for ( int columns = 0; columns < battleFieldWi
89                                      if ( battleField[columns][rows] == WAT
90                                              System.out.print(  "w" + " " )
91                                      else
92                                              System.out.print(  battleFieldO
93                              }
94                              System.out.println();
95                      }
96                      System.out.println();
97              }
98              private void printBattleFieldForPlayer(int[][] battleField) {
99                      System.out.println();
00                      System.out.println("x indicates a hit.");
01                      System.out.println("w indicates a miss, but you know n
02                      System.out.println(". indicates boat oder water.\n");
03                      System.out.print( "   " );
04                      for ( int columns = 0; columns < battleFieldWidth; col
05                              System.out.print(  " " + columns );
06                      }
07                      System.out.println(" ---> columns");
```

```
08                      for ( int rows = 0; rows < battleFieldHeight; rows ++
09                              System.out.print(rows + ": " );
10                              for ( int columns = 0; columns < battleFieldWi
11                                      if ( battleField[columns][rows] == WAT
12                                              System.out.print(  " " + "w" )
13                                      else if ( battleField[columns][rows] =
14                                              System.out.print(  " " + "x" )
15                                      else
16                                              System.out.print(  " " + "." )
17                              }
18                              System.out.println();
19                      }
20                      System.out.println();
21              }
22              private boolean isThereAboatLeft() {
23                      boolean rValue = false;
24                      for ( int rows = 0; ! rValue && rows < battleFieldHeig
25                              for ( int columns = 0; ! rValue && columns < b
26                                      if  ( battleField[columns][rows] >= 0
27                                              rValue = true;
28                              }
29                      }
30                      return rValue;
31              }
32              private boolean allWater() {
33                      for ( int column = 0; column < battleFieldWidth; colum
34                              for ( int row = 0; row < battleFieldWidth; row
35                                      if (  battleField[column][row] != WATE
36                                              return false;
37                              }
38                      }
39                      return true;
40              }
41              private void readHeightWidth() {
42                      for (int index = 0; index < 2; index ++ )       {
43                              if ( battleFieldParser.hasNextLine() )  {
44                                      String[] oneDimension = battleFieldPar
45                                      if ( oneDimension[0].equals(WIDTH) )
46                                              battleFieldWidth = Integer.par
47                                      else
48                                              battleFieldHeight = Integer.pa
49                              }
50                      }
51              }
52              private void createBattleField() {
53                      battleField             = new int[battleFieldWidth][ba
54                      originalBattleField     = new int[battleFieldWidth][ba
55                      for ( int columns = 0; columns < battleFieldWidth; col
56                              for ( int rows = 0; rows < battleFieldHeight;
57                                      battleField[columns][rows] = WATER_VAL
58                                      originalBattleField[columns][rows] = W
59                              }
60                      }
61              }
```

```
62              private void updateBattleField() {
63                      for ( int columns = 0; columns < battleFieldWidth; col
64                              for ( int rows = 0; rows < battleFieldHeight;
65                                      if ( originalBattleField[columns][rows
66                                              battleField[columns][rows] = 
67                              }
68                      }
69              }
70              private void readBattleFieldScenario() {
71                      for ( int index = 0; index <   battleFieldHeight; inde
72                              if ( battleFieldParser.hasNextLine() )  {
73                                      String[] oneRow = battleFieldParser.ne
74                                      for ( int xPosition = 1; xPosition < b
75                                              if (! oneRow[xPosition].equals
76                                                      String id = oneRow[xPo
77                                                      originalBattleField[xP
78                                              }
79                                      }
80                              }
81                      }
82              }
83              private void readBattleFieldFile(String fileName) {
84                      if ( fileName.equals("exit")) {
85                              System.exit(0);
86                      }
87                      int column = 0;
88                      try {
89                              battleFieldString = "";
90                              battleFieldParser = new Scanner(new File(fileN
91                              while ( battleFieldParser.hasNextLine() )
92                                      battleFieldString += battleFieldParser
93                              }
94
95                      } catch (FileNotFoundException e) {
96                              System.out.println("Can't find that file! Try
97                      }
98
99              }
00
01              private void readBattleFieldFromString(String theBattleFieldIn
02
03                      int column = 0;
04                      theBattleFieldInStringForm = theBattleFieldInStringFor
05                      battleFieldParser = new Scanner(theBattleFieldInString
06                      readHeightWidth();
07                      createBattleField();
08                      readBattleFieldScenario();
09                      updateBattleField();
10
11              }
12              private int readOneIntValue(Scanner readUserInput, String text
13                      System.out.print(text);
14                      if ( readUserInput.hasNextInt() )
15                              return readUserInput.nextInt();
```

```
16                      else    {
17                              System.out.println("Can't read next integer -
18                              System.exit(1);
19                      }
20                      return -1;
21              }
22              private boolean checkRange(int minValue, int value, int maxVal
23                      if ( ( minValue <= value) && ( value < maxValue ) )
24                              return true;
25                      } else  {
26                              System.out.println("Error: " + errorMessage );
27                              return false;
28                      }
29              }
30              private void readTheOceans() {                          // new
31                      if ( first )            {
32                              sendData(battleFieldString);
33
34                              String theOtherOnes =  readData();
35                              readBattleFieldFromString(theOtherOnes);
36                      } else  {
37                              String theOtherOnes =  readData();
38                              readBattleFieldFromString(theOtherOnes);
39
40                              sendData(battleFieldString);
41                      }
42              }
43              private void play() {
44                      readTheOceans();                // new
45                      Scanner readUserInput  = new Scanner( System.in);
46                      int row = 0;
47                      int column = 0;
48                      int soManyTries = 0;
49                      while ( isThereAboatLeft()      )       {
50                              soManyTries++;
51                              printBattleFieldForPlayer(battleField);
52                              column = readOneIntValue(readUserInput,
53                                              "column coordinate (0 <= colum
54                              row     = readOneIntValue(readUserInput,
55                                              "row    coordinate (0 <= row
56                              if ( checkRange(0, column, battleFieldWidth, "
57                                              checkRange(0, row, battleFieldH
58                                      if ( originalBattleField[column][row]
59                                              battleField[column][row] = WAT
60                                      } else {
61                                              System.out.println("HIT");
62                                              removeAllPartsOfBoat(column,ro
63                                      }
64                      printBattleFieldForPlayer(battleField);
65                      }
66                      printBattleFieldForPlayer(battleField);
67              }
68
69              private  void sendData(String theData) {                // new
```

```
70                      System.out.println("sendData: ");
71                      try {
72                              writeTo.println(theData);
73                      } catch ( Exception e ) {
74                              e.printStackTrace();
75                      }
76              }
77              private  String  readData() {                           // new
78                      System.out.println("readData: ");
79                      String rValue = "";
80                      try {
81                              rValue = readFrom.readLine();
82                      } catch ( Exception e ) {
83                              e.printStackTrace();
84                      }
85                      return rValue;
86              }
87              private void setUpIO()  {                               // new
88                      if ( first )    {
89                              try {
90                                      aServerSocket = new ServerSocket(port)
91                                      Socket theReadFromSocket = aServerSock
92                                      readFrom  = new BufferedReader( new In
93                                      writeTo = new PrintWriter(theReadFromS
94
95                              } catch ( Exception e ) {
96                                      e.printStackTrace();
97                              }
98                      } else {
99                              try {
00                                      Socket theReadFromSocket = new Socket(
01                                      writeTo = new PrintWriter (theReadFrom
02                                      readFrom  = new BufferedReader( new In
03
04                              } catch ( Exception e ) {
05                                      e.printStackTrace();
06                              }
07                      }
08              }
09              private void playTheGame(String[] args) {
10                      parseArgs(args);
11                      readBattleFieldFile(ocean);                     // new
12                      setUpIO();
13                      play();
14              }
15              public static void main(String[] args) {
16                      new BattleShip().playTheGame(args);
17              }
18      }
```

Source Code: Src/32_sol/BattleShip.java

Not Final

33. Homework 13

This is the last hw.

Posted: November/13/2020
Due: November/22/2020 24.00

The solutions for this homework are due November/22/2020 24.00. I recommend to submit at least one version of all homework solutions long before due date.

Snippet of the note from Provost Granberg to all faculty:

> Instead, during the final four weeks of instruction, I encourage faculty to identify opportunities to reduce your own workload and that of your students. Some faculty members are already taking this step by eliminating material they consider optional, reducing the number of assignments, or scheduling recharge days for their classes. Any of these options can provide both faculty and students with an opportunity to catch up and restore some energy.

In the spirit of her note you can choose which two of the three question you want to answer. All questions are worth 15 points. The maximum number of points for hw 11 is 30.

33.1. Homework 13.1 (15 Points)

Objective: Working with Sockts

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Objective: The objective is create a distributed Program using Sockets.

Grading:
Correctness: You can lose up to % if your solution is not correct
Quality: You can lose up to % if your solution is poorly designed
Testing: You can lose up to % if your solution is not well tested
Explanation: You can lose up to % if your solution if you can not explain your solution during the grading session

Homework Description:

Modify hw 12.1 in such a way that the program playing determine who will start first. You have to use Sockets (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/net/Socket.html)
for your implementation.

Requirements:

• You can assume that the input file is correct

• That you have a way way to communicate which port is used. In other words you do not need to send the port via your code.

• The name of the ocean must be specified on the command line.

• You have to test your solution

Submission:

Submit your files via myCourses.

33.2. Homework 13.2 (15 Points)

Objective: Working with DatagramSockets

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Objective: The objective is create a distributed Program using DatagramSockets.

Grading:
Correctness: You can lose up to % if your solution is not correct
Quality: You can lose up to % if your solution is poorly designed
Testing: You can lose up to % if your solution is not well tested
Explanation: You can lose up to % if your solution if you can not explain your solution during the grading session

Homework Description:

Modify hw 12.3 using instead of Sockets DatagramSocket. (https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/net/DatagramSocket.html)
for your implementation.

Requirements:

• You can assume that the input file is correct

• That you have a way way to communicate which port is used. In other words you do not need to send the port via your code.

• The name of the ocean must be specified on the command line.

• You have to test your solution

Submission:

Submit your files via myCourses.

33.3. Homework 13.3 (15 Points)

Objective: Working with RMI

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Objective: The objective is create a distributed Program using RMI.

Grading:
Correctness: You can lose up to % if your solution is not correct
Quality: You can lose up to % if your solution is poorly designed
Testing: You can lose up to % if your solution is not well tested
Explanation: You can lose up to % if your solution if you can not explain your solution

during the grading session

Homework Description:

Modify hw 12.3 using instead of Sockets a RMI (https://docs.oracle.com/en/java/javase/14/docs/api/java.rmi/module-summary.html) communication.

Requirements:

- You can assume that the input file is correct
- That you have a way way to communicate which port is used. In other words you do not need to send the port via your code.
- The name of the ocean must be specified on the command line.
- You have to test your solution

Submission:

Submit your files via myCourses.

34. Under Construction

35. Under Construction

36. Under Construction

37. Under Construction

38. Template - 38

38.1. Homework 39

Not Final

Posted: 1/2/3
Due: 4/5/6

The solutions for this homework are due 4/5/6. I recommend to submit at least one version of all homework solutions long before due date.

38.2. Homework 39.1 (10 Points)

Objective: Goal

Grading:
Correctness: You can lose up to 40% if your solution is not correct
Quality: You can lose up to 80% if your solution is poorly designed
Testing: You can lose up to 50% if your solution is not well tested
Explanation: You can lose up to 100% if your solution if you can not explain your solution during the grading session

Homework Description:

Explanation:

Your Work:

Requirements:

Example:
An example of a solution execution:

Submission:
Submit your files via myCourses.

39. Under Construction

40. Bridge Exam CSCI 605 Demo— August 12/2020 - version 1

Time: 50 Minutes
Total Points: 40

THIS IS NOT A REAL EXAM. THIS IS A DEMO EXAM WHICH YOU CAN TAKE AS OFTEN AS YOU WISH.

All code examples do compile.

Question 1 (8)

Take a look at the following program:

```
 1      class A1 {
 2        public static void main( String args[] ) {
 3              String aString = "12";
 4              String bString = "123";
 5              Integer aInt   = Integer.valueOf(3);
 6              Integer bInt   = Integer.valueOf(3);
 7              int number = 3;
 8
 9              System.out.println("1.  " +   (aInt == bInt) );
10              System.out.println("2.  " +   ( aString + number  == bString )
11          }
12      }
```

Source Code: Src/82/A1.java

The output of this program is:

```
% java A1
1.     true
2.     false
```

You need to explain why this is the output. If the the print statement would be:

```
System.out.println("4.  " +   ( "12" == ( "1"  + "2" ))  );
```

a possible answer would be:

Output	Explanation
4:	The strings "1" + "2" are converted to the literal "12" during compile time, and only one literal "12" exist, therefore the 'address' of the two literals is the same therefore the answer is true.

Your answer :

Output	Explanation
1:	
2:	

41. Examination I— September/16/2020 - version 1

Time: 50 Minutes
Total Points: 40

All code examples do compile.

Question 1 (8)

Take a look at the following program:

```
 1      class A {
 2
 3        public static String method()   {
 4         return ( 1 > 0 ? "1234"  : "a");
 5        }
 6        public static String met()   {
 7         return "12";
 8        }
 9        public static String hod()   {
10         return "34";
11        }
12        public static void main( String args[] ) {
13         String aString = "1234";
14         int aInt       =  34;
15
16         System.out.println(" 1:   " + ( "1234" == "1234")  );
17         System.out.println(" 2:   " + ( aString == "1234") );
18         System.out.println(" 3:   " + ( aString == "12" + "34")  );
19         System.out.println(" 4:   " + ( aString == "12" + aInt )  );
20         System.out.println(" 5:   " + ( aString.equals("12" + aInt/2 + aInt
21         System.out.println(" 6:   " + ( aString.equals("12" + ( aInt/2 + a
22         System.out.println(" 7:   " + ( aString == method())            );
23         System.out.println(" 8:   " + ( aString == ( met()) + hod())
24         System.out.println(" 9:   " + ( aString ==
25               ( aString.substring(0, 2) + aString.substring(2, 4)))   );
26         System.out.println("10:   " + ( aString.equals(
27               aString.substring(0, 2) + hod()))         );
28        }
29      }
```

Source Code: Src/82/A.java

What is the output of the program? **You need to explain your answers**. You might like to use a drawing of a memory model, but you can also explain your answers with sentences. The output lines of interest are marked in the *println* statements with numbers from 1 to 10.

If the line would be:

```
int index = 4;
System.out.println(" 1: " + ( "1234" == "123" + index)
```

You answer would be: *false*

Your explanation would be: The two references are different. During runtime the inter 4, will be converted to a string and then concatenated to the string "123", which will create a new string. This new string is not a literal.

1. True Both are the same literal, therefore the same address refernce, therefore == comparison is true

2. True Both are the same literal, therefore the same address refernce, ther efore == comparison is true

3. True "12" + "34" will be concatenated during compile time to the literal "1234". Both are the same literal, therefore the same address reference, therefore == comparison is true

4. False "12" + aInt is converted to a new string "1234", which is not a literal, therefore not the same address reference, therefore == comparison is false

5. False "12" + aInt/2 + aInt/2 will became "12" + "17" + "17" == "121717", which is not equal to "1234"

6. True, 12" + (aInt/2 + aInt/2)) will become 12" + (17 + 17) = "12" + 34 -> "1234" which is equal to "1234"

7. True method() returns the literal "1234". therefore the same address reference, therefore == comparison is true

8. False met() and hod() return a literal, but met() + hod() results in a new string, therefore the reference is not the same therefore == results in false.

9. False substring returns a new string, therefore the concatenation results in a new sting, therefore the reference is not the same therefore == results in false.

10. True aString.substring(0, 2) + hod()) result in a new string "1234" which is equal to :1234"

Question 2 (8)

The table below includes a list of verbal pattern descriptions. You have to provide the actual pattern description.

If the description be: A word which consist of exactly two characters.

The pattern would be "^..$".

Description	Example	Anti Example	Pattern
a word starting with a the character between 'a' and 'h', and ending character between 's' and 'z'.	az hs	za aj	pattern 1
a word starts with 'h' or 'H' followed by none ore one '-', followed by 'P' or 'p'	H-p h--P (incorrect)	H-t hp	pattern 2
a word which has at least two vowels back to back	book	ball	pattern 3
a pattern which can describe a 3 part word The first part is either Boat or Ship, The second part is either Had or Has, The third part is either Sails or Motor,	pattern 4 BoatHasSails ShipHasSails	 BoatHasSunk BoatSail	
an optional + or -, followed by at least one digit followed by a '.' followed by 2 digits	+0.79 0.79	0.678 =0.79	pattern 5
the same requirements as above, but a 0 can not be the single digit following a + or -	+1.79 -23.23	+0.79 -0.79	pattern 6
definition of a double in java a real, with a leading '+' or minus, followed by a lower or uppercase 'e', followed a real, with a optional leading '+' or minus.	1.7e-123 1e10	1.7Ee-123 +-1.7E+-123	pattern 7

A program similar to the home work.

```
1       import java.util.Scanner;
2       import java.util.regex.Matcher;
3       import java.util.regex.Pattern;
4       public class B
5       {
6                   static String[] allPatternsToTest = {
7                         "az",
8                                 "a word starting with a the character from th
9                                       "^[a-h].*[s-z]$",
10                        "h-p",
11                                "a word starts with 'h' or 'H' followed by no
12                                      "[hH]-?[pP]",
13                        "boook",
14                                "a word which has at least two vowels back to
15                                      "^.*[AEIOUaeiou]{2}.*$",
16                        "TheBoatHasSails", // TheShipHadSails
17                               "The Boat can be a ship, had or has a motor or
18                                ".*The(Boat|Schip)(Had|Has)(Sails|Motor).*",
19                        "+0.79",
20                                "an optional + or -, followed by at least one
21                                      "^(\\+|-)?[0-9]+\\.[0-9]{2}$",
```

```
22                      "+1.79", // but not +0.79
23                              "the same requirements as above, but a 0 can
24                                      "^(\\+|-)?[1-9]+[0-9]*\\.[0-9]{2}$",
25                      "-1.7e-123", // +4.9e-324
26                              "definition of a double in java",
27                                      "^(\\+|-)?[1-9]+[0-9]*\\.[0-9]*(e[-]?[
28                  };
29
30              public static void processStatic()      {
31                      int index = 0;
32                      while  ( index < allPatternsToTest.length )     {
33                              System.out.println("Word to test: -" + allPatt
34                              if  ( Pattern.matches(allPatternsToTest[index
35                                      System.out.println("This regular expre
36                                              + " matches the following inpu
37                                      System.out.println("    verbal explana
38                              }
39                              index += 3;
40                      }
41              }
42              public static void main(String[] args) {
43                      processStatic();
44              }
45      }
```

Source Code: Src/82/B.java

Question 3 (8)

Take a look at the follwong source code:

```
 1      public class C {
 2              public static void main(String[] args) {
 3                      DDC aDDC = new DDC();           // 1
 4                      CCC aCCC = new CCC();           // 2
 5                      aDDC = new DDC("" + 1);         // 3
 6                      aCCC = new CCC(1);              // 4
 7              }
 8      }
 9      class CC extends C {
10              CC()    {
11              }
12      }
13      class DC extends C {
14              DC()    {
15              }
16      }
17      class CCC extends CC {
18              CCC()   {
19              }
20              CCC(int aInt)   {
21              }
22              CCC(String aString)     {
23              }
24      }
25      class DDC extends DC {
26              DDC()   {
27              }
28              DDC(int aInt)   {
29              }
30              DDC(String aString)     {
31              }
32      }
```

Source Code: Src/82/C.java

Four objects will be generated during the execution of this program. What is the execution order of the constructors? Of interest are the marked lines 1-4. You need to explain your answers.

1. DDC() default constructor is called, DC() is called, C() is called, Object() is called. Object() has to complete before C() can be completed, etc

2. CCC() default constructor is called, CC() constructor is called, C() constructor is called, Object() is called. Object() has to complete before C() can be completed, etc.

3. DDC(String) is called, DC() is called, C() is called, Object() is called. Object() has to complete before C() can be completed, etc. DDC() is not called.

4. CCC(int) is called, CC() is called, C() is called, Object() is called. Object() has to complete before C() can be completed, etc

Question 4 (8)

Take a look at the following source code:

```
 1      public class  D {
 2
 3              int    speed;
 4              String aString = "you";
 5              D()     {
 6                      speed = 0;
 7              }
 8              public void speed(int speed )      {
 9                      this.speed = speed;
10              }
11              public String toString()      {
12                      return "" + speed;
13              }
14              public void testString(String arg )      {
15                      arg = "me";
16              }
17              public void testD(D arg )      {
18                      arg.speed(42);
19              }
20              public void testString()        {
21                      System.out.println("1.  " + aString);
22                      testString(aString);
23                      System.out.println("2.  " + aString);
24              }
25              public void testD()     {
26                      D aD = new D();
27                      System.out.println("3.  " + aD );
28                      testD(aD);
29                      System.out.println("4.  " + aD );
30              }
31              public static void main(String args[] )       {
32                      new D().testString();
33                      new D().testD();
34              }
35      }
```

Source Code: Src/82/D.java

What is output of this program? (The eam shows an error here). You need to explain your answers.

The output of the program is:

```
1.     you
2.     you
3.     0
4.     42
```

Explanation was given in class and during hw.

Question 5 (8)

Take a look at the following source code:

What is the out of this program, and you have to explain your answer.

```
 1      public class E {
 2              static E anEobject = null;
 3              E anOtherObject    = null;
 4              static int counter  = 0;
 5              int myLocalCounter  = 0;
 6
 7              E()     {
 8                      counter ++;
 9              }
10              public void method2() {
11                      anEobject = new E();
12                      anEobject = new E();
13              }
14              public static void method1() {
15                      anEobject = new E();
16                      anEobject.method2();
17                      // myLocalCounter = 0;
18              }
19              public static void main(String[] args) {
20                      method1();
21                      System.out.println(counter);
22              }
23      }
```

Source Code: Src/82/E.java

Output: 3

Explanation:

3 objects are created and the default constructor increments a class variable by one each time the contrctor is called. counter exists only once, therefore the value will be 3. counter is initialized with 0.

This program will not compile, if line 17 would not be commented out. Why?

Answer:

myLocalCounter is an instance variable. This varible exists only in object context.

42. Examination I— September/16/2020 - version 1

Time: 50 Minutes
Total Points: 40

All code examples do compile.

Question 1 (8)

Take a look at the following program:

```
 1      class A {
 2
 3        public static String method()   {
 4         return ( 1 > 0 ? "1234"  : "a");
 5        }
 6        public static String met()   {
 7         return "12";
 8        }
 9        public static String hod()   {
10         return "34";
11        }
12        public static void main( String args[] ) {
13         String aString = "1234";
14         int aInt       =  34;
15
16         System.out.println(" 1:   " + ( "1234" == "1234")  );
17         System.out.println(" 2:   " + ( aString == "1234") );
18         System.out.println(" 3:   " + ( aString == "12" + "34")  );
19         System.out.println(" 4:   " + ( aString == "12" + aInt )  );
20         System.out.println(" 5:   " + ( aString.equals("12" + aInt/2 + aInt
21         System.out.println(" 6:   " + ( aString.equals("12" + ( aInt/2 + a
22         System.out.println(" 7:   " + ( aString == method())            );
23         System.out.println(" 8:   " + ( aString == ( met()) + hod())
24         System.out.println(" 9:   " + ( aString ==
25               ( aString.substring(0, 2) + aString.substring(2, 4)))   );
26         System.out.println("10:   " + ( aString.equals(
27               aString.substring(0, 2) + hod()))         );
28        }
29      }
```

Source Code: Src/82/A.java

What is the output of the program? **You need to explain your answers**. You might like to use a drawing of a memory model, but you can also explain your answers with sentences. The output lines of interest are marked in the *println* statements with numbers from 1 to 10.

If the line would be:

```
int index = 4;
System.out.println(" 1: " + ( "1234" == "123" + index)
```

You answer would be: *false*

Your explanation would be: The two references are different. During runtime the inter 4, will be converted to a string and then concatenated to the string "123", which will create a new string. This new string is not a literal.

Line	Output
1.	
2.	
3.	
4.	
5.	
6.	
7.	
8.	
9.	
10.	

Question 2 (8)

The table below includes a list of verbal pattern descriptions. You have to provide the actual pattern description.

If the description be: A word which consist of exactly two characters.

The pattern would be "^..$".

Description	Example	Anti Example	Pattern
a word starting with a the character between 'a' and 'h', and ending character between 's' and 'z'.	az hs	za aj	pattern 1
a word starts with 'h' or 'H' followed by one or more '-', followed by 'P' or 'p'	H-p h--P	H-t hp	pattern 2
a word which has at least two vowels back to back	book	ball	pattern 3
a pattern which can describe a 3 part word The first part is either Boat or Ship, The second part is either Had or Has, The third part is either Sails or Motor,	pattern 4 BoatHasSails ShipHasSails	 BoatHasSunk BoatSail	
an optional + or -, followed by at least one digit followed by a '.' followed by 2 digits	+0.79 0.79	0.678 =0.79	pattern 5
the same requirements as above, but a 0 can not be the single digit following a + or -	+1.79 -23.23	+0.79 -0.79	pattern 6
definition of a double in java a real, with a leading '+' or minus, followed by a lower or uppercase 'e', followed a real, with a optional leading '+' or minus.	1.7e-123 1e10	1.7Ee-123 +-1.7E+-123	pattern 7

Question 3 (8)

Take a look at the follwong source code:

```
 1      public class C {
 2              public static void main(String[] args) {
 3                      DDC aDDC = new DDC();           // 1
 4                      CCC aCCC = new CCC();           // 2
 5                      aDDC = new DDC("" + 1);         // 3
 6                      aCCC = new CCC(1);              // 4
 7              }
 8      }
 9      class CC extends C {
10              CC()    {
11              }
12      }
13      class DC extends C {
14              DC()    {
15              }
16      }
17      class CCC extends CC {
18              CCC()   {
19              }
20              CCC(int aInt)   {
21              }
22              CCC(String aString)     {
23              }
24      }
25      class DDC extends DC {
26              DDC()   {
27              }
28              DDC(int aInt)   {
29              }
30              DDC(String aString)     {
31              }
32      }
```

Source Code: Src/82/C.java

Four objects will be generated during the execution of this program. What is the execution order of the constructors? Of interest are the marked lines 1-4. You need to explain your answers.

Marked Line	Output	Explanation
1.		
2.		
3.		

4.

Question 4 (8)

Take a look at the following source code:

```
 1      public class  D {
 2
 3              int    speed;
 4              String aString = "you";
 5              D()     {
 6                      speed = 0;
 7              }
 8              public void speed(int speed )      {
 9                      this.speed = speed;
10              }
11              public String toString()      {
12                      return "" + speed;
13              }
14              public void testString(String arg )      {
15                      arg = "me";
16              }
17              public void testD(D arg )      {
18                      arg.speed(42);
19              }
20              public void testString()        {
21                      System.out.println("1.  " + aString);
22                      testString(aString);
23                      System.out.println("2.  " + aString);
24              }
25              public void testD()     {
26                      D aD = new D();
27                      System.out.println("3.  " + aD );
28                      testD(aD);
29                      System.out.println("4.  " + aD );
30              }
31              public static void main(String args[] )       {
32                      new D().testString();
33                      new D().testD();
34              }
35      }
```

Source Code: Src/82/D.java

What is the execution order of the constructors? Of interest are the marked lines 1-4. You need to explain your answers.

Output	Explanation
1	

2

3

4

Question 5 (8)

Take a look at the following source code:

What is the out of this program, and you have to explain your answer.

```
 1      public class E {
 2              static E anEobject = null;
 3              E anOtherObject    = null;
 4              static int counter  = 0;
 5              int myLocalCounter  = 0;
 6
 7              E()     {
 8                      counter ++;
 9              }
10              public void method2() {
11                      anEobject = new E();
12                      anEobject = new E();
13              }
14              public static void method1() {
15                      anEobject = new E();
16                      anEobject.method2();
17                      // myLocalCounter = 0;
18              }
19              public static void main(String[] args) {
20                      method1();
21                      System.out.println(counter);
22              }
23      }
```

Source Code: Src/82/E.java

Output:

Explanation:

This program will not compilei, if line 17 would not be commented out. Why?

Answer:

43. Examination I— October/21/2020 - version 1

Time: 50 Minutes
Total Points: 40

All code examples do compile unless noted otherwise.

Question 1 (8)

Take a look at the following programs:

Vehicle.java

```
 1      class Vehicle {
 2        int soManyWheels       = 4;
 3        public Vehicle()      {
 4                System.out.println("Vehicle()");
 5        }
 6        public Vehicle(int soManyWheels)      {
 7              System.out.println("Vehicle(int)");
 8              this.soManyWheels = soManyWheels;
 9        }
10        public String toString ()    {
11              return "Vehicle/" + soManyWheels;
12        }
13        public void setSoManyWheels(int soManyWheels)    {
14              this.soManyWheels = soManyWheels;
15        }
16        public int soManyWheels ()    {
17              return soManyWheels;
18        }
19      }
```

Source Code: Src/84_October_21_2020/Vehicle.java

Train.java

```
 1      public class Train extends Vehicle {
 2        int soManyWheels       = 32;
 3        public Train()        {
 4              System.out.println("Train()");
 5        }
 6        public Train(int soManyWheels)        {
 7              System.out.println("Train(int)");
 8              this.soManyWheels = soManyWheels;
 9        }
10        public String toString ()    {
11              return "Train/" + soManyWheels;
12        }
13        public void setSoManyWheels(int soManyWheels)    {
14              this.soManyWheels = soManyWheels;
15        }
16        public int soManyWheels ()    {
17              return soManyWheels;
18        }
19        public static void main(String[] args )       {
20
21              Train   aTrain   = new Train(32);                    // 1
22              Vehicle aVehicle = aTrain;
23
24              System.out.println(aTrain);                         // 2
25              System.out.println(aVehicle);                       // 3
26
27              aVehicle.setSoManyWheels(1);
28              System.out.println(aVehicle.soManyWheels);          // 4
29              System.out.println(aTrain.soManyWheels);            // 5
30
31              aVehicle.soManyWheels = 2;
32              System.out.println(aVehicle.soManyWheels);          // 6
33              System.out.println(aTrain.soManyWheels);            // 7
34        }
35      }
```

Source Code: Src/84_October_21_2020/Train.java

Question

Describe the order in which the constructors will be called and completed in order to create a *Train* object. Of interest is the line marked 1.

Answer:

Train constructor calls Vehicle constructor calls Object constructor.

```
Train(int)    {
       Vehicle()
              Object()
       exectutes rest of Vehicle construtor
executes rest of Trian constructor(int)
```

Question

What is the output of this program when it gets executed? Of interest are the lines marked 2-7.

Answer:

```
2 - Train/32 - the cast will not chnage that a train method is called
3 - Train/32 - the cast will not chnage that a train method is called
4 - 4 - direct acces to the instance variable
5 - 1 direct acces to the instance variable
6 - 2 direct acces to the vehicle instance variable
7 - 1 6 does only change the vehicle instance variable
```

You need to explain your answers.

Question 2 (8)

The following program will not compile as is. Take a look at the following program:

```
 1      import java.util.LinkedList;
 2      import java.util.List;
 3      public class UseIt<T extends B> {
 4
 5          T theThing;
 6
 7          public UseIt(T theThing)  {
 8              this.theThing = theThing;
 9          }
10          static <T extends C> void methodA(T aTobject)  {            // 1
11          }
12          public static void methodB(List<? super B> list) {          // 2
13          }
14          public static void methodC(List<? extends B> list) {        // 3
15          }
16          public static void main(String[] args)      {
17                  UseIt<BBB> aBBB = new UseIt<BBB>(new BBB());
18                  // UseIt<A> aA = new UseIt<A>(new A());
19                  methodA(new C());
20                  methodA(new D());
21                  // print(new B());
22
23                  List<UseIt<B>> aList = new LinkedList<UseIt<B>>();  // 1
24                  UseIt<C> aC = new UseIt<C>(new C());
25                  aList.add(new UseIt<B>(new B()) );
26                  // aList.add(new B() );                             // 1
27
28                  List<A>   listMethodB_1  = new LinkedList<A>();
29                  List<BB>  listMethodB_2  = new LinkedList<BB>();
30                  methodB(listMethodB_1);
31                  // methodB(listMethodB_2);                          // 1
32
33                  // methodC(listMethodB_1);                          // 1
34                  methodC(listMethodB_2);
35          }
36      }
37      interface I { }
38      class A { }
39      class C extends B implements I { }
40      class D extends C { }
41      class B extends A { }
42      class BB extends B { }
43      class BBB extends BB { }
```

Source Code: Src/84_October_21_2020/UseItSol.java

Question

Explain the meaning of the method signatures of *methodA(), methodB()*, and *methodC()*.

Answer:

Question

How would a correct call of the UseIt constructor look like? See marked line 4.
How would a add a new element to aList? See marked line 5.
How would a correct call of *methodA()* look like? See marked line 6.
How would a correct call of *methodB()* look like? See marked line 7.
How would a correct call of *methodC()* look like? See marked line 8.

Answer:

Question 3 (8)

Take a look at the following program:

```
 1      // Runtime vs compile time exception
 2      import java.io.*;
 3
 4      public class ExceptionExample {
 5        static private void trouble() throws IOException {
 6              throw new IOException("something went wrong");
 7        }
 8        static private int a()         {
 9                      try {
10                              trouble();
11                              return 0;
12                      } catch (IOException e) {
13                              return 1;
14                      } catch (Exception e)   {
15                              return 2;
16                      } finally       {
17                              return 3;
18                      }
19        }
20        static private int b()         {
21                      int[] anArray  = new int[1];
22                      try {
23                              anArray[2] = 1 / 0;
24                              return 0;
25                      } catch (ArithmeticException e) {
26                              return 1;
27                      } catch (ArrayIndexOutOfBoundsException e)      {
28                              return 2;
29                      } finally       {
30                              System.exit(1);
31                      }
32        }
33
34        public static void main(String[] args) {
35              System.out.println("1: " + a() );
36              System.out.println("2: " + b() );
37        }
38      }
```

Source Code: Src/84_October_21_2020/ExceptionExample.java

Question

What is the output of the program?

Answer:

```
1: 3
```

trouble() throws excpetion, caught by } catch (IOException e) {, finlay return over writes } catch (IOException e) { return, 3. is returned.

anArray[2] = 1 / 0; causes ArithmeticException, which is caught, fainally terminates the JVM with System.exit.

Question 4 (8)

Take a look at the following program:

```
 1      import java.io.*;
 2      import java.util.*;
 3      import java.lang.StringBuilder;
 4
 5      public class Password implements Serializable {
 6        private String password;
 7        private int    hashCode;
 8
 9        public Password(String password)      {
10              this.password = password;
11              this.hashCode = password.hashCode();
12        }
13        private void writeObject(ObjectOutputStream s) throws IOException {
14              s.defaultWriteObject();
15              s.writeObject(hashCode);
16        }
17
18        private void readObject(ObjectInputStream s) throws IOException  {
19              try {
20
21                      s.defaultReadObject();
22                      int    hashCode = (int)s.readObject();
23
24                      if ( this.password.hashCode() != hashCode )
25                              this.password = "x";
26              }
27              catch ( ClassNotFoundException e)       {
28                  System.out.println(e.getMessage());
29                  e.printStackTrace();
30              }
31        }
32        public String toString()      {
33              return password;
34        }
35
36      }
```

Source Code: Src/84_October_21_2020/PasswordSol.java

Question

Assume a serialized object of Password.java has been written in 1234.ser. The password in this object is "password" 1234.ser might have been modified in such a way that the password is modified to "xassword", but the structure of the file is still intact. Meaning this file can be successfully read by the reader program. Fill in the code in *readObject()* so such "x" will be returned as the password, if the passowrd has been modified. The file wil be read with code like:

```
aPassword = (Password)inputStream.readObject();
```

aPassword will be "x" if 1234.ser was modified, otherwise the content will be the password.

Answer:

Write the code here.

Question 5 (8)

Take a look at the following program:

```
 1
 2      public class Thread_example extends Thread      {
 3
 4              int value = 0;
 5              static int counter = 0;
 6              int id;
 7
 8              public Thread_example (int id) {
 9                      this.id = id;
10                      counter ++;
11              }
12
13              public int getValue() {
14                      return value;
15              }
16              public void compute()   {
17                      if ( id == 1 )  {
18                              value = 1;
19                      } else {
20                              value = 2;
21                      }
22              }
23              public void run () {
24                      compute();
25              }
26              public static void main (String args []) {
27                      Thread_example aT1 = new Thread_example(1);
28                      Thread_example aT2 = new Thread_example(2);
29                      aT1.start();
30                      aT2.start();
31                      System.out.println(counter);
32                      System.out.println(aT1.getValue());
33                      System.out.println(aT2.getValue());
34              }
35      }
```

Source Code: Src/84_October_21_2020/Thread_example.java

Question

List all possible outputs of this program.

Answer:

```
counter will allways be 2, because construcotrs are completed before new threa
```

If the main thread is executed to completion: 0 0.

If the main thread is stopped at line 30, and 1 executes run to the point where a value is assigned (line 18), and the the main thread executes: 1 0.

If the main thread is stopped at line 30, and 2 executes run to the point where a value is assigned (line 20), and the the main thread executes: 0 2

If the main thread is stopped at line 30, and 1 and executes run to the point where a value is assigned (line 18/20), System.out.println(aT1.getValue()) 1 2

System.out.println(aT1.getValue()) 2, or 0

You need to explain your answers. You need to explain your answer with an execution order of the threads. In other words you need to explain to which point in the code the scheduler will let a thread run, stop, and which thread will continue. Words like thread_1, thread_2, scheduler, main thread might be usefull.

44. Final Examination — Dec/3/2020 - version 1

Time: 50 Minutes
Total Points: 40

All code examples do compile unless noted otherwise.

Question 1 (8)

Take a look at the following program:

```
 1      public class X1 extends Thread  {
 2              private String info;
 3              static Object o = new Object();
 4
 5              public X1 (String info) {
 6                      this.info    = info;
 7                      synchronized ( o ) {
 8                              if ( info.equals("0") )
 9                                      ( new X1("1") ).start();
10                      }
11              }
12              public void run () {
13                      for ( int index = 0; index < 3; index ++ )      {
14                              synchronized ( o ) {
15                                      System.out.println(info);
16                                      try {
17                                              o.notify();
18                                              o.wait();
19                                      } catch ( Exception e ) { }
20                              }
21                      }
22              }
23              public static void main (String args []) {
24                      new X1("0").start();
25              }
26      }
```

Source Code: Src/85_December_3/X1.java

This program can produce two, and only two, different outputs. What are the two different outputs? Explain the execution order of the threads so such these two outputs are produced.

Question 2 (8)

Take a look at the following program:

```
 1      public class X2 extends Thread    {
 2          static Object o = new Object();
 3          static int counter = 0;
 4          int id;
 5          public X2(int id)   {
 6              this.id = id;
 7              o       = new Object();
 8          }
 9          public void run () {
10              if ( id == 0 )  {
11                      new X2(1).start();
12                      new X2(2).start();
13                      return;
14              }
15              synchronized ( o ) {
16                      System.err.println(id + " --->");
17                      try {
18                              if ( counter == 0 )     {
19                                      counter = 1;
20                                      o.wait();
21                              } else
22                                      o.notify();
23                      }
24                      catch (  InterruptedException e ) {
25                      }
26                      System.err.println(id + " <---");
27                  }
28          }
29          public static void main (String args []) {
30              new X2(0).start();
31          }
32      }
```

Source Code: Src/85_December_3/X2.java

One possible exection of the program terminates and produces the following output:

```
% java X2  # this version of the execution does terminate
1 --->
2 --->
2 <---
1 <---
%
```

Explain the execution order of the threads so such this output is produced and the program does terminate.

One other possible execution of the program does **not** terminate and produces the following output:

```
% java X2  # this version of the execution does not terminate
1 --->
2 --->
2 <---
```

Question 3 (8)

Take a look at the following program:

```
 1      public class X3 extends Thread    {
 2          Object o = new Object();
 3          public void run () {
 4              synchronized ( o ) {
 5                      System.err.println(" --->");
 6                      System.err.println(" <---");
 7                  }
 8          }
 9          public static void main (String args []) {
10              X3 aX3 = new X3();
11              aX3.run();
12              aX3.start();
13          }
14      }
```

Source Code: Src/85_December_3/X3.java

Is this a possible output?

```
--->
--->
<---
<---
```

Yes/No?. Explain your answer.

Question 4 (8)

Given is the following Client code:

```
 1      import java.rmi.*;
 2
 3      public class HelloC {
 4          public static void main(String args[] ) {
 5              try {
 6                /* 1 HelloInterface obj = (HelloInterface)
 7                                          Naming.lookup("//localhost/HelloSe
 8                */
 9                /* 2 HelloInterface obj = new HelloImplementation();
10                */
11                int[] anArray = {0, 1};
12                obj.methodOne(anArray);
13                System.out.println(anArray[0]);
14              } catch (Exception e) {
15                    e.printStackTrace();
16              }
17         }
18      }
```

Source Code: Src/85_December_3_RMI/HelloC.java

Notice that the lines marked /* 1 */ and /* 2 */ are commented out. One of these lines must be commented in inorder for this code to sucessfully compile and succesfully execute. You can work under the assumption that the code will compile and sucesfully execute if line /* 1 */ or /* 2 */ is commented out.

Implmentation of the interface:

```
 1      import java.rmi.*;
 2      import java.rmi.server.UnicastRemoteObject;
 3
 4      public class HelloImplementation
 5              extends UnicastRemoteObject
 6              implements HelloInterface {
 7
 8              int state;
 9              static int counter;
10
11              public HelloImplementation() throws RemoteException {
12              }
13
14              public void methodOne(int[] anArray) throws RemoteException {
15                      anArray[0] = 42;
16              }
17      }
```

Source Code: Src/85_December_3_RMI/HelloImplementation.java

The code is compiled and executes as follows:

```
% make
javac HelloInterface.java
javac HelloImplementation.java
javac HelloC.java HelloServer.java
rmiregistry &
sleep 4
java HelloServer &
sleep 4
HelloServer bound in registry
java HelloC
42
```

Which of the two lines has been exectured in order to get the outcome 42? You need to explain your answer using words like serialization, remote execution, reference, etc.

Question 5 (8)

Given is the following scenario. You are running your rmi code on glados.cs.rit.edu for a grading session. You can work under the assumption that the code will compile and sucesfully execute. You wrote the following client:

```
 1      import java.rmi.*;
 2
 3      public class HelloC {
 4          public static void main(String args[] ) {
 5              try {
 6               HelloInterface obj = (HelloInterface)
 7                                          Naming.lookup("//localhost/HelloSe
 8                System.out.println( obj.methodOne() );
 9              } catch (Exception e) {
10                     e.printStackTrace();
11              }
12         }
13      }
```

Source Code: Src/85_December_3_RMI_2/HelloC.java

and he following implemenation:

```
 1      import java.rmi.*;
 2      import java.rmi.server.UnicastRemoteObject;
 3
 4      public class HelloImplementation
 5              extends UnicastRemoteObject
 6              implements HelloInterface {
 7
 8              int state;
 9              static int counter;
10
11              public HelloImplementation() throws RemoteException {
12              }
13
14              public String methodOne() throws RemoteException {
15                      return "blue";
16              }
17      }
```

Source Code: Src/85_December_3_RMI_2/HelloImplementation.java

The execution of your code is like this:

```
% date
Tue Nov 24 15:08:54 EST 2020
% make all
javac HelloInterface.java
javac HelloImplementation.java
# rmic HelloImplementation
```

```
javac HelloC.java HelloServer.java
rmiregistry &
sleep 4
java HelloServer &
sleep 4
HelloServer bound in registry
java HelloC
blue
```

After a few minutes your graders ask you to execute the client again. No code has been modified or recompiled. You execute excatcly the same code.

Now the execution of your code is like this:

```
% date
Tue Nov 24 15:12:35 EST 2020
% java HelloC
red
```

Question: How can it be that the unmodified client first print **blue** and then **red**. Explain your answer.

45. Final Examination — Dec/3/2020 - version 1

Time: 50 Minutes
Total Points: 40

All code examples do compile unless noted otherwise.

Question 1 (8)

Take a look at the following program:

```
 1      public class X1 extends Thread  {
 2              private String info;
 3              static Object o = new Object();
 4
 5              public X1 (String info) {
 6                      this.info    = info;
 7                      synchronized ( o ) {
 8                              if ( info.equals("0") )
 9                                      ( new X1("1") ).start();
10                      }
11              }
12              public void run () {
13                      for ( int index = 0; index < 3; index ++ )      {
14                              synchronized ( o ) {
15                                      System.out.println(info);
16                                      try {
17                                              o.notify();
18                                              o.wait();
19                                      } catch ( Exception e ) { }
20                              }
21                      }
22              }
23              public static void main (String args []) {
24                      new X1("0").start();
25              }
26      }
```

Source Code: Src/85_December_3/X1.java

This program can produce two, and only two, different outputs. What are the two different outputs? Explain the execution order of the threads so such these two outputs are produced.

```
 1      /*
 2              b is synchronized on a static object which exist exaclty once.
 3              Both threads, 0 and 1, have the same chance to enter the synch
 4
 5              a)      0 goes in first and prints out 0 and notify's nobody a
 6                      1 goes in second and prints out 1 and notify's 0 then
 7                      0 goes in again and prints out 0 and notify's 1 then g
 8                      1 goes in again and prints out 1 and notify's 0 then g
 9                      0 goes in again and prints out 0 and notify's 1 then g
10                      1 goes in again and prints out 1 and notify's 0 then g
11                      for loop terminates
12                      Output: 0 1 0 1 0 1
13              b)      1 goes in first, replace 1 with 0 and 1 with 0 from a)
14                      Output: 1 0 1 0 1 0
15
16              In either case the program will not terminate because one thre
17      */
```

```
18      public class X1 extends Thread  {
19              private String info;
20              static Object o = new Object();
21
22              public X1 (String info) {
23                      this.info    = info;
24                      synchronized ( o ) {
25                              if ( info.equals("0") )
26                                      ( new X1("1") ).start();
27                      }
28              }
29
30              private void sleep (int msec) {
31                      try {
32                              Thread.sleep(msec);
33                      } catch ( Exception e ) { }
34              }
35              public void run () {
36                      for ( int index = 0; index < 3; index ++ )      {
37                              if ( info == "1" ) sleep(100);          // 0 1
38                              //if ( info == "1" ) sleep(100);        // 1 0
39                              synchronized ( o ) {
40                                      System.out.println(info);
41                                      try {
42                                              o.notify();
43                                              o.wait();
44                                      } catch ( Exception e ) { }
45                              }
46                      }
47              }
48              public static void main (String args []) {
49                      new X1("0").start();
50              }
51      }
```

Source Code: Src/85_December_3_sol/X1.java

Question 2 (8)

Take a look at the following program:

```
 1      public class X2 extends Thread    {
 2          static Object o = new Object();
 3          static int counter = 0;
 4          int id;
 5          public X2(int id)   {
 6              this.id = id;
 7              o       = new Object();
 8          }
 9          public void run () {
10              if ( id == 0 )  {
11                      new X2(1).start();
12                      new X2(2).start();
13                      return;
14              }
15              synchronized ( o ) {
16                      System.err.println(id + " --->");
17                      try {
18                              if ( counter == 0 )     {
19                                      counter = 1;
20                                      o.wait();
21                              } else
22                                      o.notify();
23                      }
24                      catch (  InterruptedException e ) {
25                      }
26                      System.err.println(id + " <---");
27                  }
28          }
29          public static void main (String args []) {
30              new X2(0).start();
31          }
32      }
```

Source Code: Src/85_December_3/X2.java

One possible exection of the program terminates and produces the following output:

```
% java X2  # this version of the execution does terminate
1 --->
2 --->
2 <---
1 <---
%
```

Explain the execution order of the threads so such this output is produced and the program does terminate.

```
1       /*
2               There are 2 cases:
3                       a)      0 creates 1, 1 creates object value o1
4                               0 stops at line a)
5                               1 continues and stops at line d
6                               0 continues and creates 2 and object has now v
7                               0 returns
8                               2 continues to line h
9                                       -->
10                              1 can now enter the syncronized block
11                                      -->
12                                      counter is 1
13                                      sends notify to o2
14                                      and exists the synchronized block
15                                      <--
16                              2 is notified
17                                      <--
18                              program terminates
19                      b)      0 creates 1, 1 creates object value o1
20                              1 enters run enters sync and  is stopped after
21                              0 creates 2, 2 creates object value o2
22                              value of o is no o2
23                              1 continues and wait on o2 which was not used
24                              Monitor Exception.
25                      c)      0 creates 1 and 1 creates o1 and executes o1.w
26                              0 creates 2 and 2 creates o2 and executes o2.n
27                              1 will sit in the o1.wait until the program is
28                              -->
29                              -->
30                              <--
31
32
33      */
34
35      public class X2 extends Thread    {
36          static Object o = new Object();
37          static int counter = 0;
38          int id;
39          public X2(int id)   {
40              this.id = id;
41              o       = new Object();
42          }
43          public void run () {
44              if ( id == 0 )  {
45                      new X2(1).start();                      // a
46                      // try { sleep(1000); } catch (  InterruptedException
47                      new X2(2).start();                      // c
48                      return;
49              }
50                                                              // d
51              synchronized ( o ) {
52              System.out.println(id + "/" + o );
53                      System.err.println(id + " --->");               // e
54                      if ( id == 1 )  try { sleep(1000); } catch (  Interrup
```

```
55                      try {
56                              if ( counter == 0 )     {       // f
57                                      counter = 1;            // g
58                                      System.out.println(id + "/wait " + o )
59                                      o.wait();               // h
60                              } else
61                                      System.out.println(id + "/notify " + o
62                                      o.notify();             // e
63                      }
64                      catch (  InterruptedException e ) {
65                      }
66                      System.err.println(id + " <---");
67                  }
68          }
69          public static void main (String args []) {
70              new X2(0).start();
71          }
72      }
```

Source Code: Src/85_December_3_sol/X2.java

One other possible execution of the program does **not** terminate and produces the following output:

```
% java X2  # this version of the execution does not terminate
1 --->
2 --->
2 <---
```

```
 1      /*
 2       --->
 3      1/wait          java.lang.Object@3c41b509
 4       --->
 5      2/notify        java.lang.Object@58da68fa
 6       <---
 7
 8      Program does not terminate
 9
10      notify is send to the value of o generated by thread 2
11      */
12      public class X2b extends Thread    {
13          static Object o = new Object();
14          static int counter = 0;
15          int id;
16          public X2b(int id)  {
17              this.id = id;
18              o       = new Object();
19          }
20          private void sleep (int msec) {
21              try {
22                      Thread.sleep(msec);
23              } catch ( Exception e ) { }
```

```
24          }
25          public void run () {
26              if ( id == 0 )  {
27                      new X2b(1).start();                     // a
28                      sleep(100);
29                      new X2b(2).start();                     // c
30                      return;
31              }
32                                                              // d
33              synchronized ( o ) {
34                      System.err.println(id + " --->");               // e
35                      try {
36                              if ( counter == 0 )     {       // f
37                                      counter = 1;            // g
38                                      System.out.println(id + "/wait
39                                      o.wait();               // h
40                              } else
41                                      System.out.println(id + "/notify
42                                      o.notify();             // e
43                      }
44                      catch (  InterruptedException e ) {
45                      }
46                      System.err.println(id + " <---");
47                  }
48          }
49          public static void main (String args []) {
50              new X2b(0).start();
51          }
52      }
```

Source Code: Src/85_December_3_sol/X2b.java

Question 3 (8)

Take a look at the following program:

```
 1      public class X3 extends Thread    {
 2          Object o = new Object();
 3          public void run () {
 4              synchronized ( o ) {
 5                      System.err.println(" --->");
 6                      System.err.println(" <---");
 7                  }
 8          }
 9          public static void main (String args []) {
10              X3 aX3 = new X3();
11              aX3.run();      // At this point only one thread of execution
12                              // the main thread execute run and prints
13                              // --->
14                              // <--
15              aX3.start();    // the main thread creates an other thread of
16                              // this thread is put in the list of available
17                              // scheduler sends run to this thread at one p
18                              // this thread prints
19                              // --->
20                              // <--
21                              // this is the only possible output.
22                              // the output shown is not possible
23
24          }
25      }
```

Source Code: Src/85_December_3_sol/X3.java

Is this a possible output?

```
--->
--->
<---
<---
```

Yes/No?. Explain your answer.

Question 4 (8)

Given is the following Client code:

```
 1      import java.rmi.*;
 2
 3      public class HelloC {
 4          public static void main(String args[] ) {
 5              try {
 6                /* 1 HelloInterface obj = (HelloInterface)
 7                                          Naming.lookup("//localhost/HelloSe
 8                */
 9                /* 2 HelloInterface obj = new HelloImplementation();
10                */
11                int[] anArray = {0, 1};
12                obj.methodOne(anArray);
13                System.out.println(anArray[0]);
14              } catch (Exception e) {
15                    e.printStackTrace();
16              }
17         }
18      }
```

Source Code: Src/85_December_3_RMI/HelloC.java

Notice that the lines marked /* 1 */ and /* 2 */ are commented out. One of these lines must be commented in inorder for this code to sucessfully compile and succesfully execute. You can work under the assumption that the code will compile and sucesfully execute if line /* 1 */ or /* 2 */ is commented out.

Implmentation of the interface:

```
 1      import java.rmi.*;
 2      import java.rmi.server.UnicastRemoteObject;
 3
 4      public class HelloImplementation
 5              extends UnicastRemoteObject
 6              implements HelloInterface {
 7
 8              int state;
 9              static int counter;
10
11              public HelloImplementation() throws RemoteException {
12              }
13
14              public void methodOne(int[] anArray) throws RemoteException {
15                      anArray[0] = 42;
16              }
17      }
```

Source Code: Src/85_December_3_RMI_sol/HelloImplementation.java

The code is compiled and executes as follows:

```
% make
javac HelloInterface.java
javac HelloImplementation.java
javac HelloC.java HelloServer.java
rmiregistry &
sleep 4
java HelloServer &
sleep 4
HelloServer bound in registry
java HelloC
42
```

Which of the two lines has been exectured in order to get the outcome 42? You need to explain your answer using words like serialization, remote execution, reference, etc.

```
 1      import java.rmi.*;
 2
 3      public class HelloC {
 4          public static void main(String args[] ) {
 5              try {
 6                /* 1 HelloInterface obj = (HelloInterface)
 7                                          Naming.lookup("//localhost/HelloSe
 8                */
 9                HelloInterface obj = new HelloImplementation();
10                      // because the array is modified
11                      // if it owuld be 1) then the array would be serialize
12                      // send to the server, modified in the server space ar
13                int[] anArray = {0, 1};
14                obj.methodOne(anArray);
15                System.out.println(anArray[0]);
16              } catch (Exception e) {
17                      e.printStackTrace();
18              }
19          }
20      }
```

Source Code: Src/85_December_3_RMI_sol/HelloC.java

Question 5 (8)

Given is the following scenario. You are running your rmi code on glados.cs.rit.edu for a grading session. You can work under the assumption that the code will compile and sucesfully execute. You wrote the following client:

```
 1      import java.rmi.*;
 2
 3      public class HelloC {
 4          public static void main(String args[] ) {
 5              try {
 6               HelloInterface obj = (HelloInterface)
 7                                          Naming.lookup("//localhost/HelloSe
 8                System.out.println( obj.methodOne() );
 9              } catch (Exception e) {
10                     e.printStackTrace();
11              }
12         }
13      }
```

Source Code: Src/85_December_3_RMI_2/HelloC.java

and he following implemenation:

```
 1      import java.rmi.*;
 2      import java.rmi.server.UnicastRemoteObject;
 3
 4      public class HelloImplementation
 5              extends UnicastRemoteObject
 6              implements HelloInterface {
 7
 8              int state;
 9              static int counter;
10
11              public HelloImplementation() throws RemoteException {
12              }
13
14              public String methodOne() throws RemoteException {
15                      return "blue";
16              }
17      }
```

Source Code: Src/85_December_3_RMI_2/HelloImplementation.java

The execution of your code is like this:

```
% date
Tue Nov 24 15:08:54 EST 2020
% make all
javac HelloInterface.java
javac HelloImplementation.java
# rmic HelloImplementation
```

```
javac HelloC.java HelloServer.java
rmiregistry &
sleep 4
java HelloServer &
sleep 4
HelloServer bound in registry
java HelloC
blue
```

After a few minutes your graders ask you to execute the client again. No code has been modified or recompiled. You execute excatly the same code.

Now the execution of your code is like this:

```
% date
Tue Nov 24 15:12:35 EST 2020
% java HelloC
red
```

Question: How can it be that the unmodified client first print **blue** and then **red**? Explain your answer.

Answer: A other server:

```
 1      import java.rmi.*;
 2
 3      public class HelloServer2 {
 4
 5              public static void main(String args[])
 6              {
 7                      // System.setSecurityManager(new RMISecurityManager())
 8
 9                      try {
10                          HelloInterface obj = new HelloImplementation2();
11                          Naming.rebind("//localhost/HelloServer", obj);
12                          // Naming.rebind("//129.21.36.56/HelloServer", obj
13                          System.out.println("HelloServer bound in registry'
14                      } catch (Exception e) {
15                          System.out.println("HelloImpl err: " + e.getMessag
16                          e.printStackTrace();
17                          System.exit(0);
18                      }
19              }
20      }
```

Source Code: Src/85_December_3_RMI_2_sol/HelloServer2.java

with this implementation was started.

```
 1      import java.rmi.*;
 2      import java.rmi.server.UnicastRemoteObject;
 3
 4      public class HelloImplementation2
 5              extends UnicastRemoteObject
```

```
 6              implements HelloInterface {
 7
 8              int state;
 9              static int counter;
10
11              public HelloImplementation2() throws RemoteException {
12              }
13
14              public String methodOne() throws RemoteException {
15                      return "red";
16              }
17      }
```

Source Code: Src/85_December_3_RMI_2_sol/HelloImplementation2.java

The original object in rmiregistery was over written.