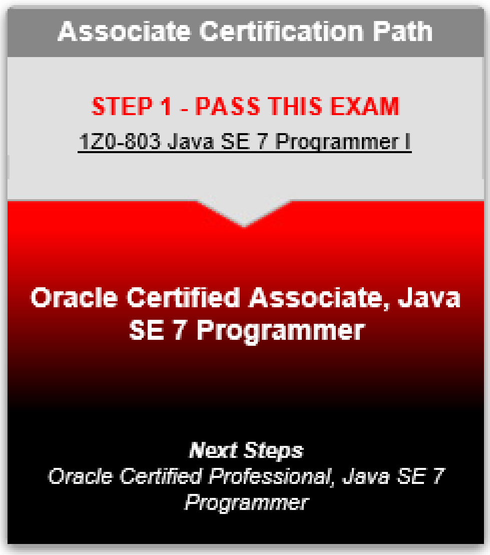
WWW.ARJUNSRIDHARUR.ORG

PIRATES OF THE KNOWLEDGE HACKER
VOYAGE TO THE CORE JAVA

Preface

INDEX

Java Certification



Associate Certification Path

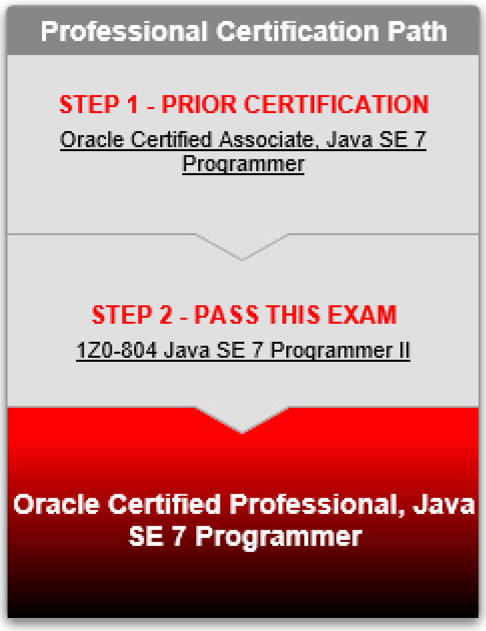
STEP 1 - PASS THIS EXAM

1Z0-803 Java SE 7 Programmer I

Oracle Certified Associate, Java SE 7 Programmer

Next Steps

Oracle Certified Professional, Java SE 7 Programmer



Professional Certification Path

STEP 1 - PRIOR CERTIFICATION

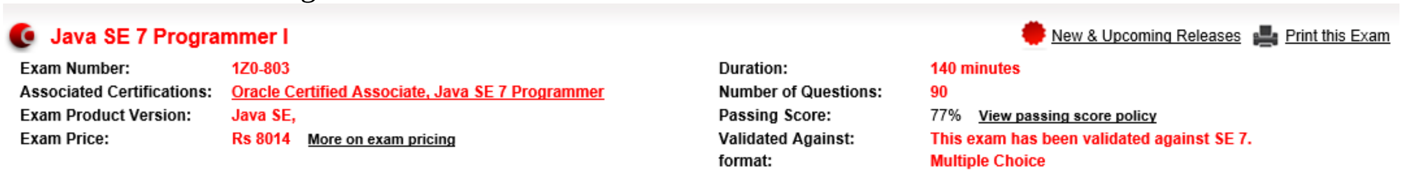
Oracle Certified Associate, Java SE 7 Programmer

STEP 2 - PASS THIS EXAM

1Z0-804 Java SE 7 Programmer II

Oracle Certified Professional, Java SE 7 Programmer

1Z0-803 Java SE 7 Programmer I Oracle Certified Associate

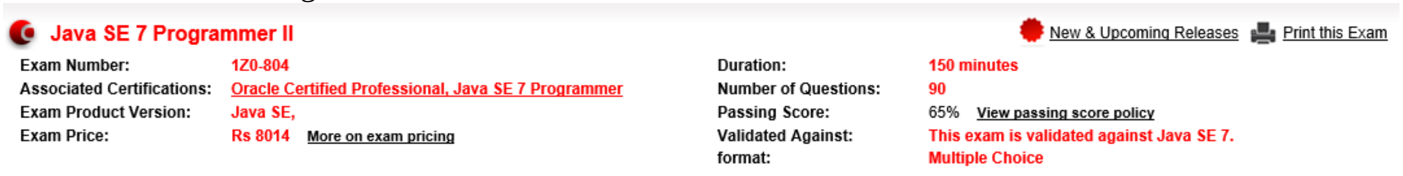


Java SE 7 Programmer I

New & Upcoming Releases Print this Exam

Exam Number:	1Z0-803	Duration:	140 minutes
Associated Certifications:	Oracle Certified Associate, Java SE 7 Programmer	Number of Questions:	90
Exam Product Version:	Java SE,	Passing Score:	77% View passing score policy
Exam Price:	Rs 8014 More on exam pricing	Validated Against:	This exam has been validated against SE 7.
		format:	Multiple Choice

1Z0-804 Java SE 7 Programmer II Oracle Certified Professional

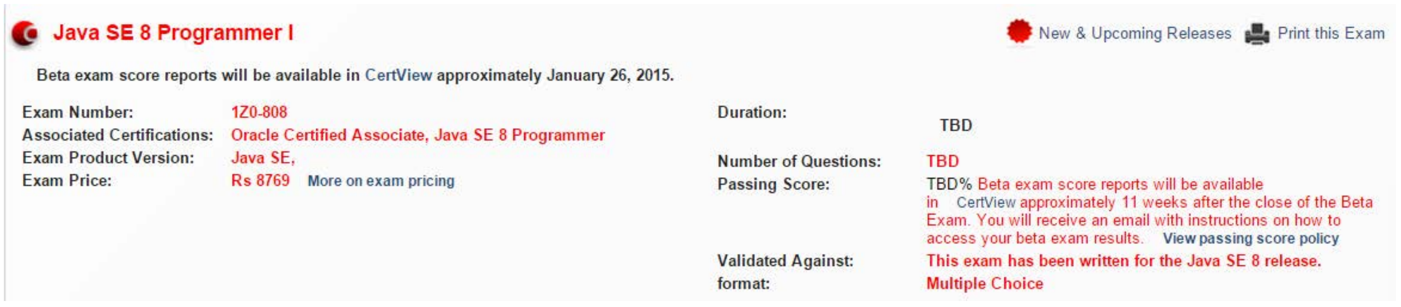


Java SE 7 Programmer II

New & Upcoming Releases Print this Exam

Exam Number:	1Z0-804	Duration:	150 minutes
Associated Certifications:	Oracle Certified Professional, Java SE 7 Programmer	Number of Questions:	90
Exam Product Version:	Java SE,	Passing Score:	65% View passing score policy
Exam Price:	Rs 8014 More on exam pricing	Validated Against:	This exam is validated against Java SE 7.
		format:	Multiple Choice

Wait For the NEW Certifitation



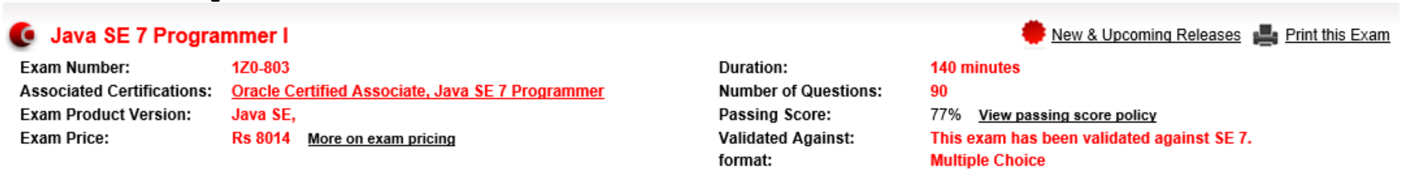
Java SE 8 Programmer I

New & Upcoming Releases Print this Exam

Beta exam score reports will be available in CertView approximately January 26, 2015.

Exam Number:	1Z0-808	Duration:	TBD
Associated Certifications:	Oracle Certified Associate, Java SE 8 Programmer	Number of Questions:	TBD
Exam Product Version:	Java SE,	Passing Score:	TBD% Beta exam score reports will be available in CertView approximately 11 weeks after the close of the Beta Exam. You will receive an email with instructions on how to access your beta exam results. View passing score policy
Exam Price:	Rs 8769 More on exam pricing	Validated Against:	This exam has been written for the Java SE 8 release.
		format:	Multiple Choice

1Z0-803 Java SE 7 Programmer I Oracle Certified Associate



Java SE 7 Programmer I

New & Upcoming Releases Print this Exam

Exam Number:	1Z0-803	Duration:	140 minutes
Associated Certifications:	Oracle Certified Associate, Java SE 7 Programmer	Number of Questions:	90
Exam Product Version:	Java SE,	Passing Score:	77% View passing score policy
Exam Price:	Rs 8014 More on exam pricing	Validated Against:	This exam has been validated against SE 7.
		format:	Multiple Choice

Java Basics

- Define the scope of variables
- Define the structure of a Java class
- Create executable Java applications with a main method
- Import other Java packages to make them accessible in your code

Working With Java Data Types

- Declare and initialize variables
- Differentiate between object reference variables and primitive variables
- Read or write to object fields
- Explain an Object's Lifecycle (creation, "dereference" and garbage collection)
- Call methods on objects
- Manipulate data using the StringBuilder class and its methods
- Creating and manipulating Strings

Using Operators and Decision Constructs

- Use Java operators
- Use parenthesis to override operator precedence
- Test equality between Strings and other objects using == and equals ()
- Create if and if/else constructs
- Use a switch statement

Creating and Using Arrays

- Declare, instantiate, initialize and use a one-dimensional array
- Declare, instantiate, initialize and use multi-dimensional array
- Declare and use an ArrayList

Using Loop Constructs

- Create and use while loops
- Create and use for loops including the enhanced for loop
- Create and use do/while loops
- Compare loop constructs
- Use break and continue

Working with Methods and Encapsulation

- Create methods with arguments and return values
- Apply the static keyword to methods and fields
- Create an overloaded method
- Differentiate between default and user defined constructors
- Create and overload constructors
- Apply access modifiers
- Apply encapsulation principles to a class

- Determine the effect upon object references and primitive values when they are passed into methods that change the values

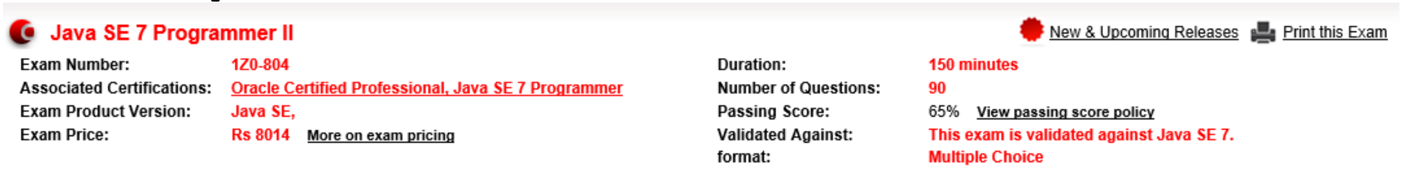
Working with Inheritance

- Implement inheritance
- Develop code that demonstrates the use of polymorphism
- Differentiate between the type of a reference and the type of an object
- Determine when casting is necessary
- Use super and this to access objects and constructors
- Use abstract classes and interfaces

Handling Exceptions

- Differentiate among checked exceptions, RuntimeExceptions and Errors
- Create a try-catch block and determine how exceptions alter normal program flow
- Describe what Exceptions are used for in Java
- Invoke a method that throws an exception
- Recognize common exception classes and categories

1Z0-804 Java SE 7 Programmer II Oracle Certified Professional



Java SE 7 Programmer II

New & Upcoming Releases Print this Exam

Exam Number:	1Z0-804	Duration:	150 minutes
Associated Certifications:	Oracle Certified Professional, Java SE 7 Programmer	Number of Questions:	90
Exam Product Version:	Java SE,	Passing Score:	65% View passing score policy
Exam Price:	Rs 8014 More on exam pricing	Validated Against:	This exam is validated against Java SE 7.
		format:	Multiple Choice

Java Class Design

- Use access modifiers: private, protected, and public
- Override methods
- Overload constructors and methods
- Use the instanceof operator and casting
- Use virtual method invocation
- Override the hashCode, equals, and toString methods from the Object class to improve the functionality of your class.
- Use package and import statements

Advanced Class Design

- Identify when and how to apply abstract classes
- Construct abstract Java classes and subclasses
- Use the static and final keywords
- Create top-level and nested classes
- Use enumerated types

Object-Oriented Design Principles

- Write code that declares, implements and/or extends interfaces
- Choose between interface inheritance and class inheritance
- Apply cohesion, low-coupling, IS-A, and HAS-A principles
- Apply object composition principles (including has-a relationships)
- Design a class using a Singleton design pattern
- Write code to implement the Data Access Object (DAO) pattern
- Design and create objects using a factory pattern

Generics and Collections

- Create a generic class
- Use the diamond for type inference
- Analyze the interoperability of collections that use raw types and generic types
- Use wrapper classes, autoboxing and unboxing
- Create and use List, Set and Deque implementations
- Create and use Map implementations
- Use java.util.Comparator and java.lang.Comparable
- Sort and search arrays and lists

String Processing

- Search, parse and build strings (including Scanner, StringTokenizer, StringBuilder, String and Formatter)
- Search, parse, and replace strings by using regular expressions, using expression patterns for matching limited to: . (dot), * (star), + (plus), ?, \d, \D, \s, \S, \w, \W, \b. \B, [], ().
- Format strings using the formatting parameters: %b, %c, %d, %f, and %s in format strings.

Exceptions and Assertions

- Use throw and throws statements

- Develop code that handles multiple Exception types in a single catch block
- Develop code that uses try-with-resources statements (including using classes that implement the AutoCloseable interface)
- Create custom exceptions
- Test invariants by using assertions

Java I/O Fundamentals

- Read and write data from the console
- Use streams to read from and write to files by using classes in the java.io package (including BufferedReader, BufferedWriter, File, FileReader, FileWriter, DataReader, ObjectOutputStream, ObjectInputStream, and PrintWriter)

Java File I/O (NIO.2)

- Operate on file and directory paths with the Path class
- Check, delete, copy, or move a file or directory with the Files class
- Read and change file and directory attributes, focusing on the BasicFileAttributes, DosFileAttributes, and PosixFileAttributes interfaces
- Recursively access a directory tree using the DirectoryStream and FileVisitor interfaces
- Find a file with the PathMatcher interface
- Watch a directory for changes with the WatchService interface

Building Database Applications with JDBC

- Describe the interfaces that make up the core of the JDBC API (including the Driver, Connection, Statement, and ResultSet interfaces and their relationship to provider implementations)
- Identify the components required to connect to a database using the DriverManager class (including the jdbc URL)
- Submit queries and read results from the database (including creating statements, returning result sets, iterating through the results, and properly closing result sets, statements, and connections)
- Use JDBC transactions (including disabling auto-commit mode, committing and rolling back transactions, and setting and rolling back to savepoints)
- Construct and use RowSet objects using the RowSetProvider class and the RowSetFactory interface
- Create and use PreparedStatement and CallableStatement objects

Threads

- Create and use the Thread class and the Runnable interface
- Manage and control thread lifecycle
- Synchronize thread access to shared data
- Identify code that may not execute correctly in a multi-threaded environment.

Concurrency

- Use collections from the java.util.concurrent package with a focus on the advantages over and differences from the traditional java.util collections.
- Use Lock, ReadWriteLock, and ReentrantLock classes in the java.util.concurrent.locks package to support lock-free thread-safe programming on single variables.
- Use Executor, ExecutorService, Executors, Callable, and Future to execute tasks using thread pools.
- Use the parallel Fork/Join Framework

Localization

- Read and set the locale by using the Locale object
- Build a resource bundle for each locale
- Call a resource bundle from an application
- Format dates, numbers, and currency values for localization with the NumberFormat and DateFormat classes (including number format patterns)
- Describe the advantages of localizing an application
- Define a locale using language and country codes

What is a platform?
A platform is the hardware or software environment in which a program runs. Most platforms can be described as a combination of the operating system and hardware, like Windows 2000/XP, Linux, Solaris, and MacOS.

What is the main difference between Java platform and other platforms?
The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.
The Java platform has two components:
The Java Virtual Machine (Java VM)
The Java Application Programming Interface (Java API)

What is the Java Virtual Machine?
The Java Virtual Machine is a software that can be ported onto various hardware-based platforms.

What is the Java API?
The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets.

What is the package?
The package is a Java namespace or part of Java libraries. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as packages.

What is native code?
The native code is code that after you compile it, the compiled code runs on a specific hardware platform.

Is Java code slower than native code?
Not really. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time bytecode compilers can bring performance close to that of native code without threatening portability.

What is the serialization?
The serialization is a kind of mechanism that makes a class or a bean persistence by having its properties or fields and state information saved and restored to and from storage.

How to make a class or a bean serializable?
By implementing either the java.io.Serializable interface, or the java.io.Externalizable interface. As long as one class in a class's inheritance hierarchy implements Serializable or Externalizable, that class is serializable.

How many methods in the Serializable interface?
There is no method in the Serializable interface. The Serializable interface acts as a marker, telling the object serialization tools that your class is serializable.

How many methods in the Externalizable interface?
There are two methods in the Externalizable interface. You have to implement these two methods in order to make your class externalizable. These two methods are readExternal() and writeExternal().

What is the difference between Serializalble and Externalizable interface?
When you use Serializable interface, your class is serialized automatically by default. But you can override writeObject() and readObject() two methods to control more complex object serailization process. When you use Externalizable interface, you have a complete control over your class's serialization process.

What is a transient variable?
A transient variable is a variable that may not be serialized. If you don't want some field to be serialized, you can mark that field transient or static.

Which containers use a border layout as their default layout?
The Window, Frame and Dialog classes use a border layout as their default layout.

How are Observer and Observable used?
Objects that subclass the Observable class maintain a list of observers. When an Observable object is updated it invokes the update() method of each of its observers to notify the observers that it has changed state. The Observer interface is implemented by objects that observe Observable objects.

What is synchronization and why is it important?

With respect to multithreading, synchronization is the capability to control the access of multiple threads to shared resources. Without synchronization, it is possible for one thread to modify a shared object while another thread is in the process of using or updating that object's value. This often causes dirty data and leads to significant errors.

What are synchronized methods and synchronized statements?

Synchronized methods are methods that are used to control access to an object. A thread only executes a synchronized method after it has acquired the lock for the method's object or class. Synchronized statements are similar to synchronized methods. A synchronized statement can only be executed after a thread has acquired the lock for the object or class referenced in the synchronized statement.

What are three ways in which a thread can enter the waiting state?

A thread can enter the waiting state by invoking its sleep() method, by blocking on I/O, by unsuccessfully attempting to acquire an object's lock, or by invoking an object's wait() method. It can also enter the waiting state by invoking its (deprecated) suspend() method.

Can a lock be acquired on a class?

Yes, a lock can be acquired on a class. This lock is acquired on the class's Class object.
What's new with the stop(), suspend() and resume() methods in JDK 1.2?
The stop(), suspend() and resume() methods have been deprecated in JDK 1.2.

What is the preferred size of a component?

The preferred size of a component is the minimum component size that will allow the component to display normally.

What method is used to specify a container's layout?

The setLayout() method is used to specify a container's layout.

Which containers use a FlowLayout as their default layout?

The Panel and Applet classes use the FlowLayout as their default layout.

What is thread?

A thread is an independent path of execution in a system.

What is multithreading?

Multithreading means various threads that run in a system.

How does multithreading take place on a computer with a single CPU?

The operating system's task scheduler allocates execution time to multiple tasks. By quickly switching between executing tasks, it creates the impression that tasks execute sequentially.

How to create multithread in a program?

You have two ways to do so. First, making your class "extends" Thread class. Second, making your class "implements" Runnable interface. Put jobs in a run() method and call start() method to start the thread.

Can Java object be locked down for exclusive use by a given thread?

Yes. You can lock an object by putting it in a "synchronized" block. The locked object is inaccessible to any thread other than the one that explicitly claimed it.

Can each Java object keep track of all the threads that want to exclusively access to it?

Yes.

What state does a thread enter when it terminates its processing?

When a thread terminates its processing, it enters the dead state.

What invokes a thread's run() method?

After a thread is started, via its start() method of the Thread class, the JVM invokes the thread's run() method when the thread is initially executed.

What is the purpose of the wait(), notify(), and notifyAll() methods?

The wait(),notify(), and notifyAll() methods are used to provide an efficient way for threads to communicate each other.

What are the high-level thread states?

The high-level thread states are ready, running, waiting, and dead.

What is the Collections API?

The Collections API is a set of classes and interfaces that support operations on collections of objects.

What is the List interface?

The List interface provides support for ordered collections of objects.

How does Java handle integer overflows and underflows?

It uses those low order bytes of the result that can fit into the size of the type allowed by the operation.

What is the Vector class?

The Vector class provides the capability to implement a growable array of objects What modifiers may be used with an inner class that is a member of an outer class? A (non-local) inner class may be declared as public, protected, private, static, final, or abstract.

If a method is declared as protected, where may the method be accessed?

A protected method may only be accessed by classes or interfaces of the same package or by subclasses of the class in which it is declared.

What is an Iterator interface?

The Iterator interface is used to step through the elements of a Collection.

How many bits are used to represent Unicode, ASCII, UTF-16, and UTF-8 characters?

Unicode requires 16 bits and ASCII require 7 bits. Although the ASCII character set uses only 7 bits, it is usually represented as 8 bits. UTF-8 represents characters using 8, 16, and 18 bit patterns. UTF-16 uses 16-bit and larger bit patterns.

What is the difference between yielding and sleeping?

When a task invokes its yield() method, it returns to the ready state. When a task invokes its sleep() method, it returns to the waiting state.

Is sizeof a keyword?

The sizeof operator is not a keyword in Java.

What are wrapped classes?

Wrapped classes are classes that allow primitive types to be accessed as objects.

Does garbage collection guarantee that a program will not run out of memory?

No, it doesn't. It is possible for programs to use up memory resources faster than they are garbage collected. It is also possible for programs to create objects that are not subject to garbage collection

What is the difference between preemptive scheduling and time slicing?

Under preemptive scheduling, the highest priority task executes until it enters the waiting or dead states or a higher priority task comes into existence. Under time slicing, a task executes for a predefined slice of time and then reenters the pool of ready tasks. The scheduler then determines which task should execute next, based on priority and other factors.

Name Component subclasses that support painting.

The Canvas, Frame, Panel, and Applet classes support painting.

What is a native method?

A native method is a method that is implemented in a language other than Java.

How can you write a loop indefinitely?

for(;;)--for loop; while(true)--always true, etc.

Can an anonymous class be declared as implementing an interface and extending a class?

An anonymous class may implement an interface or extend a superclass, but may not be declared to do both.

What is the purpose of finalization?

The purpose of finalization is to give an unreachable object the opportunity to perform any cleanup processing before the object is garbage collected.

Which class is the superclass for every class.

Object

What is the difference between the Boolean & operator and the && operator?

If an expression involving the Boolean & operator is evaluated, both operands are evaluated. Then the & operator is applied to the operand. When an expression involving the && operator is evaluated, the first operand is evaluated. If the first operand returns a value of true then the second operand is evaluated. The && operator is then applied to the first and second operands. If the first operand evaluates to false, the evaluation of the second operand is skipped.
Operator & has no chance to skip both sides evaluation and && operator does. If asked why, give details as above.

What is the GregorianCalendar class?
The GregorianCalendar provides support for traditional Western calendars.

What is the SimpleTimeZone class?
The SimpleTimeZone class provides support for a Gregorian calendar.

Which Container method is used to cause a container to be laid out and redisplayed?
validate()

What is the Properties class?
The properties class is a subclass of Hashtable that can be read from or written to a stream. It also provides the capability to specify a set of default values to be used.

What is the purpose of the Runtime class?
The purpose of the Runtime class is to provide access to the Java runtime system.

What is the purpose of the System class?
The purpose of the System class is to provide access to system resources.

What is the purpose of the finally clause of a try-catch-finally statement?
The finally clause is used to provide the capability to execute code no matter whether or not an exception is thrown or caught.

What is the Locale class?
The Locale class is used to tailor program output to the conventions of a particular geographic, political, or cultural region.

What must a class do to implement an interface?
It must provide all of the methods in the interface and identify the interface in its implements clause.

What is an abstract method?
An abstract method is a method whose implementation is deferred to a subclass. Or, a method that has no implementation (an interface of a method).

What is a static method?
A static method is a method that belongs to the class rather than any object of the class and doesn't apply to an object or even require that any objects of the class have been instantiated.

What is a protected method?
A protected method is a method that can be accessed by any method in its package and inherited by any subclass of its class.

What is the difference between a static and a non-static inner class?
A non-static inner class may have object instances that are associated with instances of the class's outer class. A static inner class does not have any object instances.

What is an object's lock and which object's have locks?
An object's lock is a mechanism that is used by multiple threads to obtain synchronized access to the object. A thread may execute a synchronized method of an object only after it has acquired the object's lock. All objects and classes have locks. A class's lock is acquired on the class's Class object.

When can an object reference be cast to an interface reference?
An object reference be cast to an interface reference when the object implements the referenced interface.

What is the difference between a Window and a Frame?
The Frame class extends Window to define a main application window that can have a menu bar.

Which package has light weight components?

javax.Swing package. All components in Swing, except JApplet, JDialog, JFrame and JWindow are lightweight components.

What happens when a thread cannot acquire a lock on an object?

If a thread attempts to execute a synchronized method or synchronized statement and is unable to acquire an object's lock, it enters the waiting state until the lock becomes available.

What is the difference between the Reader/Writer class hierarchy and the InputStream/OutputStream class hierarchy?

The Reader/Writer class hierarchy is character-oriented, and the InputStream/OutputStream class hierarchy is byte-oriented.

What classes of exceptions may be caught by a catch clause?

A catch clause can catch any exception that may be assigned to the Throwable type. This includes the Error and Exception types.

What is the difference between throw and throws keywords?

The throw keyword denotes a statement that causes an exception to be initiated. It takes the Exception object to be thrown as argument. The exception will be caught by an immediately encompassing try-catch construction or propagated further up the calling hierarchy.

The throws keyword is a modifier of a method that designates that exceptions may come out of the mehtod, either by virtue of the method throwing the exception itself or because it fails to catch such exceptions that a me thod it calls may throw.

If a class is declared without any access modifiers, where may the class be accessed?

A class that is declared without any access modifiers is said to have package or friendly access. This means that the class can only be accessed by other classes and interfaces that are defined within the same package.

What is the Map interface?

The Map interface replaces the JDK 1.1 Dictionary class and is used associate keys with values.

Does a class inherit the constructors of its superclass?

A class does not inherit constructors from any of its superclasses.

Name primitive Java types.

The primitive types are byte, char, short, int, long, float, double, and boolean.

Which class should you use to obtain design information about an object?

The Class class is used to obtain information about an object's design.

What is the difference between static and non-static variables?

A static variable is associated with the class as a whole rather than with specific instances of a class. Non-static variables take on unique values with each object instance.

What restrictions are placed on method overloading?

Two methods may not have the same name and argument list but different return types.

What restrictions are placed on method overriding?

Overridden methods must have the same name, argument list, and return type. The overriding method may not limit the access of the method it overrides. The overriding method may not throw any exceptions that may not be thrown by the overridden method.

How are this() and super() used with constructors?

this() is used to invoke a constructor of the same class. super() is used to invoke a superclass constructor.

How is it possible for two String objects with identical values not to be equal under the == operator?

The == operator compares two objects to determine if they are the same object in memory. It is possible for two String objects to have the same value, but located indifferent areas of memory.

What is the Set interface?

The Set interface provides methods for accessing the elements of a finite mathematical set. Sets do not allow duplicate elements.

What is the List interface?

The List interface provides support for ordered collections of objects.

What is the difference between the File and RandomAccessFile classes?

The File class encapsulates the files and directories of the local file system. The RandomAccessFile class provides the methods needed to directly access data contained in any part of a file.

What interface must an object implement before it can be written to a stream as an object?
An object must implement the Serializable or Externalizable interface before it can be written to a stream as an object.

What is the ResourceBundle class?
The ResourceBundle class is used to store locale-specific resources that can be loaded by a program to tailor the program's appearance to the particular locale in which it is
being run.

What is a Java package and how is it used?
A Java package is a naming context for classes and interfaces. A package is used to create a separate name space for groups of classes and interfaces. Packages are also used to organize related classes and interfaces into a single API unit and to control accessibility to these classes and interfaces.

What are the Object and Class classes used for?
The Object class is the highest-level class in the Java class hierarchy. The Class class is used to represent the classes and interfaces that are loaded by a Java program.

What is Serialization and deserialization?
Serialization is the process of writing the state of an object to a byte stream.
Deserialization is the process of restoring these objects.

what is tunnelling?
Tunnelling is a route to somewhere. For example, RMI tunnelling is a way to make RMI application get through firewall. In CS world, tunnelling means a way to transfer data.

Does the code in finally block get executed if there is an exception and a return statement in a catch block?
If an exception occurs and there is a return statement in catch block, the finally block is still executed. The finally block will not be executed when the System.exit(1) statement is executed earlier or the system shut down earlier or the memory is used up earlier before the thread goes to finally block.

How you restrict a user to cut and paste from the html page?
Using javaScript to lock keyboard keys. It is one of solutions.

Is Java a super set of JavaScript?
No. They are completely different. Some syntax may be similar.

How the object oriented approach helps us keep complexity of software development under control?
We can discuss such issue from the following aspects:
Objects allow procedures to be encapsulated with their data to reduce potential interference.
Inheritance allows well-tested procedures to be reused and enables changes to make once and have effect in all relevant places.
The well-defined separations of interface and implementation allows constraints to be imposed on inheriting classes while still allowing the flexibility of overriding and overloading.

What is polymorphism?
Polymorphism allows methods to be written that needn't be concerned about the specifics of the objects they will be applied to. That is, the method can be specified at a higher level of abstraction and can be counted on to work even on objects of yet unconceived classes.

What is design by contract?
The design by contract specifies the obligations of a method to any other methods that may use its services and also theirs to it. For example, the preconditions specify what the method required to be true when the method is called. Hence making sure that preconditions are. Similarly, postconditions specify what must be true when the method is finished, thus the called method has the responsibility of satisfying the post conditions.
In Java, the exception handling facilities support the use of design by contract, especially in the case of checked exceptions. The assert keyword can be used to make such contracts.

What are use cases?
A use case describes a situation that a program might encounter and what behavior the program should exhibit in that circumstance. It is part of the analysis of a program. The collection of use cases should, ideally, anticipate all the standard circumstances and many of the extraordinary circumstances possible so that the program will be robust.

What is the difference between interface and abstract class?

interface contains methods that must be abstract;	Abstract class may contain concrete methods.
interface contains variables that must be static and final;	Abstract class may contain non-final and final variables.

members in an interface are public by default,	Abstract class may contain non-public members.
interface is used to "implements"; whereas	Abstract class is used to "extends".
interface can be used to achieve multiple inheritance;	Abstract class can be used as a single inheritance.
interface can "extends" another interface,	Abstract class can "extends" another class and "implements" multiple interfaces.
interface is absolutely abstract;	Abstract class can be invoked if a main() exists.

interface is more flexible than abstract class because one class can only "extends" one super class, but "implements" multiple interfaces.

If given a choice, use interface instead of abstract class .Can a private method of a superclass be declared within a subclass?
Sure. A private field or method or inner class belongs to its declared class and hides from its subclasses. There is no way for private stuff to have a runtime overloading or overriding (polymorphism) features.

Why Java does not support multiple inheritance ?
Java DOES support multiple inheritance via interface implementation.

What is the difference between final, finally and finalize?
final - declare constant
finally - handles exception
finalize - helps in garbage collection

Where and how can you use a private constuctor.
Private constructor can be used if you do not want any other class to instanstiate the object , the instantiation is done from a static public method, this method is used when dealing with the factory method pattern when the designer wants only one controller (fatory method) to create the object .

In System.out.println(),what is System,out and println,pls explain?
System is a predefined final class,out is a PrintStream object and println is a built-in overloaded method in the out object.

What is meant by "Abstract Interface"?
First, an interface is abstract. That means you cannot have any implementation in an interface. All the methods declared in an interface are abstract methods or signatures of the methods.
Can you make an instance of an abstract class? For example - java.util.Calender is an abstract class with a method getInstance() which returns an instance of the Calender class.
No! You cannot make an instance of an abstract class. An abstract class has to be sub-classed. If you have an abstract class and you want to use a method which has been implemented, you may need to subclass that abstract class, instantiate your subclass and then call that method.

What is the output of x<y? a:b = p*q when x=1,y=2,p=3,q=4?
When this kind of question has been asked, find the problems you think is necessary to ask before you give an answer. Ask if variables a and b have been declared or initialized. If the answer is yes. You can say that the syntax is wrong. If the statement is rewritten as: x<y? a:(b=p*q); the return value would be variable a because the x is 1 and less than y = 2; the x < y statement return true and variable a is returned.

Why Java does not support pointers?
Because pointers are unsafe. Java uses reference types to hide pointers and programmers feel easier to deal with reference types without pointers. This is why Java and C# shine.

What is the purpose of finalization?
The purpose of finalization is to give an unreachable object the opportunity to perform any cleanup processing before the object is garbage collected.

What is the difference between the Boolean & operator and the && operator?
If an expression involving the Boolean & operator is evaluated, both operands are evaluated. Then the & operator is applied to the operand. When an expression involving the && operator is evaluated, the first operand is evaluated. If the first operand returns a value of true then the second operand is evaluated. The && operator is then applied to the first and second operands. If the first operand evaluates to false, the evaluation of the second operand is skipped.

How many times may an object’s finalize() method be invoked by the garbage collector?
An object’s finalize() method may only be invoked once by the garbage collector.

What is the purpose of the finally clause of a try-catch-finally statement?
The finally clause is used to provide the capability to execute code no matter whether or not an exception is thrown or caught.

What is the argument type of a program’s main() method?
A program’s main() method takes an argument of the String[] type.

Which Java operator is right associative?
The = operator is right associative.

Can a double value be cast to a byte?
Yes, a double value can be cast to a byte.

What is the difference between a break statement and a continue statement?
A break statement results in the termination of the statement to which it applies (switch, for, do, or while). A continue statement is used to end the current loop iteration and return control to the loop statement.

What must a class do to implement an interface?
It must provide all of the methods in the interface and identify the interface in its implements clause.

What is the advantage of the event-delegation model over the earlier event-inheritance model?
The event-delegation model has two advantages over the event-inheritance model. First, it enables event handling to be handled by objects other than the ones that generate the events (or their containers). This allows a clean separation between a component's design and its use. The other advantage of the event-delegation model is that it performs much better in applications where many events are generated. This performance improvement is due to the fact that the event-delegation model does not have to repeatedly process unhandled events, as is the case of the event-inheritance model.

How are commas used in the intialization and iteration parts of a for statement?
Commas are used to separate multiple statements within the initialization and iteration parts of a for statement.

What is an abstract method?
An abstract method is a method whose implementation is deferred to a subclass.

What value does read() return when it has reached the end of a file?
The read() method returns -1 when it has reached the end of a file.

Can a Byte object be cast to a double value?
No, an object cannot be cast to a primitive value.

What is the difference between a static and a non-static inner class?
A non-static inner class may have object instances that are associated with instances of the class's outer class. A static inner class does not have any object instances.

If a variable is declared as private, where may the variable be accessed?
A private variable may only be accessed within the class in which it is declared.

What is an object's lock and which object's have locks?
An object's lock is a mechanism that is used by multiple threads to obtain synchronized access to the object. A thread may execute a synchronized method of an object only after it has acquired the object's lock. All objects and classes have locks. A class's lock is acquired on the class's Class object.

What is the % operator?
It is referred to as the modulo or remainder operator. It returns the remainder of dividing the first operand by the second operand.

When can an object reference be cast to an interface reference?
An object reference be cast to an interface reference when the object implements the referenced interface.

Which class is extended by all other classes?
The Object class is extended by all other classes.

Can an object be garbage collected while it is still reachable?
A reachable object cannot be garbage collected. Only unreachable objects may be garbage collected.

Is the ternary operator written x : y ? z or x ? y : z ?
It is written x ? y : z.

How is rounding performed under integer division?
The fractional part of the result is truncated. This is known as rounding toward zero.

What is the difference between the Reader/Writer class hierarchy and the InputStream/OutputStream class hierarchy?

The Reader/Writer class hierarchy is character-oriented, and the InputStream/OutputStream class hierarchy is byte-oriented.

What classes of exceptions may be caught by a catch clause?

A catch clause can catch any exception that may be assigned to the Throwable type. This includes the Error and Exception types.

If a class is declared without any access modifiers, where may the class be accessed?

A class that is declared without any access modifiers is said to have package access. This means that the class can only be accessed by other classes and interfaces that are defined within the same package.

Does a class inherit the constructors of its superclass?

A class does not inherit constructors from any of its superclasses.

What is the purpose of the System class?

The purpose of the System class is to provide access to system resources.

Name the eight primitive Java types.

The eight primitive types are byte, char, short, int, long, float, double, and boolean.

Which class should you use to obtain design information about an object?

The Class class is used to obtain information about an object's design.

Can there be an abstract class with no abstract methods in it?

Yes

Can an Interface be final?

No

Can an Interface have an inner class?

Yes.

```
public interface abc
{
        static int i=0; void dd();
        class a1
        {
                a1()
                {
                        int j;
                        System.out.println("inside");
                };
                public static void main(String a1[])
                {
                        System.out.println("in interfia");
                }
        }
}
```

Can we define private and protected modifiers for variables in interfaces?

No

What is Externalizable?

Externalizable is an Interface that extends Serializable Interface. And sends data into Streams in Compressed Format. It has two methods, writeExternal(ObjectOuput out) and readExternal(ObjectInput in)

What modifiers are allowed for methods in an Interface?

Only public and abstract modifiers are allowed for methods in interfaces.

What is a local, member and a class variable?

Variables declared within a method are "local" variables. Variables declared within the class i.e not within any methods are "member" variables (global variables). Variables declared within the class i.e not within any methods and are defined as "static" are class variables

What are the different identifier states of a Thread?

The different identifiers of a Thread are: R - Running or runnable thread, S - Suspended thread, CW - Thread waiting on a condition variable, MW - Thread waiting on a monitor lock, MS - Thread suspended waiting on a monitor lock

What are some alternatives to inheritance?

Delegation is an alternative to inheritance. Delegation means that you include an instance of another class as an instance variable, and forward messages to the instance. It is often safer than inheritance because it forces you to think about each message you forward, because the instance is of a known class, rather than a new class, and because it doesn't force you to accept all the methods of the super class: you can provide only the methods that really make sense. On the other hand, it makes you write more code, and it is harder to re-use (because it is not a subclass).

Why isn't there operator overloading?

Because C++ has proven by example that operator overloading makes code almost impossible to maintain. In fact there very nearly wasn't even method overloading in Java, but it was thought that this was too useful for some very basic methods like print(). Note that some of the classes like DataOutputStream have unoverloaded methods like writeInt() and writeByte().

What does it mean that a method or field is “static"?

Static variables and methods are instantiated only once per class. In other words they are class variables, not instance variables. If you change the value of a static variable in a particular object, the value of that variable changes for all instances of that class. Static methods can be referenced with the name of the class rather than the name of a particular object of the class (though that works too). That's how library methods like System.out.println() work. out is a static field in the java.lang.System class.

How do I convert a numeric IP address like 192.18.97.39 into a hostname like java.sun.com?

String hostname = InetAddress.getByName("192.18.97.39").getHostName();

Difference between JRE/JVM/JDK?

Why do threads block on I/O?

Threads block on i/o (that is enters the waiting state) so that other threads may execute while the I/O operation is performed.

What is synchronization and why is it important?

With respect to multithreading, synchronization is the capability to control the access of multiple threads to shared resources. Without synchronization, it is possible for one thread to modify a shared object while another thread is in the process of using or updating that object's value. This often leads to significant errors.

Is null a keyword?

The null value is not a keyword.

Which characters may be used as the second character of an identifier,but not as the first character of an identifier?

The digits 0 through 9 may not be used as the first character of an identifier but they may be used after the first character of an identifier.

What modifiers may be used with an inner class that is a member of an outer class?

A (non-local) inner class may be declared as public, protected, private, static, final, or abstract.

How many bits are used to represent Unicode, ASCII, UTF-16, and UTF-8 characters?

Unicode requires 16 bits and ASCII require 7 bits. Although the ASCII character set uses only 7 bits, it is usually represented as 8 bits. UTF-8 represents characters using 8, 16, and 18 bit patterns. UTF-16 uses 16-bit and larger bit patterns.

What are wrapped classes?

Wrapped classes are classes that allow primitive types to be accessed as objects.

What restrictions are placed on the location of a package statement within a source code file?

A package statement must appear as the first line in a source code file (excluding blank lines and comments).

What is the difference between preemptive scheduling and time slicing?

Under preemptive scheduling, the highest priority task executes until it enters the waiting or dead states or a higher priority task comes into existence. Under time slicing, a task executes for a predefined slice of time and then reenters the pool of ready tasks. The scheduler then determines which task should execute next, based on priority and other factors.

What is a native method?

A native method is a method that is implemented in a language other than Java.

What are order of precedence and associativity, and how are they used?

Order of precedence determines the order in which operators are evaluated in expressions. Associatity determines whether an expression is evaluated left-to-right or right-to-left

What is the catch or declare rule for method declarations?
If a checked exception may be thrown within the body of a method, the method must either catch the exception or declare it in its throws clause.

Can an anonymous class be declared as implementing an interface and extending a class?
An anonymous class may implement an interface or extend a superclass, but may not be declared to do both.

What is the range of the char type?
The range of the char type is 0 to 2^16 - 1.

What is garbage collection? What is the process that is responsible for doing that in java?
Reclaiming the unused memory by the invalid objects. Garbage collector is responsible for this process

What kind of thread is the Garbage collector thread?
It is a daemon thread.

What is a daemon thread?
These are the threads which can run without user intervention. The JVM can exit when there are daemon thread by killing them abruptly.

How will you invoke any external process in Java?
Runtime.getRuntime().exec(….)

What is the finalize method do?
Before the invalid objects get garbage collected, the JVM give the user a chance to clean up some resources before it got garbage collected.

What is mutable object and immutable object?
If a object value is changeable then we can call it as Mutable object. (Ex., StringBuffer, …) If you are not allowed to change the value of an object, it is immutable object. (Ex., String, Integer, Float, …)

What is the basic difference between string and stringbuffer object?
String is an immutable object. StringBuffer is a mutable object.

What is the purpose of Void class?
The Void class is an uninstantiable placeholder class to hold a reference to the Class object reresenting the primitive Java type void.

What is reflection?
Reflection allows programmatic access to information about the fields, methods and constructors of loaded classes, and the use reflected fields, methods, and constructors to operate on their underlying counterparts on objects, within security restrictions.

What is the base class for Error and Exception?
Throwable

What is the byte range?
-128 to 127

What is the implementation of destroy method in java.. is it native or java code?
This method is not implemented.

What is a package?
To group set of classes into a single unit is known as packaging. Packages provides wide namespace ability.

What are the approaches that you will follow for making a program very efficient?
By avoiding too much of static methods avoiding the excessive and unnecessary use of synchronized methods Selection of related classes based on the application (meaning synchronized classes for multiuser and non-synchronized classes for single user) Usage of appropriate design patterns Using cache methodologies for remote invocations Avoiding creation of variables within a loop and lot more.

What is a DatabaseMetaData?
Comprehensive information about the database as a whole.

What is Locale?

A Locale object represents a specific geographical, political, or cultural region

How will you load a specific locale?
Using ResourceBundle.getBundle(…);

What is JIT and its use?
Really, just a very fast compiler… In this incarnation, pretty much a one-pass compiler – no offline computations. So you can't look at the whole method, rank the expressions according to which ones are re-used the most, and then generate code. In theory terms, it's an on-line problem.

Is JVM a compiler or an interpreter?
Interpreter

When you think about optimization, what is the best way to findout the time/memory consuming process?
Using profiler

What is the purpose of assert keyword used in JDK1.4.x?
In order to validate certain expressions. It effectively replaces the if block and automatically throws the AssertionError on failure. This keyword should be used for the critical arguments. Meaning, without that the method does nothing.

How will you get the platform dependent values like line separator, path separator, etc., ?
Using Sytem.getProperty(…) (line.separator, path.separator, …)

What is skeleton and stub? what is the purpose of those?
Stub is a client side representation of the server, which takes care of communicating with the remote server. Skeleton is the server side representation. But that is no more in use… it is deprecated long before in JDK.

What is the final keyword denotes?
final keyword denotes that it is the final implementation for that method or variable or class. You can't override that method/variable/class any more.

What is the significance of ListIterator?
You can iterate back and forth.

What is the major difference between LinkedList and ArrayList?
LinkedList are meant for sequential accessing. ArrayList are meant for random accessing.

What is nested class?
If all the methods of a inner class is static then it is a nested class.

What is inner class?
If the methods of the inner class can only be accessed via the instance of the inner class, then it is called inner class.

What is composition?
Holding the reference of the other class within some other class is known as composition.

What is aggregation?
It is a special type of composition. If you expose all the methods of a composite class and route the method call to the composite method through its reference, then it is called aggregation.

What are the methods in Object?
clone, equals, wait, finalize, getClass, hashCode, notify, notifyAll, toString

Can you instantiate the Math class?
You can't instantiate the math class. All the methods in this class are static. And the constructor is not public.

What is singleton?
It is one of the design pattern. This falls in the creational pattern of the design pattern. There will be only one instance for that entire JVM. You can achieve this by having the private constructor in the class. For eg., public class Singleton { private static final Singleton s = new Singleton(); private Singleton() { } public static Singleton getInstance() { return s; } // all non static methods … }

What is DriverManager?
The basic service to manage set of JDBC drivers.

What is Class.forName() does and how it is useful?

It loads the class into the ClassLoader. It returns the Class. Using that you can get the instance (“class-instance".newInstance()).

Expain the reason for each keyword of public static void main(String args[])

What are the advantages of OOPL?

Object oriented programming languages directly represent the real life objects. The features of OOPL as inhreitance, polymorphism, encapsulation makes it powerful.

What do mean by polymorphisum, inheritance, encapsulation?

Polymorhisum: is a feature of OOPl that at run time depending upon the type of object the appropriate method is called.
Inheritance: is a feature of OOPL that represents the "is a" relationship between different objects(classes). Say in real life a manager is a employee. So in OOPL manger class is inherited from the employee class.
Encapsulation: is a feature of OOPL that is used to hide the information.

What do you mean by static methods?

By using the static method there is no need creating an object of that class to use that method. We can directly call that method on that class. For example, say class A has static function f(), then we can call f() function as A.f(). There is no need of creating an object of class A.

What do you mean by virtual methods?

virtual methods are used to use the polymorhism feature in C++. Say class A is inherited from class B. If we declare say fuction f() as virtual in class B and override the same function in class A then at runtime appropriate method of the class will be called depending upon the type of the object.

Given two tables Student(SID, Name, Course) and Level(SID, level) write the SQL statement to get the name and SID of the student who are taking course = 3 and at freshman level.

```
SELECT Student.name, Student.SID
FROM Student, Level
WHERE Student.SID = Level.SID
AND Level.Level = "freshman"
AND Student.Course = 3;
```

What are the disadvantages of using threads

DeadLock.

Write the Java code to declare any constant (say gravitational constant) and to get its value

```
Class ABC
{
static final float GRAVITATIONAL_CONSTANT = 9.8;
public void getConstant()
{
system.out.println("Gravitational_Constant: " + GRAVITATIONAL_CONSTANT);
}
}
```

What do you mean by multiple inheritance in C++ ?

Multiple inheritance is a feature in C++ by which one class can be of different types. Say class teachingAssistant is inherited from two classes say teacher and Student.

Can you write Java code for declaration of multiple inheritance in Java ?

```
Class C extends A implements B
{
}
```

What is a Marker Interface?

An interface with no methods. Example: Serializable, Remote, Cloneable

What interface do you implement to do the sorting?

Comparable

What is the eligibility for a object to get cloned?

It must implement the Cloneable interface

What is the purpose of abstract class?

It is not an instantiable class. It provides the concrete implementation for some/all the methods. So that they can reuse the concrete functionality by inheriting the abstract class.

What is the difference between interface and abstract class?

Abstract class defined with methods. Interface will declare only the methods. Abstract classes are very much useful when there is a some functionality across various classes. Interfaces are well suited for the classes which varies in functionality but with the same method signatures.

What do you mean by RMI and how it is useful?

RMI is a remote method invocation. Using RMI, you can work with remote object. The function calls are as though you are invoking a local variable. So it gives you a impression that you are working really with a object that resides within your own JVM though it is somewhere.

What is the protocol used by RMI?

RMI-IIOP

What is a hashCode?

hash code value for this object which is unique for every object.

What is a thread?

Thread is a block of code which can execute concurrently with other threads in the JVM.

What is the algorithm used in Thread scheduling?

Fixed priority scheduling.

What is hash-collision in Hashtable and how it is handled in Java?

Two different keys with the same hash value. Two different entries will be kept in a single hash bucket to avoid the collision.

What are the different driver types available in JDBC?

1. A JDBC-ODBC bridge 2. A native-API partly Java technology-enabled driver 3. A net-protocol fully Java technology-enabled driver 4. A native-protocol fully Java technology-enabled driver For more information: Driver Description

Is JDBC-ODBC bridge multi-threaded?

No

Does the JDBC-ODBC Bridge support multiple concurrent open statements per connection?

No

What is the use of serializable?

To persist the state of an object into any perminant storage device.

What is the use of transient?

It is an indicator to the JVM that those variables should not be persisted. It is the users responsibility to initialize the value when read back from the storage.

What are the different level lockings using the synchronization keyword?

Class level lock Object level lock Method level lock Block level lock

What is the use of preparedstatement?

Preparedstatements are precompiled statements. It is mainly used to speed up the process of inserting/updating/deleting especially when there is a bulk processing.

What is callable statement? Tell me the way to get the callable statement?

Callablestatements are used to invoke the stored procedures. You can obtain the callablestatement from Connection using the following methods prepareCall(String sql) prepareCall(String sql, int resultSetType, int resultSetConcurrency)

In a statement, I am executing a batch. What is the result of the execution?

It returns the int array. The array contains the affected row count in the corresponding index of the SQL.

Can a abstract method have the static qualifier? No

What are the different types of qualifier and what is the default qualifier?

public, protected, private, package (default)

What is the super class of Hashtable?

Dictionary

What is a lightweight component?
Lightweight components are the one which doesn’t go with the native call to obtain the graphical units. They share their parent component graphical units to render them. Example, Swing components

What is a heavyweight component?
For every paint call, there will be a native call to get the graphical units. Example, AWT.

What is an applet?
Applet is a program which can get downloaded into a client environment and start executing there.

What do you mean by a Classloader?
Classloader is the one which loads the classes into the JVM.

What are the implicit packages that need not get imported into a class file?
java.lang

What is the difference between lightweight and heavyweight component?
Lightweight components reuses its parents graphical units. Heavyweight components goes with the native graphical unit for every component. Lightweight components are faster than the heavyweight components.

What are the ways in which you can instantiate a thread?
Using Thread class By implementing the Runnable interface and giving that handle to the Thread class.

What are the states of a thread?
1. New 2. Runnable 3. Not Runnable 4. Dead

What is a socket?
A socket is an endpoint for communication between two machines.

How will you establish the connection between the servlet and an applet?
Using the URL, I will create the connection URL. Then by openConnection method of the URL, I will establish the connection, through which I can be able to exchange data.

What are the threads will start, when you start the java program?
Finalizer, Main, Reference Handler, Signal Dispatcher

Is “abc” a primitive value?
The String literal “abc” is not a primitive value. It is a String object.

What restrictions are placed on the values of each case of a switch statement?
During compilation, the values of each case of a switch statement must evaluate to a value that can be promoted to an int value.

What modifiers may be used with an interface declaration?
An interface may be declared as public or abstract.

Is a class a subclass of itself?
A class is a subclass of itself.

What is the difference between a while statement and a do statement?
A while statement checks at the beginning of a loop to see whether the next loop iteration should occur. A do statement checks at the end of a loop to see whether the next iteration of a loop should occur. The do statement will always execute the body of a loop at least once.

What modifiers can be used with a local inner class?
A local inner class may be final or abstract.

What is the purpose of the File class?
The File class is used to create objects that provide access to the files and directories of a local file system.

Can an exception be rethrown?
Yes, an exception can be rethrown.

When does the compiler supply a default constructor for a class?

The compiler supplies a default constructor for a class if no other constructors are provided.

If a method is declared as protected, where may the method be accessed?
A protected method may only be accessed by classes or interfaces of the same package or by subclasses of the class in which it is declared.

Which non-Unicode letter characters may be used as the first character of an identifier?
The non-Unicode letter characters $ and _ may appear as the first character of an identifier

What restrictions are placed on method overloading?
Two methods may not have the same name and argument list but different return types.

What is casting?
There are two types of casting, casting between primitive numeric types and casting between object references. Casting between numeric types is used to convert larger values, such as double values, to smaller values, such as byte values. Casting between object references is used to refer to an object by a compatible class, interface, or array type reference.

What is the return type of a program's main() method?
A program's main() method has a void return type.

What class of exceptions are generated by the Java run-time system?
The Java runtime system generates RuntimeException and Error exceptions.

What class allows you to read objects directly from a stream?
The ObjectInputStream class supports the reading of objects from input streams.

What is the difference between a field variable and a local variable?
A field variable is a variable that is declared as a member of a class. A local variable is a variable that is declared local to a method.

How are this() and super() used with constructors?
this() is used to invoke a constructor of the same class. super() is used to invoke a superclass constructor.

What is the relationship between a method's throws clause and the exceptions that can be thrown during the method's execution?
A method's throws clause must declare any checked exceptions that are not caught within the body of the method.

Why are the methods of the Math class static?
So they can be invoked as if they are a mathematical code library.

What are the legal operands of the instanceof operator?
The left operand is an object reference or null value and the right operand is a class, interface, or array type.

What an I/O filter?
An I/O filter is an object that reads from one stream and writes to another, usually altering the data in some way as it is passed from one stream to another.

If an object is garbage collected, can it become reachable again?
Once an object is garbage collected, it ceases to exist. It can no longer become reachable again.

What are E and PI?
E is the base of the natural logarithm and PI is mathematical value pi.

Are true and false keywords?
The values true and false are not keywords.

What is the difference between the File and RandomAccessFile classes?
The File class encapsulates the files and directories of the local file system. The RandomAccessFile class provides the methods needed to directly access data contained in any part of a file.

What happens when you add a double value to a String?
The result is a String object.

What is your platform's default character encoding?
If you are running Java on English Windows platforms, it is probably Cp1252. If you are running Java on English Solaris platforms, it is most likely 8859_1.

Which package is always imported by default?

The java.lang package is always imported by default.

What interface must an object implement before it can be written to a stream as an object?

An object must implement the Serializable or Externalizable interface before it can be written to a stream as an object.

How can my application get to know when a HttpSession is removed?

Define a Class HttpSessionNotifier which implements HttpSessionBindingListener and implement the functionality what you need in valueUnbound() method. Create an instance of that class and put that instance in HttpSession.

Whats the difference between notify() and notifyAll()?

notify() is used to unblock one waiting thread; notifyAll() is used to unblock all of them. Using notify() is preferable (for efficiency) when only one blocked thread can benefit from the change (for example, when freeing a buffer back into a pool). notifyAll() is necessary (for correctness) if multiple threads should resume (for example, when releasing a “writer” lock on a file might permit all “readers” to resume).

Why can’t I say just abs() or sin() instead of Math.abs() and Math.sin()?

The import statement does not bring methods into your local name space. It lets you abbreviate class names, but not get rid of them altogether. That’s just the way it works, you’ll get used to it. It’s really a lot safer this way.
However, there is actually a little trick you can use in some cases that gets you what you want. If your top-level class doesn’t need to inherit from anything else, make it inherit from java.lang.Math. That *does* bring all the methods into your local name space. But you can’t use this trick in an applet, because you have to inherit from java.awt.Applet. And actually, you can’t use it on java.lang.Math at all, because Math is a “final” class which means it can’t be extended.

Why are there no global variables in Java?

Global variables are considered bad form for a variety of reasons: Adding state variables breaks referential transparency (you no longer can understand a statement or expression on its own: you need to understand it in the context of the settings of the global variables), State variables lessen the cohesion of a program: you need to know more to understand how something works. A major point of Object-Oriented programming is to break up global state into more easily understood collections of local state, When you add one variable, you limit the use of your program to one instance. What you thought was global, someone else might think of as local: they may want to run two copies of your program at once. For these reasons, Java decided to ban global variables.

What does it mean that a class or member is final?

A final class can no longer be subclassed. Mostly this is done for security reasons with basic classes like String and Integer. It also allows the compiler to make some optimizations, and makes thread safety a little easier to achieve. Methods may be declared final as well. This means they may not be overridden in a subclass. Fields can be declared final, too. However, this has a completely different meaning. A final field cannot be changed after it’s initialized, and it must include an initializer statement where it’s declared. For example, public final double c = 2.998; It’s also possible to make a static field final to get the effect of C++’s const statement or some uses of C’s #define, e.g. public static final double c = 2.998;

What does it mean that a method or class is abstract?

An abstract class cannot be instantiated. Only its subclasses can be instantiated. You indicate that a class is abstract with the abstract keyword like this:

```
public abstract class Container extends Component {
```

Abstract classes may contain abstract methods. A method declared abstract is not actually implemented in the current class. It exists only to be overridden in subclasses. It has no body. For example,

```
public abstract float price();
```

Abstract methods may only be included in abstract classes. However, an abstract class is not required to have any abstract methods, though most of them do. Each subclass of an abstract class must override the abstract methods of its superclasses or itself be declared abstract.

What is a transient variable?

Transient variable is a variable that may not be serialized.

How are Observer and Observable used?

Objects that subclass the Observable class maintain a list of observers. When an Observable object is updated it invokes the update() method of each of its observers to notify the observers that it has changed state. The Observer interface is implemented by objects that observe Observable objects.

Can a lock be acquired on a class?

Yes, a lock can be acquired on a class. This lock is acquired on the class’s Class object.

What state does a thread enter when it terminates its processing?

When a thread terminates its processing, it enters the dead state.

How does Java handle integer overflows and underflows?
It uses those low order bytes of the result that can fit into the size of the type allowed by the operation.

What is the difference between the >> and >>> operators?
The >> operator carries the sign bit when shifting right. The >>> zero-fills bits that have been shifted out.

Is sizeof a keyword?
The sizeof operator is not a keyword.

Does garbage collection guarantee that a program will not run out of memory?
Garbage collection does not guarantee that a program will not run out of memory. It is possible for programs to use up memory resources faster than they are garbage collected. It is also possible for programs to create objects that are not subject to garbage collection

Can an object's finalize() method be invoked while it is reachable?
An object's finalize() method cannot be invoked by the garbage collector while the object is still reachable. However, an object's finalize() method may be invoked by other objects.

What value does readLine() return when it has reached the end of a file?
The readLine() method returns null when it has reached the end of a file.

Can a for statement loop indefinitely?
Yes, a for statement can loop indefinitely. For example, consider the following: for(;;) ;

To what value is a variable of the String type automatically initialized?
The default value of an String type is null.

What is a task's priority and how is it used in scheduling?
A task's priority is an integer value that identifies the relative order in which it should be executed with respect to other tasks. The scheduler attempts to schedule higher priority tasks before lower priority tasks.

What is the range of the short type?
The range of the short type is -(2^15) to 2^15 - 1.

What is the purpose of garbage collection?
The purpose of garbage collection is to identify and discard objects that are no longer needed by a program so that their resources may be reclaimed and reused.

What do you understand by private, protected and public?
These are accessibility modifiers. Private is the most restrictive, while public is the least restrictive. There is no real difference between protected and the default type (also known as package protected) within the context of the same package, however the protected keyword allows visibility to a derived class in a different package.

What is Downcasting ?
Downcasting is the casting from a general to a more specific type, i.e. casting down the hierarchy

Can a method be overloaded based on different return type but same argument type ?
No, because the methods can be called without using their return type in which case there is ambiquity for the compiler

What happens to a static var that is defined within a method of a class ?
Can't do it. You'll get a compilation error

How many static init can you have ?
As many as you want, but the static initializers and class variable initializers are executed in textual order and may not refer to class variables declared in the class whose declarations appear textually after the use, even though these class variables are in scope.

What is the difference amongst JVM Spec, JVM Implementation, JVM Runtime ?
The JVM spec is the blueprint for the JVM generated and owned by Sun. The JVM implementation is the actual implementation of the spec by a vendor and the JVM runtime is the actual running instance of a JVM implementation

Describe what happens when an object is created in Java?
Several things happen in a particular order to ensure the object is constructed properly: Memory is allocated from heap to hold all instance variables and implementation-specific data of the object and its superclasses. Implemenation-specific data

includes pointers to class and method data. The instance variables of the objects are initialized to their default values. The constructor for the most derived class is invoked. The first thing a constructor does is call the consctructor for its superclasses. This process continues until the constrcutor for java.lang.Object is called, as java.lang.Object is the base class for all objects in java. Before the body of the constructor is executed, all instance variable initializers and initialization blocks are executed. Then the body of the constructor is executed. Thus, the constructor for the base class completes first and constructor for the most derived class completes last.

What does the "final" keyword mean in front of a variable? A method? A class?

FINAL for a variable: value is constant. FINAL for a method: cannot be overridden. FINAL for a class: cannot be derived

What is the difference between instanceof and isInstance?

instanceof is used to check to see if an object can be cast into a specified type without throwing a cast class exception. isInstance() Determines if the specified Object is assignment-compatible with the object represented by this Class. This method is the dynamic equivalent of the Java language instanceof operator. The method returns true if the specified Object argument is non-null and can be cast to the reference type represented by this Class object without raising a ClassCastException. It returns false otherwise.

Why does it take so much time to access an Applet having Swing Components the first time?

Because behind every swing component are many Java objects and resources. This takes time to create them in memory. JDK 1.3 from Sun has some improvements which may lead to faster execution of Swing applications.

What is the Collections API?

The Collections API is a set of classes and interfaces that support operations on collections of objects

What is the List interface?

The List interface provides support for ordered collections of objects.

What is the Vector class?

The Vector class provides the capability to implement a growable array of objects

What is an Iterator interface?

The Iterator interface is used to step through the elements of a Collection

Which java.util classes and interfaces support event handling?

The EventObject class and the EventListener interface support event processing

What is the GregorianCalendar class?

The GregorianCalendar provides support for traditional Western calendars

What is the Locale class?

The Locale class is used to tailor program output to the conventions of a particular geographic, political, or cultural region

What is the SimpleTimeZone class?

The SimpleTimeZone class provides support for a Gregorian calendar

What is the Map interface?

The Map interface replaces the JDK 1.1 Dictionary class and is used associate keys with values

What is the highest-level event class of the event-delegation model?

The java.util.EventObject class is the highest-level class in the event-delegation class hierarchy

What is the Collection interface?

The Collection interface provides support for the implementation of a mathematical bag - an unordered collection of objects that may contain duplicates

What is the Set interface?

The Set interface provides methods for accessing the elements of a finite mathematical set. Sets do not allow duplicate elements

What is the purpose of the enableEvents() method?

The enableEvents() method is used to enable an event for a particular object. Normally, an event is enabled when a listener is added to an object for a particular event. The enableEvents() method is used by objects that handle events by overriding their event-dispatch methods.

What is the ResourceBundle class?
The ResourceBundle class is used to store locale-specific resources that can be loaded by a program to tailor the program's appearance to the particular locale in which it is being run.

What is the difference between yielding and sleeping?
When a task invokes its yield() method, it returns to the ready state. When a task invokes its sleep() method, it returns to the waiting state.

When a thread blocks on I/O, what state does it enter?
A thread enters the waiting state when it blocks on I/O.

When a thread is created and started, what is its initial state?
A thread is in the ready state after it has been created and started.

What invokes a thread's run() method?
After a thread is started, via its start() method or that of the Thread class, the JVM invokes the thread's run() method when the thread is initially executed.

What method is invoked to cause an object to begin executing as a separate thread?
The start() method of the Thread class is invoked to cause an object to begin executing as a separate thread.

What is the purpose of the wait(), notify(), and notifyAll() methods?
The wait(),notify(), and notifyAll() methods are used to provide an efficient way for threads to wait for a shared resource. When a thread executes an object's wait() method, it enters the waiting state. It only enters the ready state after another thread invokes the object's notify() or notifyAll() methods.

What are the high-level thread states?
The high-level thread states are ready, running, waiting, and dead

What happens when a thread cannot acquire a lock on an object?
If a thread attempts to execute a synchronized method or synchronized statement and is unable to acquire an object's lock, it enters the waiting state until the lock becomes available.

How does multithreading take place on a computer with a single CPU?
The operating system's task scheduler allocates execution time to multiple tasks. By quickly switching between executing tasks, it creates the impression that tasks execute sequentially.

What happens when you invoke a thread's interrupt method while it is sleeping or waiting?
When a task's interrupt() method is executed, the task enters the ready state. The next time the task enters the running state, an InterruptedException is thrown.

What state is a thread in when it is executing?
An executing thread is in the running state

What are three ways in which a thread can enter the waiting state?
A thread can enter the waiting state by invoking its sleep() method, by blocking on I/O, by unsuccessfully attempting to acquire an object's lock, or by invoking an object's wait() method. It can also enter the waiting state by invoking its (deprecated) suspend() method.

What method must be implemented by all threads?
All tasks must implement the run() method, whether they are a subclass of Thread or implement the Runnable interface.

What are the two basic ways in which classes that can be run as threads may be defined?
A thread class may be declared as a subclass of Thread, or it may implement the Runnable interface.

How can you store international / Unicode characters into a cookie?
One way is, before storing the cookie URLEncode it. URLEnocder.encoder(str); And use URLDecoder.decode(str) when you get the stored cookie

What is the difference between procedural and object-oriented programs?- a) In procedural program, programming logic follows certain procedures and the instructions are executed one after another. In OOP program, unit of program is object, which is nothing but combination of data and code. b) In procedural program, data is exposed to the whole program whereas in OOPs program, it is accessible with in the object and which in turn assures the security of the code.

What are Encapsulation, Inheritance and Polymorphism?- Encapsulation is the mechanism that binds together code and data it manipulates and keeps both safe from outside interference and misuse. Inheritance is the process by which one object acquires the properties of another object. Polymorphism is the feature that allows one interface to be used for general class actions.

What is the difference between Assignment and Initialization?- Assignment can be done as many times as desired whereas initialization can be done only once.

What is OOPs?- Object oriented programming organizes a program around its data, i. e. , objects and a set of well defined interfaces to that data. An object-oriented program can be characterized as data controlling access to code.

What are Class, Constructor and Primitive data types?- Class is a template for multiple objects with similar features and it is a blue print for objects. It defines a type of object according to the data the object can hold and the operations the object can perform. Constructor is a special kind of method that determines how an object is initialized when created. Primitive data types are 8 types and they are: byte, short, int, long, float, double, boolean, char.

What is an Object and how do you allocate memory to it?- Object is an instance of a class and it is a software unit that combines a structured set of data with a set of operations for inspecting and manipulating that data. When an object is created using new operator, memory is allocated to it.

What is the difference between constructor and method?- Constructor will be automatically invoked when an object is created whereas method has to be called explicitly.

What are methods and how are they defined?- Methods are functions that operate on instances of classes in which they are defined. Objects can communicate with each other using methods and can call methods in other classes. Method definition has four parts. They are name of the method, type of object or primitive type the method returns, a list of parameters and the body of the method. A method's signature is a combination of the first three parts mentioned above.

What is the use of bin and lib in JDK?- Bin contains all tools such as javac, appletviewer, awt tool, etc., whereas lib contains API and all packages.

What is casting?- Casting is used to convert the value of one type to another.

How many ways can an argument be passed to a subroutine and explain them?- An argument can be passed in two ways. They are passing by value and passing by reference. Passing by value: This method copies the value of an argument into the formal parameter of the subroutine. Passing by reference: In this method, a reference to an argument (not the value of the argument) is passed to the parameter.

What is the difference between an argument and a parameter?- While defining method, variables passed in the method are called parameters. While using those methods, values passed to those variables are called arguments.

What are different types of access modifiers?- public: Any thing declared as public can be accessed from anywhere. private: Any thing declared as private can't be seen outside of its class. protected: Any thing declared as protected can be accessed by classes in the same package and subclasses in the other packages. default modifier : Can be accessed only to classes in the same package.

What is final, finalize() and finally?- final : final keyword can be used for class, method and variables. A final class cannot be subclassed and it prevents other programmers from subclassing a secure class to invoke insecure methods. A final method can't be overridden. A final variable can't change from its initialized value. finalize() : finalize() method is used just before an object is destroyed and can be called just prior to garbage collection. finally : finally, a key word used in exception handling, creates a block of code that will be executed after a try/catch block has completed and before the code following the try/catch block. The finally block will execute whether or not an exception is thrown. For example, if a method opens a file upon exit, then you will not want the code that closes the file to be bypassed by the exception-handling mechanism. This finally keyword is designed to address this contingency.

What is UNICODE?- Unicode is used for internal representation of characters and strings and it uses 16 bits to represent each other.

What is Garbage Collection and how to call it explicitly?- When an object is no longer referred to by any variable, java automatically reclaims memory used by that object. This is known as garbage collection. System. gc() method may be used to call it explicitly.

What is finalize() method?- finalize () method is used just before an object is destroyed and can be called just prior to garbage collection.

What are Transient and Volatile Modifiers?- Transient: The transient modifier applies to variables only and it is not stored as part of its object's Persistent state. Transient variables are not serialized. Volatile: Volatile modifier applies to variables only and it tells the compiler that the variable modified by volatile can be changed unexpectedly by other parts of the program.

What is method overloading and method overriding?- Method overloading: When a method in a class having the same method name with different arguments is said to be method overloading. Method overriding : When a method in a class having the same method name with same arguments is said to be method overriding.

What is difference between overloading and overriding?- a) In overloading, there is a relationship between methods available in the same class whereas in overriding, there is relationship between a superclass method and subclass method. b) Overloading does not block inheritance from the superclass whereas overriding blocks inheritance from the superclass. c) In overloading, separate methods share the same name whereas in overriding, subclass method replaces the superclass. d) Overloading must have different method signatures whereas overriding must have same signature.

What is meant by Inheritance and what are its advantages?- Inheritance is the process of inheriting all the features from a class. The advantages of inheritance are reusability of code and accessibility of variables and methods of the super class by subclasses.

What is the difference between this() and super()?- this() can be used to invoke a constructor of the same class whereas super() can be used to invoke a super class constructor.

What is the difference between superclass and subclass?- A super class is a class that is inherited whereas sub class is a class that does the inheriting.

What modifiers may be used with top-level class?- public, abstract and final can be used for top-level class.

What are inner class and anonymous class?- Inner class : classes defined in other classes, including those defined in methods are called inner classes. An inner class can have any accessibility including private. Anonymous class : Anonymous class is a class defined inside a method without a name and is instantiated and declared in the same place and cannot have explicit constructors.

What is a package?- A package is a collection of classes and interfaces that provides a high-level layer of access protection and name space management.

What is a reflection package?- java. lang. reflect package has the ability to analyze itself in runtime.

What is interface and its use?- Interface is similar to a class which may contain method's signature only but not bodies and it is a formal set of method and constant declarations that must be defined by the class that implements it. Interfaces are useful for: a)Declaring methods that one or more classes are expected to implement b)Capturing similarities between unrelated classes without forcing a class relationship. c)Determining an object's programming interface without revealing the actual body of the class.

What is an abstract class?- An abstract class is a class designed with implementation gaps for subclasses to fill in and is deliberately incomplete.

What is the difference between Integer and int?- a) Integer is a class defined in the java. lang package, whereas int is a primitive data type defined in the Java language itself. Java does not automatically convert from one to the other. b) Integer can be used as an argument for a method that requires an object, whereas int can be used for calculations.

What is a cloneable interface and how many methods does it contain?- It is not having any method because it is a TAGGED or MARKER interface.

What is the difference between abstract class and interface?- a) All the methods declared inside an interface are abstract whereas abstract class must have at least one abstract method and others may be concrete or abstract. b) In abstract class, key word abstract must be used for the methods whereas interface we need not use that keyword for the methods. c) Abstract class must have subclasses whereas interface can't have subclasses.

Can you have an inner class inside a method and what variables can you access?- Yes, we can have an inner class inside a method and final variables can be accessed.

What is the difference between String and String Buffer?- a) String objects are constants and immutable whereas StringBuffer objects are not. b) String class supports constant strings whereas StringBuffer class supports growable and modifiable strings.

What is the difference between Array and vector?- Array is a set of related data type and static whereas vector is a growable array of objects and dynamic.

What is the difference between exception and error?- The exception class defines mild error conditions that your program encounters. Exceptions can occur when trying to open the file, which does not exist, the network connection is disrupted, operands being manipulated are out of prescribed ranges, the class file you are interested in loading is missing. The error class defines serious error conditions that you should not attempt to recover from. In most cases it is advisable to let the program terminate when such an error is encountered.

What is the difference between process and thread?- Process is a program in execution whereas thread is a separate path of execution in a program.

What is multithreading and what are the methods for inter-thread communication and what is the class in which these methods are defined?- Multithreading is the mechanism in which more than one thread run independent of each other within the process. wait (), notify () and notifyAll() methods can be used for inter-thread communication and these methods are in Object class. wait() : When a thread executes a call to wait() method, it surrenders the object lock and enters into a waiting state. notify() or notifyAll() : To remove a thread from the waiting state, some other thread must make a call to notify() or notifyAll() method on the same object.

What is the class and interface in java to create thread and which is the most advantageous method?- Thread class and Runnable interface can be used to create threads and using Runnable interface is the most advantageous method to create threads because we need not extend thread class here.

What are the states associated in the thread?- Thread contains ready, running, waiting and dead states.

What is synchronization?- Synchronization is the mechanism that ensures that only one thread is accessed the resources at a time.

When you will synchronize a piece of your code?- When you expect your code will be accessed by different threads and these threads may change a particular data causing data corruption.

What is deadlock?- When two threads are waiting each other and can’t precede the program is said to be deadlock.

What is daemon thread and which method is used to create the daemon thread?- Daemon thread is a low priority thread which runs intermittently in the back ground doing the garbage collection operation for the java runtime system. setDaemon method is used to create a daemon thread.

Are there any global variables in Java, which can be accessed by other part of your program?- No, it is not the main method in which you define variables. Global variables is not possible because concept of encapsulation is eliminated here.

What is an applet?- Applet is a dynamic and interactive program that runs inside a web page displayed by a java capable browser.

What is the difference between applications and applets?- a)Application must be run on local machine whereas applet needs no explicit installation on local machine. b)Application must be run explicitly within a java-compatible virtual machine whereas applet loads and runs itself automatically in a java-enabled browser. d)Application starts execution with its main method whereas applet starts execution with its init method. e)Application can run with or without graphical user interface whereas applet must run within a graphical user interface.

How does applet recognize the height and width?- Using getParameters() method.

When do you use codebase in applet?- When the applet class file is not in the same directory, codebase is used.

What is the lifecycle of an applet?- init() method - Can be called when an applet is first loaded start() method - Can be called each time an applet is started. paint() method - Can be called when the applet is minimized or maximized. stop() method - Can be used when the browser moves off the applet’s page. destroy() method - Can be called when the browser is finished with the applet.

How do you set security in applets?- using setSecurityManager() method

What is an event and what are the models available for event handling?- An event is an event object that describes a state of change in a source. In other words, event occurs when an action is generated, like pressing button, clicking mouse, selecting a list, etc. There are two types of models for handling events and they are: a) event-inheritance model and b) event-delegation model

What are the advantages of the model over the event-inheritance model?- The event-delegation model has two advantages over the event-inheritance model. They are: a)It enables event handling by objects other than the ones that generate the events. This allows a clean separation between a component’s design and its use. b)It performs much better in applications where many events are generated. This performance improvement is due to the fact that the event-delegation model does not have to be repeatedly process unhandled events as is the case of the event-inheritance.

What is source and listener?- source : A source is an object that generates an event. This occurs when the internal state of that object changes in some way. listener : A listener is an object that is notified when an event occurs. It has two major requirements. First, it must have been registered with one or more sources to receive notifications about specific types of events. Second, it must implement methods to receive and process these notifications.

What is adapter class?- An adapter class provides an empty implementation of all methods in an event listener interface. Adapter classes are useful when you want to receive and process only some of the events that are handled by a particular event listener

interface. You can define a new class to act listener by extending one of the adapter classes and implementing only those events in which you are interested. For example, the MouseMotionAdapter class has two methods, mouseDragged()and mouseMoved(). The signatures of these empty are exactly as defined in the MouseMotionListener interface. If you are interested in only mouse drag events, then you could simply extend MouseMotionAdapter and implement mouseDragged() .
What is meant by controls and what are different types of controls in AWT?- Controls are components that allow a user to interact with your application and the AWT supports the following types of controls: Labels, Push Buttons, Check Boxes, Choice Lists, Lists, Scrollbars, Text Components. These controls are subclasses of Component.
What is the difference between choice and list?- A Choice is displayed in a compact form that requires you to pull it down to see the list of available choices and only one item may be selected from a choice. A List may be displayed in such a way that several list items are visible and it supports the selection of one or more list items.
What is the difference between scrollbar and scrollpane?- A Scrollbar is a Component, but not a Container whereas Scrollpane is a Conatiner and handles its own events and perform its own scrolling.
What is a layout manager and what are different types of layout managers available in java AWT?- A layout manager is an object that is used to organize components in a container. The different layouts are available are FlowLayout, BorderLayout, CardLayout, GridLayout and GridBagLayout.
How are the elements of different layouts organized?- FlowLayout: The elements of a FlowLayout are organized in a top to bottom, left to right fashion. BorderLayout: The elements of a BorderLayout are organized at the borders (North, South, East and West) and the center of a container. CardLayout: The elements of a CardLayout are stacked, on top of the other, like a deck of cards. GridLayout: The elements of a GridLayout are of equal size and are laid out using the square of a grid. GridBagLayout: The elements of a GridBagLayout are organized according to a grid. However, the elements are of different size and may occupy more than one row or column of the grid. In addition, the rows and columns may have different sizes.
Which containers use a Border layout as their default layout?- Window, Frame and Dialog classes use a BorderLayout as their layout.
Which containers use a Flow layout as their default layout?- Panel and Applet classes use the FlowLayout as their default layout.
What are wrapper classes?- Wrapper classes are classes that allow primitive types to be accessed as objects.
What are Vector, Hashtable, LinkedList and Enumeration?- Vector : The Vector class provides the capability to implement a growable array of objects. Hashtable : The Hashtable class implements a Hashtable data structure. A Hashtable indexes and stores objects in a dictionary using hash codes as the object's keys. Hash codes are integer values that identify objects. LinkedList: Removing or inserting elements in the middle of an array can be done using LinkedList. A LinkedList stores each object in a separate link whereas an array stores object references in consecutive locations. Enumeration: An object that implements the Enumeration interface generates a series of elements, one at a time. It has two methods, namely hasMoreElements() and nextElement(). HasMoreElemnts() tests if this enumeration has more elements and nextElement method returns successive elements of the series.
What is the difference between set and list?- Set stores elements in an unordered way but does not contain duplicate elements, whereas list stores elements in an ordered way but may contain duplicate elements.
What is a stream and what are the types of Streams and classes of the Streams?- A Stream is an abstraction that either produces or consumes information. There are two types of Streams and they are: Byte Streams: Provide a convenient means for handling input and output of bytes. Character Streams: Provide a convenient means for handling input & output of characters. Byte Streams classes: Are defined by using two abstract classes, namely InputStream and OutputStream. Character Streams classes: Are defined by using two abstract classes, namely Reader and Writer.
What is the difference between Reader/Writer and InputStream/Output Stream?- The Reader/Writer class is character-oriented and the InputStream/OutputStream class is byte-oriented.
What is an I/O filter?- An I/O filter is an object that reads from one stream and writes to another, usually altering the data in some way as it is passed from one stream to another.
What is serialization and deserialization?- Serialization is the process of writing the state of an object to a byte stream. Deserialization is the process of restoring these objects.
What is JDBC?- JDBC is a set of Java API for executing SQL statements. This API consists of a set of classes and interfaces to enable programs to write pure Java Database applications.
What are drivers available?- a) JDBC-ODBC Bridge driver b) Native API Partly-Java driver c) JDBC-Net Pure Java driver d) Native-Protocol Pure Java driver
What is the difference between JDBC and ODBC?- a) OBDC is for Microsoft and JDBC is for Java applications. b) ODBC can't be directly used with Java because it uses a C interface. c) ODBC makes use of pointers which have been removed totally from Java. d) ODBC mixes simple and advanced features together and has complex options for simple queries. But JDBC is designed to keep things simple while allowing advanced capabilities when required. e) ODBC requires manual installation of the ODBC driver manager and driver on all client machines. JDBC drivers are written in Java and JDBC code is automatically installable, secure, and portable on all platforms. f) JDBC API is a natural Java interface and is built on ODBC. JDBC retains some of the basic features of ODBC.
What are the types of JDBC Driver Models and explain them?- There are two types of JDBC Driver Models and they are: a) Two tier model and b) Three tier model Two tier model: In this model, Java applications interact directly with the database. A JDBC driver is required to communicate with the particular database management system that is being accessed. SQL statements are sent to the database and the results are given to user. This model is referred to as client/server configuration where user is the client and the machine that has the database is called as the server. Three tier model: A middle tier is introduced in this model. The functions of this model are: a) Collection of SQL statements from the client and handing it over to the database, b) Receiving results from database to the client and c) Maintaining control over accessing and updating of the above.
What are the steps involved for making a connection with a database or how do you connect to a database?a) Loading the driver : To load the driver, Class. forName() method is used. Class. forName("sun. jdbc. odbc. JdbcOdbcDriver"); When the driver is

loaded, it registers itself with the java. sql. DriverManager class as an available database driver. b) Making a connection with database: To open a connection to a given database, DriverManager. getConnection() method is used. Connection con = DriverManager. getConnection (”jdbc:odbc:somedb”, “user”, “password”); c) Executing SQL statements : To execute a SQL query, java. sql. statements class is used. createStatement() method of Connection to obtain a new Statement object. Statement stmt = con. createStatement(); A query that returns data can be executed using the executeQuery() method of Statement. This method executes the statement and returns a java. sql. ResultSet that encapsulates the retrieved data: ResultSet rs = stmt. executeQuery(”SELECT * FROM some table”); d) Process the results : ResultSet returns one row at a time. Next() method of ResultSet object can be called to move to the next row. The getString() and getObject() methods are used for retrieving column values: while(rs. next()) { String event = rs. getString(”event”); Object count = (Integer) rs. getObject(”count”);
What type of driver did you use in project?- JDBC-ODBC Bridge driver (is a driver that uses native(C language) libraries and makes calls to an existing ODBC driver to access a database engine).
What are the types of statements in JDBC?- Statement: to be used createStatement() method for executing single SQL statement PreparedStatement — To be used preparedStatement() method for executing same SQL statement over and over.
CallableStatement — To be used prepareCall() method for multiple SQL statements over and over.
What is stored procedure?- Stored procedure is a group of SQL statements that forms a logical unit and performs a particular task. Stored Procedures are used to encapsulate a set of operations or queries to execute on database. Stored procedures can be compiled and executed with different parameters and results and may have any combination of input/output parameters.
How to create and call stored procedures?- To create stored procedures: Create procedure procedurename (specify in, out and in out parameters) BEGIN Any multiple SQL statement; END; To call stored procedures: CallableStatement csmt = con. prepareCall(”{call procedure name(?,?)}”); csmt. registerOutParameter(column no. , data type); csmt. setInt(column no. , column name) csmt. execute();

1.what is a transient variable?
A transient variable is a variable that may not be serialized.

2.which containers use a border Layout as their default layout?
The window, Frame and Dialog classes use a border layout as their default layout.

3.Why do threads block on I/O?
Threads block on i/o (that is enters the waiting state) so that other threads may execute while the i/o Operation is performed.

4. How are Observer and Observable used?
Objects that subclass the Observable class maintain a list of observers. When an Observable object is updated it invokes the update() method of each of its observers to notify the observers that it has changed state. The Observer interface is implemented by objects that observe Observable objects.

5. What is synchronization and why is it important?
With respect to multithreading, synchronization is the capability to control the access of multiple threads to shared resources. Without synchronization, it is possible for one thread to modify a shared object while another thread is in the process of using or updating that object's value. This often leads to significant errors.

6. Can a lock be acquired on a class?
Yes, a lock can be acquired on a class. This lock is acquired on the class's Class object..

7. What's new with the stop(), suspend() and resume() methods in JDK 1.2?
The stop(), suspend() and resume() methods have been deprecated in JDK 1.2.

8. Is null a keyword?
The null value is not a keyword.

9. What is the preferred size of a component?
The preferred size of a component is the minimum component size that will allow the component to display normally.

10. What method is used to specify a container's layout?
The setLayout() method is used to specify a container's layout.

11. Which containers use a FlowLayout as their default layout?
The Panel and Applet classes use the FlowLayout as their default layout.

12. What state does a thread enter when it terminates its processing?
When a thread terminates its processing, it enters the dead state.

13. What is the Collections API?
The Collections API is a set of classes and interfaces that support operations on collections of objects.

14. Which characters may be used as the second character of an identifier,
but not as the first character of an identifier?
The digits 0 through 9 may not be used as the first character of an identifier but they may be used after the first character of an identifier.

15. What is the List interface?
The List interface provides support for ordered collections of objects.

16. How does Java handle integer overflows and underflows?
It uses those low order bytes of the result that can fit into the size of the type allowed by the operation.

17. What is the Vector class?
The Vector class provides the capability to implement a growable array of objects

18. What modifiers may be used with an inner class that is a member of an outer class?
A (non-local) inner class may be declared as public, protected, private, static, final, or abstract.

19. What is an Iterator interface?
The Iterator interface is used to step through the elements of a Collection.

20. What is the difference between the >> and >>> operators?
The >> operator carries the sign bit when shifting right. The >>> zero-fills bits that have been shifted out.

21. Which method of the Component class is used to set the position and
size of a component?
setBounds()

22. How many bits are used to represent Unicode, ASCII, UTF-16, and UTF-8 characters?
Unicode requires 16 bits and ASCII require 7 bits. Although the ASCII character set uses only 7 bits, it is usually represented as 8 bits. UTF-8 represents characters using 8, 16, and 18 bit patterns. UTF-16 uses 16-bit and larger bit patterns.

23What is the difference between yielding and sleeping?
When a task invokes its yield() method, it returns to the ready state. When a task invokes its sleep() method, it returns to the waiting state.

24. Which java.util classes and interfaces support event handling?
The EventObject class and the EventListener interface support event processing.

25. Is sizeof a keyword?
The sizeof operator is not a keyword.

26. What are wrapped classes?
Wrapped classes are classes that allow primitive types to be accessed as objects.

27. Does garbage collection guarantee that a program will not run out of memory?
Garbage collection does not guarantee that a program will not run out of memory. It is possible for programs to use up memory resources faster than they are garbage collected. It is also possible for programs to create objects that are not subject to garbage collection

28. What restrictions are placed on the location of a package statement
within a source code file?
A package statement must appear as the first line in a source code file (excluding blank lines and comments).

29. Can an object's finalize() method be invoked while it is reachable?
An object's finalize() method cannot be invoked by the garbage collector while the object is still reachable. However, an object's finalize() method may be invoked by other objects.

30. What is the immediate superclass of the Applet class?
Panel

31. What is the difference between preemptive scheduling and time slicing?
Under preemptive scheduling, the highest priority task executes until it enters the waiting or dead states or a higher priority task comes into existence. Under time slicing, a task executes for a predefined slice of time and then reenters the pool of ready tasks.

The scheduler then determines which task should execute next, based on priority and
other factors.

32. Name three Component subclasses that support painting.
The Canvas, Frame, Panel, and Applet classes support painting.

33. What value does readLine() return when it has reached the end of a file?
The readLine() method returns null when it has reached the end of a file.

34. What is the immediate superclass of the Dialog class?
Window

35. What is clipping?
Clipping is the process of confining paint operations to a limited area or shape.

36. What is a native method?
A native method is a method that is implemented in a language other than Java.

37. Can a for statement loop indefinitely?
Yes, a for statement can loop indefinitely. For example, consider the following:
for(;;) ;

38. What are order of precedence and associativity, and how are they used?
Order of precedence determines the order in which operators are evaluated in expressions. Associatity determines whether an expression is evaluated left-to-right or right-to-left

39. When a thread blocks on I/O, what state does it enter?
A thread enters the waiting state when it blocks on I/O.

40. To what value is a variable of the String type automatically initialized?
The default value of an String type is null.

41. What is the catch or declare rule for method declarations?
If a checked exception may be thrown within the body of a method, the method must either catch the exception or declare it in its throws clause.

42. What is the difference between a MenuItem and a CheckboxMenuItem?
The CheckboxMenuItem class extends the MenuItem class to support a menu item that may be checked or unchecked.

43. What is a task's priority and how is it used in scheduling?
A task's priority is an integer value that identifies the relative order in which it should be executed with respect to other tasks. The scheduler attempts to schedule higher priority tasks before lower priority tasks.

44. What class is the top of the AWT event hierarchy?
The java.awt.AWTEvent class is the highest-level class in the AWT event-class hierarchy.

45. When a thread is created and started, what is its initial state?
A thread is in the ready state after it has been created and started.

46. Can an anonymous class be declared as implementing an interface and extending a class?
An anonymous class may implement an interface or extend a superclass, but may not be declared to do both.

47. What is the range of the short type?
The range of the short type is -(2^15) to 2^15 - 1.

48. What is the range of the char type?
The range of the char type is 0 to 2^16 - 1.

49. In which package are most of the AWT events that support the event-delegation
model defined?
Most of the AWT-related events of the event-delegation model are defined in the java.awt.event package. The AWTEvent class is defined in the java.awt package.

50. What is the immediate superclass of Menu?
MenuItem

51. What is the purpose of finalization?
The purpose of finalization is to give an unreachable object the opportunity to perform any cleanup processing before the object is garbage collected.

52. Which class is the immediate superclass of the MenuComponent class.
Object

53. What invokes a thread's run() method?
After a thread is started, via its start() method or that of the Thread class, the JVM invokes the thread's run() method when the thread is initially executed.

54. What is the difference between the Boolean & operator and the && operator?
If an expression involving the Boolean & operator is evaluated, both operands are evaluated. Then the & operator is applied to the operand. When an expression involving the && operator is evaluated, the first operand is evaluated. If the first operand returns a value of true then the second operand is evaluated. The && operator is then applied to the first and second operands. If the first operand evaluates to false, the evaluation of the second operand is skipped.

55. Name three subclasses of the Component class.
Box.Filler, Button, Canvas, Checkbox, Choice, Container, Label, List, Scrollbar, or TextComponent

56. What is the GregorianCalendar class?
The GregorianCalendar provides support for traditional Western calendars.

57. Which Container method is used to cause a container to be laid out and redisplayed?
validate()

58. What is the purpose of the Runtime class?
The purpose of the Runtime class is to provide access to the Java runtime system.

59. How many times may an object's finalize() method be invoked by the
garbage collector?
An object's finalize() method may only be invoked once by the garbage collector.

60. What is the purpose of the finally clause of a try-catch-finally statement?
The finally clause is used to provide the capability to execute code no matter whether or not an exception is thrown or caught.

61. What is the argument type of a program's main() method?
A program's main() method takes an argument of the String[] type.

62. Which Java operator is right associative?
The = operator is right associative.

63. What is the Locale class?
The Locale class is used to tailor program output to the conventions of a particular geographic, political, or cultural region.

64. Can a double value be cast to a byte?
Yes, a double value can be cast to a byte.

65. What is the difference between a break statement and a continue statement?
A break statement results in the termination of the statement to which it applies (switch, for, do, or while). A continue statement is used to end the current loop iteration and return control to the loop statement.

66. What must a class do to implement an interface?
It must provide all of the methods in the interface and identify the interface in its implements clause.

67. What method is invoked to cause an object to begin executing as a separate thread?
The start() method of the Thread class is invoked to cause an object to begin executing as a separate thread.

68. Name two subclasses of the TextComponent class.
TextField and TextArea

69. What is the advantage of the event-delegation model over the earlier event-inheritance model?
The event-delegation model has two advantages over the event-inheritance model. First, it enables event handling to be handled by objects other than the ones that generate the events (or their containers). This allows a clean separation between a component's design and its use. The other advantage of the event-delegation model is that it performs much better in applications where many

events are generated. This performance improvement is due to the fact that the event-delegation model does not have to repeatedly process unhandled events, as is the case of the event-inheritance model.

70. Which containers may have a MenuBar?
Frame

71. How are commas used in the intialization and iteration parts of a for statement?
Commas are used to separate multiple statements within the initialization and iteration parts of a for statement.

72. What is the purpose of the wait(), notify(), and notifyAll() methods?
The wait(),notify(), and notifyAll() methods are used to provide an efficient way for threads to wait for a shared resource. When a thread executes an object's wait() method, it enters the waiting state. It only enters the ready state after another thread invokes the object's notify() or notifyAll() methods..

73. What is an abstract method?
An abstract method is a method whose implementation is deferred to a subclass.

74. How are Java source code files named?
A Java source code file takes the name of a public class or interface that is defined within the file. A source code file may contain at most one public class or interface. If a public class or interface is defined within a source code file, then the source code file must take the name of the public class or interface. If no public class or interface is defined within a source code file, then the file must take on a name that is different than its classes and interfaces. Source code files use the .java extension.

75. What is the relationship between the Canvas class and the Graphics class?
A Canvas object provides access to a Graphics object via its paint() method.

76. What are the high-level thread states?
The high-level thread states are ready, running, waiting, and dead.

77. What value does read() return when it has reached the end of a file?
The read() method returns -1 when it has reached the end of a file.

78. Can a Byte object be cast to a double value?
No, an object cannot be cast to a primitive value.

79. What is the difference between a static and a non-static inner class?
A non-static inner class may have object instances that are associated with instances of the class's outer class. A static inner class does not have any object instances.

80. What is the difference between the String and StringBuffer classes?
String objects are constants. StringBuffer objects are not.

81. If a variable is declared as private, where may the variable be accessed?
A private variable may only be accessed within the class in which it is declared.

82. What is an object's lock and which object's have locks?
An object's lock is a mechanism that is used by multiple threads to obtain synchronized access to the object. A thread may execute a synchronized method of an object only after it has acquired the object's lock. All objects and classes have locks. A class's lock is acquired on the class's Class object.

83. What is the Dictionary class?
The Dictionary class provides the capability to store key-value pairs.

84. How are the elements of a BorderLayout organized?
The elements of a BorderLayout are organized at the borders (North, South, East, and West) and the center of a container.

85. What is the % operator?
It is referred to as the modulo or remainder operator. It returns the remainder of dividing the first operand by the second operand.

86. When can an object reference be cast to an interface reference?
An object reference be cast to an interface reference when the object implements the referenced interface.

87. What is the difference between a Window and a Frame?
The Frame class extends Window to define a main application window that can have a menu bar.

88. Which class is extended by all other classes?

The Object class is extended by all other classes.

89. Can an object be garbage collected while it is still reachable?
A reachable object cannot be garbage collected. Only unreachable objects may be garbage collected..

90. Is the ternary operator written x : y ? z or x ? y : z ?
It is written x ? y : z.

91. What is the difference between the Font and FontMetrics classes?
The FontMetrics class is used to define implementation-specific properties, such as ascent and descent, of a Font object.

92. How is rounding performed under integer division?
The fractional part of the result is truncated. This is known as rounding toward zero.

93. What happens when a thread cannot acquire a lock on an object?
If a thread attempts to execute a synchronized method or synchronized statement and is unable to acquire an object's lock, it enters the waiting state until the lock becomes available.

94. What is the difference between the Reader/Writer class hierarchy and the
InputStream/OutputStream class hierarchy?
The Reader/Writer class hierarchy is character-oriented, and the InputStream/OutputStream class hierarchy is byte-oriented.

95. What classes of exceptions may be caught by a catch clause?
A catch clause can catch any exception that may be assigned to the Throwable type. This includes the Error and Exception types.

96. If a class is declared without any access modifiers, where may the class be accessed?
A class that is declared without any access modifiers is said to have package access. This means that the class can only be accessed by other classes and interfaces that are defined within the same package.

97. What is the SimpleTimeZone class?
The SimpleTimeZone class provides support for a Gregorian calendar.

98. What is the Map interface?
The Map interface replaces the JDK 1.1 Dictionary class and is used associate keys with values.

99. Does a class inherit the constructors of its superclass?
A class does not inherit constructors from any of its superclasses.

100. For which statements does it make sense to use a label?
The only statements for which it makes sense to use a label are those statements that can enclose a break or continue statement.

101. What is the purpose of the System class?
The purpose of the System class is to provide access to system resources.

102. Which TextComponent method is used to set a TextComponent to the read-only state?
setEditable()

103. How are the elements of a CardLayout organized?
The elements of a CardLayout are stacked, one on top of the other, like a deck of cards.

104. Is &&= a valid Java operator?
No, it is not.

105. Name the eight primitive Java types.
The eight primitive types are byte, char, short, int, long, float, double, and boolean.

106. Which class should you use to obtain design information about an object?
The Class class is used to obtain information about an object's design.

107. What is the relationship between clipping and repainting?
When a window is repainted by the AWT painting thread, it sets the clipping regions to the area of the window that requires repainting.

108. Is "abc" a primitive value?
The String literal "abc" is not a primitive value. It is a String object.

109. What is the relationship between an event-listener interface and an
event-adapter class?
An event-listener interface defines the methods that must be implemented by an event handler for a particular kind of event. An event adapter provides a default implementation of an event-listener interface.

110. What restrictions are placed on the values of each case of a switch statement?
During compilation, the values of each case of a switch statement must evaluate to a value that can be promoted to an int value.

111. What modifiers may be used with an interface declaration?
An interface may be declared as public or abstract.

112. Is a class a subclass of itself?
A class is a subclass of itself.

113. What is the highest-level event class of the event-delegation model?
The java.util.EventObject class is the highest-level class in the event-delegation class hierarchy.

114. What event results from the clicking of a button?
The ActionEvent event is generated as the result of the clicking of a button.

115. How can a GUI component handle its own events?
A component can handle its own events by implementing the required event-listener interface and adding itself as its own event listener.

116. What is the difference between a while statement and a do statement?
A while statement checks at the beginning of a loop to see whether the next loop iteration should occur. A do statement checks at the end of a loop to see whether the next iteration of a loop should occur. The do statement will always execute the body of a loop at least once.

117. How are the elements of a GridBagLayout organized?
The elements of a GridBagLayout are organized according to a grid. However, the elements are of different sizes and may occupy more than one row or column of the grid. In addition, the rows and columns may have different sizes.

118. What advantage do Java's layout managers provide over traditional windowing systems?
Java uses layout managers to lay out components in a consistent manner across all windowing platforms. Since Java's layout managers aren't tied to absolute sizing and positioning, they are able to accomodate platform-specific differences among windowing systems.

119. What is the Collection interface?
The Collection interface provides support for the implementation of a mathematical bag - an unordered collection of objects that may contain duplicates.

120. What modifiers can be used with a local inner class?
A local inner class may be final or abstract.

121. What is the difference between static and non-static variables?
A static variable is associated with the class as a whole rather than with specific instances of a class. Non-static variables take on unique values with each object instance.

122. What is the difference between the paint() and repaint() methods?
The paint() method supports painting via a Graphics object. The repaint() method is used to cause paint() to be invoked by the AWT painting thread.

123. What is the purpose of the File class?
The File class is used to create objects that provide access to the files and directories of a local file system.

124. Can an exception be rethrown?
Yes, an exception can be rethrown.
125. Which Math method is used to calculate the absolute value of a number?
The abs() method is used to calculate absolute values.

126. How does multithreading take place on a computer with a single CPU?
The operating system's task scheduler allocates execution time to multiple tasks. By quickly switching between executing tasks, it creates the impression that tasks execute sequentially.

127. When does the compiler supply a default constructor for a class?

The compiler supplies a default constructor for a class if no other constructors are provided.

128. When is the finally clause of a try-catch-finally statement executed?
The finally clause of the try-catch-finally statement is always executed unless the thread of execution terminates or an exception occurs within the execution of the finally clause.

129. Which class is the immediate superclass of the Container class?
Component

130. If a method is declared as protected, where may the method be accessed?
A protected method may only be accessed by classes or interfaces of the same package or by subclasses of the class in which it is declared.

131. How can the Checkbox class be used to create a radio button?
By associating Checkbox objects with a CheckboxGroup.

132. Which non-Unicode letter characters may be used as the first character
of an identifier?
The non-Unicode letter characters $ and _ may appear as the first character of an identifier

133. What restrictions are placed on method overloading?
Two methods may not have the same name and argument list but different return types.

134. What happens when you invoke a thread's interrupt method while it is
sleeping or waiting?
When a task's interrupt() method is executed, the task enters the ready state. The next time the task enters the running state, an InterruptedException is thrown.

135. What is casting?
There are two types of casting, casting between primitive numeric types and casting between object references. Casting between numeric types is used to convert larger values, such as double values, to smaller values, such as byte values. Casting between object references is used to refer to an object by a compatible class, interface, or array type reference.

136. What is the return type of a program's main() method?
A program's main() method has a void return type.

137. Name four Container classes.
Window, Frame, Dialog, FileDialog, Panel, Applet, or ScrollPane

138. What is the difference between a Choice and a List?
A Choice is displayed in a compact form that requires you to pull it down to see the list of available choices. Only one item may be selected from a Choice. A List may be displayed in such a way that several List items are visible. A List supports the selection of one or more List items.

139. What class of exceptions are generated by the Java run-time system?
The Java runtime system generates RuntimeException and Error exceptions.

140. What class allows you to read objects directly from a stream?
The ObjectInputStream class supports the reading of objects from input streams.

141. What is the difference between a field variable and a local variable?
A field variable is a variable that is declared as a member of a class. A local variable is a variable that is declared local to a method.

142. Under what conditions is an object's finalize() method invoked by the garbage collector?
The garbage collector invokes an object's finalize() method when it detects that the object has become unreachable.

143. How are this() and super() used with constructors?
this() is used to invoke a constructor of the same class. super() is used to invoke a superclass constructor.

144. What is the relationship between a method's throws clause and the exceptions
that can be thrown during the method's execution?
A method's throws clause must declare any checked exceptions that are not caught within the body of the method.

145. What is the difference between the JDK 1.02 event model and the event-delegation
model introduced with JDK 1.1?

The JDK 1.02 event model uses an event inheritance or bubbling approach. In this model, components are required to handle their own events. If they do not handle a particular event, the event is inherited by (or bubbled up to) the component's container. The container then either handles the event or it is bubbled up to its container and so on, until the highest-level container has been tried..
In the event-delegation model, specific objects are designated as event handlers for GUI components. These objects implement event-listener interfaces. The event-delegation model is more efficient than the event-inheritance model because it eliminates the processing required to support the bubbling of unhandled events.

146. How is it possible for two String objects with identical values not to be equal
under the == operator?
The == operator compares two objects to determine if they are the same object in memory. It is possible for two String objects to have the same value, but located indifferent areas of memory.

147. Why are the methods of the Math class static?
So they can be invoked as if they are a mathematical code library.

148. What Checkbox method allows you to tell if a Checkbox is checked?
getState()

149. What state is a thread in when it is executing?
An executing thread is in the running state.

150. What are the legal operands of the instanceof operator?
The left operand is an object reference or null value and the right operand is a class, interface, or array type.

151. How are the elements of a GridLayout organized?
The elements of a GridBad layout are of equal size and are laid out using the squares of a grid.

152. What an I/O filter?
An I/O filter is an object that reads from one stream and writes to another, usually altering the data in some way as it is passed from one stream to another.

153. If an object is garbage collected, can it become reachable again?
Once an object is garbage collected, it ceases to exist. It can no longer become reachable again.

154. What is the Set interface?
The Set interface provides methods for accessing the elements of a finite mathematical set. Sets do not allow duplicate elements.

155. What classes of exceptions may be thrown by a throw statement?
A throw statement may throw any expression that may be assigned to the Throwable type.

156. What are E and PI?
E is the base of the natural logarithm and PI is mathematical value pi.

157. Are true and false keywords?
The values true and false are not keywords.

158. What is a void return type?
A void return type indicates that a method does not return a value.

159. What is the purpose of the enableEvents() method?
The enableEvents() method is used to enable an event for a particular object. Normally, an event is enabled when a listener is added to an object for a particular event. The enableEvents() method is used by objects that handle events by overriding their event-dispatch methods.

160. What is the difference between the File and RandomAccessFile classes?
The File class encapsulates the files and directories of the local file system. The RandomAccessFile class provides the methods needed to directly access data contained in any part of a file.

161. What happens when you add a double value to a String?
The result is a String object.

162. What is your platform's default character encoding?
If you are running Java on English Windows platforms, it is probably Cp1252. If you are running Java on English Solaris platforms, it is most likely 8859_1..

163. Which package is always imported by default?
The java.lang package is always imported by default.

164. What interface must an object implement before it can be written to a
stream as an object?
An object must implement the Serializable or Externalizable interface before it can be written to a stream as an object.

165. How are this and super used?
this is used to refer to the current object instance. super is used to refer to the variables and methods of the superclass of the current object instance.

166. What is the purpose of garbage collection?
The purpose of garbage collection is to identify and discard objects that are no longer needed by a program so that their resources may be reclaimed and reused.

167. What is a compilation unit?
A compilation unit is a Java source code file.

168. What interface is extended by AWT event listeners?
All AWT event listeners extend the java.util.EventListener interface.

169. What restrictions are placed on method overriding?
Overridden methods must have the same name, argument list, and return type.
The overriding method may not limit the access of the method it overrides.
The overriding method may not throw any exceptions that may not be thrown
by the overridden method.

170. How can a dead thread be restarted?
A dead thread cannot be restarted.

171. What happens if an exception is not caught?
An uncaught exception results in the uncaughtException() method of the thread's ThreadGroup being invoked, which eventually results in the termination of the program in which it is thrown.

172. What is a layout manager?
A layout manager is an object that is used to organize components in a container.

173. Which arithmetic operations can result in the throwing of an ArithmeticException?
Integer / and % can result in the throwing of an ArithmeticException.

174. What are three ways in which a thread can enter the waiting state?
A thread can enter the waiting state by invoking its sleep() method, by blocking on I/O, by unsuccessfully attempting to acquire an object's lock, or by invoking an object's wait() method. It can also enter the waiting state by invoking its (deprecated) suspend() method.

175. Can an abstract class be final?
An abstract class may not be declared as final.

176. What is the ResourceBundle class?
The ResourceBundle class is used to store locale-specific resources that can be loaded by a program to tailor the program's appearance to the particular locale in which it is being run.

177. What happens if a try-catch-finally statement does not have a catch clause to handle an exception that is thrown within the body of the try statement?
The exception propagates up to the next higher level try-catch statement (if any) or results in the program's termination.

178. What is numeric promotion?
Numeric promotion is the conversion of a smaller numeric type to a larger numeric type, so that integer and floating-point operations may take place. In numerical promotion, byte, char, and short values are converted to int values. The int values are also converted to long values, if necessary. The long and float values are converted to double values, as required.

179. What is the difference between a Scrollbar and a ScrollPane?
A Scrollbar is a Component, but not a Container. A ScrollPane is a Container. A ScrollPane handles its own events and performs its own scrolling.

180. What is the difference between a public and a non-public class?

A public class may be accessed outside of its package. A non-public class may not be accessed outside of its package.

181. To what value is a variable of the boolean type automatically initialized?
The default value of the boolean type is false.

182. Can try statements be nested?
Try statements may be tested.

183. What is the difference between the prefix and postfix forms of the ++ operator?
The prefix form performs the increment operation and returns the value of the increment operation. The postfix form returns the current value all of the expression and then performs the increment operation on that value.

184. What is the purpose of a statement block?
A statement block is used to organize a sequence of statements as a single statement group.

185. What is a Java package and how is it used?
A Java package is a naming context for classes and interfaces. A package is used to create a separate name space for groups of classes and interfaces. Packages are also used to organize related classes and interfaces into a single API unit and to control accessibility to these classes and interfaces.

186. What modifiers may be used with a top-level class?
A top-level class may be public, abstract, or final.

187. What are the Object and Class classes used for?
The Object class is the highest-level class in the Java class hierarchy. The Class class is used to represent the classes and interfaces that are loaded by a Java program..

188. How does a try statement determine which catch clause should be used to handle an exception?
When an exception is thrown within the body of a try statement, the catch clauses of the try statement are examined in the order in which they appear. The first catch clause that is capable of handling the exception is executed. The remaining catch clauses are ignored.

189. Can an unreachable object become reachable again?
An unreachable object may become reachable again. This can happen when the object's finalize() method is invoked and the object performs an operation which causes it to become accessible to reachable objects.

190. When is an object subject to garbage collection?
An object is subject to garbage collection when it becomes unreachable to the program in which it is used.

191. What method must be implemented by all threads?
All tasks must implement the run() method, whether they are a subclass of Thread or implement the Runnable interface.

192. What methods are used to get and set the text label displayed by a Button object?
getLabel() and setLabel()

193. Which Component subclass is used for drawing and painting?
Canvas

194. What are synchronized methods and synchronized statements?
Synchronized methods are methods that are used to control access to an object. A thread only executes a synchronized method after it has acquired the lock for the method's object or class. Synchronized statements are similar to synchronized methods. A synchronized statement can only be executed after a thread has acquired the lock for the object or class referenced in the synchronized statement.

195. What are the two basic ways in which classes that can be run as threads may be defined?
A thread class may be declared as a subclass of Thread, or it may implement the Runnable interface.

196. What are the problems faced by Java programmers who don't use layout managers?
Without layout managers, Java programmers are faced with determining how their GUI will be displayed across multiple windowing systems and finding a common sizing and positioning that will work within the constraints imposed by each windowing system.

197. What is the difference between an if statement and a switch statement?
The if statement is used to select among two alternatives. It uses a boolean expression to decide which alternative should be executed. The switch statement is used to select among multiple alternatives. It uses an int expression to determine which alternative should be executed.

198. What happens when you add a double value to a String?
The result is a String object.

199. What is the List interface?
The List interface provides support for ordered collections of objects.

729 what is socket? and server socket?
Socket is end part of communication system.It is virtually exits.In socket function it contain two parameter first one is ip address and second is post no.. Port no is simply id of process. Server socket is server part it bind and create the server socket. further

730 Why Concept Of Pointer is not there in java
Pointers are not there mainly to avoid security threats. As JVM restricts access to the file system, but thru pointers you can access any memory location. This is the main reason, other is to avoid the complexity.

731 when does JIT plays its role. Does JIT come along...
JIT - Just In Time compiler, is embedded in the web browser, like the internet explorer!
The interpreter for Java, which executes the Byte Codes, is the JVM - Java Virtual Machine.
JIT is also a part of the JVM, and it compiles bytecodes into real-time code, when needed as on demand. The entire Java program is not compiled into an executable code at once, coz there need to performed various run-time checks. However, JIT compiles code as and when needed, during execution. To put it simply, the JIT compiler is faster than the JVM.
JIT comes into play when you execute applications written in JavaScript.
JIT is java interpreter. The running of a java program is a two step process
Step-1 : The source code is compiled using the Java compiler which generates bytecode
Step-2 : The bytecode is then transfered to JIT which changes it into object code adding the configurations of the machine in which the code is to be executed
Yes JIT comes along with JDK

732 Will class level variables are garbage collected?
yes
733 Vector or ArrayList -- which is better and why?
Sometimes Vector is better; sometimes ArrayList is better; sometimes you don't want to use either. I hope you weren't looking for an easy answer because the answer depends upon what you are doing.
Vector and ArrayList both can expand size if required. A Vector defaults to doubling the size of its array, while the ArrayList increases its array size by 50 percent. If you don't know how much data you'll have, but you do know the rate at which it grows, Vector does possess a slight advantage since you can set the increment value.
Basically Vector is Syncronized Array list is Not. We can make array List also as Syncronized by using :-
Collections.SyncronizedList(new ArryList())
Vikas J Mandal Wrote:
Sometimes Vector is better; sometimes ArrayList is better; sometimes you don't want to use either. I hope you weren't looking for an easy answer because the answer depends upon what you are doing.
The key difference between Arrays and Vectors in Java is that Vectors are dynamically-allocated.They aren't declared to contain a type of variable; instead, each Vector contains a dynamic list of references to other objects. The Vector class is found in the java.util package, and extends java.util.Abstractlist.
The big advantage of using Vectors is that the size of the vector can change as needed.Vector and ArrayList both can expand size if required. A Vector defaults to doubling the size of its array, while the ArrayList increases its array size by 50 percent. If you don't know how much data you'll have, but you do know the rate at which it grows, Vector does possess a slight advantage since you can set the increment value.
Basically Vector is Syncronized Array list is Not. We can make array List also as Syncronized by using :-
Collections.SyncronizedList(new ArryList())
Because Vectors are dynamically-allocated, they offer a relatively memory-efficient way of handling lists whose size could change drastically.Because Vectors are dynamically-allocated, they offer a relatively memory-efficient way of handling lists whose size could change drastically.Vectors have some useful member functions that make coding simpler.Vector class is based on an array of object references, these methods are generally no more efficient than array-based algorithms.Thus, Vectors are easier to use than arrays for most applications, but they do not offer all the performance advantages of fully-dynamic storage
734 Mention 5 basic difference between Array List and Vector in Java Colletion FrameWork.

Vector is unsynchronized wheras ArrayList is synchronized.

001 Question: Tell me the differences between enumeration and iteration?Which can use where in realtime?
Answer:Enumeration deals with objects while iteration deals with values only. Enumeration is used when we use vector, hashtable, etc while iteration are used in while loop, for loop etc

002 Question: What are the differences between ArrayList and a Vector
Answesr:the basic differences are
vector is a old collectiion came with jdk 1.0 or later.
so, it comes under legacy classes.
it allows synchronized way of accessing the elements.

where as arraylist allows elements to be made synchronized as well asynchronized.
its is dynamic collectiion.
u have more flexibilty of using the arraylist collection.
----
1. Arraylist is not synchronized while vector is.
2. Arraylist has no default size while vector has a default size of 10.

Note: Methods in Vector are synchronised which means they are thread-safe and thus preclude access to the Vector elements by concurrent threads.Bu this imposes additional overhead on the JVM as it has to acquire and release locks on the vector objects under consideration.

This is not possible in ArrayList since those methods are not synchronised and hence are faster in performance.

Use Vector only if it will be accessed by multiple threads at a time else ArrayList is always better.

Both represent growable array of objects. The difference is Vector is Synchronized and Arraylist is not.

ArrayList will grow half the size what you initialise ..where as vector will grow doble the initial size..

One Difference between Vector and ArrayList is In vector the data is retrive through elementAt() method while in ArrayList through get()

The default size of arraylist is 0 but for vector the default size is 10.
----
vector has two advantages,first they can dynamically grow and second its method are synchronized.so if more then one object want to access your vector,no inconsistancy will appear
----
In ArrayList we can store similar data types
but in Vectror we can store different types of data.
----
array list list is static where we cant change the size of array.but where as vector is dynamic whose size increases or shrink dynmically as the program data storage requirement changes
----
U should always us an ArrayList.The reason dates to JDK implementation over time. Vector is a legacy class in JDK which is still there for giving a backward compatability. The only self-expanding sequence in Java 1.0/1.1 was the Vector, so it saw a lot of use. Its flaws are too numerous to describe here. Basically, you can think of it as an ArrayList with long, awkward method names. Unfortunately, a lot of code was written using the Java 1.0/1.1 containers, and even new code is sometimes written using these classes. So although you should never use the old containers when writing new code, you'll still need to be aware of themThe Java 1.0/1.1 version of the iterator chose to invent a new name, "enumeration," instead of using a term that everyone was already familiar with. The Enumeration interface is smaller than Iterator, with only two methods, and it uses longer method names: boolean hasMoreElements( ) produces true if this enumeration contains more elements, and Object nextElement( ) returns the next element of this enumeration if there are any more (otherwise it throws an exception).Enumeration is only an interface, not an implementation, and even new libraries sometimes still use the old Enumeration, which is unfortunate but generally harmless. Even though you should always use Iterator when you can in your own code, you must be prepared for libraries that want to hand you an Enumeration.In addition, you can produce an Enumeration for any Collection by using the Collections.enumeration( ) method. (Please see for the failfast mechanism of the iterators of the Collection framework below.)As of the Java 2 platform v1.2, this class has been retrofitted to implement List, so that it becomes a part of Java's collection framework. Unlike the new collection implementations, Vector is synchronized.The Java 2 containers represent a thorough redesign of the rather poor showings in Java 1.0 and 1.1. Some of the redesign makes things tighter and more sensible. It also fills out the functionality of the containers library, providing the behavior of linked lists, queues, and deques (double-ended queues, pronounced "decks")In any container class, you must have a way to put things in and a way to get things out. After all, that's the primary job of a container—to hold things. In the ArrayList, add( ) is the way that you insert objects, and get( ) is one way to get things out. ArrayList is quite flexible; you can select anything at any time, and select multiple elements at once using different indexes.More compelleing reason for whcih we should not use the previous version classes i.e Hashtable, Vector, Stack whcih implements Vector is to do away with legacy classes in the newer inplementation. The concept of the stack was introduced earlier, with the LinkedList. What's rather odd about the Java 1.0/1.1 Stack is that instead of using a Vector as a building block, Stack is inherited from Vector. So it has all of the characteristics and behaviors of a Vector plus some extra Stack behaviors. It's difficult to know whether the designers consciously thought that this was an especially useful way of doing things, or whether it was just a naïve design; in any event it was clearly not reviewed before it was rushed into distribution, so this bad design is still hanging around (but you should never use it).Apart from that the Java containers also have a mechanism to prevent more than one process from

modifying the contents of a container. The problem occurs if you're iterating through a container, and some other process steps in and inserts, removes, or changes an object in that container. Maybe you've already passed that object, maybe it's ahead of you, maybe the size of the container shrinks after you call size( )—there are many scenarios for disaster. The Java containers library incorporates a fail-fast mechanism that looks for any changes to the container other than the ones your process is personally responsible for. If it detects that someone else is modifying the container, it immediately produces a ConcurrentModificationException. This is the "fail-fast" aspect—it doesn't try to detect a problem later on using a more complex algorithm.With a Collection, List, Set, or Map, the compiler still restricts you to calling only the methods in that interface, so it's not like Smalltalk (in which you can call any method for any object, and find out only when you run the program whether your call does anything). In addition, most methods that take a Collection as an argument only read from that Collection—all the "read" methods of Collection are not optional.This approach prevents an explosion of interfaces in the design. Other designs for container libraries always seem to end up with a confusing plethora of interfaces to describe each of the variations on the main theme, and are thus difficult to learn. It's not even possible to capture all of the special cases in interfaces, because someone can always invent a new interface. The "unsupported operation" approach achieves an important goal of the Java containers library: The containers are simple to learn and use; unsupported operations are a special case that can be learned later. For this approach to work, however: 1. The UnsupportedOperationException must be a rare event. That is, for most classes all operations should work, and only in special cases should an operation be unsupported. This is true in the Java containers library, since the classes you'll use 99 percent of the time—ArrayList, LinkedList, HashSet, and HashMap, as well as the other concrete implementations—support all of the operations. The design does provide a "back door" if you want to create a new Collection without providing meaningful definitions for all the methods in the Collection interface, and yet still fit it into the existing library. 2. When an operation is unsupported, there should be reasonable likelihood that an UnsupportedOperationException will appear at implementation time, rather than after you've shipped the product to the customer. After all, it indicates a programming error: You've used an implementation incorrectly. This point is less certain and is where the experimental nature of this design comes into play. Only over time will we find out how well it works. As far as the Synchronization is concerned 5.0 comes with java.util.concurrent package. You have many classes there which are tested and thread safe. They r built with Multi Threading built in. Its better that we should use instead of writing our own synchronization stuff.from java 1.2 we can create synchronized List usingList list = java.util.Collections.SynchronizedList(new ArrayList();But in 5.0 new package java.util.concurrent included in which it provides threadsafe Lists and sets
----

003 Question: How are memory leaks possible in Java

If any object variable is still pointing to some object which
is of no use, then JVM will not garbage collect that object
and object will remain in memory creating memory leak

Perhaps memory leaks can occur because of thread deadlock. When thread A acquires a lock on object C and thread B acquires a lock on object D. Then thread A tries to acquire a lock on object D while thread B tries to acquire a lock on object C. Both thread A and B will remain in a blocked state and I think objects C and D will not become eligible to be garbage collected as they still have live threads attached to them. This can be avoided using synchronized methods or blocks in objects C and D.

004 Question: What are the differences between EJB and Java beans
the main difference is Ejb componenets are distributed which means develop once and run anywhere.
java beans are not distributed. which means the beans cannot be shared .

JAVA BEAN COMPONENTS ARE VISUAL COMPONENTS CAN CUSTOMISED THROUGH THROUGH EVENTS AND PROPERTIES,EJBS USED FOR MULTIUSER,DISTRIBUTED,TRANSACTIONAL SERVICES AND PROCESSES NOT VISIBLE BY USER.

005 Question: What would happen if you say this = null
this will give a compilation error as follows
cannot assign value to final variable this

006 Question: Will there be a performance penalty if you make a method synchronized? If so, can you make any design changes to improve the performance

yes.the performance will be down if we use synchronization.
one can minimise the penalty by including garbage collection algorithm, which reduces the cost of collecting large numbers of short- lived objects. and also by using Improved thread synchronization for invoking the synchronized methods.the invoking will be faster.

007 Question: How would you implement a thread pool

public class ThreadPool extends java.lang.Object implements ThreadPoolInt

This class is an generic implementation of a thread pool, which takes the following input
a) Size of the pool to be constructed

b) Name of the class which implements Runnable (which has a visible default constructor)
and constructs a thread pool with active threads that are waiting for activation. once the threads have finished processing they come back and wait once again in the pool.

This thread pool engine can be locked i.e. if some internal operation is performed on the pool then it is preferable that the thread engine be locked. Locking ensures that no new threads are issued by the engine. However, the currently executing threads are allowed to continue till they come back to the passivePool

001 Question: Can wehave run() method directly without start() method in threads?
Start method is method of Thread which is implemented for create and run thread. Which will call run() where you put your implementation. If you directly call run() method it is just normal method call and not a new thread. That call will be in main thread.

002 Question: which class is the super class for all classes in java.lang package?
the object class is super or boss class of all classes in java all the classes by default extends this class and can uses the methods of these classexmp. equals,compareto etc

003 Question: What are differences between Enumeration, ArrayList, Hashtable and Collections and Collection?
enmuration is a copy of any colletion.
Arraylist is a Dynamic array. it stores elements in array format. It is a dynamic array.
Hashtable is Key value pair.
Collection is a class which stores the data temprarley.
All the above Collection are used to strore the data temprarly.

004 Question: How many JVM could be run on an operating system. if only one then what is the logical reason.
only one jvm

005 Question: What is JVM Heap Size? How does it affect the performance of the Application?
The heap is the runtime data area from which memory for all class instances and arrays is allocated. The heap may be of a fixed size or may be expanded. The heap is created on virtual machine start-up. If you have complicated algorithms or big caching which might create lot of objects in memory you may need bigger heap size.

006 Question: we know that Object class is super class of every class & a class extends only one class. so
how is it possible to a class to extend other than Object class?
Answer: Because java supports multilevel inheritance.

case 1
suppose you have a class Test, which doesn't extend any other class. So by default, Object becomes it's super class.

Object->Test

case 2
If your Test class wants to extend some other class, for example Hashtable, your class will become a sub class of Object through Hashtable because Hashtable is a subclass of Object. Any class you are trying to extend will be a subclass of Object directly or due to hierarchy.

Object->Map->Hashtable->Test

Object is the superclass of Test even when you have Test extend HahTable

So according to case 1 Test is a subclass of Object directly.

According to case 2 Test is a subclass of Object due to the hierarchy.

so no matter you extend a class or not, your class will always be a subclass of Object.

007 Question: what is difference between string and stringtokenizer?
Obviously StringTokenizer as it is suggested by its name tokenizes a String supplied to it as an argument to its constructor and the character based on which tokens of that string are to be made.The default tokenizing character is space " ".

What is a platform?
Answer:
A platform is the hardware or software environment in which a
program runs. Most
platforms can be described as a combination of the operating system and

hardware, like Windows
2000 and XP, Linux, Solaris, and MacOS.
What is the main difference between Java platform and other platforms?
Answer:
The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms. The Java platform has
two components: 1. The Java
Virtual Machine (Java VM) 2. The Java Application Programming Interface
(Java API)
What is the Java Virtual Machine?
Answer:
The Java Virtual Machine is a software that can be ported onto various
hardware-based platforms.

What is the Java API?
Answer:
The Java API is a large collection of ready-made software components
that provide many useful
capabilities, such as graphical user interface (GUI) widgets
What is the package?
Answer:
The package is a Java namespace or part of Java libraries. The Java API
is grouped into
libraries of related classes and interfaces; these libraries are known
as packages.
What is native code?
Answer:
The native code is code that after you compile it, the compiled code
runs on a specific
hardware platform.
Is Java code slower than native code? Can main() method be
overloaded?
Answer:
Yes. the main() method is a special method for a program entry. You can
overload main()
method in any ways. But if you change the signature of the main method,
the entry point for
the program will be gone.
What is the serialization?
Answer:
The serialization is a kind of mechanism that makes a class or a bean
persistence by having
its properties or fields and state information saved and restored to
and from storage.
What is a transient variable?
Answer:
A transient variable is a variable that may not be serialized. If you
don\'t want some field
to be serialized, you can mark that field transient or static.
Which containers use a border layout as their default layout?
Answer:
The Window, Frame and Dialog classes use a border layout as their
default layout.

Answer:
Not really. As a platform-independent environment, the Java platform
can be a bit slower
than native code. However, smart compilers, well-tuned interpreters,
and just-in-time bytecode
compilers can bring performance close to that of native code without
threatening portability.
What are three ways in which a thread can enter the waiting state?
Answer:
A thread can enter the waiting state by invoking its sleep() method, by
blocking on IO, by
unsuccessfully attempting to acquire an object's lock, or by invoking

an object's wait() method.
It can also enter the
waiting state by invoking its (deprecated) suspend() method.
Can a lock be acquired on a class?
Answer:
Yes, a lock can be acquired on a class. This lock is acquired on the class's Class object.
What method is used to specify a container's layout?
Answer:
The setLayout() method is used to specify a container's layout.
Which containers use a FlowLayout as their default layout?
Answer:
The Panel and Applet classes use the FlowLayout as their default layout.
Can Java object be locked down for exclusive use by a given thread?
Answer:
Yes. You can lock an object by putting it in a "synchronized" block. The locked object is
inaccessible to any thread other than the one that explicitly claimed it.
What state does a thread enter when it terminates its processing?
Answer:
When a thread terminates its processing, it enters the dead state.
What is the difference between Process and Thread?(donated in April 2005)
Answer:
A process can contain multiple threads. In most multithreading operating systems, a process
gets its own memory address space; a thread doesn't. Threads typically share the heap belonging
to their parent process. For instance, a JVM runs in a single process in the host O/S.
Threads in the JVM share the heap belonging to that process; that's why several threads
may access the same object.Typically, even though they share a common heap, threads have
their own stack space. This is how one thread's invocation of a method is kept separate
from another's.This is all a gross oversimplification, but it's accurate enough at a high level.
Lots of details differ between operating systems.Process vs. Thread A program Similar to a
sequential program vs. Can run on its own Cannot run on its own Unit of allocation Unit of
execution vs. Have its own memory space Share with others Each process has one or more
threads vs. Each thread belongs to one process Expensive, need to context switch
vs. Cheap, can use process memory and may not need to context switch More secure. One process
cannot corrupt another process Less secure.vs. A thread can write the memory used by another
thread
If a method is declared as protected, where may the method be accessed?
Answer:
A protected method may only be accessed by classes or interfaces of the same package or by
subclasses of the class in which it is declared.
What is the purpose of finalization?
Answer:
The purpose of finalization is to give an unreachable object the opportunity to perform any
cleanup processing before the object is garbage collected.
Which class is the superclass for every class.
Answer:
Object.

What is the purpose of the finally clause of a try-catch-finally
statement?
Answer:
The finally clause is used to provide the capability to execute code no
matter whether or
not an exception is thrown or caught.
What is a static method?
Answer:
A static method is a method that belongs to the class rather than any
object of the class
and doesn't apply to an object or even require that any objects of the
class have been
instantiated.
What is the difference between a Window and a Frame?
Answer:
The Frame class extends Window to define a main application window that
can have a menu bar.

What do heavy weight components mean?

Answer:

Heavy weight components like Abstract Window Toolkit (AWT), depend on
the local windowing
toolkit. For example, java.awt.Button is a heavy weight component, when
it is running on the
Java platform for Unix platform, it maps to a real Motif button. In
this relationship, the
Motif button is called the peer to the java.awt.Button. If you create
two Buttons, two peers
and hence two Motif Buttons are also created. The Java platform
communicates with the Motif
Buttons using the Java Native Interface. For each and every component
added to the application,
there is an additional overhead tied to the local windowing system,
which is why these
components are called heavy weight.

Which package has light weight components?

Answer:

javax.Swing package. All components in Swing, except JApplet, JDialog,
JFrame and JWindow are
lightweight components.

Does a class inherit the constructors of its superclass?

Answer:

A class does not inherit constructors from any of its superclasses.

Name primitive Java types.

Answer:

The primitive types are byte, char, short, int, long, float, double,
and boolean.

What restrictions are placed on method overloading?

Answer:

Two methods may not have the same name and argument list but different
return types.

What restrictions are placed on method overriding?

Answer:

Overridden methods must have the same name, argument list, and return type. The overriding
method may not limit the access of the method it overrides. The overriding method may not
throw any exceptions that may not be thrown by the overridden method.

What is casting?

Answer:

There are two types of casting, casting between primitive numeric types and casting between
object references. Casting between numeric types is used to convert larger values, such as
double values, to smaller values, such as byte values. Casting between object references is
used to refer to an object by a compatible class, interface, or array type reference.

What is the difference between a Scrollbar and a ScrollPane?

Answer:

A Scrollbar is a Component, but not a Container. A ScrollPane is a Container. A ScrollPane
handles its own events and performs its own scrolling.

What is tunnelling?

Answer:

Tunnelling is a route to somewhere. For example, RMI tunnelling is a way to make RMI
application get through firewall. In CS world, tunnelling means a way to transfer data.

Java Language
Question:

Does the code in finally block get executed if there is an exception and a return statement
in a catch block?

Answer:

If an exception occurs and there is a return statement in catch block, the finally block is
still executed. The finally block will not be executed when the System.exit(1) statement is
executed earlier or the system shut down earlier or the memory is used up earlier before the
thread goes to finally block.

What are use cases?

Answer:

A use case describes a situation that a program might encounter and what behavior the program
should exhibit in that circumstance. It is part of the analysis of a program. The collection of

use cases should, ideally, anticipate all the standard circumstances and many of the
extraordinary circumstances possible so that the program will be robust.

What is scalability and performance?

Answer:

Performance is a measure of "how fast can you perform this task." and scalability describes how
an application behaves as its workload and available computing resources increase.

Java Language
Question:

What is the benefit of subclass?

Answer:

Generally:

The sub class inherits all the public methods and the implementation.
The sub class inherits all the protected methods and their implementation.
The sub class inherits all the default(non-access modifier) methods and their implementation.
The sub class also inherits all the public, protected and default member variables from the
super class.
The constructors are not part of this inheritance model.

In System.out.println(),what is System,out and println,pls explain?

Answer:

System is a predefined final class,out is a PrintStream object acting as a field member and
println is a built-in overloaded method in the out object.

What is meant by "Abstract Interface"?

Answer:

First, an interface is abstract. That means you cannot have any implementation in an interface.
All the methods declared in an interface are abstract methods or signatures of the methods.

Why Java does not support pointers?

Answer:

Because pointers are unsafe. Java uses reference types to hide pointers and programmers feel
easier to deal with reference types without pointers. This is why Java and C-sharp shine.

Can you declare a class as private? (donated in April, 2005)

Answer:

Yes, we can declare a private class as an inner class. For example,

```
class MyPrivate {
    private static class MyKey {
        String key = "12345";
    }
    public static void main(String[] args) {
        System.out.println(new MyKey().key);//prints 12345
    }
}
```

Can Java code be compiled to machine dependent executable file?

Answer:

Yes. There are many tools out there. If you did so, the generated exe
file would be run in the
specific platform, not cross-platform.
What are the drawbacks of inheritance? (donated by RS in Mar. 2005)

Answer:

Since inheritance inherits everything from the super class and
interface, it may make the
subclass too clustering and sometimes error-prone when dynamic
overriding or dynamic overloading
in some situation. In addition, the inheritance may make peers hardly
understand your code if
they don't know how your super-class acts.

Usually, when you want to use a functionality of a class, you may use
subclass to inherit such
function or use an instance of this class in your class. Which is
better, depends on your
specification.

115 questions total, not for the weak. Covers everything from basics to JDBC connectivity, AWT and JSP.
What is the difference between procedural and object-oriented programs?- a) In procedural program, programming logic follows certain procedures and the instructions are executed one after another. In OOP program, unit of program is object, which is nothing but combination of data and code. b) In procedural program, data is exposed to the whole program whereas in OOPs program, it is accessible with in the object and which in turn assures the security of the code.
What are Encapsulation, Inheritance and Polymorphism?- Encapsulation is the mechanism that binds together code and data it manipulates and keeps both safe from outside interference and misuse. Inheritance is the process by which one object acquires the properties of another object. Polymorphism is the feature that allows one interface to be used for general class actions.
What is the difference between Assignment and Initialization?- Assignment can be done as many times as desired whereas initialization can be done only once.
What is OOPs?- Object oriented programming organizes a program around its data, i. e. , objects and a set of well defined interfaces to that data. An object-oriented program can be characterized as data controlling access to code.
What are Class, Constructor and Primitive data types?- Class is a template for multiple objects with similar features and it is a blue print for objects. It defines a type of object according to the data the object can hold and the operations the object can perform. Constructor is a special kind of method that determines how an object is initialized when created. Primitive data types are 8 types and they are: byte, short, int, long, float, double, boolean, char.
What is an Object and how do you allocate memory to it?- Object is an instance of a class and it is a software unit that combines a structured set of data with a set of operations for inspecting and manipulating that data. When an object is created using new operator, memory is allocated to it.
What is the difference between constructor and method?- Constructor will be automatically invoked when an object is created whereas method has to be called explicitly.
What are methods and how are they defined?- Methods are functions that operate on instances of classes in which they are defined. Objects can communicate with each other using methods and can call methods in other classes. Method definition has four parts. They are name of the method, type of object or primitive type the method returns, a list of parameters and the body of the method. A method's signature is a combination of the first three parts mentioned above.
What is the use of bin and lib in JDK?- Bin contains all tools such as javac, appletviewer, awt tool, etc., whereas lib contains API and all packages.
What is casting?- Casting is used to convert the value of one type to another.
How many ways can an argument be passed to a subroutine and explain them?- An argument can be passed in two ways. They are passing by value and passing by reference. Passing by value: This method copies the value of an argument into the formal parameter of the subroutine. Passing by reference: In this method, a reference to an argument (not the value of the argument) is passed to the parameter.

What is the difference between an argument and a parameter?- While defining method, variables passed in the method are called parameters. While using those methods, values passed to those variables are called arguments.
What are different types of access modifiers?- public: Any thing declared as public can be accessed from anywhere. private: Any thing declared as private can't be seen outside of its class. protected: Any thing declared as protected can be accessed by classes in the same package and subclasses in the other packages. default modifier : Can be accessed only to classes in the same package.
What is final, finalize() and finally?- final : final keyword can be used for class, method and variables. A final class cannot be subclassed and it prevents other programmers from subclassing a secure class to invoke insecure methods. A final method can't be overridden. A final variable can't change from its initialized value. finalize() : finalize() method is used just before an object is destroyed and can be called just prior to garbage collection. finally : finally, a key word used in exception handling, creates a block of code that will be executed after a try/catch block has completed and before the code following the try/catch block. The finally block will execute whether or not an exception is thrown. For example, if a method opens a file upon exit, then you will not want the code that closes the file to be bypassed by the exception-handling mechanism. This finally keyword is designed to address this contingency.
What is UNICODE?- Unicode is used for internal representation of characters and strings and it uses 16 bits to represent each other.
What is Garbage Collection and how to call it explicitly?- When an object is no longer referred to by any variable, java automatically reclaims memory used by that object. This is known as garbage collection. System. gc() method may be used to call it explicitly.
What is finalize() method?- finalize () method is used just before an object is destroyed and can be called just prior to garbage collection.
What are Transient and Volatile Modifiers?- Transient: The transient modifier applies to variables only and it is not stored as part of its object's Persistent state. Transient variables are not serialized. Volatile: Volatile modifier applies to variables only and it tells the compiler that the variable modified by volatile can be changed unexpectedly by other parts of the program.
What is method overloading and method overriding?- Method overloading: When a method in a class having the same method name with different arguments is said to be method overloading. Method overriding : When a method in a class having the same method name with same arguments is said to be method overriding.
What is difference between overloading and overriding?- a) In overloading, there is a relationship between methods available in the same class whereas in overriding, there is relationship between a superclass method and subclass method. b) Overloading does not block inheritance from the superclass whereas overriding blocks inheritance from the superclass. c) In overloading, separate methods share the same name whereas in overriding, subclass method replaces the superclass. d) Overloading must have different method signatures whereas overriding must have same signature.
What is meant by Inheritance and what are its advantages?- Inheritance is the process of inheriting all the features from a class. The advantages of inheritance are reusability of code and accessibility of variables and methods of the super class by subclasses.
What is the difference between this() and super()?- this() can be used to invoke a constructor of the same class whereas super() can be used to invoke a super class constructor.
What is the difference between superclass and subclass?- A super class is a class that is inherited whereas sub class is a class that does the inheriting.
What modifiers may be used with top-level class?- public, abstract and final can be used for top-level class.
What are inner class and anonymous class?- Inner class : classes defined in other classes, including those defined in methods are called inner classes. An inner class can have any accessibility including private. Anonymous class : Anonymous class is a class defined inside a method without a name and is instantiated and declared in the same place and cannot have explicit constructors.
What is a package?- A package is a collection of classes and interfaces that provides a high-level layer of access protection and name space management.
What is a reflection package?- java. lang. reflect package has the ability to analyze itself in runtime.
What is interface and its use?- Interface is similar to a class which may contain method's signature only but not bodies and it is a formal set of method and constant declarations that must be defined by the class that implements it. Interfaces are useful for: a)Declaring methods that one or more classes are expected to implement b)Capturing similarities between unrelated classes without forcing a class relationship. c)Determining an object's programming interface without revealing the actual body of the class.
What is an abstract class?- An abstract class is a class designed with implementation gaps for subclasses to fill in and is deliberately incomplete.
What is the difference between Integer and int?- a) Integer is a class defined in the java. lang package, whereas int is a primitive data type defined in the Java language itself. Java does not automatically convert from one to the other. b) Integer can be used as an argument for a method that requires an object, whereas int can be used for calculations.
What is a cloneable interface and how many methods does it contain?- It is not having any method because it is a TAGGED or MARKER interface.
What is the difference between abstract class and interface?- a) All the methods declared inside an interface are abstract whereas abstract class must have at least one abstract method and others may be concrete or abstract. b) In abstract class, key word abstract must be used for the methods whereas interface we need not use that keyword for the methods. c) Abstract class must have subclasses whereas interface can't have subclasses.
Can you have an inner class inside a method and what variables can you access?- Yes, we can have an inner class inside a method and final variables can be accessed.
What is the difference between String and String Buffer?- a) String objects are constants and immutable whereas StringBuffer objects are not. b) String class supports constant strings whereas StringBuffer class supports growable and modifiable strings.
What is the difference between Array and vector?- Array is a set of related data type and static whereas vector is a growable array of objects and dynamic.

What is the difference between exception and error?- The exception class defines mild error conditions that your program encounters. Exceptions can occur when trying to open the file, which does not exist, the network connection is disrupted, operands being manipulated are out of prescribed ranges, the class file you are interested in loading is missing. The error class defines serious error conditions that you should not attempt to recover from. In most cases it is advisable to let the program terminate when such an error is encountered.
What is the difference between process and thread?- Process is a program in execution whereas thread is a separate path of execution in a program.
What is multithreading and what are the methods for inter-thread communication and what is the class in which these methods are defined?- Multithreading is the mechanism in which more than one thread run independent of each other within the process. wait (), notify () and notifyAll() methods can be used for inter-thread communication and these methods are in Object class. wait() : When a thread executes a call to wait() method, it surrenders the object lock and enters into a waiting state. notify() or notifyAll() : To remove a thread from the waiting state, some other thread must make a call to notify() or notifyAll() method on the same object.
What is the class and interface in java to create thread and which is the most advantageous method?- Thread class and Runnable interface can be used to create threads and using Runnable interface is the most advantageous method to create threads because we need not extend thread class here.
What are the states associated in the thread?- Thread contains ready, running, waiting and dead states.
What is synchronization?- Synchronization is the mechanism that ensures that only one thread is accessed the resources at a time.
When you will synchronize a piece of your code?- When you expect your code will be accessed by different threads and these threads may change a particular data causing data corruption.
What is deadlock?- When two threads are waiting each other and can't precede the program is said to be deadlock.
What is daemon thread and which method is used to create the daemon thread?- Daemon thread is a low priority thread which runs intermittently in the back ground doing the garbage collection operation for the java runtime system. setDaemon method is used to create a daemon thread.
Are there any global variables in Java, which can be accessed by other part of your program?- No, it is not the main method in which you define variables. Global variables is not possible because concept of encapsulation is eliminated here.
What is an applet?- Applet is a dynamic and interactive program that runs inside a web page displayed by a java capable browser.
What is the difference between applications and applets?- a)Application must be run on local machine whereas applet needs no explicit installation on local machine. b)Application must be run explicitly within a java-compatible virtual machine whereas applet loads and runs itself automatically in a java-enabled browser. d)Application starts execution with its main method whereas applet starts execution with its init method. e)Application can run with or without graphical user interface whereas applet must run within a graphical user interface.
How does applet recognize the height and width?- Using getParameters() method.
When do you use codebase in applet?- When the applet class file is not in the same directory, codebase is used.
What is the lifecycle of an applet?- init() method - Can be called when an applet is first loaded start() method - Can be called each time an applet is started. paint() method - Can be called when the applet is minimized or maximized. stop() method - Can be used when the browser moves off the applet's page. destroy() method - Can be called when the browser is finished with the applet.
How do you set security in applets?- using setSecurityManager() method
What is an event and what are the models available for event handling?- An event is an event object that describes a state of change in a source. In other words, event occurs when an action is generated, like pressing button, clicking mouse, selecting a list, etc. There are two types of models for handling events and they are: a) event-inheritance model and b) event-delegation model
What are the advantages of the model over the event-inheritance model?- The event-delegation model has two advantages over the event-inheritance model. They are: a)It enables event handling by objects other than the ones that generate the events. This allows a clean separation between a component's design and its use. b)It performs much better in applications where many events are generated. This performance improvement is due to the fact that the event-delegation model does not have to be repeatedly process unhandled events as is the case of the event-inheritance.
What is source and listener?- source : A source is an object that generates an event. This occurs when the internal state of that object changes in some way. listener : A listener is an object that is notified when an event occurs. It has two major requirements. First, it must have been registered with one or more sources to receive notifications about specific types of events. Second, it must implement methods to receive and process these notifications.
What is adapter class?- An adapter class provides an empty implementation of all methods in an event listener interface. Adapter classes are useful when you want to receive and process only some of the events that are handled by a particular event listener interface. You can define a new class to act listener by extending one of the adapter classes and implementing only those events in which you are interested. For example, the MouseMotionAdapter class has two methods, mouseDragged()and mouseMoved(). The signatures of these empty are exactly as defined in the MouseMotionListener interface. If you are interested in only mouse drag events, then you could simply extend MouseMotionAdapter and implement mouseDragged() .
What is meant by controls and what are different types of controls in AWT?- Controls are components that allow a user to interact with your application and the AWT supports the following types of controls: Labels, Push Buttons, Check Boxes, Choice Lists, Lists, Scrollbars, Text Components. These controls are subclasses of Component.
What is the difference between choice and list?- A Choice is displayed in a compact form that requires you to pull it down to see the list of available choices and only one item may be selected from a choice. A List may be displayed in such a way that several list items are visible and it supports the selection of one or more list items.
What is the difference between scrollbar and scrollpane?- A Scrollbar is a Component, but not a Container whereas Scrollpane is a Conatiner and handles its own events and perform its own scrolling.

What is a layout manager and what are different types of layout managers available in java AWT?- A layout manager is an object that is used to organize components in a container. The different layouts are available are FlowLayout, BorderLayout, CardLayout, GridLayout and GridBagLayout.
How are the elements of different layouts organized?- FlowLayout: The elements of a FlowLayout are organized in a top to bottom, left to right fashion. BorderLayout: The elements of a BorderLayout are organized at the borders (North, South, East and West) and the center of a container. CardLayout: The elements of a CardLayout are stacked, on top of the other, like a deck of cards. GridLayout: The elements of a GridLayout are of equal size and are laid out using the square of a grid. GridBagLayout: The elements of a GridBagLayout are organized according to a grid. However, the elements are of different size and may occupy more than one row or column of the grid. In addition, the rows and columns may have different sizes.
Which containers use a Border layout as their default layout?- Window, Frame and Dialog classes use a BorderLayout as their layout.
Which containers use a Flow layout as their default layout?- Panel and Applet classes use the FlowLayout as their default layout.
What are wrapper classes?- Wrapper classes are classes that allow primitive types to be accessed as objects.
What are Vector, Hashtable, LinkedList and Enumeration?- Vector : The Vector class provides the capability to implement a growable array of objects. Hashtable : The Hashtable class implements a Hashtable data structure. A Hashtable indexes and stores objects in a dictionary using hash codes as the object’s keys. Hash codes are integer values that identify objects. LinkedList: Removing or inserting elements in the middle of an array can be done using LinkedList. A LinkedList stores each object in a separate link whereas an array stores object references in consecutive locations. Enumeration: An object that implements the Enumeration interface generates a series of elements, one at a time. It has two methods, namely hasMoreElements() and nextElement(). HasMoreElemnts() tests if this enumeration has more elements and nextElement method returns successive elements of the series.
What is the difference between set and list?- Set stores elements in an unordered way but does not contain duplicate elements, whereas list stores elements in an ordered way but may contain duplicate elements.
What is a stream and what are the types of Streams and classes of the Streams?- A Stream is an abstraction that either produces or consumes information. There are two types of Streams and they are: Byte Streams: Provide a convenient means for handling input and output of bytes. Character Streams: Provide a convenient means for handling input & output of characters. Byte Streams classes: Are defined by using two abstract classes, namely InputStream and OutputStream. Character Streams classes: Are defined by using two abstract classes, namely Reader and Writer.
What is the difference between Reader/Writer and InputStream/Output Stream?- The Reader/Writer class is character-oriented and the InputStream/OutputStream class is byte-oriented.
What is an I/O filter?- An I/O filter is an object that reads from one stream and writes to another, usually altering the data in some way as it is passed from one stream to another.
What is serialization and deserialization?- Serialization is the process of writing the state of an object to a byte stream. Deserialization is the process of restoring these objects.
What is JDBC?- JDBC is a set of Java API for executing SQL statements. This API consists of a set of classes and interfaces to enable programs to write pure Java Database applications.
What are drivers available?- a) JDBC-ODBC Bridge driver b) Native API Partly-Java driver c) JDBC-Net Pure Java driver d) Native-Protocol Pure Java driver
What is the difference between JDBC and ODBC?- a) OBDC is for Microsoft and JDBC is for Java applications. b) ODBC can’t be directly used with Java because it uses a C interface. c) ODBC makes use of pointers which have been removed totally from Java. d) ODBC mixes simple and advanced features together and has complex options for simple queries. But JDBC is designed to keep things simple while allowing advanced capabilities when required. e) ODBC requires manual installation of the ODBC driver manager and driver on all client machines. JDBC drivers are written in Java and JDBC code is automatically installable, secure, and portable on all platforms. f) JDBC API is a natural Java interface and is built on ODBC. JDBC retains some of the basic features of ODBC.
What are the types of JDBC Driver Models and explain them?- There are two types of JDBC Driver Models and they are: a) Two tier model and b) Three tier model Two tier model: In this model, Java applications interact directly with the database. A JDBC driver is required to communicate with the particular database management system that is being accessed. SQL statements are sent to the database and the results are given to user. This model is referred to as client/server configuration where user is the client and the machine that has the database is called as the server. Three tier model: A middle tier is introduced in this model. The functions of this model are: a) Collection of SQL statements from the client and handing it over to the database, b) Receiving results from database to the client and c) Maintaining control over accessing and updating of the above.
What are the steps involved for making a connection with a database or how do you connect to a database?a) Loading the driver : To load the driver, Class. forName() method is used. Class. forName(”sun. jdbc. odbc. JdbcOdbcDriver”); When the driver is loaded, it registers itself with the java. sql. DriverManager class as an available database driver. b) Making a connection with database: To open a connection to a given database, DriverManager. getConnection() method is used. Connection con = DriverManager. getConnection (”jdbc:odbc:somedb”, “user”, “password”); c) Executing SQL statements : To execute a SQL query, java. sql. statements class is used. createStatement() method of Connection to obtain a new Statement object. Statement stmt = con. createStatement(); A query that returns data can be executed using the executeQuery() method of Statement. This method executes the statement and returns a java. sql. ResultSet that encapsulates the retrieved data: ResultSet rs = stmt. executeQuery(”SELECT * FROM some table”); d) Process the results : ResultSet returns one row at a time. Next() method of ResultSet object can be called to move to the next row. The getString() and getObject() methods are used for retrieving column values: while(rs. next()) { String event = rs. getString(”event”); Object count = (Integer) rs. getObject(”count”);
What type of driver did you use in project?- JDBC-ODBC Bridge driver (is a driver that uses native(C language) libraries and makes calls to an existing ODBC driver to access a database engine).

What are the types of statements in JDBC?- Statement: to be used createStatement() method for executing single SQL statement PreparedStatement — To be used preparedStatement() method for executing same SQL statement over and over. CallableStatement — To be used prepareCall() method for multiple SQL statements over and over.
What is stored procedure?- Stored procedure is a group of SQL statements that forms a logical unit and performs a particular task. Stored Procedures are used to encapsulate a set of operations or queries to execute on database. Stored procedures can be compiled and executed with different parameters and results and may have any combination of input/output parameters.
How to create and call stored procedures?- To create stored procedures: Create procedure procedurename (specify in, out and in out parameters) BEGIN Any multiple SQL statement; END; To call stored procedures: CallableStatement csmt = con. prepareCall("{call procedure name(?,?)}"); csmt. registerOutParameter(column no. , data type); csmt. setInt(column no. , column name) csmt. execute();
What is servlet?- Servlets are modules that extend request/response-oriented servers, such as java-enabled web servers. For example, a servlet might be responsible for taking data in an HTML order-entry form and applying the business logic used to update a company's order database.
What are the classes and interfaces for servlets?- There are two packages in servlets and they are javax. servlet and
What is the difference between an applet and a servlet?- a) Servlets are to servers what applets are to browsers. b) Applets must have graphical user interfaces whereas servlets have no graphical user interfaces.
What is the difference between doPost and doGet methods?- a) doGet() method is used to get information, while doPost() method is used for posting information. b) doGet() requests can't send large amount of information and is limited to 240-255 characters. However, doPost()requests passes all of its data, of unlimited length. c) A doGet() request is appended to the request URL in a query string and this allows the exchange is visible to the client, whereas a doPost() request passes directly over the socket connection as part of its HTTP request body and the exchange are invisible to the client.
What is the life cycle of a servlet?- Each Servlet has the same life cycle: a) A server loads and initializes the servlet by init () method. b) The servlet handles zero or more client's requests through service() method. c) The server removes the servlet through destroy() method.
Who is loading the init() method of servlet?- Web server
What are the different servers available for developing and deploying Servlets?- a) Java Web Server b) JRun g) Apache Server h) Netscape Information Server i) Web Logic
How many ways can we track client and what are they?- The servlet API provides two ways to track client state and they are: a) Using Session tracking and b) Using Cookies.
What is session tracking and how do you track a user session in servlets?- Session tracking is a mechanism that servlets use to maintain state about a series requests from the same user across some period of time. The methods used for session tracking are: a) User Authentication - occurs when a web server restricts access to some of its resources to only those clients that log in using a recognized username and password. b) Hidden form fields - fields are added to an HTML form that are not displayed in the client's browser. When the form containing the fields is submitted, the fields are sent back to the server. c) URL rewriting - every URL that the user clicks on is dynamically modified or rewritten to include extra information. The extra information can be in the form of extra path information, added parameters or some custom, server-specific URL change. d) Cookies - a bit of information that is sent by a web server to a browser and which can later be read back from that browser. e) HttpSession- places a limit on the number of sessions that can exist in memory. This limit is set in the session. maxresidents property.
What is Server-Side Includes (SSI)?- Server-Side Includes allows embedding servlets within HTML pages using a special servlet tag. In many servlets that support servlets, a page can be processed by the server to include output from servlets at certain points inside the HTML page. This is accomplished using a special internal SSINCLUDE, which processes the servlet tags. SSINCLUDE servlet will be invoked whenever a file with an. shtml extension is requested. So HTML files that include server-side includes must be stored with an . shtml extension.
What are cookies and how will you use them?- Cookies are a mechanism that a servlet uses to have a client hold a small amount of state-information associated with the user. a) Create a cookie with the Cookie constructor: public Cookie(String name, String value) b) A servlet can send a cookie to the client by passing a Cookie object to the addCookie() method of HttpServletResponse: public void HttpServletResponse. addCookie(Cookie cookie) c) A servlet retrieves cookies by calling the getCookies() method of HttpServletRequest: public Cookie[ ] HttpServletRequest. getCookie().
Is it possible to communicate from an applet to servlet and how many ways and how?- Yes, there are three ways to communicate from an applet to servlet and they are: a) HTTP Communication(Text-based and object-based) b) Socket Communication c) RMI Communication
What is connection pooling?- With servlets, opening a database connection is a major bottleneck because we are creating and tearing down a new connection for every page request and the time taken to create connection will be more. Creating a connection pool is an ideal approach for a complicated servlet. With a connection pool, we can duplicate only the resources we need to duplicate rather than the entire servlet. A connection pool can also intelligently manage the size of the pool and make sure each connection remains valid. A number of connection pool packages are currently available. Some like DbConnectionBroker are freely available from Java Exchange Works by creating an object that dispenses connections and connection Ids on request. The ConnectionPool class maintains a Hastable, using Connection objects as keys and Boolean values as stored values. The Boolean value indicates whether a connection is in use or not. A program calls getConnection() method of the ConnectionPool for getting Connection object it can use; it calls returnConnection() to give the connection back to the pool.
Why should we go for interservlet communication?- Servlets running together in the same server communicate with each other in several ways. The three major reasons to use interservlet communication are: a) Direct servlet manipulation - allows to gain access to the other currently loaded servlets and perform certain tasks (through the ServletContext object) b) Servlet reuse - allows the servlet to reuse the public methods of another servlet. c) Servlet collaboration - requires to communicate with each other by sharing specific information (through method invocation)

Is it possible to call servlet with parameters in the URL?- Yes. You can call a servlet with parameters in the syntax as (?Param1 = xxx || m2 = yyy).
What is Servlet chaining?- Servlet chaining is a technique in which two or more servlets can cooperate in servicing a single request. In servlet chaining, one servlet's output is piped to the next servlet's input. This process continues until the last servlet is reached. Its output is then sent back to the client.
How do servlets handle multiple simultaneous requests?- The server has multiple threads that are available to handle requests. When a request comes in, it is assigned to a thread, which calls a service method (for example: doGet(), doPost() and service()) of the servlet. For this reason, a single servlet object can have its service methods called by many threads at once.
What is the difference between TCP/IP and UDP?- TCP/IP is a two-way communication between the client and the server and it is a reliable and there is a confirmation regarding reaching the message to the destination. It is like a phone call. UDP is a one-way communication only between the client and the server and it is not a reliable and there is no confirmation regarding reaching the message to the destination. It is like a postal mail.
What is Inet address?- Every computer connected to a network has an IP address. An IP address is a number that uniquely identifies each computer on the Net. An IP address is a 32-bit number.
What is Domain Naming Service(DNS)?- It is very difficult to remember a set of numbers(IP address) to connect to the Internet. The Domain Naming Service(DNS) is used to overcome this problem. It maps one particular IP address to a string of characters. For example, www. mascom. com implies com is the domain name reserved for US commercial sites, moscom is the name of the company and www is the name of the specific computer, which is mascom's server.
What is URL?- URL stands for Uniform Resource Locator and it points to resource files on the Internet. URL has four components: http://www. address. com:80/index.html, where http - protocol name, address - IP address or host name, 80 - port number and index.html - file path.
What is RMI and steps involved in developing an RMI object?- Remote Method Invocation (RMI) allows java object that executes on one machine and to invoke the method of a Java object to execute on another machine. The steps involved in developing an RMI object are: a) Define the interfaces b) Implementing these interfaces c) Compile the interfaces and their implementations with the java compiler d) Compile the server implementation with RMI compiler e) Run the RMI registry f) Run the application
What is RMI architecture?- RMI architecture consists of four layers and each layer performs specific functions: a) Application layer - contains the actual object definition. b) Proxy layer - consists of stub and skeleton. c) Remote Reference layer - gets the stream of bytes from the transport layer and sends it to the proxy layer. d) Transportation layer - responsible for handling the actual machine-to-machine communication.
what is UnicastRemoteObject?- All remote objects must extend UnicastRemoteObject, which provides functionality that is needed to make objects available from remote machines.
Explain the methods, rebind() and lookup() in Naming class?- rebind() of the Naming class(found in java. rmi) is used to update the RMI registry on the server machine. Naming. rebind("AddSever", AddServerImpl); lookup() of the Naming class accepts one argument, the rmi URL and returns a reference to an object of type AddServerImpl.
What is a Java Bean?- A Java Bean is a software component that has been designed to be reusable in a variety of different environments.
What is a Jar file?- Jar file allows to efficiently deploying a set of classes and their associated resources. The elements in a jar file are compressed, which makes downloading a Jar file much faster than separately downloading several uncompressed files. The package java. util. zip contains classes that read and write jar files.
What is BDK?- BDK, Bean Development Kit is a tool that enables to create, configure and connect a set of set of Beans and it can be used to test Beans without writing a code.
What is JSP?- JSP is a dynamic scripting capability for web pages that allows Java as well as a few special tags to be embedded into a web file (HTML/XML, etc). The suffix traditionally ends with .jsp to indicate to the web server that the file is a JSP files. JSP is a server side technology - you can't do any client side validation with it. The advantages are: a) The JSP assists in making the HTML more functional. Servlets on the other hand allow outputting of HTML but it is a tedious process. b) It is easy to make a change and then let the JSP capability of the web server you are using deal with compiling it into a servlet and running it.
What are JSP scripting elements?- JSP scripting elements lets to insert Java code into the servlet that will be generated from the current JSP page. There are three forms: a) Expressions of the form <%= expression %> that are evaluated and inserted into the output, b) Scriptlets of the formthat are inserted into the servlet's service method, and c) Declarations of the form <%! Code %>that are inserted into the body of the servlet class, outside of any existing methods.
What are JSP Directives?- A JSP directive affects the overall structure of the servlet class. It usually has the following form:<%@ directive attribute="value" %> However, you can also combine multiple attribute settings for a single directive, as follows:<%@ directive attribute1="value1″ attribute 2="value2″ . . . attributeN ="valueN" %> There are two main types of directive: page, which lets to do things like import classes, customize the servlet superclass, and the like; and include, which lets to insert a file into the servlet class at the time the JSP file is translated into a servlet
What are Predefined variables or implicit objects?- To simplify code in JSP expressions and scriptlets, we can use eight automatically defined variables, sometimes called implicit objects. They are request, response, out, session, application, config, pageContext, and page.
What are JSP ACTIONS?- JSP actions use constructs in XML syntax to control the behavior of the servlet engine. You can dynamically insert a file, reuse JavaBeans components, forward the user to another page, or generate HTML for the Java plugin. Available actions include: jsp:include - Include a file at the time the page is requested. jsp:useBean - Find or instantiate a JavaBean. jsp:setProperty - Set the property of a JavaBean. jsp:getProperty - Insert the property of a JavaBean into the output. jsp:forward - Forward the requester to a newpage. Jsp: plugin - Generate browser-specific code that makes an OBJECT or EMBED

How do you pass data (including JavaBeans) to a JSP from a servlet?- (1) Request Lifetime: Using this technique to pass beans, a request dispatcher (using either "include" or forward") can be called. This bean will disappear after processing this request has been completed. Servlet: request. setAttribute(”theBean”, myBean); RequestDispatcher rd = getServletContext(). getRequestDispatcher(”thepage. jsp”); rd. forward(request, response); JSP PAGE:<jsp: useBean id=”theBean” scope=”request” class=”. . . . . ” />(2) Session Lifetime: Using this technique to pass beans that are relevant to a particular session (such as in individual user login) over a number of requests. This bean will disappear when the session is invalidated or it times out, or when you remove it. Servlet: HttpSession session = request. getSession(true); session. putValue(”theBean”, myBean); /* You can do a request dispatcher here, or just let the bean be visible on the next request */ JSP Page:<jsp:useBean id=”theBean” scope=”session” class=”. . . ” /> 3) Application Lifetime: Using this technique to pass beans that are relevant to all servlets and JSP pages in a particular app, for all users. For example, I use this to make a JDBC connection pool object available to the various servlets and JSP pages in my apps. This bean will disappear when the servlet engine is shut down, or when you remove it. Servlet: GetServletContext(). setAttribute(”theBean”, myBean); JSP PAGE:<jsp:useBean id=”theBean” scope=”application” class=”. . . ” />
How can I set a cookie in JSP?- response. setHeader(”Set-Cookie”, “cookie string”); To give the response-object to a bean, write a method setResponse (HttpServletResponse response) - to the bean, and in jsp-file:<% bean. setResponse (response); %>
How can I delete a cookie with JSP?- Say that I have a cookie called “foo, ” that I set a while ago & I want it to go away. I simply: <% Cookie killCookie = new Cookie(”foo”, null); KillCookie. setPath(”/”); killCookie. setMaxAge(0); response. addCookie(killCookie); %>
How are Servlets and JSP Pages related?- JSP pages are focused around HTML (or XML) with Java codes and JSP tags inside them. When a web server that has JSP support is asked for a JSP page, it checks to see if it has already compiled the page into a servlet. Thus, JSP pages become servlets and are transformed into pure Java and then compiled, loaded into the server and executed.

What is the difference between procedural and object-oriented programs?-
a) In procedural program, programming logic follows certain procedures and the instructions are executed one after another. In OOP program, unit of program is object, which is nothing but combination of data and code.
b) In procedural program, data is exposed to the whole program whereas in OOPs program, it is accessible with in the object and which in turn assures the security of the code.
What are Encapsulation, Inheritance and Polymorphism?- Encapsulation is the mechanism that binds together code and data it manipulates and keeps both safe from outside interference and misuse. Inheritance is the process by which one object acquires the properties of another object. Polymorphism is the feature that allows one interface to be used for general class actions.
What is the difference between Assignment and Initialization?- Assignment can be done as many times as desired whereas initialization can be done only once.
What is OOPs?- Object oriented programming organizes a program around its data, i. e. , objects and a set of well defined interfaces to that data. An object-oriented program can be characterized as data controlling access to code.
What are Class, Constructor and Primitive data types?- Class is a template for multiple objects with similar features and it is a blue print for objects. It defines a type of object according to the data the object can hold and the operations the object can perform. Constructor is a special kind of method that determines how an object is initialized when created. Primitive data types are 8 types and they are: byte, short, int, long, float, double, boolean, char.
What is an Object and how do you allocate memory to it?- Object is an instance of a class and it is a software unit that combines a structured set of data with a set of operations for inspecting and manipulating that data. When an object is created using new operator, memory is allocated to it.
What is the difference between constructor and method?- Constructor will be automatically invoked when an object is created whereas method has to be called explicitly.
What are methods and how are they defined?- Methods are functions that operate on instances of classes in which they are defined. Objects can communicate with each other using methods and can call methods in other classes. Method definition has four parts. They are name of the method, type of object or primitive type the method returns, a list of parameters and the body of the method. A method’s signature is a combination of the first three parts mentioned above.
What is the use of bin and lib in JDK?- Bin contains all tools such as javac, appletviewer, awt tool, etc., whereas lib contains API and all packages.
What is casting?- Casting is used to convert the value of one type to another.
How many ways can an argument be passed to a subroutine and explain them?- An argument can be passed in two ways. They are passing by value and passing by reference. Passing by value: This method copies the value of an argument into the formal parameter of the subroutine. Passing by reference: In this method, a reference to an argument (not the value of the argument) is passed to the parameter.
What is the difference between an argument and a parameter?- While defining method, variables passed in the method are called parameters. While using those methods, values passed to those variables are called arguments.
What are different types of access modifiers?- public: Any thing declared as public can be accessed from anywhere. private: Any thing declared as private can’t be seen outside of its class. protected: Any thing declared as protected can be accessed by classes in the same package and subclasses in the other packages. default modifier : Can be accessed only to classes in the same package.
What is final, finalize() and finally?- final : final keyword can be used for class, method and variables. A final class cannot be subclassed and it prevents other programmers from subclassing a secure class to invoke insecure methods. A final method can’t be overridden. A final variable can’t change from its initialized value. finalize() : finalize() method is used just before an object is destroyed and can be called just prior to garbage collection. finally : finally, a key word used in exception handling, creates a block of code that will be executed after a try/catch block has completed and before the code following the try/catch block. The

finally block will execute whether or not an exception is thrown. For example, if a method opens a file upon exit, then you will not want the code that closes the file to be bypassed by the exception-handling mechanism. This finally keyword is designed to address this contingency.
What is UNICODE?- Unicode is used for internal representation of characters and strings and it uses 16 bits to represent each other.
What is Garbage Collection and how to call it explicitly?- When an object is no longer referred to by any variable, java automatically reclaims memory used by that object. This is known as garbage collection. System. gc() method may be used to call it explicitly.
What is finalize() method?- finalize () method is used just before an object is destroyed and can be called just prior to garbage collection.
What are Transient and Volatile Modifiers?- Transient: The transient modifier applies to variables only and it is not stored as part of its object's Persistent state. Transient variables are not serialized. Volatile: Volatile modifier applies to variables only and it tells the compiler that the variable modified by volatile can be changed unexpectedly by other parts of the program.
What is method overloading and method overriding?- Method overloading: When a method in a class having the same method name with different arguments is said to be method overloading. Method overriding : When a method in a class having the same method name with same arguments is said to be method overriding.
What is difference between overloading and overriding?- a) In overloading, there is a relationship between methods available in the same class whereas in overriding, there is relationship between a superclass method and subclass method. b) Overloading does not block inheritance from the superclass whereas overriding blocks inheritance from the superclass. c) In overloading, separate methods share the same name whereas in overriding, subclass method replaces the superclass. d) Overloading must have different method signatures whereas overriding must have same signature.
What is meant by Inheritance and what are its advantages?- Inheritance is the process of inheriting all the features from a class. The advantages of inheritance are reusability of code and accessibility of variables and methods of the super class by subclasses.
What is the difference between this() and super()?- this() can be used to invoke a constructor of the same class whereas super() can be used to invoke a super class constructor.
What is the difference between superclass and subclass?- A super class is a class that is inherited whereas sub class is a class that does the inheriting.
What modifiers may be used with top-level class?- public, abstract and final can be used for top-level class.
What are inner class and anonymous class?- Inner class : classes defined in other classes, including those defined in methods are called inner classes. An inner class can have any accessibility including private. Anonymous class : Anonymous class is a class defined inside a method without a name and is instantiated and declared in the same place and cannot have explicit constructors.
What is a package?- A package is a collection of classes and interfaces that provides a high-level layer of access protection and name space management.
What is a reflection package?- java. lang. reflect package has the ability to analyze itself in runtime.
What is interface and its use?- Interface is similar to a class which may contain method's signature only but not bodies and it is a formal set of method and constant declarations that must be defined by the class that implements it. Interfaces are useful for: a)Declaring methods that one or more classes are expected to implement b)Capturing similarities between unrelated classes without forcing a class relationship. c)Determining an object's programming interface without revealing the actual body of the class.
What is an abstract class?- An abstract class is a class designed with implementation gaps for subclasses to fill in and is deliberately incomplete.
What is the difference between Integer and int?- a) Integer is a class defined in the java. lang package, whereas int is a primitive data type defined in the Java language itself. Java does not automatically convert from one to the other. b) Integer can be used as an argument for a method that requires an object, whereas int can be used for calculations.
What is a cloneable interface and how many methods does it contain?- It is not having any method because it is a TAGGED or MARKER interface.
What is the difference between abstract class and interface?- a) All the methods declared inside an interface are abstract whereas abstract class must have at least one abstract method and others may be concrete or abstract. b) In abstract class, key word abstract must be used for the methods whereas interface we need not use that keyword for the methods. c) Abstract class must have subclasses whereas interface can't have subclasses.
Can you have an inner class inside a method and what variables can you access?- Yes, we can have an inner class inside a method and final variables can be accessed.
What is the difference between String and String Buffer?- a) String objects are constants and immutable whereas StringBuffer objects are not. b) String class supports constant strings whereas StringBuffer class supports growable and modifiable strings.
What is the difference between Array and vector?- Array is a set of related data type and static whereas vector is a growable array of objects and dynamic.
What is the difference between exception and error?- The exception class defines mild error conditions that your program encounters. Exceptions can occur when trying to open the file, which does not exist, the network connection is disrupted, operands being manipulated are out of prescribed ranges, the class file you are interested in loading is missing. The error class defines serious error conditions that you should not attempt to recover from. In most cases it is advisable to let the program terminate when such an error is encountered.
What is the difference between process and thread?- Process is a program in execution whereas thread is a separate path of execution in a program.
What is multithreading and what are the methods for inter-thread communication and what is the class in which these methods are defined?- Multithreading is the mechanism in which more than one thread run independent of each other within the process. wait (), notify () and notifyAll() methods can be used for inter-thread communication and these methods are in Object class. wait() :

When a thread executes a call to wait() method, it surrenders the object lock and enters into a waiting state. notify() or notifyAll() : To remove a thread from the waiting state, some other thread must make a call to notify() or notifyAll() method on the same object.
What is the class and interface in java to create thread and which is the most advantageous method?- Thread class and Runnable interface can be used to create threads and using Runnable interface is the most advantageous method to create threads because we need not extend thread class here.
What are the states associated in the thread?- Thread contains ready, running, waiting and dead states.
What is synchronization?- Synchronization is the mechanism that ensures that only one thread is accessed the resources at a time.
When you will synchronize a piece of your code?- When you expect your code will be accessed by different threads and these threads may change a particular data causing data corruption.
What is deadlock?- When two threads are waiting each other and can’t precede the program is said to be deadlock.
What is daemon thread and which method is used to create the daemon thread?- Daemon thread is a low priority thread which runs intermittently in the back ground doing the garbage collection operation for the java runtime system. setDaemon method is used to create a daemon thread.
Are there any global variables in Java, which can be accessed by other part of your program?- No, it is not the main method in which you define variables. Global variables is not possible because concept of encapsulation is eliminated here.
What is an applet?- Applet is a dynamic and interactive program that runs inside a web page displayed by a java capable browser.
What is the difference between applications and applets?- a)Application must be run on local machine whereas applet needs no explicit installation on local machine. b)Application must be run explicitly within a java-compatible virtual machine whereas applet loads and runs itself automatically in a java-enabled browser. d)Application starts execution with its main method whereas applet starts execution with its init method. e)Application can run with or without graphical user interface whereas applet must run within a graphical user interface.
How does applet recognize the height and width?- Using getParameters() method.
When do you use codebase in applet?- When the applet class file is not in the same directory, codebase is used.
What is the lifecycle of an applet?- init() method - Can be called when an applet is first loaded start() method - Can be called each time an applet is started. paint() method - Can be called when the applet is minimized or maximized. stop() method - Can be used when the browser moves off the applet’s page. destroy() method - Can be called when the browser is finished with the applet.
How do you set security in applets?- using setSecurityManager() method
What is an event and what are the models available for event handling?- An event is an event object that describes a state of change in a source. In other words, event occurs when an action is generated, like pressing button, clicking mouse, selecting a list, etc. There are two types of models for handling events and they are: a) event-inheritance model and b) event-delegation model
What are the advantages of the model over the event-inheritance model?- The event-delegation model has two advantages over the event-inheritance model. They are: a)It enables event handling by objects other than the ones that generate the events. This allows a clean separation between a component’s design and its use. b)It performs much better in applications where many events are generated. This performance improvement is due to the fact that the event-delegation model does not have to be repeatedly process unhandled events as is the case of the event-inheritance.
What is source and listener?- source : A source is an object that generates an event. This occurs when the internal state of that object changes in some way. listener : A listener is an object that is notified when an event occurs. It has two major requirements. First, it must have been registered with one or more sources to receive notifications about specific types of events. Second, it must implement methods to receive and process these notifications.
What is adapter class?- An adapter class provides an empty implementation of all methods in an event listener interface. Adapter classes are useful when you want to receive and process only some of the events that are handled by a particular event listener interface. You can define a new class to act listener by extending one of the adapter classes and implementing only those events in which you are interested. For example, the MouseMotionAdapter class has two methods, mouseDragged()and mouseMoved(). The signatures of these empty are exactly as defined in the MouseMotionListener interface. If you are interested in only mouse drag events, then you could simply extend MouseMotionAdapter and implement mouseDragged() .
What is meant by controls and what are different types of controls in AWT?- Controls are components that allow a user to interact with your application and the AWT supports the following types of controls: Labels, Push Buttons, Check Boxes, Choice Lists, Lists, Scrollbars, Text Components. These controls are subclasses of Component.
What is the difference between choice and list?- A Choice is displayed in a compact form that requires you to pull it down to see the list of available choices and only one item may be selected from a choice. A List may be displayed in such a way that several list items are visible and it supports the selection of one or more list items.
What is the difference between scrollbar and scrollpane?- A Scrollbar is a Component, but not a Container whereas Scrollpane is a Conatiner and handles its own events and perform its own scrolling.
What is a layout manager and what are different types of layout managers available in java AWT?- A layout manager is an object that is used to organize components in a container. The different layouts are available are FlowLayout, BorderLayout, CardLayout, GridLayout and GridBagLayout.
How are the elements of different layouts organized?- FlowLayout: The elements of a FlowLayout are organized in a top to bottom, left to right fashion. BorderLayout: The elements of a BorderLayout are organized at the borders (North, South, East and West) and the center of a container. CardLayout: The elements of a CardLayout are stacked, on top of the other, like a deck of cards. GridLayout: The elements of a GridLayout are of equal size and are laid out using the square of a grid. GridBagLayout: The elements of a GridBagLayout are organized according to a grid. However, the elements are of different size and may occupy more than one row or column of the grid. In addition, the rows and columns may have different sizes.
Which containers use a Border layout as their default layout?- Window, Frame and Dialog classes use a BorderLayout as their layout.
Which containers use a Flow layout as their default layout?- Panel and Applet classes use the FlowLayout as their default layout.

What are wrapper classes?- Wrapper classes are classes that allow primitive types to be accessed as objects.
What are Vector, Hashtable, LinkedList and Enumeration?- Vector : The Vector class provides the capability to implement a growable array of objects. Hashtable : The Hashtable class implements a Hashtable data structure. A Hashtable indexes and stores objects in a dictionary using hash codes as the object's keys. Hash codes are integer values that identify objects. LinkedList: Removing or inserting elements in the middle of an array can be done using LinkedList. A LinkedList stores each object in a separate link whereas an array stores object references in consecutive locations. Enumeration: An object that implements the Enumeration interface generates a series of elements, one at a time. It has two methods, namely hasMoreElements() and nextElement(). HasMoreElemnts() tests if this enumeration has more elements and nextElement method returns successive elements of the series.
What is the difference between set and list?- Set stores elements in an unordered way but does not contain duplicate elements, whereas list stores elements in an ordered way but may contain duplicate elements.
What is a stream and what are the types of Streams and classes of the Streams?- A Stream is an abstraction that either produces or consumes information. There are two types of Streams and they are: Byte Streams: Provide a convenient means for handling input and output of bytes. Character Streams: Provide a convenient means for handling input & output of characters. Byte Streams classes: Are defined by using two abstract classes, namely InputStream and OutputStream. Character Streams classes: Are defined by using two abstract classes, namely Reader and Writer.
What is the difference between Reader/Writer and InputStream/Output Stream?- The Reader/Writer class is character-oriented and the InputStream/OutputStream class is byte-oriented.
What is an I/O filter?- An I/O filter is an object that reads from one stream and writes to another, usually altering the data in some way as it is passed from one stream to another.
What is serialization and deserialization?- Serialization is the process of writing the state of an object to a byte stream. Deserialization is the process of restoring these objects.

CORE JAVA QUESTIONS by Systems Technology Group
1.what is a transient variable?
A transient variable is a variable that may not be serialized.

2.which containers use a border Layout as their default layout?
The window, Frame and Dialog classes use a border layout as their default layout.

3.Why do threads block on I/O?
Threads block on i/o (that is enters the waiting state) so that other threads may execute while the i/o Operation is performed.

4. How are Observer and Observable used?
Objects that subclass the Observable class maintain a list of observers. When an Observable object is updated it invokes the update() method of each of its observers to notify the observers that it has changed state. The Observer interface is implemented by objects that observe Observable objects.

5. What is synchronization and why is it important?
With respect to multithreading, synchronization is the capability to control the access of multiple threads to shared resources. Without synchronization, it is possible for one thread to modify a shared object while another thread is in the process of using or updating that object's value. This often leads to significant errors.

6. Can a lock be acquired on a class?
Yes, a lock can be acquired on a class. This lock is acquired on the class's Class object..

7. What's new with the stop(), suspend() and resume() methods in JDK 1.2?
The stop(), suspend() and resume() methods have been deprecated in JDK 1.2.

8. Is null a keyword?
The null value is not a keyword.

9. What is the preferred size of a component?
The preferred size of a component is the minimum component size that will allow the component to display normally.

10. What method is used to specify a container's layout?
The setLayout() method is used to specify a container's layout.

11. Which containers use a FlowLayout as their default layout?
The Panel and Applet classes use the FlowLayout as their default layout.

12. What state does a thread enter when it terminates its processing?
When a thread terminates its processing, it enters the dead state.

13. What is the Collections API?
The Collections API is a set of classes and interfaces that support operations on collections of objects.

14. Which characters may be used as the second character of an identifier,
but not as the first character of an identifier?
The digits 0 through 9 may not be used as the first character of an identifier but they may be used after the first character of an identifier.

15. What is the List interface?
The List interface provides support for ordered collections of objects.

16. How does Java handle integer overflows and underflows?
It uses those low order bytes of the result that can fit into the size of the type allowed by the operation.

17. What is the Vector class?
The Vector class provides the capability to implement a growable array of objects

18. What modifiers may be used with an inner class that is a member of an outer class?
A (non-local) inner class may be declared as public, protected, private, static, final, or abstract.

19. What is an Iterator interface?
The Iterator interface is used to step through the elements of a Collection.

20. What is the difference between the >> and >>> operators?
The >> operator carries the sign bit when shifting right. The >>> zero-fills bits that have been shifted out.

21. Which method of the Component class is used to set the position and
size of a component?
setBounds()

22. How many bits are used to represent Unicode, ASCII, UTF-16, and UTF-8 characters?
Unicode requires 16 bits and ASCII require 7 bits. Although the ASCII character set uses only 7 bits, it is usually represented as 8 bits. UTF-8 represents characters using 8, 16, and 18 bit patterns. UTF-16 uses 16-bit and larger bit patterns.

23What is the difference between yielding and sleeping?
When a task invokes its yield() method, it returns to the ready state. When a task invokes its sleep() method, it returns to the waiting state.

24. Which java.util classes and interfaces support event handling?
The EventObject class and the EventListener interface support event processing.

25. Is sizeof a keyword?
The sizeof operator is not a keyword.

26. What are wrapped classes?
Wrapped classes are classes that allow primitive types to be accessed as objects.

27. Does garbage collection guarantee that a program will not run out of memory?
Garbage collection does not guarantee that a program will not run out of memory. It is possible for programs to use up memory resources faster than they are garbage collected. It is also possible for programs to create objects that are not subject to garbage collection

28. What restrictions are placed on the location of a package statement
within a source code file?
A package statement must appear as the first line in a source code file (excluding blank lines and comments).

29. Can an object's finalize() method be invoked while it is reachable?
An object's finalize() method cannot be invoked by the garbage collector while the object is still reachable. However, an object's finalize() method may be invoked by other objects.

30. What is the immediate superclass of the Applet class?
Panel

31. What is the difference between preemptive scheduling and time slicing?

Under preemptive scheduling, the highest priority task executes until it enters the waiting or dead states or a higher priority task comes into existence. Under time slicing, a task executes for a predefined slice of time and then reenters the pool of ready tasks. The scheduler then determines which task should execute next, based on priority and
other factors.

32. Name three Component subclasses that support painting.
The Canvas, Frame, Panel, and Applet classes support painting.

33. What value does readLine() return when it has reached the end of a file?
The readLine() method returns null when it has reached the end of a file.

34. What is the immediate superclass of the Dialog class?
Window

35. What is clipping?
Clipping is the process of confining paint operations to a limited area or shape.

36. What is a native method?
A native method is a method that is implemented in a language other than Java.

37. Can a for statement loop indefinitely?
Yes, a for statement can loop indefinitely. For example, consider the following:
for(;;) ;

38. What are order of precedence and associativity, and how are they used?
Order of precedence determines the order in which operators are evaluated in expressions. Associatity determines whether an expression is evaluated left-to-right or right-to-left

39. When a thread blocks on I/O, what state does it enter?
A thread enters the waiting state when it blocks on I/O.

40. To what value is a variable of the String type automatically initialized?
The default value of an String type is null.

41. What is the catch or declare rule for method declarations?
If a checked exception may be thrown within the body of a method, the method must either catch the exception or declare it in its throws clause.

42. What is the difference between a MenuItem and a CheckboxMenuItem?
The CheckboxMenuItem class extends the MenuItem class to support a menu item that may be checked or unchecked.

43. What is a task's priority and how is it used in scheduling?
A task's priority is an integer value that identifies the relative order in which it should be executed with respect to other tasks. The scheduler attempts to schedule higher priority tasks before lower priority tasks.

44. What class is the top of the AWT event hierarchy?
The java.awt.AWTEvent class is the highest-level class in the AWT event-class hierarchy.

45. When a thread is created and started, what is its initial state?
A thread is in the ready state after it has been created and started.

46. Can an anonymous class be declared as implementing an interface and extending a class?
An anonymous class may implement an interface or extend a superclass, but may not be declared to do both.

47. What is the range of the short type?
The range of the short type is -(2^15) to 2^15 - 1.

48. What is the range of the char type?
The range of the char type is 0 to 2^16 - 1.

49. In which package are most of the AWT events that support the event-delegation
model defined?

Most of the AWT-related events of the event-delegation model are defined in the java.awt.event package. The AWTEvent class is defined in the java.awt package.

50. What is the immediate superclass of Menu?
MenuItem

51. What is the purpose of finalization?
The purpose of finalization is to give an unreachable object the opportunity to perform any cleanup processing before the object is garbage collected.

52. Which class is the immediate superclass of the MenuComponent class.
Object

53. What invokes a thread's run() method?
After a thread is started, via its start() method or that of the Thread class, the JVM invokes the thread's run() method when the thread is initially executed.

54. What is the difference between the Boolean & operator and the && operator?
If an expression involving the Boolean & operator is evaluated, both operands are evaluated. Then the & operator is applied to the operand. When an expression involving the && operator is evaluated, the first operand is evaluated. If the first operand returns a value of true then the second operand is evaluated. The && operator is then applied to the first and second operands. If the first operand evaluates to false, the evaluation of the second operand is skipped.

55. Name three subclasses of the Component class.
Box.Filler, Button, Canvas, Checkbox, Choice, Container, Label, List, Scrollbar, or TextComponent

56. What is the GregorianCalendar class?
The GregorianCalendar provides support for traditional Western calendars.

57. Which Container method is used to cause a container to be laid out and redisplayed?
validate()

58. What is the purpose of the Runtime class?
The purpose of the Runtime class is to provide access to the Java runtime system.

59. How many times may an object's finalize() method be invoked by the
garbage collector?
An object's finalize() method may only be invoked once by the garbage collector.

60. What is the purpose of the finally clause of a try-catch-finally statement?
The finally clause is used to provide the capability to execute code no matter whether or not an exception is thrown or caught.

61. What is the argument type of a program's main() method?
A program's main() method takes an argument of the String[] type.

62. Which Java operator is right associative?
The = operator is right associative.

63. What is the Locale class?
The Locale class is used to tailor program output to the conventions of a particular geographic, political, or cultural region.

64. Can a double value be cast to a byte?
Yes, a double value can be cast to a byte.

65. What is the difference between a break statement and a continue statement?
A break statement results in the termination of the statement to which it applies (switch, for, do, or while). A continue statement is used to end the current loop iteration and return control to the loop statement.

66. What must a class do to implement an interface?
It must provide all of the methods in the interface and identify the interface in its implements clause.

67. What method is invoked to cause an object to begin executing as a separate thread?
The start() method of the Thread class is invoked to cause an object to begin executing as a separate thread.

68. Name two subclasses of the TextComponent class.
TextField and TextArea

69. What is the advantage of the event-delegation model over the earlier event-inheritance model?
The event-delegation model has two advantages over the event-inheritance model. First, it enables event handling to be handled by objects other than the ones that generate the events (or their containers). This allows a clean separation between a component's design and its use. The other advantage of the event-delegation model is that it performs much better in applications where many events are generated. This performance improvement is due to the fact that the event-delegation model does not have to repeatedly process unhandled events, as is the case of the event-inheritance model.

70. Which containers may have a MenuBar?
Frame

71. How are commas used in the intialization and iteration parts of a for statement?
Commas are used to separate multiple statements within the initialization and iteration parts of a for statement.

72. What is the purpose of the wait(), notify(), and notifyAll() methods?
The wait(),notify(), and notifyAll() methods are used to provide an efficient way for threads to wait for a shared resource. When a thread executes an object's wait() method, it enters the waiting state. It only enters the ready state after another thread invokes the object's notify() or notifyAll() methods..

73. What is an abstract method?
An abstract method is a method whose implementation is deferred to a subclass.

74. How are Java source code files named?
A Java source code file takes the name of a public class or interface that is defined within the file. A source code file may contain at most one public class or interface. If a public class or interface is defined within a source code file, then the source code file must take the name of the public class or interface. If no public class or interface is defined within a source code file, then the file must take on a name that is different than its classes and interfaces. Source code files use the .java extension.

75. What is the relationship between the Canvas class and the Graphics class?
A Canvas object provides access to a Graphics object via its paint() method.

76. What are the high-level thread states?
The high-level thread states are ready, running, waiting, and dead.

77. What value does read() return when it has reached the end of a file?
The read() method returns -1 when it has reached the end of a file.

78. Can a Byte object be cast to a double value?
No, an object cannot be cast to a primitive value.

79. What is the difference between a static and a non-static inner class?
A non-static inner class may have object instances that are associated with instances of the class's outer class. A static inner class does not have any object instances.

80. What is the difference between the String and StringBuffer classes?
String objects are constants. StringBuffer objects are not.

81. If a variable is declared as private, where may the variable be accessed?
A private variable may only be accessed within the class in which it is declared.

82. What is an object's lock and which object's have locks?
An object's lock is a mechanism that is used by multiple threads to obtain synchronized access to the object. A thread may execute a synchronized method of an object only after it has acquired the object's lock. All objects and classes have locks. A class's lock is acquired on the class's Class object.

83. What is the Dictionary class?
The Dictionary class provides the capability to store key-value pairs.

84. How are the elements of a BorderLayout organized?
The elements of a BorderLayout are organized at the borders (North, South, East, and West) and the center of a container.

85. What is the % operator?

It is referred to as the modulo or remainder operator. It returns the remainder of dividing the first operand by the second operand.

86. When can an object reference be cast to an interface reference?
An object reference be cast to an interface reference when the object implements the referenced interface.

87. What is the difference between a Window and a Frame?
The Frame class extends Window to define a main application window that can have a menu bar.

88. Which class is extended by all other classes?
The Object class is extended by all other classes.

89. Can an object be garbage collected while it is still reachable?
A reachable object cannot be garbage collected. Only unreachable objects may be garbage collected..

90. Is the ternary operator written x : y ? z or x ? y : z ?
It is written x ? y : z.

91. What is the difference between the Font and FontMetrics classes?
The FontMetrics class is used to define implementation-specific properties, such as ascent and descent, of a Font object.

92. How is rounding performed under integer division?
The fractional part of the result is truncated. This is known as rounding toward zero.

93. What happens when a thread cannot acquire a lock on an object?
If a thread attempts to execute a synchronized method or synchronized statement and is unable to acquire an object's lock, it enters the waiting state until the lock becomes available.

94. What is the difference between the Reader/Writer class hierarchy and the InputStream/OutputStream class hierarchy?
The Reader/Writer class hierarchy is character-oriented, and the InputStream/OutputStream class hierarchy is byte-oriented.

95. What classes of exceptions may be caught by a catch clause?
A catch clause can catch any exception that may be assigned to the Throwable type. This includes the Error and Exception types.

96. If a class is declared without any access modifiers, where may the class be accessed?
A class that is declared without any access modifiers is said to have package access. This means that the class can only be accessed by other classes and interfaces that are defined within the same package.

97. What is the SimpleTimeZone class?
The SimpleTimeZone class provides support for a Gregorian calendar.

98. What is the Map interface?
The Map interface replaces the JDK 1.1 Dictionary class and is used associate keys with values.

99. Does a class inherit the constructors of its superclass?
A class does not inherit constructors from any of its superclasses.

100. For which statements does it make sense to use a label?
The only statements for which it makes sense to use a label are those statements that can enclose a break or continue statement.

101. What is the purpose of the System class?
The purpose of the System class is to provide access to system resources.

102. Which TextComponent method is used to set a TextComponent to the read-only state?
setEditable()

103. How are the elements of a CardLayout organized?
The elements of a CardLayout are stacked, one on top of the other, like a deck of cards.

104. Is &&= a valid Java operator?
No, it is not.

105. Name the eight primitive Java types.
The eight primitive types are byte, char, short, int, long, float, double, and boolean.

106. Which class should you use to obtain design information about an object?
The Class class is used to obtain information about an object's design.

107. What is the relationship between clipping and repainting?
When a window is repainted by the AWT painting thread, it sets the clipping regions to the area of the window that requires repainting.

108. Is "abc" a primitive value?
The String literal "abc" is not a primitive value. It is a String object.

109. What is the relationship between an event-listener interface and an
event-adapter class?
An event-listener interface defines the methods that must be implemented by an event handler for a particular kind of event. An event adapter provides a default implementation of an event-listener interface.

110. What restrictions are placed on the values of each case of a switch statement?
During compilation, the values of each case of a switch statement must evaluate to a value that can be promoted to an int value.

111. What modifiers may be used with an interface declaration?
An interface may be declared as public or abstract.

112. Is a class a subclass of itself?
A class is a subclass of itself.

113. What is the highest-level event class of the event-delegation model?
The java.util.EventObject class is the highest-level class in the event-delegation class hierarchy.

114. What event results from the clicking of a button?
The ActionEvent event is generated as the result of the clicking of a button.

115. How can a GUI component handle its own events?
A component can handle its own events by implementing the required event-listener interface and adding itself as its own event listener.

116. What is the difference between a while statement and a do statement?
A while statement checks at the beginning of a loop to see whether the next loop iteration should occur. A do statement checks at the end of a loop to see whether the next iteration of a loop should occur. The do statement will always execute the body of a loop at least once.

117. How are the elements of a GridBagLayout organized?
The elements of a GridBagLayout are organized according to a grid. However, the elements are of different sizes and may occupy more than one row or column of the grid. In addition, the rows and columns may have different sizes.

118. What advantage do Java's layout managers provide over traditional windowing systems?
Java uses layout managers to lay out components in a consistent manner across all windowing platforms. Since Java's layout managers aren't tied to absolute sizing and positioning, they are able to accomodate platform-specific differences among windowing systems.

119. What is the Collection interface?
The Collection interface provides support for the implementation of a mathematical bag - an unordered collection of objects that may contain duplicates.

120. What modifiers can be used with a local inner class?
A local inner class may be final or abstract.

121. What is the difference between static and non-static variables?
A static variable is associated with the class as a whole rather than with specific instances of a class. Non-static variables take on unique values with each object instance.

122. What is the difference between the paint() and repaint() methods?

The paint() method supports painting via a Graphics object. The repaint() method is used to cause paint() to be invoked by the AWT painting thread.

123. What is the purpose of the File class?
The File class is used to create objects that provide access to the files and directories of a local file system.

124. Can an exception be rethrown?
Yes, an exception can be rethrown.

125. Which Math method is used to calculate the absolute value of a number?
The abs() method is used to calculate absolute values.

126. How does multithreading take place on a computer with a single CPU?
The operating system's task scheduler allocates execution time to multiple tasks. By quickly switching between executing tasks, it creates the impression that tasks execute sequentially.

127. When does the compiler supply a default constructor for a class?
The compiler supplies a default constructor for a class if no other constructors are provided.

128. When is the finally clause of a try-catch-finally statement executed?
The finally clause of the try-catch-finally statement is always executed unless the thread of execution terminates or an exception occurs within the execution of the finally clause.

129. Which class is the immediate superclass of the Container class?
Component

130. If a method is declared as protected, where may the method be accessed?
A protected method may only be accessed by classes or interfaces of the same package or by subclasses of the class in which it is declared.

131. How can the Checkbox class be used to create a radio button?
By associating Checkbox objects with a CheckboxGroup.

132. Which non-Unicode letter characters may be used as the first character
of an identifier?
The non-Unicode letter characters $ and _ may appear as the first character of an identifier

133. What restrictions are placed on method overloading?
Two methods may not have the same name and argument list but different return types.

134. What happens when you invoke a thread's interrupt method while it is
sleeping or waiting?
When a task's interrupt() method is executed, the task enters the ready state. The next time the task enters the running state, an InterruptedException is thrown.

135. What is casting?
There are two types of casting, casting between primitive numeric types and casting between object references. Casting between numeric types is used to convert larger values, such as double values, to smaller values, such as byte values. Casting between object references is used to refer to an object by a compatible class, interface, or array type reference.

136. What is the return type of a program's main() method?
A program's main() method has a void return type.

137. Name four Container classes.
Window, Frame, Dialog, FileDialog, Panel, Applet, or ScrollPane

138. What is the difference between a Choice and a List?
A Choice is displayed in a compact form that requires you to pull it down to see the list of available choices. Only one item may be selected from a Choice. A List may be displayed in such a way that several List items are visible. A List supports the selection of one or more List items.

139. What class of exceptions are generated by the Java run-time system?

The Java runtime system generates RuntimeException and Error exceptions.

140. What class allows you to read objects directly from a stream?
The ObjectInputStream class supports the reading of objects from input streams.

141. What is the difference between a field variable and a local variable?
A field variable is a variable that is declared as a member of a class. A local variable is a variable that is declared local to a method.

142. Under what conditions is an object's finalize() method invoked by the garbage collector?
The garbage collector invokes an object's finalize() method when it detects that the object has become unreachable.

143. How are this() and super() used with constructors?
this() is used to invoke a constructor of the same class. super() is used to invoke a superclass constructor.

144. What is the relationship between a method's throws clause and the exceptions
that can be thrown during the method's execution?
A method's throws clause must declare any checked exceptions that are not caught within the body of the method.

145. What is the difference between the JDK 1.02 event model and the event-delegation
model introduced with JDK 1.1?
The JDK 1.02 event model uses an event inheritance or bubbling approach. In this model, components are required to handle their own events. If they do not handle a particular event, the event is inherited by (or bubbled up to) the component's container. The container then either handles the event or it is bubbled up to its container and so on, until the highest-level container has been tried..

In the event-delegation model, specific objects are designated as event handlers for GUI components. These objects implement event-listener interfaces. The event-delegation model is more efficient than the event-inheritance model because it eliminates the processing required to support the bubbling of unhandled events.

146. How is it possible for two String objects with identical values not to be equal
under the == operator?
The == operator compares two objects to determine if they are the same object in memory. It is possible for two String objects to have the same value, but located indifferent areas of memory.

147. Why are the methods of the Math class static?
So they can be invoked as if they are a mathematical code library.

148. What Checkbox method allows you to tell if a Checkbox is checked?
getState()

149. What state is a thread in when it is executing?
An executing thread is in the running state.

150. What are the legal operands of the instanceof operator?
The left operand is an object reference or null value and the right operand is a class, interface, or array type.

151. How are the elements of a GridLayout organized?
The elements of a GridBad layout are of equal size and are laid out using the squares of a grid.

152. What an I/O filter?
An I/O filter is an object that reads from one stream and writes to another, usually altering the data in some way as it is passed from one stream to another.

153. If an object is garbage collected, can it become reachable again?
Once an object is garbage collected, it ceases to exist. It can no longer become reachable again.

154. What is the Set interface?
The Set interface provides methods for accessing the elements of a finite mathematical set. Sets do not allow duplicate elements.

155. What classes of exceptions may be thrown by a throw statement?
A throw statement may throw any expression that may be assigned to the Throwable type.

156. What are E and PI?
E is the base of the natural logarithm and PI is mathematical value pi.

157. Are true and false keywords?
The values true and false are not keywords.

158. What is a void return type?
A void return type indicates that a method does not return a value.

159. What is the purpose of the enableEvents() method?
The enableEvents() method is used to enable an event for a particular object. Normally, an event is enabled when a listener is added to an object for a particular event. The enableEvents() method is used by objects that handle events by overriding their event-dispatch methods.

160. What is the difference between the File and RandomAccessFile classes?
The File class encapsulates the files and directories of the local file system. The RandomAccessFile class provides the methods needed to directly access data contained in any part of a file.

161. What happens when you add a double value to a String?
The result is a String object.

162. What is your platform's default character encoding?
If you are running Java on English Windows platforms, it is probably Cp1252. If you are running Java on English Solaris platforms, it is most likely 8859_1..

163. Which package is always imported by default?
The java.lang package is always imported by default.

164. What interface must an object implement before it can be written to a
stream as an object?
An object must implement the Serializable or Externalizable interface before it can be written to a stream as an object.

165. How are this and super used?
this is used to refer to the current object instance. super is used to refer to the variables and methods of the superclass of the current object instance.

166. What is the purpose of garbage collection?
The purpose of garbage collection is to identify and discard objects that are no longer needed by a program so that their resources may be reclaimed and reused.

167. What is a compilation unit?
A compilation unit is a Java source code file.

168. What interface is extended by AWT event listeners?
All AWT event listeners extend the java.util.EventListener interface.

169. What restrictions are placed on method overriding?
Overridden methods must have the same name, argument list, and return type.
The overriding method may not limit the access of the method it overrides.
The overriding method may not throw any exceptions that may not be thrown
by the overridden method.

170. How can a dead thread be restarted?
A dead thread cannot be restarted.

171. What happens if an exception is not caught?
An uncaught exception results in the uncaughtException() method of the thread's ThreadGroup being invoked, which eventually results in the termination of the program in which it is thrown.

172. What is a layout manager?
A layout manager is an object that is used to organize components in a container.

173. Which arithmetic operations can result in the throwing of an ArithmeticException?

Integer / and % can result in the throwing of an ArithmeticException.

174. What are three ways in which a thread can enter the waiting state?
A thread can enter the waiting state by invoking its sleep() method, by blocking on I/O, by unsuccessfully attempting to acquire an object's lock, or by invoking an object's wait() method. It can also enter the waiting state by invoking its (deprecated) suspend() method.

175. Can an abstract class be final?
An abstract class may not be declared as final.

176. What is the ResourceBundle class?
The ResourceBundle class is used to store locale-specific resources that can be loaded by a program to tailor the program's appearance to the particular locale in which it is being run.

177. What happens if a try-catch-finally statement does not have a catch clause to handle an exception that is thrown within the body of the try statement?
The exception propagates up to the next higher level try-catch statement (if any) or results in the program's termination.

178. What is numeric promotion?
Numeric promotion is the conversion of a smaller numeric type to a larger numeric type, so that integer and floating-point operations may take place. In numerical promotion, byte, char, and short values are converted to int values. The int values are also converted to long values, if necessary. The long and float values are converted to double values, as required.

179. What is the difference between a Scrollbar and a ScrollPane?
A Scrollbar is a Component, but not a Container. A ScrollPane is a Container. A ScrollPane handles its own events and performs its own scrolling.

180. What is the difference between a public and a non-public class?
A public class may be accessed outside of its package. A non-public class may not be accessed outside of its package.

181. To what value is a variable of the boolean type automatically initialized?
The default value of the boolean type is false.

182. Can try statements be nested?
Try statements may be tested.

183. What is the difference between the prefix and postfix forms of the ++ operator?
The prefix form performs the increment operation and returns the value of the increment operation. The postfix form returns the current value all of the expression and then performs the increment operation on that value.

184. What is the purpose of a statement block?
A statement block is used to organize a sequence of statements as a single statement group.

185. What is a Java package and how is it used?
A Java package is a naming context for classes and interfaces. A package is used to create a separate name space for groups of classes and interfaces. Packages are also used to organize related classes and interfaces into a single API unit and to control accessibility to these classes and interfaces.

186. What modifiers may be used with a top-level class?
A top-level class may be public, abstract, or final.

187. What are the Object and Class classes used for?
The Object class is the highest-level class in the Java class hierarchy. The Class class is used to represent the classes and interfaces that are loaded by a Java program..

188. How does a try statement determine which catch clause should be used to handle an exception?
When an exception is thrown within the body of a try statement, the catch clauses of the try statement are examined in the order in which they appear. The first catch clause that is capable of handling the exception is executed. The remaining catch clauses are ignored.

189. Can an unreachable object become reachable again?

An unreachable object may become reachable again. This can happen when the object's finalize() method is invoked and the object performs an operation which causes it to become accessible to reachable objects.

190. When is an object subject to garbage collection?
An object is subject to garbage collection when it becomes unreachable to the program in which it is used.

191. What method must be implemented by all threads?
All tasks must implement the run() method, whether they are a subclass of Thread or implement the Runnable interface.

192. What methods are used to get and set the text label displayed by a Button object?
getLabel() and setLabel()

193. Which Component subclass is used for drawing and painting?
Canvas

194. What are synchronized methods and synchronized statements?
Synchronized methods are methods that are used to control access to an object. A thread only executes a synchronized method after it has acquired the lock for the method's object or class. Synchronized statements are similar to synchronized methods. A synchronized statement can only be executed after a thread has acquired the lock for the object or class referenced in the synchronized statement.

195. What are the two basic ways in which classes that can be run as threads may be defined?
A thread class may be declared as a subclass of Thread, or it may implement the Runnable interface.

196. What are the problems faced by Java programmers who don't use layout managers?
Without layout managers, Java programmers are faced with determining how their GUI will be displayed across multiple windowing systems and finding a common sizing and positioning that will work within the constraints imposed by each windowing system.

197. What is the difference between an if statement and a switch statement?
The if statement is used to select among two alternatives. It uses a boolean expression to decide which alternative should be executed. The switch statement is used to select among multiple alternatives. It uses an int expression to determine which alternative should be executed.

198. What happens when you add a double value to a String?
The result is a String object.

199. What is the List interface?
The List interface provides support for ordered collections of objects.

FREQUENTLY ASKED QUESTIONS (JAVA) IN INTERVIEWS

1)What is OOPs?

Ans: Object oriented programming organizes a program around its data,i.e.,objects and a set of well defined interfaces to that data.An object-oriented program can be characterized as data controlling access to code.

2)what is the difference between Procedural and OOPs?
Ans: a) In procedural program, programming logic follows certain procedures and the instructions are executed one after another. In OOPs program, unit of program is object, which is nothing but combination of data and code.
b) In procedural program,data is exposed to the whole program whereas in OOPs program,it is accessible with in the object and which in turn assures the security of the code.

3)What are Encapsulation, Inheritance and Polymorphism?
Ans: Encapsulation is the mechanism that binds together code and data it manipulates and keeps both safe from outside interference and misuse.
Inheritance is the process by which one object acquires the properties of another object.
Polymorphism is the feature that allows one interface to be used for general class actions.

4)What is the difference between Assignment and Initialization?
Ans: Assignment can be done as many times as desired whereas initialization can be done only once.

5)What are Class, Constructor and Primitive data types?

Ans: Class is a template for multiple objects with similar features and it is a blue print for objects. It defines a type of object according to the data the object can hold and the operations the object can perform.

Constructor is a special kind of method that determines how an object is initialized when created.

Primitive data types are 8 types and they are:
byte, short, int, long
float, double
boolean
char

6)What is an Object and how do you allocate memory to it?
Ans: Object is an instance of a class and it is a software unit that combines a structured set of data with a set of operations for inspecting and manipulating that data. When an object is created using new operator, memory is allocated to it.

7)What is the difference between constructor and method?
Ans: Constructor will be automatically invoked when an object is created whereas method has to be called explicitly.

8)What are methods and how are they defined?
Ans: Methods are functions that operate on instances of classes in which they are defined. Objects can communicate with each other using methods and can call methods in other classes.
Method definition has four parts. They are name of the method, type of object or primitive type the method returns, a list of parameters and the body of the method. A method's signature is a combination of the first three parts mentioned above.

9)What is the use of bin and lib in JDK?
Ans: Bin contains all tools such as javac, appletviewer, awt tool, etc., whereas lib contains API and all packages.

10)What is casting?
Ans: Casting is used to convert the value of one type to another.

11)How many ways can an argument be passed to a subroutine and explain them?
Ans: An argument can be passed in two ways. They are passing by value and passing by reference.
Passing by value: This method copies the value of an argument into the formal parameter of the subroutine.
Passing by reference: In this method, a reference to an argument (not the value of the argument) is passed to the parameter.

12)What is the difference between an argument and a parameter?
Ans: While defining method, variables passed in the method are called parameters. While using those methods, values passed to those variables are called arguments.

13)What are different types of access modifiers?
Ans: public: Any thing declared as public can be accessed from anywhere.
private: Any thing declared as private can't be seen outside of its class.
protected: Any thing declared as protected can be accessed by classes in the same package and subclasses in the other packages.
default modifier : Can be accessed only to classes in the same package.

14)What is final, finalize() and finally?
Ans: final : final keyword can be used for class, method and variables.
A final class cannot be subclassed and it prevents other programmers from subclassing a secure class to invoke insecure methods.
A final method can' t be overridden
A final variable can't change from its initialized value.

finalize( ) : finalize( ) method is used just before an object is destroyed and can be called just prior to garbage collection.
finally : finally, a key word used in exception handling, creates a block of code that will be executed after a try/catch block has completed and before the code following the try/catch block. The finally block will execute whether or not an exception is thrown.
For example, if a method opens a file upon exit, then you will not want the code that closes the file to be bypassed by the exception-handling mechanism. This finally keyword is designed to address this contingency.

15)What is UNICODE?

Ans: Unicode is used for internal representation of characters and strings and it uses 16 bits to represent each other.

16)What is Garbage Collection and how to call it explicitly?
Ans: When an object is no longer referred to by any variable, java automatically reclaims memory used by that object. This is known as garbage collection.
System.gc() method may be used to call it explicitly.

17)What is finalize() method ?
Ans: finalize () method is used just before an object is destroyed and can be called just prior to garbage collection.

18)What are Transient and Volatile Modifiers?
Ans: Transient: The transient modifier applies to variables only and it is not stored as part of its object's Persistent state. Transient variables are not serialized.
Volatile: Volatile modifier applies to variables only and it tells the compiler that the variable modified by volatile can be changed unexpectedly by other parts of the program.

19)What is method overloading and method overriding?
Ans: Method overloading: When a method in a class having the same method name with different arguments is said to be method overloading.

Method overriding : When a method in a class having the same method name with same arguments is said to be method overriding.

20)What is difference between overloading and overriding?
Ans: a) In overloading, there is a relationship between methods available in the same class whereas in overriding, there is relationship between a superclass method and subclass method.
b) Overloading does not block inheritance from the superclass whereas overriding blocks inheritance from the superclass.
c) In overloading, separate methods share the same name whereas in overriding,subclass method replaces the superclass.
d) Overloading must have different method signatures whereas overriding must have same signature.

21) What is meant by Inheritance and what are its advantages?
Ans: Inheritance is the process of inheriting all the features from a class. The advantages of inheritance are reusability of code and accessibility of variables and methods of the super class by subclasses.

22)What is the difference between this() and super()?
Ans: this() can be used to invoke a constructor of the same class whereas super() can be used to invoke a super class constructor.

23)What is the difference between superclass and subclass?
Ans: A super class is a class that is inherited whereas sub class is a class that does the inheriting.

24) What modifiers may be used with top-level class?
Ans: public, abstract and final can be used for top-level class.

25)What are inner class and anonymous class?
Ans: Inner class : classes defined in other classes, including those defined in methods are called inner classes.
An inner class can have any accessibility including private.

Anonymous class : Anonymous class is a class defined inside a method without a name and is instantiated and declared in the same place and cannot have explicit constructors.

26)What is a package?
Ans: A package is a collection of classes and interfaces that provides a high-level layer of access protection and name space management.

27) What is a reflection package?
Ans: java.lang.reflect package has the ability to analyze itself in runtime.

28) What is interface and its use?
Ans:

Interface is similar to a class which may contain method's signature
only but not bodies and it is a formal set of method and constant declarations that must be defined by the class that implements it.

Interfaces are useful for:
a)Declaring methods that one or more classes are expected to implement

b)Capturing similarities between unrelated classes without forcing a class relationship.
c)Determining an object's programming interface without revealing the actual body of the class.

29) What is an abstract class?
Ans: An abstract class is a class designed with implementation gaps for subclasses to fill in and is deliberately incomplete.

30) What is the difference between Integer and int?
Ans: a) Integer is a class defined in the java.lang package, whereas int is a primitive data type defined in the Java language itself. Java does not automatically convert from one to the other.
b) Integer can be used as an argument for a method that requires an object, whereas int can be used for calculations.

31) What is a cloneable interface and how many methods does it contain?
Ans- It is not having any method because it is a TAGGED or MARKER interface.

32) What is the difference between abstract class and interface?
Ans: a) All the methods declared inside an interface are abstract whereas abstract class must have at least one abstract method and others may be concrete or abstract.

b) In abstract class, key word abstract must be used for the methods
whereas interface we need not use that keyword for the methods.

c) Abstract class must have subclasses whereas interface can't have subclasses.

33) Can you have an inner class inside a method and what variables can you access?
Ans: Yes, we can have an inner class inside a method and final variables can be accessed.

34) What is the difference between String and String Buffer?
Ans: a) String objects are constants and immutable whereas
StringBuffer objects are not.
b) String class supports constant strings whereas
StringBuffer class supports growable and modifiable strings.

35) What is the difference between Array and vector?
Ans: Array is a set of related data type and static whereas vector
is a growable array of objects and dynamic.

36) What is the difference between exception and error?
Ans: The exception class defines mild error conditions that your program encounters.
Ex: Arithmetic Exception, FilenotFound exception
Exceptions can occur when
-- try to open the file, which does not exist
-- the network connection is disrupted
-- operands being manipulated are out of prescribed ranges
-- the class file you are interested in loading is missing
The error class defines serious error conditions that you should not attempt to recover from. In most cases it is advisable to let the program terminate when such an error is encountered.
Ex: Running out of memory error, Stack overflow error.

37) What is the difference between process and thread?
Ans: Process is a program in execution whereas thread is a separate path of execution in a program.

38) What is multithreading and what are the methods for inter-thread communication and what is the class
in which these methods are defined?
Ans: Multithreading is the mechanism in which more than one thread run independent of each other within the process.
wait (), notify () and notifyAll() methods can be used for inter-thread communication and these methods are in Object class.

wait( ) : When a thread executes a call to wait( ) method, it surrenders the object lock and enters into a waiting state.
notify( ) or notifyAll( ) : To remove a thread from the waiting state, some other thread must make a call to notify( ) or notifyAll( ) method on the same object.

39) What is the class and interface in java to create thread and which is the most advantageous method?
Ans: Thread class and Runnable interface can be used to create threads and using Runnable interface is the most advantageous method to create threads because we need not extend thread class here.

40) What are the states associated in the thread?
Ans: Thread contains ready, running, waiting and dead states.

41) What is synchronization?
Ans: Synchronization is the mechanism that ensures that only one thread is accessed the resources at a time.

42) When you will synchronize a piece of your code?
Ans: When you expect your code will be accessed by different threads and these threads may change a particular data causing data corruption.

43) What is deadlock?
Ans: When two threads are waiting each other and can't precede the program is said to be deadlock.

44) What is daemon thread and which method is used to create the daemon thread?
Ans: Daemon thread is a low priority thread which runs intermittently in the back ground doing the garbage collection operation for the java runtime system. setDaemon method is used to create a daemon thread.

45) Are there any global variables in Java, which can be accessed by other part of your program?
Ans: No, it is not the main method in which you define variables. Global variables is not possible because concept of encapsulation is eliminated here.

46)What is an applet?
Ans: Applet is a dynamic and interactive program that runs inside a web page displayed by a java capable browser.

47)What is the difference between applications and applets?
Ans: a)Application must be run on local machine whereas applet needs no explicit installation on local machine.
b)Application must be run explicitly within a java-compatible virtual machine whereas applet loads and runs itself automatically in a java-enabled browser.
d)Application starts execution with its main method whereas applet starts execution with its init method.
e)Application can run with or without graphical user interface whereas applet must run within a graphical user interface.

48)How does applet recognize the height and width?
Ans:Using getParameters() method.

49)When do you use codebase in applet?
Ans:When the applet class file is not in the same directory, codebase is used.

50)What is the lifecycle of an applet?
Ans:init( ) method - Can be called when an applet is first loaded
start( ) method - Can be called each time an applet is started
paint( ) method - Can be called when the applet is minimized or maximized
stop( ) method - Can be used when the browser moves off the applet's page
destroy( ) method - Can be called when the browser is finished with the applet

51)How do you set security in applets?
Ans: using setSecurityManager() method

52) What is an event and what are the models available for event handling?
Ans: An event is an event object that describes a state of change in a source. In other words, event occurs when an action is generated, like pressing button, clicking mouse, selecting a list, etc.
There are two types of models for handling events and they are:
a) event-inheritance model and b) event-delegation model
53) What are the advantages of the model over the event-inheritance model?
Ans: The event-delegation model has two advantages over the event-inheritance model. They are:
a)It enables event handling by objects other than the ones that generate the events. This allows a clean separation between a component's design and its use.
b)It performs much better in applications where many events are generated. This performance improvement is due to the fact that the event-delegation model does not have to be repeatedly process unhandled events as is the case of the event-inheritance.

54)What is source and listener ?
Ans: source : A source is an object that generates an event. This occurs when the internal state of that object changes in some way.
listener : A listener is an object that is notified when an event occurs. It has two major requirements. First, it must have been registered with one or more sources to receive notifications about specific types of events.
Second, it must implement methods to receive and process these notifications.

55) What is adapter class?
Ans: An adapter class provides an empty implementation of all methods in an event listener interface. Adapter

classes are useful when you want to receive and process only some of the events that are handled by a particular event listener interface. You can define a new class to act listener by extending one of the adapter classes and implementing only those events in which you are interested.
For example, the MouseMotionAdapter class has two methods, mouseDragged( )and mouseMoved(). The signatures of these empty are exactly as defined in the MouseMotionListener interface. If you are interested in only mouse drag events, then you could simply extend MouseMotionAdapter and implement mouseDragged( ) .

56)What is meant by controls and what are different types of controls in AWT?
Ans: Controls are components that allow a user to interact with your application and the AWT supports the following types of controls:
Labels, Push Buttons, Check Boxes, Choice Lists, Lists, Scrollbars, Text Components.
These controls are subclasses of Component.

57) What is the difference between choice and list?
Ans: A Choice is displayed in a compact form that requires you to pull it down to see the list of available choices and only one item may be selected from a choice.
A List may be displayed in such a way that several list items are visible and it supports the selection of one or more list items.

58) What is the difference between scrollbar and scrollpane?
Ans: A Scrollbar is a Component, but not a Container whereas Scrollpane is a Conatiner and handles its own events and perform its own scrolling.

59) What is a layout manager and what are different types of layout managers available in java.awt?
Ans: A layout manager is an object that is used to organize components in a container. The different layouts are available are FlowLayout, BorderLayout, CardLayout, GridLayout and GridBagLayout.

60) How are the elements of different layouts organized?
Ans: FlowLayout: The elements of a FlowLayout are organized in a top to bottom, left to right fashion.
BorderLayout: The elements of a BorderLayout are organized at the borders (North, South, East and West) and the center of a container.
CardLayout: The elements of a CardLayout are stacked, on top of the other, like a deck of cards.
GridLayout: The elements of a GridLayout are of equal size and are laid out using the square of a grid.
GridBagLayout: The elements of a GridBagLayout are organized according to a grid.
However, the elements are of different size and may occupy more than one row or column of the grid. In addition, the rows and columns may have different sizes.

61) Which containers use a Border layout as their default layout?
Ans: Window, Frame and Dialog classes use a BorderLayout as their layout.

62) Which containers use a Flow layout as their default layout?
Ans: Panel and Applet classes use the FlowLayout as their default layout.

63) What are wrapper classes?
Ans: Wrapper classes are classes that allow primitive types to be accessed as objects.

64) What are Vector, Hashtable, LinkedList and Enumeration?
Ans: Vector : The Vector class provides the capability to implement a growable array of objects.
Hashtable : The Hashtable class implements a Hashtable data structure. A Hashtable indexes and stores objects in a dictionary using hash codes as the object's keys. Hash codes are integer values that identify objects.
LinkedList: Removing or inserting elements in the middle of an array can be done using LinkedList. A LinkedList stores each object in a separate link whereas an array stores object references in consecutive locations.
Enumeration: An object that implements the Enumeration interface generates a series of elements, one at a time. It has two methods, namely hasMoreElements( ) and nextElement( ). HasMoreElemnts( ) tests if this enumeration has more elements and nextElement method returns successive elements of the series.

65) What is the difference between set and list?
Ans: Set stores elements in an unordered way but does not contain duplicate elements, whereas list stores elements in an ordered way but may contain duplicate elements.

66) What is a stream and what are the types of Streams and classes of the Streams?
Ans: A Stream is an abstraction that either produces or consumes information. There are two types of Streams

and they are:
Byte Streams: Provide a convenient means for handling input and output of bytes.
Character Streams: Provide a convenient means for handling input & output of characters.
Byte Streams classes: Are defined by using two abstract classes, namely InputStream and OutputStream.
Character Streams classes: Are defined by using two abstract classes, namely Reader and Writer.

67) What is the difference between Reader/Writer and InputStream/Output Stream?
Ans: The Reader/Writer class is character-oriented and the InputStream/OutputStream class is byte-oriented.

68) What is an I/O filter?
Ans: An I/O filter is an object that reads from one stream and writes to another, usually altering the data in some way as it is passed from one stream to another.

69) What is serialization and deserialization?
Ans: Serialization is the process of writing the state of an object to a byte stream.
Deserialization is the process of restoring these objects.

*Q1. How could Java classes direct program messages to the system console, but error messages, say to a file?

A. The class System has a variable out that represents the standard output, and the variable err that represents the standard error device. By default, they both point at the system console. This how the standard output could be re-directed:
Stream st = new Stream(new FileOutputStream("output.txt")); System.setErr(st); System.setOut(st);

*Q2. What's the difference between an interface and an abstract class?

A. An abstract class may contain code in method bodies, which is not allowed in an interface. With abstract classes, you have to inherit your class from it and Java does not allow multiple inheritance. On the other hand, you can implement multiple interfaces in your class.

*Q3. Why would you use a synchronized block vs. synchronized method?

A. Synchronized blocks place locks for shorter periods than synchronized methods.

*Q4. Explain the usage of the keyword transient?

A. This keyword indicates that the value of this member variable does not have to be serialized with the object. When the class will be de-serialized, this variable will be initialized with a default value of its data type (i.e. zero for integers).

*Q5. How can you force garbage collection?

A. You can't force GC, but could request it by calling System.gc(). JVM does not guarantee that GC will be started immediately.

*Q6. How do you know if an explicit object casting is needed?

A. If you assign a superclass object to a variable of a subclass's data type, you need to do explicit casting. For example:
Object a; Customer b; b = (Customer) a;
When you assign a subclass to a variable having a supeclass type, the casting is performed automatically.

*Q7. What's the difference between the methods sleep() and wait()

A. The code sleep(1000); puts thread aside for exactly one second. The code wait(1000), causes a wait of up to one second. A thread could stop waiting earlier if it receives the notify() or notifyAll() call. The method wait() is defined in the class Object and the method sleep() is defined in the class Thread.

*Q8. Can you write a Java class that could be used both as an applet as well as an application?

A. Yes. Add a main() method to the applet.

*Q9. What's the difference between constructors and other methods?

A. Constructors must have the same name as the class and can not return a value. They are only called once while regular methods could be called many times.

*Q10. Can you call one constructor from another if a class has multiple constructors

A. Yes. Use this() syntax.

*Q11. Explain the usage of Java packages.

A. This is a way to organize files when a project consists of multiple modules. It also helps resolve naming conflicts when different packages have classes with the same names. Packages access level also allows you to protect data from being used by the non-authorized classes.

*Q12. If a class is located in a package, what do you need to change in the OS environment to be able to use it?

A. You need to add a directory or a jar file that contains the package directories to the CLASSPATH environment variable. Let's say a class Employee belongs to a package com.xyz.hr; and is located in the file c:\dev\com\xyz\hr\Employee.java. In this case, you'd need to add c:\dev to the variable CLASSPATH. If this class contains the method main(), you could test it from a command prompt window as follows:
c:\>java com.xyz.hr.Employee

*Q13. What's the difference between J2SDK 1.5 and J2SDK 5.0?

A.There's no difference, Sun Microsystems just re-branded this version.

*Q14. What would you use to compare two String variables - the operator == or the method equals()?

A. I'd use the method equals() to compare the values of the Strings and the == to check if two variables point at the same instance of a String object.

*Q15. Does it matter in what order catch statements for FileNotFoundException and IOExceptipon are written?

A. Yes, it does. The FileNoFoundException is inherited from the IOException. Exception's subclasses have to be caught first.

*Q16. Can an inner class declared inside of a method access local variables of this method?

A. It's possible if these variables are final.

*Q17. What can go wrong if you replace && with & in the following code:
String a=null; if (a!=null && a.length()>10) {...}
A. A single ampersand here would lead to a NullPointerException.

*Q18. What's the main difference between a Vector and an ArrayList

A. Java Vector class is internally synchronized and ArrayList is not.

*Q19. When should the method invokeLater()be used?

A. This method is used to ensure that Swing components are updated through the event-dispatching thread.

*Q20. How can a subclass call a method or a constructor defined in a superclass?

A. Use the following syntax: super.myMethod(); To call a constructor of the superclass, just write super(); in the first line of the subclass's constructor

For senior-level developers:

**Q21. What's the difference between a queue and a stack?

A. Stacks works by last-in-first-out rule (LIFO), while queues use the FIFO rule

**Q22. You can create an abstract class that contains only abstract methods. On the other hand, you can create an interface that declares the same methods. So can you use abstract classes instead of interfaces?

A. Sometimes. But your class may be a descendent of another class and in this case the interface is your only option.

**Q23. What comes to mind when you hear about a young generation in Java?

A. Garbage collection.

**Q24. What comes to mind when someone mentions a shallow copy in Java?

A. Object cloning.

**Q25. If you're overriding the method equals() of an object, which other method you might also consider?

A. hashCode()

**Q26. You are planning to do an indexed search in a list of objects. Which of the two Java collections should you use: ArrayList or LinkedList?

A. ArrayList

**Q27. How would you make a copy of an entire Java object with its state?

A. Have this class implement Cloneable interface and call its method clone().
**Q28. How can you minimize the need of garbage collection and make the memory use more effective?

A. Use object pooling and weak object references.
**Q29. There are two classes: A and B. The class B need to inform a class A when some important event has happened. What Java technique would you use to implement it?

A. If these classes are threads I'd consider notify() or notifyAll(). For regular classes you can use the Observer interface.

*Q30. What access level do you need to specify in the class declaration to ensure that only classes from the same directory can access it?

A. You do not need to specify any access level, and Java will use a default package access level.
Q - The Java read() method reads and returns a single byte from the standard input device. It
stores that byte according to what type. What does the method return if the user enters an eof?

A - The Java read() method reads and returns a single byte from the standard input device and stores that byte
in an integer. It returns an integer value of -1 if the user enters an eof.

Q - What keystroke combination can be used to simulate an eof at the keyboard of a DOS system?

A - An eof can be simulated on a DOS system keyboard by holding down the ctrl key and pressing the z key.

Q - Provide a Java code fragment illustrating how you would read a stream of bytes from the
standard input device until encountering an eof and quit reading when the eof is encountered.

A - The following Java code fragment will read a stream of bytes from the standard input device until
encountering an eof.

```
 while(System.in.read() != -1) { //do something }
```

This code fragment accesses the read()method of the object referred to by the class variable named in of the
class named System.

Q - Provide a Java code fragment illustrating two different ways to display a String argument on the
Java standard output device. Explain how your code works in object-oriented terms. Make certain
that you explain the difference between the two.

A - The following two code fragments will each display a string argument on the Java standard output device.

```
 System.out.println("String argument")

 System.out.print("String argument")
```

In the first case, the code fragment accesses the println() method of the object referred to by the class variable named out of the class named System. In the second case, the print() method is accessed instead of the println() method.

The difference between the two is that the println() method automatically inserts a newlineat the end of the string argument whereas the print() method leaves the display cursor at the end of the string argument.
===============
OOP

Q - In Object-Oriented Programming, an object is often said to be an ____________ of a class.

A - An object is often said to be an instance of a class.

Q - In Object-Oriented Programming, an object is often said to have s_______ and b_______.
Provide the missing words which begin with the letters shown.

A - In OOP, an object is often said to have state and behavior.

Q - An object's state is contained in its ________ and its behavior is implemented through its ________.
A - An object's state is contained in its member variables (or data members) and its behavior is implemented through its methods (or member functions).

Q - The member variables of an object can be either ____________ or _________ .

A - Its member variables can be either instance variables or class variables.

Q - What is generally meant by the terminology "sending a message to an object?"

A - We activate the behavior of an object by invoking one of its methods (sending it a message).

Q - What are the two things that can usually happen when an object receives a message?

A - When an object receives a message, it usually either performs an action, or modifies its state, or both.

Q - What happens to the memory occupied by an object in Java when the object is no longer needed, and what do we normally do to make that happen?

A - When an object is no longer needed in Java, we simply forget it. Eventually, the garbage collector may (or may not) come by and pick it up for recycling.

Q - Identify as the stages of an object's life?

A - The stages of an Object's life are:

Creation
Use
Cleanup

Q - The creation of an object involves three steps (which are often combined). What are the three steps?

A - The three steps are:

declaration (providing a name for the object)
instantiation (setting aside memory for the object)
optional initialization (providing initial values for the object's instance variables)

Q - Java allows the instantiation of variables of primitive types in dynamic memory: True or False?
If false, explain why and what you might be able to do to achieve almost the same result.

A - False. Java does not allow the instantiation of primitive variables in dynamic memory. (However, there are wrapper classes for primitive types which can be used to turn them into objects for this purpose.)

Q - An array of objects in Java is instantiated as an array of reference variables where each reference variable can then be used to instantiate an object pointed to by the reference variable:
True or False. If false, explain why and either provide a code fragment that illustrates your answer

A - True.

Q - In Java, it is always necessary to declare (give a name to) all new objects: True or False? If false, explain why and either provide a code fragment that illustrates your answer

A - It is not always necessary in Java to declare an object (to give it a name). Consider, for example a case where a new object is instantiated to be used in an expression and there is no requirement to be able to access that object outside of the expression.

Q - Instance variables and instance methods can be accessed using an object as the access mechanism. What is the difference in the syntax used to access an instance variable and an instance method.

A - None. There is essentially no difference in the syntax used to access a variable or a method.

Q - Once you have instantiated an object, it is always possible to access all of the instance variables and instance methods of that object by joining the name of the object to the name of the variable or method using a period: True or False? If false, explain why.

A - False. Sometimes variables or methods may be hidden so as to make it impossible to access them. It is very common to hide the variables and to provide methods which can be accessed to serve as a pathway to the variables.

Q - Given an object named obj that has a public instance method named myMethod(), provide a code fragment that shows the proper syntax for accessing the method.

A - The proper syntax for accessing an instance method named myMethod() follows:

```
obj.myMethod()
```

Q - The object-oriented approach normally recommends hiding instance variables behind access methods: True or False? If true, explain why.

A - True. The object-oriented approach normally recommends hiding instance variables behind access methods. There are a variety of reasons why. One important reason is that hiding the instance variables makes it possible to later modify the implementation of instance variables to improve the behavior of objects of the class, without a requirement for modifying code that uses the clsss, provided that the access methods are not modified.

Q - The returning of memory (to the operating system) occupied by objects that are no longer needed is automatically accomplished in Java by a feature commonly known as the ____________________.

A - The returning of memory to the operating system is taken care of automatically by a feature of Java known as the garbage collector.

Q - All necessary cleanup in a Java program is performed automatically by the garbage collector: True or False? If false, explain why.

A - False. Java does not support anything like a destructor that is guaranteed to be called whenever the object is no longer needed. Therefore, other than returning allocated memory, it is the responsibility of the programmer to explicitly perform any other required cleanup at the appropriate point in time.

Q - When does an object become eligible for garbage collection?

A - An object becomes eligible for garbage collection when there are no more references to that object.

Q - What can your program do to purposely make an object eligible for garbage collection.

A - Your program can make an object eligible for garbage collection by assigning null to all references to the object.

Q - The purpose of garbage collection in Java is to perform all necessary cleanup and the memory occupied by objects that are no longer needed will always be reclaimed by the garbage collector: True or False? If false, explain why.

A - False. The sole purpose of garbage collection is to reclaim memory occupied by objects that are no longer needed, and it has no other purpose relative to necessary cleanup. An object becomes eligible for garbage collection when there are no more references to that object. However, just because an object is eligible for garbage collection doesn't mean that it will be reclaimed. The garbage collector runs in a low-priority thread, and may not run at all unless a memory shortage is detected.

Q - Before the garbage collector reclaims the memory occupied by an object, it always calls the object's _________ method. (Provide the name of the method.) Explain why this method is one of the methods in all new classes that you define.

A - Before the garbage collector reclaims the memory occupied by an object, it calls the object's finalizemethod. The finalize method is a member of the Object class. Since all classes inherit from the Object class, your classes also contain the default finalize method.

Q - What must you do to make effective use of the finalize method? Explain why you might want to do this.

A - In order to make use of the finalize method, you must override it, providing the code that you want to have executed before the memory is reclaimed.

Q - You can always be confident that the finalize method will be promptly executed to perform necessary cleanup in a Java program when an object becomes eligible for garbage collection: True or False? If false, explain why.

A - False. Although you can be confident that the finalize method will be called before the garbage collector reclaims the memory occupied by a particular object, you cannot be certain when, or if that memory will be reclaimed. There is no guarantee that the memory will be reclaimed by the garbage collector during the execution of your program.

Q - Provide a code fragment illustrating the method call that you can make to ask the garbage collector to run. This guarantees that garbage collection will take place: True or False? If false, explain why.

A - False. Campione and Walrath indicate that you can ask the garbage collector to run at any time by calling the method shown below. They further point out, however, that making the request does not guarantee that your objects will be reclaimed through garbage collection.   System.gc();

======

Q 1What do we call an operator that operates on only one operand?
A - An operator that operates on only one operand is called a unary operator.

Q 2- What do we call an operator that operates on two operands?
A - An operator that operates on two operands is called a binary operator.

Q 3- Is the minus sign a unary or a binary operator, or both? Explain your answer.
A - Both. As a binary operator, the minus sign causes its right operand to be subtracted from its left operand. As a unary operator, the minus sign causes the algebraic sign of the right operand to be changed.

Q 4- Describe operator overloading.
A - For those languages that support it (such as C++) operator overloading means that the programmer can redefine the behavior of an operator with respect to objects of a new type defined by that program.

Q5- Java programmers may overload operators: True or False?
A 6- False: Unfortunately, Java does not support operator overloading.

Q7 - Show the symbols used for the following operators in Java: assignment, not equal, addition,cast.
A - The above listed operators in order are: = != + (type)

Q8 - Is any operator automatically overloaded in Java? If so, identify it and describe its overloaded behavior.
A - The plus sign (+) is automatically overloaded in Java. The plus sign can be used to perform arithmetic addition. It can also be used to concatenate strings. However, the plus sign does more than concatenate strings. It also performs a conversion to String type. When the plus sign is used to concatenate strings, the operand on the right is automatically converted to a character string before being concatenated with the operand on the left. This assumes that the compiler knows enough about the operand on the right to be able to successfully perform the conversion. It has that knowledge for all of the primitive types and most or all of the built-in reference types.

Q 9- What is the purpose of the cast operator?
A - The cast operator is used to purposely convert from one type to another.

Q10 - The increment operator is a binary operator: True or False?

A - False: The increment operator is a unary operator.

Q11 - Show the symbol for the increment operator.
A - The symbol for the increment operator is two plus signs with nothing between them (++).

Q 12- Describe the appearance and the behavior of the increment operator with both prefix and postfix notation. Show example, possibly incomplete, code fragments illustrating both notational forms.
A - The increment operator may be used with both prefix and postfix notation. Basically, the increment operator causes the value of the variable to which it is applied to be increased by one. With prefix notation, the operand appears to the right of the operator (++X), while with postfix notation, the operand appears to the left of the operator (X++).
The difference in behavior has to do with the point in time that the increment actually occurs if the operator and its operand appear as part of a larger overall expression. With the prefix version, the variable is incremented before it is used to evaluate the larger overall expression. With the postfix version, the variable is used to evaluate the larger overall expression and then it is incremented.

Q13 - Show the output that would be produced by the following Java application.

```
 class prg1 { //define the controlling class
   public static void main(String[] args){ //define main method
     int x = 5, X = 5, y = 5, Y = 5;
     System.out.println("x = " + x );

     System.out.println("X = " + X );
     System.out.println("x + X++ = " + (x + X++) );
     System.out.println("X = " + X );
     System.out.println();
     System.out.println("y = " + y );
     System.out.println("Y = " + Y );
     System.out.println("y + ++Y = " + (y + ++Y) );
     System.out.println("Y = " + Y );
   }//end main
 }//End  class.  Note no semicolon required
 //End Java application
```

A - The output from this Java application follows:

```
x = 5
X = 5
x + X++ = 10
X = 6
y = 5
Y = 5
y + ++Y = 11
Y = 6
```

Q 14 - Binary operators use outfix notation: True or False? If your answer is False, explain why.
A - False: Binary operators use infix notation, which means that the operator appears between its operands.

Q15 - In practice, what does it mean to say that an operator that has performed an action returns a value (or evaluates to a value) of a given type?
A - As a result of performing the specified action, an operator can be said to return a value (or evaluate to a value) of a given type. The type depends on the operator and the type of the operands. To evaluate to a value means that after the action is performed, the operator and its operands are effectively replaced in the expression by the value that is returned.

Q 16 - What are the four categories of operators described in Baldwin's Java tutorial on operators? Do you agree with this categorization? If not, explain why not.
A - Some authors divide Java's operators into the following categories:
    arithmetic
    relational and conditional (typically called relational and logical in C++)
    bitwise and logical
    assignment

Q17 - Show and describe at least five of the binary arithmetic operators supported by Java (Clarification: binary operators does not mean bitwise operators).
A - Java support various arithmetic operators on floating point and integer numbers. The following table lists five of the binary arithmetic operators supported by Java.

Operator	Description
+	Adds its operands

-	Subtracts the right operand from the left operand
*	Multiplies the operands
/	Divides the left operand by the right operand
%	Remainder of dividing the left operand by the right operand

Q18 - In addition to arithmetic addition, what is another use for the plus operator (+)? Show an example code fragment to illustrate your answer. The code fragment need not be a complete statement.
A - The plus operator (+) is also used to concatenate strings : "IMR " + " global Ltd."

Q19 - When the plus operator (+) is used as a concatenation operator, what is the nature of its behavior if its right operand is not of type String? If the right operand is a variable that is not of type String, what is the impact of this behavior on that variable.
A - In this case, the operator also coerces the value of the right operand to a string representation for use in the expression only. If the right operand is a variable, the value stored in the variable is not modified in any way.

Q 20- Show and describe four unary arithmetic operators supported by Java.
A - Java supports the following four unary arithmetic operators.

Operator	Description
+	Indicates a positive value
-	Negates, or changes algebraic sign
++	Adds one to the operand, both prefix and postfix
--	Subtracts one from operand, prefix and postfix

Q20 - What is the type returned by relational operators in Java?
A - Relational operators return the boolean type in Java.

Q 21- Show and describe six different relational operators supported by Java.
A - Java supports the following set of relational operators:

Operator	Returns true if
>	Left operand is greater than right operand
>=	Left operand is greater than or equal to right operand
<	Left operand is less than right operand
<=	Left operand is less than or equal to right operand
==	Left operand is equal to right operand
!=	Left operand is not equal to right operand

Q 22- Show the output that would be produced by the following Java application:

```
 class prg2 { //define the controlling class
   public static void main(String[] args){ //define main method
     System.out.println("The relational 6<5 is " + (6<5 ) );
     System.out.println("The relational 6>5 is " + (6>5 ) );
   }//end main
 }//End  class.  Note no semicolon required
 //End Java application
```

A - This program produces the following output:

```
The relational 6<5 is false
The relational 6>5 is true
```

Q23 - Show and describe three operators (frequently referred to as conditional operators in Java and logical operators in C++) which are often combined with relational operators to construct more complex expressions (often called conditional expressions). Hint: The && operator returns true if the left and right operands are both true. What are the other two and how do they behave?
A - The following three logical or conditional operators are supported by Java.

Operator	Typical Use	Returns true if
&&	Left && Right	Left and Right are both true
||	Left || Right	Either Left or Right is true
!	! Right	Right is false

Q24 - Describe the special behavior of the || operator in the following expression for the case where the value of the variable a is less than the value of the variable b.
(a < b) || (c < d)
A - An important characteristic of the behavior of the && and || operators in Java is that the expressions are evaluated from left to right, and the evaluation of the expression is terminated as soon as the result of evaluating the expression can be determined. For example, in the above expression, if the variable a is less than the variable b , there is no need to evaluate

the right operand of the || to determine the value of the entire expression. Thus, evaluation will terminate as soon as it is determined that a is less than b.

Q25 - Show the symbols used for the bitwise and operator and the or operator.
A - The & operator in Java is a bitwise and and the | operator is a bitwise or.

Q26 - The logical and operator is represented in Java by the && symbol. What is the representation of the bitwise and operator in Java?
A - The bitwise and operator is represented by the & symbol in Java.

Q27 - Show and describe five operators in Java that perform actions on the operands one bit at a time (bitwise operators).
A - The following table shows the seven bitwise operators supported by Java.

Operator	Typical Use	Operation
>>	OpLeft >> Dist	Shift bits of OpLeft right by Dist bits (signed)
<<	OpLeft << Dist	Shift bits of OpLeft left by Dist bits
>>>	OpLeft >>> Dist	Shift bits of OpLeft right by Dist bits (unsigned)
&	OpLeft & OpRight	Bitwise and of the two operands
|	OpLeft | OpRight	Bitwise inclusive or of the two operands
^	OpLeft ^ OpRight	Bitwise exclusive or (xor) of the two operands
~	~ OpRight	Bitwise complement of the right operand (unary)

Q28 - In Java, the signed right shift operation populates the vacated bits with the zeros, while the left shift and the unsigned right shift populate the vacated bits with the sign bit: True or False. If your answer is False, explain why.
A - False: In Java, the signed right shift operation populates the vacated bits with the sign bit, while the left shift and the unsigned right shift populate the vacated bits with zeros.

Q29 - In a signed right-shift operation in Java, the bits shifted off the right end are lost: True or False. If your answer is False, explain why.
A - True: For both Java and C++, bits shifted off the right end are lost.

Q30 - Using the symbols 1 and 0 construct a table showing the four possible combinations of 1 and 0. Using a 1 or a 0, show the result of the bitwise and operation on these four combinations of 1 and 0.
A - The answer is:
1 and 1 produces 1
1 and 0 produces 0
0 and 1 produces 0
0 and 0 produces 0

Q 31- Using the symbols 1 and 0 construct a truth table showing the four possible combinations of 1 and 0. Using a 1 or a 0, show the result of the bitwise inclusive or operation on these four combinations on these four combinations of 1 and 0.
A - The answer for the inclusive or is:
1 or 1 produces 1
1 or 0 produces 1
0 or 1 produces 1
0 or 0 produces 0

Q 32- Using the symbols 1 and 0 construct a truth table showing the four possible combinations of 1 and 0. Using a 1 or a 0, show the result of the bitwise exclusive or operation on these four combinations on these four combinations of 1 and 0.
A - The answer for the exclusive or is:
1 xor 1 produces 0
1 xor 0 produces 1
0 xor 1 produces 1
0 xor 0 produces 0

Q 33- For the exclusive or, if the two bits are different, the result is a 1. If the two bits are the same, the result is a 0. True or False? If your answer is False, explain why.
A - True.

Q 34- Is the assignment operator a unary operator or a binary operator. Select one or the other.
A - The assignment operator is a binary operator.

Q 35- In Java, when using the assignment operator, the value stored in memory and represented by the

right operand is copied into the memory represented by the left operand: True or False? If your answer is False, explain why.
A - True.

Q 36- Show two of the shortcut assignment operators and explain how they behave by comparing them with the regular (nonshortcut) versions. Hint: The (^=) operator is a shortcut assignment operator.
A - Java supports the following list of shortcut assignment operators. These operators allow you to perform an assignment and another operation with a single operator. += -= *= /= %= &= |= ^= <<= >>= >>>=
For example, the two statements which follow perform the same operation.

```
 x += y;     x = x + y;
```

The behavior of all the shortcut assignment operators follows this same pattern.

Q 37 - Write a Java application illustrates the difference between the prefix and the postfix versions of the increment operator.

```
 class prog3{
   static public void main(String[] args){
     int x = 3;
     int y = 3;
     int z = 10;
     System.out.println("Prefix version gives  " + (z + ++x));
     System.out.println("Postfix version gives " + (z + y++));
   }//end main
 }//end class
```

Q38 Write a Java application that illustrates the use of the following relational operators: < > <= >= == !=

```
class Prog4 { //define the controlling class
   public static void main(String[] args){ //define main method
     System.out.println("The relational 6<5 is " + (6<5 ) );
     System.out.println("The relational 6>5 is " + (6>5 ) );
     System.out.println("The relational 5>=5 is " + (5>=5 ) );
     System.out.println("The relational 5<=5 is " + (5<=5 ) );
     System.out.println("The relational 6==5 is " + (6==5 ) );
     System.out.println("The relational 6!=5 is " + (6!=5 ) );
   }//end main
 }//End prog4 class.  Note no semicolon required
```

Q39 - Write a Java application that illustrates the use of the following logical or conditional operators:
&& || !

```
 class prg5 { //define the controlling class
   public static void main(String[] args){ //define main method
     System.out.println("true and true is " + (true && true) );
     System.out.println("true and false is " + (true && false) );

     System.out.println("true or true is " + (true || true) );
     System.out.println("true or false is " + (true || false) );
     System.out.println("false or false is " + (false || false) );
     System.out.println("not true is " + (! true) );
     System.out.println("not false is " + (! false) );
}//end main
}
```

Q40 - Java supports a constant type: True or False. If false, explain why.
A - Java does not support a constant type. However, in Java, it is possible to achieve the same result by declaring and initializing a variable and making it final.

Q41 Provide a code fragment that illustrates the syntax for creating a named constant in Java.
A - The syntax for creating a named constant in Java is as follows: final float PI = 3.14159;

Q42 - What is the common method of controlling the order of evaluation of expressions in Java?
A - you can control the order of evaluation by the use of parentheses.

Q43 - What are the three actions normally involved in the operation of a loop (in addition to executing the code in the body of the loop)?

A - The operation of a loop normally involves the following three actions in addition to executing the code in the body of the loop:

Initialize a control variable.
Test the control variable in a conditional expression.
Update the control variable.

Strings

Q - Java provides two different string classes from which string objects can be instantiated. What are they?

A - The two classes are:

String
StringBuffer

Q - The StringBuffer class is used for strings that are not allowed to change. The String class is used for strings that are modified by the program: True or False. If false, explain why.

A - False. This statement is backwards. The String class is used for strings that are not allowed to change. The StringBuffer class is used for strings that are modified by the program.

Q - While the contents of a String object cannot be modified, a reference to a String object can be caused to point to a different String object: True or False. If false, explain why.

A - True.

Q - The use of the new operator is required for instantiation of objects of type String: True or False? If false, explain your answer.

A - False. A String object can be instantiated using either of the following statements:

```
String str1 = new String("String named str2");



String str2 = "String named str1";
```

Q - The use of the new operator is required for instantiation of objects of type StringBuffer: True or False? If false, explain your answer.

A - True.

Q - Provide a code fragment that illustrates how to instantiate an empty StringBuffer object of a default length and then use a version of the append() method to put some data into the object.

A - See code fragment below:

```
StringBuffer str5 = new StringBuffer();//accept default initial length

str5.append("StringBuffer named str5");//modify length as needed
```

Q - Without specifying any explicit numeric values, provide a code fragment that will instantiate an empty StringBuffer object of the correct initial length to contain the string "StringBuffer named str6" and then store that string in the object.

A - See the following code fragment:

```
StringBuffer str6 = new StringBuffer("StringBuffer named str6".length());

str6.append("StringBuffer named str6");
```

Q - Provide a code fragment consisting of a single statement showing how to use the Integer wrapper class to convert a string containing digits to an integer and store it in a variable of type int.

A - See code fragment below

```
    int num = new Integer("3625").intValue();
```

Q - Explain the difference between the capacity() method and the length() methods of the StringBuffer class.

A - The capacity() method returns the amount of space currently allocated for the StringBuffer object. The length() method returns the amount of space used.

Q - The following is a valid code fragment: True or False? If false, explain why.

```
 StringBuffer str6 = new StringBuffer("StringBuffer named str6".length());
```

A - True.

Q - Which of the following code fragments is the most efficient, first or second?

```
 String str1 = "THIS STRING IS NAMED str1";
 String str1 = new String("THIS STRING IS NAMED str1");
```

A - The first code fragment is the most efficient.

System

Java provides the System class which provides a platform-dependent interface between your program and various system resources: True or False? If false, explain why.

A - False. Java provides the System class which provides a platform-independent interface between your program and those resources.

Q - You must instantiate an object of the System class in order to use it: True or False? If false, explain why.

A - False. You don't need to instantiate an object of the System class to use it, because all of its variables and methods are class variables and methods.

Q - The following code fragment can be used to instantiate an object of the System class: True or False? If false, explain why.

```
 System mySystemObject = new System();
```

A - False. You cannot instantiate an object of the System class. It is a final class, and all of its constructors are private.

Q - What is the purpose of the write() method of the PrintStream class?

A - The write() method is used to write bytes to the stream. You can use write() to write data which is not intended to be interpreted as text (such as bit-mapped graphics data).

Exceptions

Q - The exception-handling capability of Java makes it possible for you to monitor for exceptional conditions within your program, and to transfer control to special exception-handling code which you design. List five keywords that are used for this purpose.

A - try, throw, catch, finally, and throws

Q - All exceptions in Java are thrown by code that you write: True or False? If false, explain why.

A - False. There are situations where an exceptional condition automatically transfers control to special exception-handling code which you write (cases where you don't provide the code to throw the exception object).

Q - When an exceptional condition causes an exception to be thrown, that exception is an object derived, either directly, or indirectly from the class Exception: True or False? If false, explain why.

A - False. When an exceptional condition causes an exception to be thrown, that exception is an object derived, either directly, or indirectly from the class Throwable.

Q - All exceptions other than those in the RuntimeException class must be either caught, or declared in a throws clause of any method that can throw them: True or False? If false, explain why.

A - True.

Q - What method of which class would you use to extract the message from an exception object?

A - The getMessage() method of the Throwable class.

Q - Normally, those exception handlers designed to handle exceptions closest to the root of the exception class hierarchy should be placed first in the list of exception handlers: True or False? If false, explain why.

A - False. The above statement has it backwards. Those handlers designed to handle exceptions furthermost from the root of the hierarchy tree should be placed first in the list of exception handlers.

Q - Explain why you should place exception handlers furthermost from the root of the exception hierarchy tree first in the list of exception handlers.

A - An exception hander designed to handle a specialized "leaf" object may be preempted by another handler whose exception object type is closer to the root of the exception hierarchy tree if the second exception handler appears earlier in the list of exception handlers.

Q - In addition to writing handlers for very specialized exception objects, the Java language allows you to write general exception handlers that handle multiple types of exceptions: True or False? If false, explain why.

A - True.

Q - Your exception handler can be written to handle any class that inherits from Throwable. If you write a handler for a node class (a class with no subclasses), you've written a specialized handler: it will only handle exceptions of that specific type. If you write a handler for a leaf class (a class with subclasses), you've written a general handler: it will handle any exception whose type is the node class or any of its subclasses. True or False? If false, explain why.

A - False. "Leaf" and "node" are reversed in the above statement. If you write a handler for a "leaf" class (a class with no subclasses), you've written a specialized handler: it will only handle exceptions of that specific type. If you write a handler for a "node" class (a class with subclasses), you've written a general handler: it will handle any exception whose type is the node class or any of its subclasses."

Q - Java's finally block provides a mechanism that allows your method to clean up after itself regardless of what happens within the try block. True or False? If false, explain why.

A - True.

Q - Explain how you would specify that a method throws one or more exceptions.

A - To specify that a method throws one or more exceptions, you add a throws clause to the method signature for the method. The throws clause is composed of the throws keyword followed by a comma-separated list of all the exceptions thrown by that method.

Q - Provide a code fragment that illustrates how you would specify that a method throws more than one exception.

A - See code fragment below.

```
 void myMethod() throws InterruptedException, MyException,

      HerException, UrException

 {

 //method body

 }
```

Q - What type of argument is required by the throw statement?

A - The throw statement requires a single argument, which must be an object derived either directly or indirectly from the class Throwable.

Q - Some exception objects are automatically thrown by the system. It is also possible for you to define your own exception classes, and to cause objects of those classes to be thrown whenever an exception occurs. True or False? If false, explain why.

A - True.

=======

Threads

Q - What is the definition of multi-threaded programming according to Patrick Naughton?

A - According to The Java Handbook, by Patrick Naughton,

"Multi-threaded programming is a conceptual paradigm for programming where you divide programs into two or more processes which can be run in parallel."

Q - Multithreading refers to two or more programs executing, "apparently" concurrently, under control of the operating system. The programs need have no relationship with each other, other than the fact that you want to start and run them all concurrently. True or False? If false, explain why.

A - False. That is a description of multiprocessing, not multithreading. Multithreading refers to two or more tasks executing, "apparently" concurrently, within a single program.

Q - According to current terminology, the term blocked means that the thread is waiting for something to happen and is not consuming computer resources. True or False? If false, explain why.

A - True.

Q - What are the two ways to create threaded programs in Java?

A - In Java, there are two ways to create threaded programs:

Implement the Runnable interface
Extend the Thread class

Q - What two steps are required to spawn a thread in Java?

A - The two steps necessary to spawn a thread in Java are:

instantiate an object of type Thread and
invoke its run() method.

Q - How do you start a thread actually running in Java?

A - Invoke the start() method on object of the Thread class or of a subclass of the Thread class.

Q - It is always possible to extend the Thread class in your Java applications and applets. True or False? If false, explain why.

A - False. Sometimes it is not possible to extend the Thread class, because you must extend some other class and Java does not support multiple inheritance.

Q - Although multithreaded programming in Java is possible, it is also possible to write Java programs that do not involve threads: True or False? If false, explain why.

A - False. The main method itself runs in a thread which is started by the interpreter.

Q - What is the name of the method that can be used to determine if a thread is alive?

A - The name of the method is isAlive().

Q - Once you start two or more threads running, unless you specify otherwise, they run synchronously and independently of one another: True or False? If false, explain why.

A - False. Once you start two or more threads running, unless you specify otherwise, they run asynchronously and independently of one another.

Q - The process of keeping one thread from corrupting the data while it is being processed by another thread is known as synchronization: True or False? If false, explain why.

A - True.

Q - Java allows you to specify the absolute priority of each thread: True or False? If false, explain why.

A - False. Java allows you to specify the priority of each thread relative to other threads but not on an absolute basis.

Q - Thread synchronization can be achieved using wait(), notify(), and notifyAll() which are methods of the Thread class: True or False? If false, explain why.

A - False. wait(), notify(), and notifyAll() are not methods of the Thread class, but rather are methods of the Object class.

Q - When you implement a threaded program, you will always override the _____________ method of the Thread class and build the functionality of your threaded program into that method. What is the name of the method?

A - The run() method.

Q - In a multithreaded program, you will start a thread running by invoking the __________ method on your Thread object which will in turn invoke the ___________ method. What are the names of the missing methods, and what are the required parameters for each method?

A - In a multithreaded program, you will start a thread running by invoking the start() method on your Thread object which will in turn invoke the run() method. Neither method takes any parameters.

Q - What do Campione and Walrath list as the four possible states of a thread?

A -
  New Thread
  Runnable
  Not Runnable
  Dead

Q - What methods can be invoked on a thread object which is in the state that Campione and Walrath refer to as a New Thread and what will happen if you invoke any other method on the thread?

A - When a thread is in this state, you can only start the thread or stop it. Calling any method other than start() or stop() will cause an IllegalThreadStateException.

Q - What, according to Campione and Walrath, will cause a thread to become Not Runnable?

A - a thread becomes Not Runnable when one of the following four events occurs:

Someone invokes its sleep() method.
Someone invokes its suspend() method.
The thread uses its wait() method to wait on a condition variable.
The thread is blocking on I/O.

Q1 - Three keywords are used in Java to specify access control. What are they?
A - The three keywords used to specify access control in Java are public, private, and protected.

Q2 - In Java, special access privileges are afforded to other members of the same package: True or False? If false, explain your answer.
A - True. In Java, special access privileges are afforded to other members of the same package.

Q3 - In Java, class variables are often used with the __________ keyword to create variables that act like constants.

A - The final keyword.

Q4 - In Java, the ___________ keyword is used to declare a class variable.

A - The static keyword.

Q5 - In Java, the ___________ keyword is used to declare a class method.

A - The static keyword.

Q6 - When you include a method in a Java class definition without use of static keyword, this will result in objects of that class containing an instance method: True or False? If false, explain why.

A 7- True.

Q8 - Normally each object contains its own copy of each instance method: True or False?

A - False, multiple copies of the method do not normally exist in memory.

Q9- When you invoke an instance method using a specific object, if that method refers to instance variables of the class, that method is caused to refer to the specific instance variables of the specific object for which it was invoked: True or False? If false, explain why.

A - True.

Q10 - Instance methods are invoked in Java using the name of the object, the colon, and the name of the method as shown below: True or False? If false, explain why.

```
myObject:myInstanceMethod( )
```

A - False. Use the period or dot operator, not the colon.

Q11 - Instance methods have access to both instance variables and class variables in Java: True or False. If false, explain why.

A - True.

Q12 - Class methods have access to both instance variables and class variables in Java: True or False. If false, explain why.

A - False. Class method can only access other class members.

Q13 - What are the two most significant characteristics of class methods?

A - 1. Class methods can only access other class members. 2. Class methods can be accessed using only the name of the class. An object of the class is not required to access class methods.

Q14 - In Java, a class method can be invoked using the name of the class, the colon, and the name of the method as shown below: True or False? If false, explain why.

```
MyClass:myClassMethod()
```

A - False. You must use the period or dot operator, not the colon.

Q15 - What is meant by overloaded methods?

A - The term overloaded methods means that two or more methods may have the same name so long as they have different argument lists.

Q16 - If you overload a method name, the compiler determines at run time, on the basis of the arguments provided to the invocation of the method, which version of the method to call in that instance: True or False? If false, explain why.

A - False. The determination is made at compile time.

Q16 - A constructor is a special method which is used to construct an object. A constructor always has the same name as the class in which it is defined, and has no return type specified. True or False? If false, explain why.

A - True.

Q17 - Constructors may be overloaded, so a single class may have more than one constructor, all of which have the same name, but different argument lists: True or False. If false, explain why.

A - True

Q18 - What is the purpose of a parameterized constructor?

A - The purpose of a parameterized constructor is to initialize the instance variables of an object when the object is instantiated.

Q 19 - The same set of instance variables can often be initialized in more than one way using overloaded constructors: True or False? If false, explain why.

A - True.

Q20 - It is not necessary to provide a constructor in Java. True or False? If false, explain why.

A - True. .

Q 21 You can think of the default constructor as a constructor which doesn't take any parameters: True or False? If false, explain why.

A - True.

Q 22- In Java, if you provide any constructors, the default constructor is no longer provided automatically: True or False? If false, explain why.

A - True.

Q 23- In Java, if you need both a parameterized constructor and a constructor which doesn't take parameters (often called a default constructor), you must provide them both: True or False? If false, explain why.

A - True.

Q 24- In Java, you can instantiate objects in static memory at compile time, or you can use the new operator to request memory from the operating system at runtime and use the constructor to instantiate the object in that memory: True or False? If false, explain why.

A - False. In Java, objects can only be instantiated on the heap at runtime.

Q25 - Provide a code fragment consisting of a single statement that illustrates how the constructor is typically used in Java to declare, instantiate, and initialize an object. Assume a parameterized constructor with three parameters of type int.

A - MyClass myObject = new MyClass(1,2,3);

Q26 - Provide a code fragment consisting of a single statement that illustrates how the default constructor is typically used in Java to declare and instantiate an object.

A - MyClass myObject = new MyClass();

Q 27- What are the three actions performed by the following statement?

MyClass myObject = new MyClass(1,2,3);

A - This statement performs three actions in one.

The object is declared by notifying the compiler of the name of the object.
The object is instantiated by using the new operator to allocate memory space to contain the new object.
The object is initialized by making a call to the constructor named MyClass.

Q 28- In Java, if you attempt to instantiate an object and the Java Virtual Machine cannot allocate the requisite memory, the system will: ____.

A - Throw an OutOfMemoryError.

Q 29- The following is a valid method call: True or False. If false, explain why.

obj.myFunction(new myClassConstructor(1,2,3));//Java version

A - True.

Q30 - In Java, when a method begins execution, all of the parameters are created as local automatic variables: True or False? If false, explain why.

A - True.

Q31 - In the following statement, an object is instantiated and initialized and passed as a parameter to a function. What will happen to that object when the function terminates?

obj.myFunction(new myClassConstructor(1,2,3));//Java version

A - It will become eligible for garbage collection.

Q 32- In Java, you declare and implement a constructor just like you would implement any other method in your class, except that: ____

A - you do not specify a return type and must not include a return statement.

Q33 - The name of the constructor must be the same as the name of the ___________________.

A - class.

Q 34- Usually in cases of inheritance, you will want the subclass to cause the constructor for the superclass to execute last to initialize those instance variables which derive from the superclass: True or False? If false, explain why.

A - False. You will want the subclass to cause the constructor for the superclass to execute first.

Q 35- Provide a code fragment that you would include at the beginning of your constructor for a subclass to cause the constructor for the superclass to be invoked prior to the execution of the body of the constructor.

A - super(optional parameters);

Q 36- Every object has a finalize method which is inherited from the class named ________________.

A - object.

Q 37- Before an object is reclaimed by the garbage collector, the _______________ method for the object is called.

A - finalize

Q 38- In Java, the destructor is always called when an object goes out of scope: True or False? If false, explain why.

A - False. Java does not support the concept of a destructor.

=====

Q 39- The class at the top of the inheritance hierarchy is the Object class and this class is defined in the package named java.Object: True or False? If false, explain why.

A - False. The Object class is defined in the package named java.lang.

Q40 - We say that an object has state and behavior: True or False? If false, explain why.

A - True.

Q 41- In Java, a method can be defined as an empty method, normally indicating that it is intended to be overridden: True or False? If false, explain why.

A - True.

Q 42- Including an empty method in a class definition will make it impossible to instantiate an object of that class: True or False.

A - False.

Q 43- A subclass can invoke the constructor for the immediate superclass by causing the last line of of the subclass constructor to contain the keyword super followed by a parameter list as though calling a function named super() and the parameter list must match the method signature of the superclass constructor: True or False? If false, explain why.

A - False. This can only be accomplished by causing the first line of the constructor to contain the keyword super followed by a parameter list as though calling a function named super(). The parameter list must match the method signature of the superclass constructor.

Q 43 The equals() method is used to determine if two reference variables point to the same object: True or False? If false, explain why.

A- False. You can use the equals() method to compare two objects for equality. You can use the equality operator (==) to determine if two reference variables point to the same object.

Q 44- The equals() method is used to determine if two separate objects are of the same type and contain the same data. The method returns false if the objects are equal and true otherwise. True or False? If false, explain why.

A - False. The method returns true if the objects are equal and false otherwise.

Q 45 - The equals() method is defined in the Object class: True or False? If false, explain why.

A - True.

Q 46- Your classes can override the equals() method to make an appropriate comparison between two objects of a type that you define: True or False? If false, explain why.

A - True.

Q 47- You must override the equals() method to determine if two string objects contain the same data: True or False? If false, explain why.

A - False. The system already knows how to apply the equals() method to all of the standard classes and objects of which the compiler has knowledge. For example, you can already use the method to test two String objects or two array objects for equality.

Q 48- Given an object named obj1, provide a code fragment that shows how to obtain the name of the class from which obj1 was instantiated and the name of the superclass of that class.

A - See code fragment below:

```
    System.out.println("Name of class for obj1: "          + obj1.getClass().getName());

    System.out.println("Name of superclass for obj1: "         + obj1.getClass().getSuperclass());
```

Q 49- Given an object named obj2, provide a code fragment that shows how to use the newInstance() method to create a new object of the same type

A - See code fragment below:

```
        obj2 = obj1.getClass().newInstance();
```

Q 50- By overriding the getClass() method, you can use that method to determine the name of the class from which an object was instantiated: True or False? If false, explain why.

A - False. The getClass() method is a final method and cannot be overridden.

Q 51- You must use the new operator to instantiate an object of type Class: True or False? If false, explain why.

A - False. There is no public constructor for the class Class. Class objects are constructed automatically by the Java Virtual Machine as classes are loaded and or by calls to the defineClass method in the class loader.

Q 52- The Class class provides a toString() method which can be used to convert all objects known to the compiler to some appropriate string representation. The actual string representation depends on the type of object: True or False? If false, explain why.

A - False. The Object class (not the Class class) provides a toString() method which can be used to convert all objects known to the compiler to some appropriate string representation. The actual string representation depends on the type of object.

Q 53- You can override the toString() method of the Class class to cause it to convert objects of your design to strings: True or False? If false, explain why.

A - False. The toString() method is a method of the Object class, not the Class class.

Q 54- By default, all classes in Java are either direct or indirect descendants of the Class class which is at the top of the inheritance hierarchy: True or False? If false, explain why.

A - False. By default, all classes in Java are either direct or indirect descendants of the Object class (not the Class class) which is at the top of the inheritance hierarchy.

=====

Q 55 - To a limited extent, the interface concept allows you to treat a number of objects, instantiated from different classes, as if they were all of the same type: True or False? If false, explain why.

A - True.

Q 56- At its simplest level, an interface definition has a name, and declares one or more methods: True or False? If false, explain why.

A - True.

Q57- In an interface definition, both the method signatures and the actual implementations (bodies) of the methods are provided: True or False? If false, explain why.

A - False. Only the method signatures are provided. The actual implementations (bodies) of the methods are not provided.

Q 58- An interface definition can contain only method declarations: True or False? If false, explain why.

A - False. In addition to method declarations, an interface can also declare constants. Nothing else may be included inside the body of an interface definition.

Q59 - If classes P, D, and Q all implement interface X, a reference variable for an object of class P, D, or Q could be assigned to a reference variable of type X: True or False? If false, explain why.

A - True.

Q 60- If classes P, D, and Q all implement interface X, then all of the methods declared in X must be exactly the same in classes P, D, and Q: True or False? If false, explain why.

A - False. The interface simply declares the signatures for methods. Classes that implement the interface are free to provide a body for those methods which best suits the needs of the class.

Q 61- If classes P, D, and Q all implement interface X a reference variable for an object of class P, D, or Q could be assigned to a reference variable of type X and that reference variable could be used to access all of the methods of the class (which are not excluded using public, private, or protected): True or False? If false, explain why.

A - False. If one or more of the classes P, D, and Q, define instance methods which are not declared in the interface X, then a variable of type X cannot be used to access those instance methods. Those methods can only be accessed using a reference variable of the class in which the method is defined. Reference variables of the type X can only be used to access methods declared in the interface X (or one of its superinterfaces).

Q 61 - The new operator must be used to instantiate an object which is of the type of an interface: True or False? If false, explain why.

A - False. Even though you can consider the interface name as a type for purposes of storing references to objects, you cannot instantiate an object of the interface type itself.

Q 63- One of the difficulties of implementing interfaces is the requirement to coordinate the definition of interface methods among the classes that implement the interface: True or False? If false, explain why.

A - False. In defining interface methods, each class defines the methods in a manner that is appropriate to its own class without concern for how it is defined in other classes.

Q 64- As with classes, multiple interface definitions can be combined into the same source file: True or False? If false, explain why.

A- False. The compiler requires interface definitions to be in separate files.

Q 65- List four ways in which interfaces are useful:

A - See the following list:

To a limited extent, the interface concept allows you to treat a number of objects, instantiated from different classes, as if they were all of the same type
Capturing similarities between unrelated classes without forcing a class relationship

Declaring methods that one or more classes are expected to implement
Revealing an object's programming interface without revealing its class (objects such as these are called anonymous objects and can be useful when shipping a package of classes to other developers)

Q 66- A minimum interface declaration contains the Java keyword interface, the name of the interface, and the name of the interface that it extends: True or False? If false, explain why.

A - False. A minimum interface declaration contains the Java keyword interface and the name of the interface. There is no requirement to specify the name of the interface that it extends, because it may not extend another interface.

Q 67 - An interface can extend any number of other interfaces: True or False? If false, explain why.

A - True.

Q 68- Just like a class definition can extend any number of other classes, an interface can extend any number of other interfaces: True or False? If false, explain why.

A - False. A class can extend only one other class.

Q 69- An interface can extend any number of other interfaces but not more than one class: True or False? If false, explain why.

A - False. An interface cannot extend a class.

Q 70- An interface inherits all constants and methods from its superinterface: True or False? If false, explain why.

A - False. See reasons below:

An interface inherits all constants and methods from its superinterface unless:

the interface hides a constant with another of the same name, or
redeclares a method with a new method declaration.

Q 71- The method declaration in an interface consists of the method signature followed by a pair of empty curly braces: True or False? If false, explain why.

A - False. The method declaration is terminated by a semicolon and no body (no curly braces) is provided for the method.

Q 72- The keyword private is used to restrict access to the members of an interface only to classes within the same package: True or False? If false, explain why.

A - False. You may not use private in a member declaration in an interface.

Q 73- All methods declared in an interface are implicitly public and abstract: True or False? If false, explain why.

A - True.

Q 74- In addition to declaring methods, the body of the interface may also define constants. Constant values defined in an interface are implicitly public, static, and final: True or False? If false, explain why.

A - True.

Q 75- You use an interface by defining a class that extends the interface by name: True or False? If false, explain why.

A - False. You use an interface by defining a class that implements (not extends) the interface by name.

Q 76- When a class claims to implement an interface, it must provide a full definition for all the methods declared in the interface as well as all of the methods declared in all of the superinterfaces of that interface: True or False? If false, explain why.

A - True.

Q 77- A class can implement more than one interface by including several interface names in a comma-separated list of interface names, and by providing a full definition for all of the methods declared in all of the interfaces listed as well as all of the superinterfaces of those interfaces: True or False? If false, explain why.

A - True.

Q 78- Whenever a class implements an interface, it is allowed to define only those methods declared in the interface: True or False? If false, explain why.

A - False. Whenever a class implements an interface, the class must define all of the methods declared in the interface, but is also free to define other methods as well.

Q 79- The definition of an interface is a definition of a new reference data type. You can use interface names just about anywhere that you would use other type names, except that you cannot ____________________.

A - You cannot instantiate objects of the interface type.

Q 80- Explain in your own words the "bottom line" benefits of the use of an interface.

A - The interface makes it possible for a method in one class to invoke methods on objects of other classes,
without the requirement to know the true class of those objects, provided that those objects are all instantiated from classes that implement one or more specified interfaces. In other words, objects of classes that implement specified interfaces can be passed into methods of other objects as the generic type Object, and the methods of the other objects can invoke methods on the incoming objects by first casting them as the interface type.

The following programs have to be used along with Java course material.
Each of these programs illustrate a particular point/objective.

JT	-	Java Training
M1	-	OOPs and Environment
M2 & M3	-	Language Elements
M4	-	Input / Output
M5	-	OOPs Using Java
M6	-	Web Programming
M7	-	Windows Programming
M8 & M9	-	Advanced topics

Pgm: JTM1P1
Objective: To show the use of System.in.read() method.

```
//Import java.io for read method.
import java.io.*;
public class MyCat
{
        public static void main(String args[]) throws IOException
        {
                int b;
                // Key values are received as ascii ints.
                // read() throws IOException but is not caught.
                while((b=System.in.read()) != 65)
                {
                        //Output the character read
                        System.out.print((char)b);
         }
         //Output a blank line
         System.out.println();
        }
}
```

Pgm: JTM1P2

Objective: To understand the use of static modifier. Note that those static blocks without any name / parameter /or mdifier will be executed before even main method.

```
public class UnderstandingStaticDemo
{
        static int length = 10;
        static int width = 20;
        static int height = 30;
        static int volume,area;

        public static void main(String a[])
        {
                System.out.println("This is main method.");
                callStaticMethod(42);
        }

        static void         callStaticMethod(int x)
        {
                        System.out.println("This is static method.");
                        System.out.println("area : "+area);
                  System.out.println("volume :"+volume);
        }

        static
  {
                System.out.println("This is static first block.");
                area = length * width;
  }

 static
 {
                System.out.println("This is static second block.");
                volume = area * height;
 }

}

//OUTPUT:
//This is static first block.
//This is static second block.
//This is main method.
//This is static method.
//area : 200
//volume :6000
```

Pgm-JTM1P3

Obejective:

// Calling Static methods/variables in different classes through the class names itself.

```
class Box
{
        static int length=10;
        static int width=20;
        int height=30;

        static void area()
        {
                System.out.println("The area is :" + length*width);
        }

        void volume()
        {
                System.out.println("The volume is :" + length*width*height);
        }
}

public class StaticMethodCallDemo
{
	public static void main(String args[])
	{
		Box.area();
		System.out.println("The length is : " + Box.length);
		System.out.println("The width is : " + Box.width);
		// Box.volume(); // Error. You cannot refer to a non-static method like this.
		Box obj=new Box();  // You have to instantiate an object
		obj.volume();	// Access a non-static method through the object
		obj.area();	// You can access a static method also like this.
	}
}

//OUTPUT:
//The area is :200
//The length is : 10
//The width is : 20
//The volume is :6000
//The area is :200
```

Pgm : JTM5P1

```
 class SimpleClass
{
	int intVar;
	float floatVar;
	double doubleVar;

	SimpleClass(int i,float f,double d)
		{
			intVar=i;
			floatVar=f;
			doubleVar=d;
			System.out.println("SimpleClass(int i,float f,double d)");
		}

	SimpleClass(int i,float f)
		{
			intVar=i;
			floatVar=f;
			doubleVar=0;
			System.out.println("SimpleClass(int i,float f)");
		}

	SimpleClass(int i)
		{
			intVar=i;
			floatVar=0;
			doubleVar=0;
			System.out.println("SimpleClass(int i)");
		}

	SimpleClass(float f)
		{
			intVar=0;
			floatVar=f;
			doubleVar=0;
			System.out.println("SimpleClass(float f)");
		}

	SimpleClass(double d)
```

```
                {
                        intVar=0;
                        floatVar=0;
                        doubleVar=d;
                        System.out.println("SimpleClass(double d)");
                }

        SimpleClass()
                {
                        intVar=0;
                        floatVar=0;
                        doubleVar=0;
                        System.out.println("SimpleClass()");
         }
}

public  class OverLoadedConstructorsDemo1
{
        public static void main(String args[])
        {
                SimpleClass SC1=new SimpleClass(10,9.81f,1.11);
                SimpleClass SC2=new SimpleClass(10,9.81f);
        SimpleClass SC3=new SimpleClass(10);
                SimpleClass SC4=new SimpleClass(9.81f);
                SimpleClass SC5=new SimpleClass(1.11);
                SimpleClass SC6=new SimpleClass();
        }
}
//Output:
//SimpleClass(int i,float f, double d);
//SimpleClass(int i,float f);
//SimpleClass(int i);
//SimpleClass(float f);
//SimpleClass(double d);
//SimpleClass();
```

Pgm: JTM5P2

```
// Avoiding zero assignment as java does it for you
class SimpleClass
{
        int intVar;
        float floatVar;
        double doubleVar;

        SimpleClass(int i,float f,double d)
                {
                        intVar=i;
                        floatVar=f;
                        doubleVar=d;
                        System.out.println("SimpleClass(int i,float f,double d)");
                }

        SimpleClass(int i,float f)
                {
                        intVar=i;
                        floatVar=f;
                        System.out.println("SimpleClass(int i,float f)");
                }

        SimpleClass(int i)
                {
                        intVar=i;
                        floatVar=0;
                        doubleVar=0;
                        System.out.println("SimpleClass(int i)");
                }



        SimpleClass(float f)
                {
                        intVar=0;
                        floatVar=f;
                        doubleVar=0;
                        System.out.println("SimpleClass(float f)");
                }

        SimpleClass(double d)
                {
                        intVar=0;
                        floatVar=0;
                        doubleVar=d;
                        System.out.println("SimpleClass(double d)");
                }

        SimpleClass()
                {
                        intVar=0;
                        floatVar=0;
                        doubleVar=0;
                        System.out.println("SimpleClass()");
         }
}

public  class OverLoadedConstructorsDemo1
{
        public static void main(String args[])
        {
                SimpleClass SC1=new SimpleClass(10,9.81f,1.11);
                SimpleClass SC2=new SimpleClass(10,9.81f);
        SimpleClass SC3=new SimpleClass(10);
                SimpleClass SC4=new SimpleClass(9.81f);
                SimpleClass SC5=new SimpleClass(1.11);
                SimpleClass SC6=new SimpleClass();
        }
}

//Output:
//SimpleClass(int i,float f, double d);
//SimpleClass(int i,float f);
//SimpleClass(int i);
//SimpleClass(float f);
//SimpleClass(double d);
//SimpleClass();
```

Pgm: JTM5P3

```
        // Use of 'this' keyword to refer to the same class constructor
class SimpleClass
{
        int intVar;
        float floatVar;
        double doubleVar;

        SimpleClass(int i,float f,double d)
                {
                        intVar=i;
                        floatVar=f;
                        doubleVar=d;
```

```
                              System.out.println("SimpleClass(int i,float f,double d)");
                    }

          SimpleClass(int i,float f)
                    {
    this(9.81);
                              intVar=i;
                              floatVar=f;
                              System.out.println("SimpleClass(int i,float f)");
                    }

          SimpleClass(int i)
                    {
                              intVar=i;
                              System.out.println("SimpleClass(int i)");
                    }

          SimpleClass(float f)
                    {
                              floatVar=f;
                              System.out.println("SimpleClass(float f)");
                    }

          SimpleClass(double d)
                    {
                              doubleVar=d;
                              System.out.println("SimpleClass(double d)");
                    }

          SimpleClass()
                    {
    this(10,1.11f,9.81);
                              System.out.println("SimpleClass()");
           }
}

public    class OverLoadedConstructorsDemo1
{
          public static void main(String args[])
          {
                    SimpleClass SC1=new SimpleClass(10,9.81f,1.11);
                    SimpleClass SC2=new SimpleClass(10,9.81f);
            SimpleClass SC3=new SimpleClass(10);
                    SimpleClass SC4=new SimpleClass(9.81f);
                    SimpleClass SC5=new SimpleClass(1.11);
                    SimpleClass SC6=new SimpleClass();
          }
}
//Output:
//SimpleClass(int i,float f, double d);
//SimpleClass(double d);
//SimpleClass(int i,float f);
//SimpleClass(int i);
//SimpleClass(float f);
//SimpleClass(double d);
//SimpleClass(int i,float f, double d);
//SimpleClass();
```

Pgm: JTM5P4

```
          // Passing the same class object as an argument in the class method.
  class MyClass
{
          int i,j;
          MyClass(int x,int y)
          {
                    i=x;
                    j=y;
          }

          boolean equals(MyClass MC)
          {
                    if(MC.i==i && MC.j==j)
                              return true;
                    else
                              return false;
          }

}

public class ObjectsAsParametersDemo
{
          public static void main(String args[])
                    {
                              MyClass MC1=new MyClass(10,20);
                              MyClass MC2=new MyClass(10,20);
                              MyClass MC3=new MyClass(1,2);
                              System.out.println("MC1==MC2 ? " + MC1.equals(MC2));
                              System.out.println("MC1==MC3 ? " + MC1.equals(MC3));
                    }
}

          //Output:
//MC1==MC2 ? true
//MC1==MC3 ? false
```

Pgm : JTM5P5

```
class MyClass
{
          int a;

          MyClass(int i)
          {
                    a=i;
          }

          MyClass multiply()
          {
                    MyClass temp=new MyClass(a*10);
                    return temp;
          }
}

public class ReturningObjectDemo
{
          public static void main(String a[])
          {
                    MyClass obj1=new MyClass(2);
                    MyClass obj2=obj1.multiply();
                    MyClass obj3=obj2.multiply();

                    System.out.println("obj1.a = " + obj1.a);
                    System.out.println("obj2.a = " + obj2.a);
                    System.out.println("obj3.a = " + obj3.a);
          }
}


//OUTPUT:
```

```
//obj1.a =2
//obj2.a =20
//obj3.a =200
```

Pgm - JTM5P6

```
// Public Private Access Specifier

class Box
{
	int length;
	public int width;
	private int height;

	void setHeight(int i)
	{
		height=i;
	}


	int getHeight()
  {
	return height;
  }
}

public class PublicPrivateAccessDemo
{
	public static void main(String args[])
	{
		Box obj = new Box();
		obj.length=10;
		obj.width=20;
		// obj.height = 30;    // Error ! U can't access private variables
		// U can access private variables thro' public methods
		obj.setHeight(30);
		System.out.println("length, width and height : "+obj.length+":"+obj.width+
				":"+obj.getHeight());
  }
}

// OUTPUT:
// length, width and height : 10:20:30
```

Pgm-JTM5P7

```
// A simple inheritance example: Accessing
//    super class variables and methods from the sub class

class SuperClass
{
	int superInt;
	double superDouble;
	void printSuperVariable()
	{
		System.out.println("superInt = " + superInt + " superDouble = " + superDouble);
	}
}

class SubClass extends SuperClass
{
	boolean subBoolean;
	void printSubVariable()
	{
		System.out.println("subBoolean = " + subBoolean);
```

```
        }
}

class SimpleInheritance1
{
        public static void main(String args[])
                {
                 SuperClass sup=new SuperClass();
                 SubClass sub=new SubClass();
                 sup.superInt=10;
                 sup.superDouble=9.81;
                 sup.printSuperVariable();

                 sub.superInt=100;
                 sub.superDouble=98.1;
                 sub.subBoolean=true;
                 sub.printSuperVariable();
                 sub.printSubVariable();
                }
 }

//Output:
//superInt = 10 superDouble = 9.81
//superInt = 100 superDouble = 98.1
//subBoolean = true
```

Pgm-JTM5P8

```
// A simple Inheritance Example: Declaring the same super class variable in the sub-class

class SuperClass
{
  int superInt=0;
        double superDouble;

        void printSuperVariable()
        {
                System.out.println("\nSUPER CLASS: ");
                System.out.println("superInt = " + superInt);
                System.out.println("superDouble = " + superDouble);
        }
}

class SubClass extends SuperClass
{
        int superInt;
        boolean subBoolean;

        void printSubVariable()
        {
                System.out.println("\nSUB CLASS: ");
                System.out.println("subBoolean = " + subBoolean);
                System.out.println("superInt = " +superInt);
        }
}

class SimpleInheritance2
{
        public static void main(String args[])
                {
                        SuperClass sup= new SuperClass();
                        SubClass sub=new SubClass();

                 sup.superInt=10;
                sup.superDouble=9.81;
                sup.printSuperVariable();

                        sub.superInt=100;
                        sub.superDouble=98.1;
                        sub.subBoolean=true;
                        sub.printSuperVariable();
                        sub.printSubVariable();
                }
        }

//OUTPUT :
//SUPER CLASS:
//superInt = 10
//superDouble = 9.81
//
//SUPER CLASS:
//superInt = 0
//superDouble = 98.1
//
//SUB CLASS:
//subBoolean = true
//superInt = 100
```

Pgm-JTM5P9

```
// Assignment of object references in three different styles
//  super class object = sub class object; -->No problem
//  sub class object = super class object; -->Compiler error
//  sub class object = (sub class casting)super class object; -->Runtime error

class SuperClass
{
        int superInt;
        float superFloat;

        float multiply()
        {
                return superInt*superFloat;
        }
}

class SubClass extends SuperClass
{
        SubClass(int i, float f)
        {
                superInt=i;
                superFloat=f;
        }
}

class SimpleInheritance3
{
        public static void main(String a[])
        {
                SuperClass sup=new SuperClass();
                SubClass sub=new SubClass(10,1.56f);

                float subtemp=sub.multiply();
                System.out.println("Point 1: "+subtemp);

                subtemp=sup.multiply();
                System.out.println("Point 2: "+subtemp);

                sup=sub;                        //No Problem.
```

```
                        subtemp=sup.multiply();
                        System.out.println("Point 3: "+subtemp);

                        // sub=sup;          //Compile Time Error.
                        //sub=(SubClass)sup;  //Run Time Exception.
                        }
}

//OUTPUT :
//Point 1: 15.599999
//Point 2: 0.0
//Point 3: 15.599999
```

Pgm-JTM5P10

```
// Constructors Inheritance Demo:
// Finding out what constructors are being called by default,
// if you instantiate a sub class

class SuperClass
{
            int superInt;

            SuperClass()
            {
                        System.out.println("SuperClass()");
            }

            SuperClass(int i)
            {
                        System.out.println("SuperClass(int i)");
            }
}

class SubClass extends SuperClass
{
            double subDouble;
            SubClass()
  {
                        System.out.println("SubClass()");
            }

            SubClass(double d)
            {
                        System.out.println("SubClass(double d)");
            }
}

public class ConstructorInheritance
{
            public static void main(String A[])
            {
                        System.out.println("SuperClass sup1=new SuperClass()");
                        SuperClass sup1=new SuperClass();
              System.out.println();

                        System.out.println("SuperClass sup2=new SuperClass(10)");
                        SuperClass sup2=new SuperClass(10);
              System.out.println();

              System.out.println("SubClass sub1=new SubClass()");
                        SubClass sub1=new SubClass();
               System.out.println();

              System.out.println("SubClass sub2=new SubClass(20)");
                        SubClass sub2=new SubClass(20); // U can't invoke a superclass constructor.
            System.out.println();                   // It will type cast it to double.

                    System.out.println("SubClass sub3=new SubClass(9.81)");
                    SubClass sub3=new SubClass(9.81);
                    System.out.println();
  }
}

//OUTPUT :
//SuperClass sup1=new SuperClass()
//SuperClass()
//
//SuperClass sup2=new SuperClass(10)
//SuperClass(int i)
//
//SubClass sub1=new SubClass()
//SuperClass()
//SubClass()
//
//SubClass sub2=new SubClass(20)
//SuperClass()
//SubClass(double d)
//
//SubClass sub3=new SubClass(9.81)
//SuperClass()
//SubClass(double d)
```

Pgm- JTM5P11

```
// Constructor Inheritance Demo: Using 'this' and 'super' keywords for constructors
// 'this' or 'super' should be first exceutable statement in a constructor
// In a constructor, only one ('this' or 'super') should be present

class   SuperClass
{
        int intVar;

        SuperClass()
        {
                this(0);
                System.out.println("SuperClass()");
        }

        SuperClass(int i)
  {
   this.intVar=i;
   System.out.println("SuperClass(int i)");
        }
}

        class SubClass extends SuperClass
        {
                double doubleVar;

                SubClass()
                {
                        super();
                        doubleVar=0.0;
                        System.out.println("SubClass()");
                }

                SubClass(int i,double d)
                {
                        super(i);
                        this.doubleVar=d;
```

```
                    System.out.println("SubClass(int i,double d)");
            }
    }

    public class SuperDemo1
    {
            public static void main(String[] args)
                    {
                                    System.out.println("SuperClass sup1=new SuperClass()");
                                    SuperClass sup1=new SuperClass();
                                    System.out.println();

                                    System.out.println("SuperClass sup2=new SuperClass(10)");
                                    SuperClass sup2=new SuperClass(10);
                                    System.out.println();

                                    System.out.println("SubClass sub1=new SubClass()");
                                    SubClass sub1=new SubClass();
                                    System.out.println();

                                    System.out.println("SubClass sub2=new SubClass(10,9.81)");
                                    SubClass sub2=new SubClass(10,9.81);
                    }
    }

//OUTPUT:
//SuperClass sup1=new SuperClass()
//SuperClass(int i)
//SuperClass()

//SuperClass sup2=new SuperClass(10)
//SuperClass(int i)

//SubClass sub1=new SubClass()
//SuperClass(int i)
//SuperClass()
//SubClass()

//SubClass sub2=new SubClass(10,9.81)
//SuperClass(int i)
//SubClass(int i,double d)
```

Pgm-JTM5P12

```
// Constructors Inheritance Demo: Using this and super keywords
// for class instance variables.
class SuperClass
{
    int intVar;

    SuperClass()
 {
     intVar=10;
     System.out.println("SuperClass() : intvar = " + intVar);
    }

    SuperClass(int i)
    {
        intVar=i;
        System.out.println("SuperClass(int i) :  intvar = " + intVar);
  }
 }

 class SubClass extends SuperClass
 {
     int intVar;
```

```
            SubClass()
            {
              super.intVar=25;
              this.intVar=35;
              System.out.println("SubClass() : super.intVar = " + super.intVar +

                                                        " this.intVar = "+this.intVar);
  }


  SubClass(int i, int j)
  {
          super.intVar=i;
          this.intVar=j;
          System.out.println("SubClass() : super.intVar = " + super.intVar +

                                                        " this.intVar = "+this.intVar);
  }
 }


 public class SuperDemo2
 {
            public static void main(String a[])
            {
                    System.out.println("SuperClass sup1=new SuperClass()");
                    SuperClass sup1=new SuperClass();
                    System.out.println();

                    System.out.println("SuperClass sup2=new SuperClass(99)");
                    SuperClass sup2=new SuperClass(99);
                    System.out.println();

                    System.out.println("SubClass sub1=new SubClass()");
                    SubClass sub1=new SubClass();
                    System.out.println();

                    System.out.println("SubClass sub2=new SubClass(199,299)");
                    SuperClass sub2=new SubClass(199,299);
            }
 }

//OUTPUT:
//SuperClass sup1=new SuperClass()
//SuperClass() : intvar = 10
//
//SuperClass sup2=new SuperClass(99)
//SuperClass(int i) :  intvar = 99
//
//SubClass sub1=new SubClass()
//SuperClass() : intvar = 10
//SubClass() : super.intVar = 25 this.intVar = 35
//
//SubClass sub2=new SubClass(199,299)
//SuperClass() : intvar = 10
//SubClass() : super.intVar = 199 this.intVar = 299
```

Pgm- JTM5P13

```
//Dynamic Method Dispatch Demo: Creating each class object separately

class A
{
        void show()                                                                                    //
Classes
```

```
        {
                                        //              A
                System.out.println("This is A Class show()");                             //
          /   \
        }
                                        //          B   C
}
                                                                                          //
|
                                                                                          //
D
class B extends A
{
        void show()
        {
                System.out.println("This is B Class show()");
        }
}

class C extends A
{
        void show()
        {
                System.out.println("This is C Class show()");
        }
}

class D extends B
{
        void show()
        {
                System.out.println("This is D Class show()");
        }
}

class DynamicMethodDispatchDemo1
{
        public static void main(String[] args)
        {
                A aobj=new A();
                B bobj=new B();
                C cobj=new C();
                D dobj=new D();

                A ref;
                ref=aobj;
                ref.show();

                ref=bobj;
                ref.show();

                ref=cobj;
                ref.show();

                ref=dobj;
                ref.show();
        }
}

// OUTPUT:
//This is A Class show()
//This is B Class show()
//This is C Class show()
//This is D Class show()
```

Pgm-JTM5P14

```
// Overriding Demo : Converting onre of the subclass to a final class.
// All methods in a final class are implicitly final
// A final class cannot be subclassed.
// Converting one subclass method as final method. A final method
//    cannot be overrriden

abstract class Figure                                   // Figure class doesn't do anything and
{
        // the describes the fundamental behavior
        double dim1;                                                    //      of future classes.
        double dim2;

        abstract double area();
}

final class Rectangle extends Figure
{
        Rectangle(double a, double b)
        {
                dim1 = a;
                dim2 = b;
        }

        double area()
        {
                return dim1*dim2;
        }
}

final class Triangle extends Figure
{
        Triangle(double a, double b)
        {
                dim1 = a;
                dim2 = b;
        }

        final double area()                     // Sub classes of triangle cannot override
        {                                                               // this method.
                return 0.5*dim1*dim2;
        }
}

class FinalClassDemo
{
        static public void main(String a[])
        {
                Figure fig[]=new Figure[3];
                fig[0]=new Rectangle(2,5);
                fig[1]=new Triangle(5,6);

                System.out.println("Rectangle area = "+fig[0].area());
                System.out.println("Triangle area = "+fig[1].area());
        }
}
```

```
//OUTPUT :
//Rectangle area = 10.0
//Triangle area = 15.0
```

Strings

Q - Java provides two different string classes from which string objects can be instantiated. What are they?

A - The two classes are:

    String
    StringBuffer

Q - The StringBuffer class is used for strings that are not allowed to change. The String class is used for strings that are modified by the program: True or False. If false, explain why.

A - False. This statement is backwards. The String class is used for strings that are not allowed to change. The StringBuffer class is used for strings that are modified by the program.

Q - While the contents of a String object cannot be modified, a reference to a String object can be caused to point to a different String object: True or False. If false, explain why.

A - True.

Q - The use of the new operator is required for instantiation of objects of type String: True or False? If false, explain your answer.

A - False. A String object can be instantiated using either of the following statements:

```
String str1 = new String("String named str2");



String str2 = "String named str1";
```

Q - The use of the new operator is required for instantiation of objects of type StringBuffer: True or False? If false, explain your answer.

A - True.

Q - Provide a code fragment that illustrates how to instantiate an empty StringBuffer object of a default length and then use a version of the append() method to put some data into the object.

A - See code fragment below:

```
StringBuffer str5 = new StringBuffer();//accept default initial length

str5.append("StringBuffer named str5");//modify length as needed
```

Q - Without specifying any explicit numeric values, provide a code fragment that will instantiate an empty StringBuffer object of the correct initial length to contain the string "StringBuffer named str6" and then store that string in the object.

A - See the following code fragment:

```
StringBuffer str6 = new StringBuffer("StringBuffer named str6".length());

str6.append("StringBuffer named str6");
```

Q - Provide a code fragment consisting of a single statement showing how to use the Integer wrapper class to convert a string containing digits to an integer and store it in a variable of type int.

A - See code fragment below

```
int num = new Integer("3625").intValue();
```

Q - Explain the difference between the capacity() method and the length() methods of the StringBuffer class.

A - The capacity() method returns the amount of space currently allocated for the StringBuffer object. The length() method returns the amount of space used.

Q - The following is a valid code fragment: True or False? If false, explain why.

```
StringBuffer str6 = new StringBuffer("StringBuffer named str6".length());
```

A - True.

Q - Which of the following code fragments is the most efficient, first or second?

```
String str1 = "THIS STRING IS NAMED str1";
String str1 = new String("THIS STRING IS NAMED str1");
```

A - The first code fragment is the most efficient.

System

Java provides the System class which provides a platform-dependent interface between your program and various system resources: True or False? If false, explain why.

A - False. Java provides the System class which provides a platform-independent interface between your program and those resources.

Q - You must instantiate an object of the System class in order to use it: True or False? If false, explain why.

A - False. You don't need to instantiate an object of the System class to use it, because all of its variables and methods are class variables and methods.

Q - The following code fragment can be used to instantiate an object of the System class: True or False? If false, explain why.

```
System mySystemObject = new System();
```

A - False. You cannot instantiate an object of the System class. It is a final class, and all of its constructors are private.

Q - What is the purpose of the write() method of the PrintStream class?

A - The write() method is used to write bytes to the stream. You can use write() to write data which is not intended to be interpreted as text (such as bit-mapped graphics data).

Exceptions

Q - The exception-handling capability of Java makes it possible for you to monitor for exceptional conditions within your program, and to transfer control to special exception-handling code which you design. List five keywords that are used for this purpose.

A - try, throw, catch, finally, and throws

Q - All exceptions in Java are thrown by code that you write: True or False? If false, explain why.

A - False. There are situations where an exceptional condition automatically transfers control to special exception-handling code which you write (cases where you don't provide the code to throw the exception object).

Q - When an exceptional condition causes an exception to be thrown, that exception is an object derived, either directly, or indirectly from the class Exception: True or False? If false, explain why.

A - False. When an exceptional condition causes an exception to be thrown, that exception is an object derived, either directly, or indirectly from the class Throwable.

Q - All exceptions other than those in the RuntimeException class must be either caught, or declared in a throws clause of any method that can throw them: True or False? If false, explain why.

A - True.

Q - What method of which class would you use to extract the message from an exception object?

A - The getMessage() method of the Throwable class.

Q - Normally, those exception handlers designed to handle exceptions closest to the root of the exception class hierarchy should be placed first in the list of exception handlers: True or False? If false, explain why.

A - False. The above statement has it backwards. Those handlers designed to handle exceptions furthermost from the root of the hierarchy tree should be placed first in the list of exception handlers.

Q - Explain why you should place exception handlers furthermost from the root of the exception hierarchy tree first in the list of exception handlers.

A - An exception hander designed to handle a specialized "leaf" object may be preempted by another handler whose exception object type is closer to the root of the exception hierarchy tree if the second exception handler appears earlier in the list of exception handlers.

Q - In addition to writing handlers for very specialized exception objects, the Java language allows you to write general exception handlers that handle multiple types of exceptions: True or False? If false, explain why.

A - True.

Q - Your exception handler can be written to handle any class that inherits from Throwable. If you write a handler for a node class (a class with no subclasses), you've written a specialized handler: it will only handle exceptions of that specific type. If you write a handler for a leaf class (a class with subclasses), you've written a general handler: it will handle any exception whose type is the node class or any of its subclasses. True or False? If false, explain why.

A - False. "Leaf" and "node" are reversed in the above statement. If you write a handler for a "leaf" class (a class with no subclasses), you've written a specialized handler: it will only handle exceptions of that specific type. If you write a handler for a "node" class (a class with subclasses), you've written a general handler: it will handle any exception whose type is the node class or any of its subclasses."

Q - Java's finally block provides a mechanism that allows your method to clean up after itself regardless of what happens within the try block. True or False? If false, explain why.

A - True.

Q - Explain how you would specify that a method throws one or more exceptions.

A - To specify that a method throws one or more exceptions, you add a throws clause to the method signature for the method. The throws clause is composed of the throws keyword followed by a comma-separated list of all the exceptions thrown by that method.

Q - Provide a code fragment that illustrates how you would specify that a method throws more than one exception.

A - See code fragment below.

```
 void myMethod() throws InterruptedException, MyException,

      HerException, UrException

 {

 //method body

 }
```

Q - What type of argument is required by the throw statement?

A - The throw statement requires a single argument, which must be an object derived either directly or indirectly from the class Throwable.

Q - Some exception objects are automatically thrown by the system. It is also possible for you to define your own exception classes, and to cause objects of those classes to be thrown whenever an exception occurs. True or False? If false, explain why.

A - True.

=======

Threads

Q - What is the definition of multi-threaded programming according to Patrick Naughton?

A - According to The Java Handbook, by Patrick Naughton,

"Multi-threaded programming is a conceptual paradigm for programming where you divide programs into two or more processes which can be run in parallel."

Q - Multithreading refers to two or more programs executing, "apparently" concurrently, under control of the operating system. The programs need have no relationship with each other, other than the fact that you want to start and run them all concurrently. True or False? If false, explain why.

A - False. That is a description of multiprocessing, not multithreading. Multithreading refers to two or more tasks executing, "apparently" concurrently, within a single program.

Q - According to current terminology, the term blocked means that the thread is waiting for something to happen and is not consuming computer resources. True or False? If false, explain why.

A - True.

Q - What are the two ways to create threaded programs in Java?

A - In Java, there are two ways to create threaded programs:

Implement the Runnable interface
Extend the Thread class

Q - What two steps are required to spawn a thread in Java?

A - The two steps necessary to spawn a thread in Java are:

instantiate an object of type Thread and
invoke its run() method.

Q - How do you start a thread actually running in Java?

A - Invoke the start() method on object of the Thread class or of a subclass of the Thread class.

Q - It is always possible to extend the Thread class in your Java applications and applets. True or False? If false, explain why.

A - False. Sometimes it is not possible to extend the Thread class, because you must extend some other class and Java does not support multiple inheritance.

Q - Although multithreaded programming in Java is possible, it is also possible to write Java programs that do not involve threads: True or False? If false, explain why.

A - False. The main method itself runs in a thread which is started by the interpreter.

Q - What is the name of the method that can be used to determine if a thread is alive?

A - The name of the method is isAlive().

Q - Once you start two or more threads running, unless you specify otherwise, they run synchronously and independently of one another: True or False? If false, explain why.

A - False. Once you start two or more threads running, unless you specify otherwise, they run asynchronously and independently of one another.

Q - The process of keeping one thread from corrupting the data while it is being processed by another thread is known as synchronization: True or False? If false, explain why.

A - True.

Q - Java allows you to specify the absolute priority of each thread: True or False? If false, explain why.

A - False. Java allows you to specify the priority of each thread relative to other threads but not on an absolute basis.

Q - Thread synchronization can be achieved using wait(), notify(), and notifyAll() which are methods of the Thread class: True or False? If false, explain why.

A - False. wait(), notify(), and notifyAll() are not methods of the Thread class, but rather are methods of the Object class.

Q - When you implement a threaded program, you will always override the _____________ method of the Thread class and build the functionality of your threaded program into that method. What is the name of the method?

A - The run() method.

Q - In a multithreaded program, you will start a thread running by invoking the __________ method on your Thread object which will in turn invoke the ___________ method. What are the names of the missing methods, and what are the required parameters for each method?

A - In a multithreaded program, you will start a thread running by invoking the start() method on your Thread object which will in turn invoke the run() method. Neither method takes any parameters.

Q - What do Campione and Walrath list as the four possible states of a thread?

A -
New Thread
Runnable
Not Runnable
Dead

Q - What methods can be invoked on a thread object which is in the state that Campione and Walrath refer to as a New Thread and what will happen if you invoke any other method on the thread?

A - When a thread is in this state, you can only start the thread or stop it. Calling any method other than start() or stop() will cause an IllegalThreadStateException.

Q - What, according to Campione and Walrath, will cause a thread to become Not Runnable?

A - a thread becomes Not Runnable when one of the following four events occurs:

Someone invokes its sleep() method.
Someone invokes its suspend() method.
The thread uses its wait() method to wait on a condition variable.
The thread is blocking on I/O.

Q1 - Three keywords are used in Java to specify access control. What are they?
A - The three keywords used to specify access control in Java are public, private, and protected.

Q2 - In Java, special access privileges are afforded to other members of the same package: True or False? If false, explain your answer.
A - True. In Java, special access privileges are afforded to other members of the same package.

Q3 - In Java, class variables are often used with the __________ keyword to create variables that act like constants.

A - The final keyword.

Q4 - In Java, the ___________ keyword is used to declare a class variable.

A - The static keyword.

Q5 - In Java, the ___________ keyword is used to declare a class method.

A - The static keyword.

Q6 - When you include a method in a Java class definition without use of static keyword, this will result in objects of that class containing an instance method: True or False? If false, explain why.

A 7- True.

Q8 - Normally each object contains its own copy of each instance method: True or False?

A - False, multiple copies of the method do not normally exist in memory.

Q9- When you invoke an instance method using a specific object, if that method refers to instance variables of the class, that method is caused to refer to the specific instance variables of the specific object for which it was invoked: True or False? If false, explain why.

A - True.

Q10 - Instance methods are invoked in Java using the name of the object, the colon, and the name of the method as shown below: True or False? If false, explain why.

```
myObject:myInstanceMethod( )
```

A - False. Use the period or dot operator, not the colon.

Q11 - Instance methods have access to both instance variables and class variables in Java: True or False. If false, explain why.

A - True.

Q12 - Class methods have access to both instance variables and class variables in Java: True or False. If false, explain why.

A - False. Class method can only access other class members.

Q13 - What are the two most significant characteristics of class methods?

A - 1. Class methods can only access other class members. 2. Class methods can be accessed using only the name of the class. An object of the class is not required to access class methods.

Q14 - In Java, a class method can be invoked using the name of the class, the colon, and the name of the method as shown below: True or False? If false, explain why.

```
MyClass:myClassMethod()
```

A - False. You must use the period or dot operator, not the colon.

Q15 - What is meant by overloaded methods?

A - The term overloaded methods means that two or more methods may have the same name so long as they have different argument lists.

Q16 - If you overload a method name, the compiler determines at run time, on the basis of the arguments provided to the invocation of the method, which version of the method to call in that instance: True or False? If false, explain why.

A - False. The determination is made at compile time.

Q16 - A constructor is a special method which is used to construct an object. A constructor always has the same name as the class in which it is defined, and has no return type specified. True or False? If false, explain why.

A - True.

Q17 - Constructors may be overloaded, so a single class may have more than one constructor, all of which have the same name, but different argument lists: True or False. If false, explain why.

A - True

Q18 - What is the purpose of a parameterized constructor?

A - The purpose of a parameterized constructor is to initialize the instance variables of an object when the object is instantiated.

Q 19 - The same set of instance variables can often be initialized in more than one way using overloaded constructors: True or False? If false, explain why.

A - True.

Q20 - It is not necessary to provide a constructor in Java. True or False? If false, explain why.

A - True. .

Q 21 You can think of the default constructor as a constructor which doesn't take any parameters: True or False? If false, explain why.

A - True.

Q 22- In Java, if you provide any constructors, the default constructor is no longer provided automatically: True or False? If false, explain why.

A - True.

Q 23- In Java, if you need both a parameterized constructor and a constructor which doesn't take parameters (often called a default constructor), you must provide them both: True or False? If false, explain why.

A - True.

Q 24- In Java, you can instantiate objects in static memory at compile time, or you can use the new operator to request memory from the operating system at runtime and use the constructor to instantiate the object in that memory: True or False? If false, explain why.

A - False. In Java, objects can only be instantiated on the heap at runtime.

Q25 - Provide a code fragment consisting of a single statement that illustrates how the constructor is typically used in Java to declare, instantiate, and initialize an object. Assume a parameterized constructor with three parameters of type int.

A - MyClass myObject = new MyClass(1,2,3);

Q26 - Provide a code fragment consisting of a single statement that illustrates how the default constructor is typically used in Java to declare and instantiate an object.

A - MyClass myObject = new MyClass();

Q 27- What are the three actions performed by the following statement?

MyClass myObject = new MyClass(1,2,3);

A - This statement performs three actions in one.

The object is declared by notifying the compiler of the name of the object.
The object is instantiated by using the new operator to allocate memory space to contain the new object.
The object is initialized by making a call to the constructor named MyClass.

Q 28- In Java, if you attempt to instantiate an object and the Java Virtual Machine cannot allocate the requisite memory, the system will: ____.

A - Throw an OutOfMemoryError.

Q 29- The following is a valid method call: True or False. If false, explain why.

obj.myFunction(new myClassConstructor(1,2,3));//Java version

A - True.

Q30 - In Java, when a method begins execution, all of the parameters are created as local automatic variables: True or False? If false, explain why.

A - True.

Q31 - In the following statement, an object is instantiated and initialized and passed as a parameter to a function. What will happen to that object when the function terminates?

obj.myFunction(new myClassConstructor(1,2,3));//Java version

A - It will become eligible for garbage collection.

Q 32- In Java, you declare and implement a constructor just like you would implement any other method in your class, except that: ____

A - you do not specify a return type and must not include a return statement.

Q33 - The name of the constructor must be the same as the name of the ___________________.

A - class.

Q 34- Usually in cases of inheritance, you will want the subclass to cause the constructor for the superclass to execute last to initialize those instance variables which derive from the superclass: True or False? If false, explain why.

A - False. You will want the subclass to cause the constructor for the superclass to execute first.

Q 35- Provide a code fragment that you would include at the beginning of your constructor for a subclass to cause the constructor for the superclass to be invoked prior to the execution of the body of the constructor.

A - super(optional parameters);

Q 36- Every object has a finalize method which is inherited from the class named ________________.

A - object.

Q 37- Before an object is reclaimed by the garbage collector, the _______________ method for the object is called.

A - finalize

Q 38- In Java, the destructor is always called when an object goes out of scope: True or False? If false, explain why.

A - False. Java does not support the concept of a destructor.

=====

Q 39- The class at the top of the inheritance hierarchy is the Object class and this class is defined in the package named java.Object: True or False? If false, explain why.

A - False. The Object class is defined in the package named java.lang.

Q40 - We say that an object has state and behavior: True or False? If false, explain why.

A - True.

Q 41- In Java, a method can be defined as an empty method, normally indicating that it is intended to be overridden: True or False? If false, explain why.

A - True.

Q 42- Including an empty method in a class definition will make it impossible to instantiate an object of that class: True or False.

A - False.

Q 43- A subclass can invoke the constructor for the immediate superclass by causing the last line of of the subclass constructor to contain the keyword super followed by a parameter list as though calling a function named super() and the parameter list must match the method signature of the superclass constructor: True or False? If false, explain why.

A - False. This can only be accomplished by causing the first line of the constructor to contain the keyword super followed by a parameter list as though calling a function named super(). The parameter list must match the method signature of the superclass constructor.

Q 43 The equals() method is used to determine if two reference variables point to the same object: True or False? If false, explain why.

A- False. You can use the equals() method to compare two objects for equality. You can use the equality operator (==) to determine if two reference variables point to the same object.

Q 44- The equals() method is used to determine if two separate objects are of the same type and contain the same data. The method returns false if the objects are equal and true otherwise. True or False? If false, explain why.

A - False. The method returns true if the objects are equal and false otherwise.

Q 45 - The equals() method is defined in the Object class: True or False? If false, explain why.

A - True.

Q 46- Your classes can override the equals() method to make an appropriate comparison between two objects of a type that you define: True or False? If false, explain why.

A - True.

Q 47- You must override the equals() method to determine if two string objects contain the same data: True or False? If false, explain why.

A - False. The system already knows how to apply the equals() method to all of the standard classes and objects of which the compiler has knowledge. For example, you can already use the method to test two String objects or two array objects for equality.

Q 48- Given an object named obj1, provide a code fragment that shows how to obtain the name of the class from which obj1 was instantiated and the name of the superclass of that class.

A - See code fragment below:

```
System.out.println("Name of class for obj1: "          + obj1.getClass().getName());

System.out.println("Name of superclass for obj1: "         + obj1.getClass().getSuperclass());
```

Q 49- Given an object named obj2, provide a code fragment that shows how to use the newInstance() method to create a new object of the same type

A - See code fragment below:

```
obj2 = obj1.getClass().newInstance();
```

Q 50- By overriding the getClass() method, you can use that method to determine the name of the class from which an object was instantiated: True or False? If false, explain why.

A - False. The getClass() method is a final method and cannot be overridden.

Q 51- You must use the new operator to instantiate an object of type Class: True or False? If false, explain why.

A - False. There is no public constructor for the class Class. Class objects are constructed automatically by the Java Virtual Machine as classes are loaded and or by calls to the defineClass method in the class loader.

Q 52- The Class class provides a toString() method which can be used to convert all objects known to the compiler to some appropriate string representation. The actual string representation depends on the type of object: True or False? If false, explain why.

A - False. The Object class (not the Class class) provides a toString() method which can be used to convert all objects known to the compiler to some appropriate string representation. The actual string representation depends on the type of object.

Q 53- You can override the toString() method of the Class class to cause it to convert objects of your design to strings: True or False? If false, explain why.

A - False. The toString() method is a method of the Object class, not the Class class.

Q 54- By default, all classes in Java are either direct or indirect descendants of the Class class which is at the top of the inheritance hierarchy: True or False? If false, explain why.

A - False. By default, all classes in Java are either direct or indirect descendants of the Object class (not the Class class) which is at the top of the inheritance hierarchy.

=====

Q 55 - To a limited extent, the interface concept allows you to treat a number of objects, instantiated from different classes, as if they were all of the same type: True or False? If false, explain why.

A - True.

Q 56- At its simplest level, an interface definition has a name, and declares one or more methods: True or False? If false, explain why.

A - True.

Q57- In an interface definition, both the method signatures and the actual implementations (bodies) of the methods are provided: True or False? If false, explain why.

A - False. Only the method signatures are provided. The actual implementations (bodies) of the methods are not provided.

Q 58- An interface definition can contain only method declarations: True or False? If false, explain why.

A - False. In addition to method declarations, an interface can also declare constants. Nothing else may be included inside the body of an interface definition.

Q59 - If classes P, D, and Q all implement interface X, a reference variable for an object of class P, D, or Q could be assigned to a reference variable of type X: True or False? If false, explain why.

A - True.

Q 60- If classes P, D, and Q all implement interface X, then all of the methods declared in X must be exactly the same in classes P, D, and Q: True or False? If false, explain why.

A - False. The interface simply declares the signatures for methods. Classes that implement the interface are free to provide a body for those methods which best suits the needs of the class.

Q 61- If classes P, D, and Q all implement interface X a reference variable for an object of class P, D, or Q could be assigned to a reference variable of type X and that reference variable could be used to access all of the methods of the class (which are not excluded using public, private, or protected): True or False? If false, explain why.

A - False. If one or more of the classes P, D, and Q, define instance methods which are not declared in the interface X, then a variable of type X cannot be used to access those instance methods. Those methods can only be accessed using a reference variable of the class in which the method is defined. Reference variables of the type X can only be used to access methods declared in the interface X (or one of its superinterfaces).

Q 61 - The new operator must be used to instantiate an object which is of the type of an interface: True or False? If false, explain why.

A - False. Even though you can consider the interface name as a type for purposes of storing references to objects, you cannot instantiate an object of the interface type itself.

Q 63- One of the difficulties of implementing interfaces is the requirement to coordinate the definition of interface methods among the classes that implement the interface: True or False? If false, explain why.

A - False. In defining interface methods, each class defines the methods in a manner that is appropriate to its own class without concern for how it is defined in other classes.

Q 64- As with classes, multiple interface definitions can be combined into the same source file: True or False? If false, explain why.

A- False. The compiler requires interface definitions to be in separate files.

Q 65- List four ways in which interfaces are useful:

A - See the following list:

To a limited extent, the interface concept allows you to treat a number of objects, instantiated from different classes, as if they were all of the same type
Capturing similarities between unrelated classes without forcing a class relationship

Declaring methods that one or more classes are expected to implement
Revealing an object's programming interface without revealing its class (objects such as these are called anonymous objects and can be useful when shipping a package of classes to other developers)

Q 66- A minimum interface declaration contains the Java keyword interface, the name of the interface, and the name of the interface that it extends: True or False? If false, explain why.

A - False. A minimum interface declaration contains the Java keyword interface and the name of the interface. There is no requirement to specify the name of the interface that it extends, because it may not extend another interface.

Q 67 - An interface can extend any number of other interfaces: True or False? If false, explain why.

A - True.

Q 68- Just like a class definition can extend any number of other classes, an interface can extend any number of other interfaces: True or False? If false, explain why.

A - False. A class can extend only one other class.

Q 69- An interface can extend any number of other interfaces but not more than one class: True or False? If false, explain why.

A - False. An interface cannot extend a class.

Q 70- An interface inherits all constants and methods from its superinterface: True or False? If false, explain why.

A - False. See reasons below:

An interface inherits all constants and methods from its superinterface unless:

the interface hides a constant with another of the same name, or
redeclares a method with a new method declaration.

Q 71- The method declaration in an interface consists of the method signature followed by a pair of empty curly braces: True or False? If false, explain why.

A - False. The method declaration is terminated by a semicolon and no body (no curly braces) is provided for the method.

Q 72- The keyword private is used to restrict access to the members of an interface only to classes within the same package: True or False? If false, explain why.

A - False. You may not use private in a member declaration in an interface.

Q 73- All methods declared in an interface are implicitly public and abstract: True or False? If false, explain why.

A - True.

Q 74- In addition to declaring methods, the body of the interface may also define constants. Constant values defined in an interface are implicitly public, static, and final: True or False? If false, explain why.

A - True.

Q 75- You use an interface by defining a class that extends the interface by name: True or False? If false, explain why.

A - False. You use an interface by defining a class that implements (not extends) the interface by name.

Q 76- When a class claims to implement an interface, it must provide a full definition for all the methods declared in the interface as well as all of the methods declared in all of the superinterfaces of that interface: True or False? If false, explain why.

A - True.

Q 77- A class can implement more than one interface by including several interface names in a comma-separated list of interface names, and by providing a full definition for all of the methods declared in all of the interfaces listed as well as all of the superinterfaces of those interfaces: True or False? If false, explain why.

A - True.

Q 78- Whenever a class implements an interface, it is allowed to define only those methods declared in the interface: True or False? If false, explain why.

A - False. Whenever a class implements an interface, the class must define all of the methods declared in the interface, but is also free to define other methods as well.

Q 79- The definition of an interface is a definition of a new reference data type. You can use interface names just about anywhere that you would use other type names, except that you cannot ____________________.

A - You cannot instantiate objects of the interface type.

Q 80- Explain in your own words the "bottom line" benefits of the use of an interface.

A - The interface makes it possible for a method in one class to invoke methods on objects of other classes, without the requirement to know the true class of those objects, provided that those objects are all instantiated from classes that implement one or more specified interfaces. In other words, objects of classes that implement specified interfaces can be passed into methods of other objects as the generic type Object, and the methods of the other objects can invoke methods on the incoming objects by first casting them as the interface type.

Q - The Java read() method reads and returns a single byte from the standard input device. It stores that byte according to what type. What does the method return if the user enters an eof?

A - The Java read() method reads and returns a single byte from the standard input device and stores that byte in an integer. It returns an integer value of -1 if the user enters an eof.

Q - What keystroke combination can be used to simulate an eof at the keyboard of a DOS system?

A - An eof can be simulated on a DOS system keyboard by holding down the ctrl key and pressing the z key.

Q - Provide a Java code fragment illustrating how you would read a stream of bytes from the standard input device until encountering an eof and quit reading when the eof is encountered.

A - The following Java code fragment will read a stream of bytes from the standard input device until encountering an eof.

```
 while(System.in.read() != -1) { //do something }
```

This code fragment accesses the read()method of the object referred to by the class variable named in of the class named System.

Q - Provide a Java code fragment illustrating two different ways to display a String argument on the Java standard output device. Explain how your code works in object-oriented terms. Make certain that you explain the difference between the two.

A - The following two code fragments will each display a string argument on the Java standard output device.

```
 System.out.println("String argument")

 System.out.print("String argument")
```

In the first case, the code fragment accesses the println() method of the object referred to by the class variable named out of the class named System. In the second case, the print() method is accessed instead of the println() method.

The difference between the two is that the println() method automatically inserts a newlineat the end of the string argument whereas the print() method leaves the display cursor at the end of the string argument.
===============

OOP

Q - In Object-Oriented Programming, an object is often said to be an ____________ of a class.

A - An object is often said to be an instance of a class.

Q - In Object-Oriented Programming, an object is often said to have s_______ and b_______.
Provide the missing words which begin with the letters shown.

A - In OOP, an object is often said to have state and behavior.

Q - An object's state is contained in its ________ and its behavior is implemented through its ________.
A - An object's state is contained in its member variables (or data members) and its behavior is implemented through its methods (or member functions).

Q - The member variables of an object can be either ____________ or _________ .

A - Its member variables can be either instance variables or class variables.

Q - What is generally meant by the terminology "sending a message to an object?"

A - We activate the behavior of an object by invoking one of its methods (sending it a message).

Q - What are the two things that can usually happen when an object receives a message?

A - When an object receives a message, it usually either performs an action, or modifies its state, or both.

Q - What happens to the memory occupied by an object in Java when the object is no longer needed, and what do we normally do to make that happen?

A - When an object is no longer needed in Java, we simply forget it. Eventually, the garbage collector may (or may not) come by and pick it up for recycling.

Q - Identify as the stages of an object's life?

A - The stages of an Object's life are:

Creation
Use
Cleanup

Q - The creation of an object involves three steps (which are often combined). What are the three steps?

A - The three steps are:

declaration (providing a name for the object)
instantiation (setting aside memory for the object)
optional initialization (providing initial values for the object's instance variables)

Q - Java allows the instantiation of variables of primitive types in dynamic memory: True or False?
If false, explain why and what you might be able to do to achieve almost the same result.

A - False. Java does not allow the instantiation of primitive variables in dynamic memory. (However, there are wrapper classes for primitive types which can be used to turn them into objects for this purpose.)

Q - An array of objects in Java is instantiated as an array of reference variables where each reference variable can then be used to instantiate an object pointed to by the reference variable:
True or False. If false, explain why and either provide a code fragment that illustrates your answer

A - True.

Q - In Java, it is always necessary to declare (give a name to) all new objects: True or False? If false, explain why and either provide a code fragment that illustrates your answer

A - It is not always necessary in Java to declare an object (to give it a name). Consider, for example a case where a new object is instantiated to be used in an expression and there is no requirement to be able to access that object outside of the expression.

Q - Instance variables and instance methods can be accessed using an object as the access mechanism. What is the difference in the syntax used to access an instance variable and an instance method.

A - None. There is essentially no difference in the syntax used to access a variable or a method.

Q - Once you have instantiated an object, it is always possible to access all of the instance variables and instance methods of that object by joining the name of the object to the name of the variable or method using a period: True or False? If false, explain why.

A - False. Sometimes variables or methods may be hidden so as to make it impossible to access them. It is very common to hide the variables and to provide methods which can be accessed to serve as a pathway to the variables.

Q - Given an object named obj that has a public instance method named myMethod(), provide a code fragment that shows the proper syntax for accessing the method.

A - The proper syntax for accessing an instance method named myMethod() follows:

```
obj.myMethod()
```

Q - The object-oriented approach normally recommends hiding instance variables behind access methods: True or False? If true, explain why.

A - True. The object-oriented approach normally recommends hiding instance variables behind access methods. There are a variety of reasons why. One important reason is that hiding the instance variables makes it possible to later modify the implementation of instance variables to improve the behavior of objects of the class, without a requirement for modifying code that uses the clsss, provided that the access methods are not modified.

Q - The returning of memory (to the operating system) occupied by objects that are no longer needed is automatically accomplished in Java by a feature commonly known as the ____________________.

A - The returning of memory to the operating system is taken care of automatically by a feature of Java known as the garbage collector.

Q - All necessary cleanup in a Java program is performed automatically by the garbage collector: True or False? If false, explain why.

A - False. Java does not support anything like a destructor that is guaranteed to be called whenever the object is no longer needed. Therefore, other than returning allocated memory, it is the responsibility of the programmer to explicitly perform any other required cleanup at the appropriate point in time.

Q - When does an object become eligible for garbage collection?

A - An object becomes eligible for garbage collection when there are no more references to that object.

Q - What can your program do to purposely make an object eligible for garbage collection.

A - Your program can make an object eligible for garbage collection by assigning null to all references to the object.

Q - The purpose of garbage collection in Java is to perform all necessary cleanup and the memory occupied by objects that are no longer needed will always be reclaimed by the garbage collector: True or False? If false, explain why.

A - False. The sole purpose of garbage collection is to reclaim memory occupied by objects that are no longer needed, and it has no other purpose relative to necessary cleanup. An object becomes eligible for garbage collection when there are no more references to that object. However, just because an object is eligible for garbage collection doesn't mean that it will be reclaimed. The garbage collector runs in a low-priority thread, and may not run at all unless a memory shortage is detected.

Q - Before the garbage collector reclaims the memory occupied by an object, it always calls the object's _________ method. (Provide the name of the method.) Explain why this method is one of the methods in all new classes that you define.

A - Before the garbage collector reclaims the memory occupied by an object, it calls the object's finalizemethod. The finalize method is a member of the Object class. Since all classes inherit from the Object class, your classes also contain the default finalize method.

Q - What must you do to make effective use of the finalize method? Explain why you might want to do this.

A - In order to make use of the finalize method, you must override it, providing the code that you want to have executed before the memory is reclaimed.

Q - You can always be confident that the finalize method will be promptly executed to perform necessary cleanup in a Java program when an object becomes eligible for garbage collection: True or False? If false, explain why.

A - False. Although you can be confident that the finalize method will be called before the garbage collector reclaims the memory occupied by a particular object, you cannot be certain when, or if that memory will be reclaimed. There is no guarantee that the memory will be reclaimed by the garbage collector during the execution of your program.

Q - Provide a code fragment illustrating the method call that you can make to ask the garbage collector to run. This guarantees that garbage collection will take place: True or False? If false, explain why.

A - False. Campione and Walrath indicate that you can ask the garbage collector to run at any time by calling the method shown below. They further point out, however, that making the request does not guarantee that your objects will be reclaimed through garbage collection.   System.gc();

======

Q 1What do we call an operator that operates on only one operand?
A - An operator that operates on only one operand is called a unary operator.

Q 2- What do we call an operator that operates on two operands?
A - An operator that operates on two operands is called a binary operator.

Q 3- Is the minus sign a unary or a binary operator, or both? Explain your answer.
A - Both. As a binary operator, the minus sign causes its right operand to be subtracted from its left operand. As a unary operator, the minus sign causes the algebraic sign of the right operand to be changed.

Q 4- Describe operator overloading.
A - For those languages that support it (such as C++) operator overloading means that the programmer can redefine the behavior of an operator with respect to objects of a new type defined by that program.

Q5- Java programmers may overload operators: True or False?
A 6- False: Unfortunately, Java does not support operator overloading.

Q7 - Show the symbols used for the following operators in Java: assignment, not equal, addition,cast.
A - The above listed operators in order are: = != + (type)

Q8 - Is any operator automatically overloaded in Java? If so, identify it and describe its overloaded behavior.
A - The plus sign (+) is automatically overloaded in Java. The plus sign can be used to perform arithmetic addition. It can also be used to concatenate strings. However, the plus sign does more than concatenate strings. It also performs a conversion to String type. When the plus sign is used to concatenate strings, the operand on the right is automatically converted to a character string before being concatenated with the operand on the left. This assumes that the compiler knows enough about the operand on the right to be able to successfully perform the conversion. It has that knowledge for all of the primitive types and most or all of the built-in reference types.

Q 9- What is the purpose of the cast operator?
A - The cast operator is used to purposely convert from one type to another.

Q10 - The increment operator is a binary operator: True or False?
A - False: The increment operator is a unary operator.

Q11 - Show the symbol for the increment operator.
A - The symbol for the increment operator is two plus signs with nothing between them (++).

Q 12- Describe the appearance and the behavior of the increment operator with both prefix and postfix notation. Show example, possibly incomplete, code fragments illustrating both notational forms.
A - The increment operator may be used with both prefix and postfix notation. Basically, the increment operator causes the value of the variable to which it is applied to be increased by one. With prefix notation, the operand appears to the right of the operator (++X), while with postfix notation, the operand appears to the left of the operator (X++).
The difference in behavior has to do with the point in time that the increment actually occurs if the operator and its operand appear as part of a larger overall expression. With the prefix version, the variable is incremented before it is used to evaluate the larger overall expression. With the postfix version, the variable is used to evaluate the larger overall expression and then it is incremented.

Q13 - Show the output that would be produced by the following Java application.

```
 class prg1 { //define the controlling class
   public static void main(String[] args){ //define main method
     int x = 5, X = 5, y = 5, Y = 5;
     System.out.println("x = " + x );

     System.out.println("X = " + X );
     System.out.println("x + X++ = " + (x + X++) );
     System.out.println("X = " + X );
     System.out.println();
     System.out.println("y = " + y );
     System.out.println("Y = " + Y );
     System.out.println("y + ++Y = " + (y + ++Y) );
     System.out.println("Y = " + Y );
   }//end main
 }//End  class.  Note no semicolon required
 //End Java application
```

A - The output from this Java application follows:

```
x = 5
X = 5
x + X++ = 10
X = 6
y = 5
Y = 5
y + ++Y = 11
Y = 6
```

Q 14 - Binary operators use outfix notation: True or False? If your answer is False, explain why.
A - False: Binary operators use infix notation, which means that the operator appears between its operands.

Q15 - In practice, what does it mean to say that an operator that has performed an action returns a value (or evaluates to a value) of a given type?
A - As a result of performing the specified action, an operator can be said to return a value (or evaluate to a value) of a given type. The type depends on the operator and the type of the operands. To evaluate to a value means that after the action is performed, the operator and its operands are effectively replaced in the expression by the value that is returned.

Q 16 - What are the four categories of operators described in Baldwin's Java tutorial on operators? Do you agree with this categorization? If not, explain why not.
A - Some authors divide Java's operators into the following categories:
arithmetic
relational and conditional (typically called relational and logical in C++)
bitwise and logical
assignment

Q17 - Show and describe at least five of the binary arithmetic operators supported by Java (Clarification: binary operators does not mean bitwise operators).
A - Java support various arithmetic operators on floating point and integer numbers. The following table lists five of the binary arithmetic operators supported by Java.

Operator	Description
+	Adds its operands
-	Subtracts the right operand from the left operand
*	Multiplies the operands
/	Divides the left operand by the right operand
%	Remainder of dividing the left operand by the right operand

Q18 - In addition to arithmetic addition, what is another use for the plus operator (+)? Show an example code fragment to illustrate your answer. The code fragment need not be a complete statement.
A - The plus operator (+) is also used to concatenate strings : "IMR " + " global Ltd."

Q19 - When the plus operator (+) is used as a concatenation operator, what is the nature of its behavior if its right operand is not of type String? If the right operand is a variable that is not of type String, what is the impact of this behavior on that variable.
A - In this case, the operator also coerces the value of the right operand to a string representation for use in the expression only. If the right operand is a variable, the value stored in the variable is not modified in any way.

Q 20- Show and describe four unary arithmetic operators supported by Java.
A - Java supports the following four unary arithmetic operators.

Operator	Description
+	Indicates a positive value
-	Negates, or changes algebraic sign
++	Adds one to the operand, both prefix and postfix
--	Subtracts one from operand, prefix and postfix

Q20 - What is the type returned by relational operators in Java?
A - Relational operators return the boolean type in Java.

Q 21- Show and describe six different relational operators supported by Java.
A - Java supports the following set of relational operators:

Operator	Returns true if
>	Left operand is greater than right operand
>=	Left operand is greater than or equal to right operand
<	Left operand is less than right operand
<=	Left operand is less than or equal to right operand
==	Left operand is equal to right operand
!=	Left operand is not equal to right operand

Q 22- Show the output that would be produced by the following Java application:

```
 class prg2 { //define the controlling class
   public static void main(String[] args){ //define main method
     System.out.println("The relational 6<5 is " + (6<5 ) );
     System.out.println("The relational 6>5 is " + (6>5 ) );
   }//end main
 }//End  class.  Note no semicolon required
 //End Java application
```

A - This program produces the following output:

```
The relational 6<5 is false
The relational 6>5 is true
```

Q23 - Show and describe three operators (frequently referred to as conditional operators in Java and logical operators in C++) which are often combined with relational operators to construct more complex expressions (often called conditional expressions). Hint: The && operator returns true if the left and right operands are both true. What are the other two and how do they behave?
A - The following three logical or conditional operators are supported by Java.

Operator	Typical Use	Returns true if
&&	Left && Right	Left and Right are both true
||	Left || Right	Either Left or Right is true
!	! Right	Right is false

Q24 - Describe the special behavior of the || operator in the following expression for the case where the value of the variable a is less than the value of the variable b.
(a < b) || (c < d)
A - An important characteristic of the behavior of the && and || operators in Java is that the expressions are evaluated from left to right, and the evaluation of the expression is terminated as soon as the result of evaluating the expression can be determined. For example, in the above expression, if the variable a is less than the variable b , there is no need to evaluate the right operand of the || to determine the value of the entire expression. Thus, evaluation will terminate as soon as it is determined that a is less than b.

Q25 - Show the symbols used for the bitwise and operator and the or operator.
A - The & operator in Java is a bitwise and and the | operator is a bitwise or.

Q26 - The logical and operator is represented in Java by the && symbol. What is the representation of the bitwise and operator in Java?
A - The bitwise and operator is represented by the & symbol in Java.

Q27 - Show and describe five operators in Java that perform actions on the operands one bit at a time (bitwise operators).
A - The following table shows the seven bitwise operators supported by Java.

Operator	Typical Use	Operation
>>	OpLeft >> Dist	Shift bits of OpLeft right by Dist bits (signed)
<<	OpLeft << Dist	Shift bits of OpLeft left by Dist bits
>>>	OpLeft >>> Dist	Shift bits of OpLeft right by Dist bits (unsigned)
&	OpLeft & OpRight	Bitwise and of the two operands
|	OpLeft | OpRight	Bitwise inclusive or of the two operands
^	OpLeft ^ OpRight	Bitwise exclusive or (xor) of the two operands
~	~ OpRight	Bitwise complement of the right operand (unary)

Q28 - In Java, the signed right shift operation populates the vacated bits with the zeros, while the left shift and the unsigned right shift populate the vacated bits with the sign bit: True or False. If your answer is False, explain why.
A - False: In Java, the signed right shift operation populates the vacated bits with the sign bit, while the left shift and the unsigned right shift populate the vacated bits with zeros.

Q29 - In a signed right-shift operation in Java, the bits shifted off the right end are lost: True or False. If your answer is False, explain why.
A - True: For both Java and C++, bits shifted off the right end are lost.

Q30 - Using the symbols 1 and 0 construct a table showing the four possible combinations of 1 and 0. Using a 1 or a 0, show the result of the bitwise and operation on these four combinations of 1 and 0.
A - The answer is:
1 and 1 produces 1
1 and 0 produces 0
0 and 1 produces 0
0 and 0 produces 0

Q 31- Using the symbols 1 and 0 construct a truth table showing the four possible combinations of 1 and 0. Using a 1 or a 0, show the result of the bitwise inclusive or operation on these four combinations on these four combinations of 1 and 0.
A - The answer for the inclusive or is:
1 or 1 produces 1
1 or 0 produces 1
0 or 1 produces 1
0 or 0 produces 0

Q 32- Using the symbols 1 and 0 construct a truth table showing the four possible combinations of 1 and 0. Using a 1 or a 0, show the result of the bitwise exclusive or operation on these four combinations on these four combinations of 1 and 0.
A - The answer for the exclusive or is:
1 xor 1 produces 0
1 xor 0 produces 1
0 xor 1 produces 1
0 xor 0 produces 0

Q 33- For the exclusive or, if the two bits are different, the result is a 1. If the two bits are the same, the result is a 0. True or False? If your answer is False, explain why.
A - True.

Q 34- Is the assignment operator a unary operator or a binary operator. Select one or the other.
A - The assignment operator is a binary operator.

Q 35- In Java, when using the assignment operator, the value stored in memory and represented by the right operand is copied into the memory represented by the left operand: True or False? If your answer is False, explain why.
A - True.

Q 36- Show two of the shortcut assignment operators and explain how they behave by comparing them with the regular (nonshortcut) versions. Hint: The (^=) operator is a shortcut assignment operator.

A - Java supports the following list of shortcut assignment operators. These operators allow you to perform an assignment and another operation with a single operator. += -= *= /= %= &= |= ^= <<= >>= >>>=

For example, the two statements which follow perform the same operation.

```
x += y;   x = x + y;
```

The behavior of all the shortcut assignment operators follows this same pattern.

Q 37 - Write a Java application illustrates the difference between the prefix and the postfix versions of the increment operator.

```
 class prog3{
   static public void main(String[] args){
     int x = 3;
     int y = 3;
     int z = 10;
     System.out.println("Prefix version gives  " + (z + ++x));
     System.out.println("Postfix version gives " + (z + y++));
   }//end main
 }//end class
```

Q38 Write a Java application that illustrates the use of the following relational operators: < > <= >= == !=

```
class Prog4 { //define the controlling class
   public static void main(String[] args){ //define main method
     System.out.println("The relational 6<5 is " + (6<5 ) );
     System.out.println("The relational 6>5 is " + (6>5 ) );
     System.out.println("The relational 5>=5 is " + (5>=5 ) );
     System.out.println("The relational 5<=5 is " + (5<=5 ) );
     System.out.println("The relational 6==5 is " + (6==5 ) );
     System.out.println("The relational 6!=5 is " + (6!=5 ) );
   }//end main
 }//End prog4 class.  Note no semicolon required
```

Q39 - Write a Java application that illustrates the use of the following logical or conditional operators:
&& || !

```
 class prg5 { //define the controlling class
   public static void main(String[] args){ //define main method
     System.out.println("true and true is " + (true && true) );
     System.out.println("true and false is " + (true && false) );

     System.out.println("true or true is " + (true || true) );
     System.out.println("true or false is " + (true || false) );
     System.out.println("false or false is " + (false || false) );
     System.out.println("not true is " + (! true) );
     System.out.println("not false is " + (! false) );
}//end main
}
```

Q40 - Java supports a constant type: True or False. If false, explain why.

A - Java does not support a constant type. However, in Java, it is possible to achieve the same result by declaring and initializing a variable and making it final.

Q41 Provide a code fragment that illustrates the syntax for creating a named constant in Java.

A - The syntax for creating a named constant in Java is as follows: final float PI = 3.14159;

Q42 - What is the common method of controlling the order of evaluation of expressions in Java?

A - you can control the order of evaluation by the use of parentheses.

Q43 - What are the three actions normally involved in the operation of a loop (in addition to executing the code in the body of the loop)?

A - The operation of a loop normally involves the following three actions in addition to executing the code in the body of the loop:

Initialize a control variable.
Test the control variable in a conditional expression.
Update the control variable.

1.The Java interpreter is used for the execution of the source code.
True
False
Ans: a.
2) On successful compilation a file with the class extension is created.
a) True
b) False
Ans: a.
3) The Java source code can be created in a Notepad editor.
a) True
b) False
Ans: a.
4) The Java Program is enclosed in a class definition.
a) True
b) False
Ans: a.
5) What declarations are required for every Java application?
Ans: A class and the main( ) method declarations.
6) What are the two parts in executing a Java program and their purposes?
Ans: Two parts in executing a Java program are:
Java Compiler and Java Interpreter.
The Java Compiler is used for compilation and the Java Interpreter is used for execution of the application.
7) What are the three OOPs principles and define them?
Ans : Encapsulation, Inheritance and Polymorphism are the three OOPs
Principles.
Encapsulation:
Is the Mechanism that binds together code and the data it manipulates, and keeps both safe from outside interference and misuse.
Inheritance:
Is the process by which one object acquires the properties of another object.
Polymorphism:
Is a feature that allows one interface to be used for a general class of actions.

8) What is a compilation unit?
Ans : Java source code file.
9) What output is displayed as the result of executing the following statement?
System.out.println("// Looks like a comment.");
// Looks like a comment
The statement results in a compilation error
Looks like a comment
No output is displayed
Ans : a.
10) In order for a source code file, containing the public class Test, to successfully compile, which of the following must be true?
It must have a package statement
It must be named Test.java
It must import java.lang
It must declare a public class named Test
Ans : b
11) What are identifiers and what is naming convention?
Ans : Identifiers are used for class names, method names and variable names. An identifier may be any descriptive sequence of upper case & lower case letters,numbers or underscore or dollar sign and must not begin with numbers.
12) What is the return type of program's main( ) method?
Ans : void
13) What is the argument type of program's main( ) method?
Ans : string array.
14) Which characters are as first characters of an identifier?
Ans : A – Z, a – z, _ ,$
15) What are different comments?
Ans : 1) // -- single line comment
2) /* --
*/ multiple line comment
3) /** --
*/ documentation
16) What is the difference between constructor method and method?
Ans : Constructor will be automatically invoked when an object is created. Whereas method has to be call explicitly.

17) What is the use of bin and lib in JDK?
Ans : Bin contains all tools such as javac, applet viewer, awt tool etc., whereas Lib
contains all packages and variables.

Data types,variables and Arrays
1) What is meant by variable?
Ans: Variables are locations in memory that can hold values. Before assigning any value to a variable, it must be declared.
2) What are the kinds of variables in Java? What are their uses?
Ans: Java has three kinds of variables namely, the instance variable, the local variable and the class variable.
Local variables are used inside blocks as counters or in methods as temporary variables and are used to store information needed by a single method.
Instance variables are used to define attributes or the state of a particular object and are used to store information needed by multiple methods in the objects.
Class variables are global to a class and to all the instances of the class and are useful for communicating between different objects of all the same class or keeping track of global states.
3) How are the variables declared?
Ans: Variables can be declared anywhere in the method definition and can be initialized during their declaration.They are commonly declared before usage at the beginning of the definition.
Variables with the same data type can be declared together. Local variables must be given a value before usage.
4) What are variable types?
Ans: Variable types can be any data type that java supports, which includes the eight primitive data types, the name of a class or interface and an array.
5) How do you assign values to variables?
Ans: Values are assigned to variables using the assignment operator =.
6) What is a literal? How many types of literals are there?
Ans: A literal represents a value of a certain type where the type describes how that value behaves.
There are different types of literals namely number literals, character literals,
boolean literals, string literals,etc.
7) What is an array?
Ans: An array is an object that stores a list of items.
8) How do you declare an array?
Ans: Array variable indicates the type of object that the array holds.
Ex: int arr[];
9) Java supports multidimensional arrays.
a)True
b)False
Ans: a.
10) An array of arrays can be created.
a)True
b)False
Ans: a.
11) What is a string?
Ans: A combination of characters is called as string.
12) Strings are instances of the class String.
a)True
b)False
Ans: a.
13) When a string literal is used in the program, Java automatically creates instances of the string class.
a)True
b)False
Ans: a.
14) Which operator is to create and concatenate string?
Ans: Addition operator(+).
15) Which of the following declare an array of string objects?
String[ ] s;
String [ ]s:
String[s]:
String s[ ]:
Ans : a, b and d
16) What is the value of a[3] as the result of the following array declaration?
1
2
3
4
Ans : d

17) Which of the following are primitive types?
byte
String
integer
Float
Ans : a.
18) What is the range of the char type?
0 to 216
0 to 215
0 to 216-1
0 to 215-1
Ans. d
19) What are primitive data types?
Ans : byte, short, int, long
float, double
boolean
char
20) What are default values of different primitive types?
Ans : int - 0
short - 0
byte - 0
long - 0 l
float - 0.0 f
double - 0.0 d
boolean - false
char - null
21) Converting of primitive types to objects can be explicitly.
a)True
b)False
Ans: b.
22) How do we change the values of the elements of the array?
Ans : The array subscript expression can be used to change the values of the elements of the array.
23) What is final varaible?
Ans : If a variable is declared as final variable, then you can not change its value. It becomes constant.
24) What is static variable?
Ans : Static variables are shared by all instances of a class.

Operators
1) What are operators and what are the various types of operators available in Java?
Ans: Operators are special symbols used in expressions.
The following are the types of operators:
Arithmetic operators,
Assignment operators,
Increment & Decrement operators,
Logical operators,
Biwise operators,
Comparison/Relational operators and
Conditional operators
2) The ++ operator is used for incrementing and the -- operator is used for
decrementing.
a)True
b)False
Ans: a.
3) Comparison/Logical operators are used for testing and magnitude.
a)True
b)False
Ans: a.
4) Character literals are stored as unicode characters.
a)True
b)False
Ans: a.
5) What are the Logical operators?
Ans: OR(|), AND(&), XOR(^) AND NOT(~).
6) What is the % operator?

Ans : % operator is the modulo operator or reminder operator. It returns the reminder of dividing the first operand by second operand.
7) What is the value of 111 % 13?
3
5
7
9
Ans : c.
8) Is &&= a valid operator?
Ans : No.
9) Can a double value be cast to a byte?
Ans : Yes
10) Can a byte object be cast to a double value ?
Ans : No. An object cannot be cast to a primitive value.
11) What are order of precedence and associativity?
Ans : Order of precedence the order in which operators are evaluated in expressions.
Associativity determines whether an expression is evaluated left-right or right-left.
12) Which Java operator is right associativity?
Ans : = operator.
13) What is the difference between prefix and postfix of -- and ++ operators?
Ans : The prefix form returns the increment or decrement operation and returns the value of the increment or decrement operation.
The postfix form returns the current value of all of the expression and then
performs the increment or decrement operation on that value.
14) What is the result of expression 5.45 + "3,2"?
The double value 8.6
The string ""8.6"
The long value 8.
The String "5.453.2"
Ans : d
15) What are the values of x and y ?
x = 5; y = ++x;
Ans : x = 6; y = 6
16) What are the values of x and z?
x = 5; z = x++;
Ans : x = 6; z = 5

Control Statements
1) What are the programming constructs?
Ans: a) Sequential
b) Selection -- if and switch statements
c) Iteration -- for loop, while loop and do-while loop
2) class conditional {
public static void main(String args[]) {
int i = 20;
int j = 55;
int z = 0;
z = i < j ? i : j; // ternary operator
System.out.println("The value assigned is " + z);
}
}
What is output of the above program?
Ans: The value assigned is 20
3) The switch statement does not require a break.
a)True
b)False
Ans: b.
4) The conditional operator is otherwise known as the ternary operator.
a)True
b)False
Ans: a.

5) The while loop repeats a set of code while the condition is false.
a)True
b)False
Ans: b.
6) The do-while loop repeats a set of code atleast once before the condition is tested.
a)True
b)False
Ans: a.
7) What are difference between break and continue?
Ans: The break keyword halts the execution of the current loop and forces control out of the loop.
The continue is similar to break, except that instead of halting the execution of the loop, it starts the next iteration.

8) The for loop repeats a set of statements a certain number of times until a condition is matched.
a)True
b)False
Ans: a.
9) Can a for statement loop indefintely?
Ans : Yes.
10) What is the difference between while statement and a do statement/
Ans : A while statement checks at the beginning of a loop to see whether the next loop iteration should occur.
A do statement checks at the end of a loop to see whether the next iteration of a loop should occur. The do statement will always execute the body of a loop at least once.

Introduction to Classes and Methods
1) Which is used to get the value of the instance variables?
Ans: Dot notation.
2) The new operator creates a single instance named class and returns a
reference to that object.
a)True
b)False
Ans: a.
3) A class is a template for multiple objects with similar features.
a)True
b)False
Ans: a.
4) What is mean by garbage collection?
Ans: When an object is no longer referred to by any variable, Java automatically
reclaims memory used by that object. This is known as garbage collection.
5) What are methods and how are they defined?
Ans: Methods are functions that operate on instances of classes in which they are defined.Objects can communicate with each other using methods and can call methods in other classes.
Method definition has four parts. They are name of the method, type of object or primitive type the method returns, a list of parameters and the body of the method.
A method's signature is a combination of the first three parts mentioned above.
6) What is calling method?
Ans: Calling methods are similar to calling or referring to an instance variable. These methods are accessed using dot notation.
Ex: obj.methodname(param1,param2)
7) Which method is used to determine the class of an object?
Ans: getClass( ) method can be used to find out what class the belongs to. This class is defined in the object class and is available to all objects.
8) All the classes in java.lang package are automatically imported when
a program is compiled.
a)True
b)False
Ans: a.
9) How can class be imported to a program?
Ans: To import a class, the import keyword should be used as shown.;
import classname;
10) How can class be imported from a package to a program?
Ans: import java . packagename . classname (or) import java.package name.*;
11) What is a constructor?
Ans: A constructor is a special kind of method that determines how an object is
initialized when created.
12) Which keyword is used to create an instance of a class?
Ans: new.

13) Which method is used to garbage collect an object?
Ans: finalize ().
14) Constructors can be overloaded like regular methods.
a)True
b)False
Ans: a.
15) What is casting?
Ans: Casting is bused to convert the value of one type to another.

16) Casting between primitive types allows conversion of one primitive type to another.
a)True
b)False
Ans: a.
17) Casting occurs commonly between numeric types.
a)True
b)False
Ans: a.
18) Boolean values can be cast into any other primitive type.
a)True
b)False
Ans: b.
19) Casting does not affect the original object or value.
a)True
b)False
Ans: a.
20) Which cast must be used to convert a larger value into a smaller one?
Ans: Explicit cast.
21) Which cast must be used to cast an object to another class?
Ans: Specific cast.
22) Which of the following features are common to both Java & C++?
A.The class declaration
b.The access modifiers
c.The encapsulation of data & methods with in objects
d.The use of pointers
Ans: a,b,c.
23) Which of the following statements accurately describe the use of access modifiers within a class definition?
a.They can be applied to both data & methods
b.They must precede a class's data variables or methods
c.They can follow a class's data variables or methods
d.They can appear in any order
e.They must be applied to data variables first and then to methods
Ans: a,b,d.
24) Suppose a given instance variable has been declared private.
Can this instance variable be manipulated by methods out side its class?
a.yes
b.no
Ans: b.
25) Which of the following statements can be used to describe a public method?
a.It is accessible to all other classes in the hierarchy
b.It is accessablde only to subclasses of its parent class
c.It represents the public interface of its class
d.The only way to gain access to this method is by calling one of the public class
methods
Ans: a,c.
26) Which of the following types of class members can be part of the internal part of a class?
a.Public instance variables
b.Private instance variables
c.Public methods
d.Private methods
Ans: b,d.
27) You would use the ____ operator to create a single instance of a named class.
a.new
b.dot
Ans: a.

28) Which of the following statements correctly describes the relation between an object and the instance variable it stores?
a.Each new object has its own distinctive set of instance variables
b.Each object has a copy of the instance variables of its class
c.the instance variable of each object are seperate from the variables of other objects
d.The instance variables of each object are stored together with the variables of other objects
Ans: a,b,c.
29) If no input parameters are specified in a method declaration then the declaration will include __.
a.an empty set of parantheses
b.the term void
Ans: a.
30) What are the functions of the dot(.) operator?
a.It enables you to access instance variables of any objects within a class
b.It enables you to store values in instance variables of an object
c.It is used to call object methods
d.It is to create a new object
Ans: a,b,c.
31) Which of the following can be referenced by this variable?
a.The instance variables of a class only
b.The methods of a class only
c.The instance variables and methods of a class
Ans: c.
32) The this reference is used in conjunction with ___methods.
a.static
b.non-static
Ans: b.
33) Which of the following operators are used in conjunction with the this and super references?
a.The new operator
b.The instanceof operator
c.The dot operator
Ans: c.
34) A constructor is automatically called when an object is instantiated
a. true
b. false
Ans: a.
35) When may a constructor be called without specifying arguments?
a. When the default constructor is not called
b. When the name of the constructor differs from that of the class
c. When there are no constructors for the class
Ans: c.
36) Each class in java can have a finalizer method
a. true
b.false
Ans: a.
37) When an object is referenced, does this mean that it has been identified by the finalizer method for garbage collection?
a.yes
b.no
Ans: b.
38) Because finalize () belongs to the java.lang.Object class, it is present in all ___.
a.objects
b.classes
c.methods
Ans: b.
39) Identify the true statements about finalization.
a.A class may have only one finalize method
b.Finalizers are mostly used with simple classes
c.Finalizer overloading is not allowed
Ans: a,c.
40) When you write finalize() method for your class, you are overriding a finalizer
inherited from a super class.
a.true
b.false
Ans: a.
41) Java memory management mechanism garbage collects objects which are no longer referenced
a true
b.false
Ans: a.

42) are objects referenced by a variable candidates for garbage collection when the variable goes out of scope?
a yes
b. no
Ans: a.
43) Java's garbage collector runs as a ___ priority thread waiting for __priority threads to relinquish the processor.
a.high
b.low
Ans: a,b.
44) The garbage collector will run immediately when the system is out of memory
a.true
b.false
Ans: a.
45) You can explicitly drop a object reference by setting the value of a variable whose data type is a reference type to ___
Ans: null
46) When might your program wish to run the garbage collecter?
a. before it enters a compute-intense section of code
b. before it enters a memory-intense section of code
c. before objects are finalized
d. when it knows there will be some idle time
Ans: a,b,d
47) For externalizable objects the class is solely responsible for the external format of its contents
a.true
b.false
Ans: a
48) When an object is stored, are all of the objects that are reachable from that object stored as well?
a.true
b.false
Ans: a
49) The default__ of objects protects private and trancient data, and supports the __ of the classes
a.evolution
b.encoding
Ans: b,a.
50) Which are keywords in Java?
a) NULL
b) sizeof
c) friend
d) extends
e) synchronized
Ans : d and e
51) When must the main class and the file name coincide?
Ans :When class is declared public.
52) What are different modifiers?
Ans : public, private, protected, default, static, trancient, volatile, final, abstract.
53) What are access modifiers?
Ans : public, private, protected, default.
54) What is meant by "Passing by value" and " Passing by reference"?
Ans : objects – pass by referrence
Methods - pass by value
55) Is a class a subclass of itself?
Ans : A class is a subclass itself.

56) What modifiers may be used with top-level class?
Ans : public, abstract, final.
57) What is an example of polymorphism?
Inner class
Anonymous classes
Method overloading
Method overriding
Ans : c

Packages and interface
1) What are packages ? what is use of packages ?
Ans :The package statement defines a name space in which classes are stored.If you omit the package, the classes are put into the default package.
Signature... package pkg;

Use: * It specifies to which package the classes defined in a file belongs to. * Package is both naming and a visibility control mechanism.
2) What is difference between importing "java.applet.Applet" and "java.applet.*;" ?
Ans :"java.applet.Applet" will import only the class Applet from the package java.applet
Where as "java.applet.*" will import all the classes from java.applet package.
3) What do you understand by package access specifier?
Ans : public: Anything declared as public can be accessed from anywhere
private: Anything declared in the private can't be seen outside of its class.
default: It is visible to subclasses as well as to other classes in the same package.
4) What is interface? What is use of interface?
Ans : It is similar to class which may contain method's signature only but not bodies.
Methods declared in interface are abstract methods. We can implement many interfaces on a class which support the multiple inheritance.
5) Is it is necessary to implement all methods in an interface?
Ans : Yes. All the methods have to be implemented.
6) Which is the default access modifier for an interface method?
Ans : public.
7) Can we define a variable in an interface ?and what type it should be ?
Ans : Yes we can define a variable in an interface. They are implicitly final and static.
8) What is difference between interface and an abstract class?
Ans : All the methods declared inside an Interface are abstract. Where as abstract class must have at least one abstract method and others may be concrete or abstract.
In Interface we need not use the keyword abstract for the methods.
9) By default, all program import the java.lang package.
True/False
Ans : True
10) Java compiler stores the .class files in the path specified in CLASSPATH
environmental variable.
True/False
Ans : False

11) User-defined package can also be imported just like the standard packages.
True/False
Ans : True
12) When a program does not want to handle exception, the ______class is used.
Ans : Throws
13) The main subclass of the Exception class is _______ class.
Ans : RuntimeException
14) Only subclasses of ______class may be caught or thrown.
Ans : Throwable
15) Any user-defined exception class is a subclass of the _____ class.
Ans : Exception
16) The catch clause of the user-defined exception class should ______ its
Base class catch clause.
Ans : Exception
17) A _______ is used to separate the hierarchy of the class while declaring an
Import statement.
Ans : Package

18) All standard classes of Java are included within a package called _____.
Ans : java.lang
19) All the classes in a package can be simultaneously imported using ____.
Ans : *
20) Can you define a variable inside an Interface. If no, why? If yes, how?
Ans.: YES. final and static
21) How many concrete classes can you have inside an interface?
Ans.: None
22) Can you extend an interface?
Ans.: Yes
23) Is it necessary to implement all the methods of an interface while implementing the interface?
Ans.: No
24) If you do not implement all the methods of an interface while implementing , what specifier should you use for the class ?
Ans.: abstract
25) How do you achieve multiple inheritance in Java?
Ans: Using interfaces.
26) How to declare an interface example?

Ans : access class classname implements interface.
27) Can you achieve multiple interface through interface?
a)True
b) false
Ans : a.
28) Can variables be declared in an interface ? If so, what are the modifiers?
Ans : Yes. final and static are the modifiers can be declared in an interface.
29) What are the possible access modifiers when implementing interface methods?
Ans : public.
30) Can anonymous classes be implemented an interface?
Ans : Yes.
31) Interfaces can't be extended.
a)True
b)False
Ans : b.
32) Name interfaces without a method?
Ans : Serializable, Cloneble & Remote.
33) Is it possible to use few methods of an interface in a class ? If so, how?
Ans : Yes. Declare the class as abstract.

Exception Handling
1) What is the difference between 'throw' and 'throws' ?And it's application?
Ans : Exceptions that are thrown by java runtime systems can be handled by Try and catch blocks. With throw exception we can handle the exceptions thrown by the program itself. If a method is capable of causing an exception that it does not
handle, it must specify this behavior so the callers of the method can guard
against that exception.
2) What is the difference between 'Exception' and 'error' in java?
Ans : Exception and Error are the subclasses of the Throwable class. Exception class is used for exceptional conditions that user program should catch. With exception class we can subclass to create our own custom exception.
Error defines exceptions that are not excepted to be caught by you program. Example is Stack Overflow.
3) What is 'Resource leak'?
Ans : Freeing up other resources that might have been allocated at the beginning of a method.
4)What is the 'finally' block?
Ans : Finally block will execute whether or not an exception is thrown. If an exception is thrown, the finally block will execute even if no catch statement match the exception. Any time a method is about to return to the caller from inside try/catch block, via an uncaught exception or an explicit return statement, the finally clause is also execute.
5) Can we have catch block with out try block? If so when?
Ans : No. Try/Catch or Try/finally form a unit.
6) What is the difference between the following statements?
Catch (Exception e),
Catch (Error err),
Catch (Throwable t)
Ans :

7) What will happen to the Exception object after exception handling?
Ans : It will go for Garbage Collector. And frees the memory.
8) How many Exceptions we can define in 'throws' clause?
Ans : We can define multiple exceptions in throws clause.
Signature is..
type method-name (parameter-list) throws exception-list

9) The finally block is executed when an exception is thrown, even if no catch matches it.
True/False
Ans : True
10) The subclass exception should precede the base class exception when used within the catch clause.
True/False
Ans : True
11) Exceptions can be caught or rethrown to a calling method.
True/False
Ans : True
12) The statements following the throw keyword in a program are not executed.
True/False
Ans : True
13) The toString ( ) method in the user-defined exception class is overridden.

True/False
Ans : True

MULTI THREADING
1) What are the two types of multitasking?
Ans : 1.process-based
2.Thread-based
2) What are the two ways to create the thread?
Ans : 1.by implementing Runnable
2.by extending Thread
3) What is the signature of the constructor of a thread class?
Ans : Thread(Runnable threadob,String threadName)
4) What are all the methods available in the Runnable Interface?
Ans : run()
5) What is the data type for the method isAlive() and this method is
available in which class?
Ans : boolean, Thread
6) What are all the methods available in the Thread class?
Ans : 1.isAlive()
2.join()
3.resume()
4.suspend()
5.stop()
6.start()
7.sleep()
8.destroy()
7) What are all the methods used for Inter Thread communication and what is the class in which these methods are defined?
Ans :1. wait(),notify() & notifyall()
2. Object class
8) What is the mechanisam defind by java for the Resources to be used by only one Thread at a time?
Ans : Synchronisation
9) What is the procedure to own the moniter by many threads?
Ans : not possible
10) What is the unit for 1000 in the below statement?
ob.sleep(1000)
Ans : long milliseconds
11) What is the data type for the parameter of the sleep() method?
Ans : long
12) What are all the values for the following level?
max-priority
min-priority
normal-priority
Ans : 10,1,5
13) What is the method available for setting the priority?
Ans : setPriority()
14) What is the default thread at the time of starting the program?
Ans : main thread
15) The word synchronized can be used with only a method.
True/ False
Ans : False
16) Which priority Thread can prompt the lower primary Thread?
Ans : Higher Priority
17) How many threads at a time can access a monitor?
Ans : one
18) What are all the four states associated in the thread?

Ans : 1. new 2. runnable 3. blocked 4. dead
19) The suspend()method is used to teriminate a thread?
True /False
Ans : False
20) The run() method should necessary exists in clases created as subclass of thread?
True /False
Ans : True
21) When two threads are waiting on each other and can't proceed the programe is said to be in a deadlock?
True/False
Ans : True
22) Which method waits for the thread to die ?
Ans : join() method

23) Which of the following is true?
1) wait(),notify(),notifyall() are defined as final & can be called only from with in a synchronized method
2) Among wait(),notify(),notifyall() the wait() method only throws IOException
3) wait(),notify(),notifyall() & sleep() are methods of object class
1
2
3
1 & 2
1,2 & 3
Ans : D
24) Garbage collector thread belongs to which priority?
Ans : low-priority
25) What is meant by timeslicing or time sharing?
Ans : Timeslicing is the method of allocating CPU time to individual threads in a priority schedule.
26) What is meant by daemon thread? In java runtime, what is it's role?
Ans : Daemon thread is a low priority thread which runs intermittently in the background doing the garbage collection operation for the java runtime system.

Inheritance
1) What is the difference between superclass & subclass?
Ans : A super class is a class that is inherited whereas subclass is a class that does the inheriting.
2) Which keyword is used to inherit a class?
Ans : extends
3) Subclasses methods can access superclass members/ attributes at all times?
True/False
Ans : False

4) When can subclasses not access superclass members?
Ans : When superclass is declared as private.
5) Which class does begin Java class hierarchy?
Ans : Object class
6) Object class is a superclass of all other classes?
True/False
Ans : True
7) Java supports multiple inheritance?
True/False
Ans : False
8) What is inheritance?
Ans : Deriving an object from an existing class. In the other words, Inheritance is the process of inheriting all the features from a class
9) What are the advantages of inheritance?
Ans : Reusability of code and accessibility of variables and methods of the superclass by subclasses.
10) Which method is used to call the constructors of the superclass from the subclass?
Ans : super(argument)
11) Which is used to execute any method of the superclass from the subclass?
Ans : super.method-name(arguments)
12) Which methods are used to destroy the objects created by the constructor methods?
Ans : finalize()
13) What are abstract classes?
Ans : Abstract classes are those for which instances can't be created.

14) What must a class do to implement an interface?
Ans: It must provide all of the methods in the interface and identify the interface in its implements clause.
15) Which methods in the Object class are declared as final?
Ans : getClass(), notify(), notifyAll(), and wait()
16) Final methods can be overridden.
True/False
Ans : False
17) Declaration of methods as final results in faster execution of the program?
True/False
Ans: True
18) Final variables should be declared in the beginning?
True/False
Ans : True
19) Can we declare variable inside a method as final variables? Why?
Ans : Cannot because, local variable cannot be declared as final variables.
20) Can an abstract class may be final?
Ans : An abstract class may not be declared as final.
21) Does a class inherit the constructors of it's super class?
Ans: A class does not inherit constructors from any of it's super classes.
22) What restrictions are placed on method overloading?
Ans: Two methods may not have the same name and argument list but different return types.
23) What restrictions are placed on method overriding?
Ans : Overridden methods must have the same name , argument list , and return type. The overriding method may not limit the access of the method it overridees.The overriding method may not throw any exceptions that may not be thrown by the overridden method.
24) What modifiers may be used with an inner class that is a member of an outer class?
Ans : a (non-local) inner class may be declared as public, protected, private, static, final or abstract.
25) How this() is used with constructors?
Ans: this() is used to invoke a constructor of the same class
26) How super() used with constructors?
Ans : super() is used to invoke a super class constructor
27) Which of the following statements correctly describes an interface?
a)It's a concrete class
b)It's a superclass
c)It's a type of abstract class
Ans: c
28) An interface contains __ methods
a)Non-abstract
b)Implemented
c)unimplemented
Ans:c
STRING HANDLING
Which package does define String and StringBuffer classes?
Ans : java.lang package.
Which method can be used to obtain the length of the String?
Ans : length( ) method.
How do you concatenate Strings?
Ans : By using " + " operator.
Which method can be used to compare two strings for equality?
Ans : equals( ) method.
Which method can be used to perform a comparison between strings that ignores case differences?
Ans : equalsIgnoreCase( ) method.
What is the use of valueOf( ) method?
Ans : valueOf( ) method converts data from its internal format into a human-readable form.
What are the uses of toLowerCase( ) and toUpperCase( ) methods?
Ans : The method toLowerCase( ) converts all the characters in a string from uppercase to
lowercase.
The method toUpperCase( ) converts all the characters in a string from lowercase to
uppercase.
Which method can be used to find out the total allocated capacity of a StrinBuffer?
Ans : capacity( ) method.
Which method can be used to set the length of the buffer within a StringBuffer object?
Ans : setLength( ).
What is the difference between String and StringBuffer?
Ans : String objects are constants, whereas StringBuffer objects are not.
String class supports constant strings, whereas StringBuffer class supports growable, modifiable strings.

What are wrapper classes?
Ans : Wrapper classes are classes that allow primitive types to be accessed as objects.
Which of the following is not a wrapper class?
String
Integer
Boolean
Character
Ans : a.
What is the output of the following program?

```
public class Question {
public static void main(String args[]) {
String s1 = "abc";
String s2 = "def";
String s3 = s1.concat(s2.toUpperCase( ) );
System.out.println(s1+s2+s3);
}
}
```

abcdefabcdef
abcabcDEFDEF
abcdefabcDEF
None of the above
ANS : c.
Which of the following methods are methods of the String class?
delete( )
append( )
reverse( )
replace( )
Ans : d.
Which of the following methods cause the String object referenced by s to be changed?
s.concat( )
s.toUpperCase( )
s.replace( )
s.valueOf( )
Ans : a and b.
String is a wrapper class?
True
False
Ans : b.
17) If you run the code below, what gets printed out?

```
String s=new String("Bicycle");
int iBegin=1;
char iEnd=3;
System.out.println(s.substring(iBegin,iEnd));
```

Bic
ic
c) icy
d) error: no method matching substring(int,char)
Ans : b.
18) Given the following declarations

```
String s1=new String("Hello")
String s2=new String("there");
String s3=new String();
```

Which of the following are legal operations?
s3=s1 + s2;
s3=s1 - s2;
c) s3=s1 & s2
d) s3=s1 && s2
Ans : a.
19) Which of the following statements are true?
The String class is implemented as a char array, elements are addressed using the stringname[] convention
b) Strings are a primitive type in Java that overloads the + operator for concatenation
c) Strings are a primitive type in Java and the StringBuffer is used as the matching wrapper type
d) The size of a string can be retrieved using the length property.
Ans : b.

EXPLORING JAVA.LANG
java.lang package is automatically imported into all programs.
True
False
Ans : a
What are the interfaces defined by java.lang?
Ans : Cloneable, Comparable and Runnable.
What are the constants defined by both Flaot and Double classes?
Ans : MAX_VALUE,
MIN_VALUE,
NaN,
POSITIVE_INFINITY,
NEGATIVE_INFINITY and
TYPE.
What are the constants defined by Byte, Short, Integer and Long?
Ans : MAX_VALUE,
MIN_VALUE and
TYPE.
What are the constants defined by both Float and Double classes?
Ans : MAX_RADIX,
MIN_RADIX,
MAX_VALUE,
MIN_VALUE and
TYPE.
What is the purpose of the Runtime class?
Ans : The purpose of the Runtime class is to provide access to the Java runtime system.
What is the purpose of the System class?
Ans : The purpose of the System class is to provide access to system resources.
Which class is extended by all other classes?
Ans : Object class is extended by all other classes.
Which class can be used to obtain design information about an object?
Ans : The Class class can be used to obtain information about an object's design.
Which method is used to calculate the absolute value of a number?
Ans : abs( ) method.
What are E and PI?
Ans : E is the base of the natural logarithm and PI is the mathematical value pi.
Which of the following classes is used to perform basic console I/O?
System
SecurityManager
Math
Runtime
Ans : a.
Which of the following are true?
The Class class is the superclass of the Object class.
The Object class is final.
The Class class can be used to load other classes.
The ClassLoader class can be used to load other classes.
Ans : c and d.
Which of the following methods are methods of the Math class?
absolute( )
log( )
cosine( )
sine( )
Ans : b.
Which of the following are true about the Error and Exception classes?
Both classes extend Throwable.
The Error class is final and the Exception class is not.
The Exception class is final and the Error is not.
Both classes implement Throwable.
Ans : a.
Which of the following are true?
The Void class extends the Class class.
The Float class extends the Double class.
The System class extends the Runtime class.
The Integer class extends the Number class.

Ans : d.

17) Which of the following will output -4.0
System.out.println(Math.floor(-4.7));
System.out.println(Math.round(-4.7));
System.out.println(Math.ceil(-4.7));
d) System.out.println(Math.Min(-4.7));
Ans : c.
18) Which of the following are valid statements
a) public class MyCalc extends Math
b) Math.max(s);
c) Math.round(9.99,1);
d) Math.mod(4,10);
e) None of the above.
Ans : e.
19) What will happen if you attempt to compile and run the following code?
Integer ten=new Integer(10);
Long nine=new Long (9);
System.out.println(ten + nine);
int i=1;
System.out.println(i + ten);
19 followed by 20
19 followed by 11
Error: Can't convert java lang Integer
d) 10 followed by 1
Ans : c.

INPUT / OUTPUT : EXPLORING JAVA.IO
What is meant by Stream and what are the types of Streams and classes of the Streams?
Ans : A Stream is an abstraction that either produces or consumes information.
There are two types of Streams. They are:
Byte Streams : Byte Streams provide a convenient means for handling input and output of bytes.
Character Streams : Character Streams provide a convenient means for handling input and output of characters.
Byte Stream classes : Byte Streams are defined by using two abstract classes. They are:InputStream and OutputStream.
Character Stream classes : Character Streams are defined by using two abstract classes. They are : Reader and Writer.
Which of the following statements are true?
UTF characters are all 8-bits.
UTF characters are all 16-bits.
UTF characters are all 24-bits.
Unicode characters are all 16-bits.
Bytecode characters are all 16-bits.
Ans : d.
Which of the following statements are true?
When you construct an instance of File, if you do not use the filenaming semantics of the local machine, the constructor will throw an IOException.
When you construct an instance of File, if the corresponding file does not exist on the local file system, one will be created.
When an instance of File is garbage collected, the corresponding file on the local file system is deleted.
None of the above.
Ans : a,b and c.
The File class contains a method that changes the current working directory.
True
False
Ans : b.
It is possible to use the File class to list the contents of the current working directory.
True
False
Ans : a.
Readers have methods that can read and return floats and doubles.
True
False
Ans : b.
You execute the code below in an empty directory. What is the result?
File f1 = new File("dirname");
File f2 = new File(f1, "filename");

A new directory called dirname is created in the current working directory.
A new directory called dirname is created in the current working directory. A new file called filename is created in directory dirname.
A new directory called dirname and a new file called filename are created, both in the current working directory.
A new file called filename is created in the current working directory.
No directory is created, and no file is created.
Ans : e.
What is the difference between the Reader/Writer class hierarchy and the
InputStream/OutputStream class hierarchy?
Ans : The Reader/Writer class hierarchy is character-oriented and the InputStream/OutputStream class hierarchy is byte-oriented.
What is an I/O filter?
Ans : An I/O filter is an object that reads from one stream and writes to another, usually altering the data in some way as it is passed from one stream to another.
What is the purpose of the File class?
Ans : The File class is used to create objects that provide access to the files and directories of a local file system.
What interface must an object implement before it can be written to a stream as an object?
Ans : An object must implement the Serializable or Externalizable interface before it can be written to a stream as an object.
What is the difference between the File and RandomAccessFile classes?
Ans : The File class encapsulates the files and directories of the local file system. The RandomAccessFile class provides the methods needed to directly access data contained in any part of a file.
What class allows you to read objects directly from a stream?
Ans : The ObjectInputStream class supports the reading of objects from input streams.
What value does read( ) return when it has reached the end of a file?
Ans : The read( ) method returns – 1 when it has reached the end of a file.
What value does readLine( ) return when it has reached the end of a file?
Ans : The readLine( ) method returns null when it has reached the end of a file.
How many bits are used to represent Unicode, ASCII, UTF-16 and UTF-8 characters?
Ans : Unicode requires 16-bits and ASCII requires 8-bits. Although the ASCII character set uses only 1-bits, it is usually represented as 8-bits. UTF-8 represents characters using 8, 16 and 18-bit patterns. UTF-16 uses 16-bit and larger bit patterns.
Which of the following are true?
The InputStream and OutputStream classes are byte-oriented.
The ObjectInputStream and ObjectOutputStream do not support serialized object input and output.
The Reader and Writer classes are character-oriented.
The Reader and Writer classes are the preferred solution to serialized object output.
Ans : a and c.
Which of the following are true about I/O filters?
Filters are supported on input, but not on output.
Filters are supported by the InputStream/OutputStream class hierarchy, but not by the Reader/Writer class hierarchy.
Filters read from one stream and write to another.
A filter may alter data that is read from one stream and written to another.
Ans : c and d.
Which of the following are true?
Any Unicode character is represented using 16-bits.
7-bits are needed to represent any ASCII character.
UTF-8 characters are represented using only 8-bits.
UTF-16 characters are represented using only 16-bits.
Ans : a and b.
Which of the following are true?
The Serializable interface is used to identify objects that may be written to an output stream.
The Externalizable interface is implemented by classes that control the way in which their objects are serialized.
The Serializable interface extends the Externalizable interface.
The Externalizable interface extends the Serializable interface.
Ans : a, b and d.
Which of the following are true about the File class?
A File object can be used to change the current working directory.
A File object can be used to access the files in the current directory.
When a File object is created, a corresponding directory or file is created in the local file system.
File objects are used to access files and directories on the local file system.
File objects can be garbage collected.
When a File object is garbage collected, the corresponding file or directory is deleted.
Ans : b, d and e.
How do you create a Reader object from an InputStream object?
Use the static createReader( ) method of InputStream class.
Use the static createReader( ) method of Reader class.
Create an InputStreamReader object, passing the InputStream object as an argument to the InputStreamReader constructor.
Create an OutputStreamReader object, passing the InputStream object as an argument to the OutputStreamReader constructor.

Ans : c.
Which of the following are true?
Writer classes can be used to write characters to output streams using different character encodings.
Writer classes can be used to write Unicode characters to output streams.
Writer classes have methods that support the writing of the values of any Java primitive type to output streams.
Writer classes have methods that support the writing of objects to output streams.
Ans : a and b.
The isFile( ) method returns a boolean value depending on whether the file object is a file or a directory.
True.
False.
Ans : a.
Reading or writing can be done even after closing the input/output source.
True.
False.
Ans : b.

The ________ method helps in clearing the buffer.
Ans : flush( ).
The System.err method is used to print error message.
True.
False.
Ans : a.
What is meant by StreamTokenizer?
Ans : StreamTokenizer breaks up InputStream into tokens that are delimited by sets of characters.
It has the constructor : StreamTokenizer(Reader inStream).
Here inStream must be some form of Reader.
What is Serialization and deserialization?
Ans : Serialization is the process of writing the state of an object to a byte stream.
Deserialization is the process of restoring these objects.
30) Which of the following can you perform using the File class?
a) Change the current directory
b) Return the name of the parent directory
c) Delete a file
d) Find if a file contains text or binary information
Ans : b and c.
31)How can you change the current working directory using an instance of the File class called FileName?
FileName.chdir("DirName").
FileName.cd("DirName").
FileName.cwd("DirName").
The File class does not support directly changing the current directory.
Ans : d.

EVENT HANDLING
The event delegation model, introduced in release 1.1 of the JDK, is fully compatible with the
event model.
True
False
Ans : b.
A component subclass that has executed enableEvents( ) to enable processing of a certain kind of event cannot also use an adapter as a listener for the same kind of event.
True
False
Ans : b.
What is the highest-level event class of the event-delegation model?
Ans : The java.util.eventObject class is the highest-level class in the event-delegation hierarchy.
What interface is extended by AWT event listeners?
Ans : All AWT event listeners extend the java.util.EventListener interface.
What class is the top of the AWT event hierarchy?
Ans : The java.awt.AWTEvent class is the highest-level class in the AWT event class hierarchy.
What event results from the clicking of a button?
Ans : The ActionEvent event is generated as the result of the clicking of a button.
What is the relationship between an event-listener interface and an event-adapter class?
Ans : An event-listener interface defines the methods that must be implemented by an event
handler for a particular kind of event.
An event adapter provides a default implementation of an event-listener interface.

In which package are most of the AWT events that support the event-delegation model defined?
Ans : Most of the AWT–related events of the event-delegation model are defined in the java.awt.event package. The AWTEvent class is defined in the java.awt package.
What is the advantage of the event-delegation model over the earlier event-inheritance model?
Ans : The event-delegation has two advantages over the event-inheritance model. They are :
It enables event handling by objects other than the ones that generate the events. This allows a clean separation between a component's design and its use.
It performs much better in applications where many events are generated. This performance improvement is due to the fact that the event-delegation model does not have to repeatedly process unhandled events, as is the case of the event-inheritance model.
What is the purpose of the enableEvents( ) method?
Ans :The enableEvents( ) method is used to enable an event for a particular object.
Which of the following are true?
The event-inheritance model has replaced the event-delegation model.
The event-inheritance model is more efficient than the event-delegation model.
The event-delegation model uses event listeners to define the methods of event-handling classes.
The event-delegation model uses the handleEvent( ) method to support event handling.
Ans : c.
Which of the following is the highest class in the event-delegation model?
java.util.EventListener
java.util.EventObject
java.awt.AWTEvent
java.awt.event.AWTEvent
Ans : b.
When two or more objects are added as listeners for the same event, which listener is first invoked to handle the event?
The first object that was added as listener.
The last object that was added as listener.
There is no way to determine which listener will be invoked first.
It is impossible to have more than one listener for a given event.
Ans : c.
Which of the following components generate action events?
Buttons
Labels
Check boxes
Windows
Ans : a.
Which of the following are true?
A TextField object may generate an ActionEvent.
A TextArea object may generate an ActionEvent.
A Button object may generate an ActionEvent.
A MenuItem object may generate an ActionEvent.
Ans : a,c and d.
Which of the following are true?
The MouseListener interface defines methods for handling mouse clicks.
The MouseMotionListener interface defines methods for handling mouse clicks.
The MouseClickListener interface defines methods for handling mouse clicks.
The ActionListener interface defines methods for handling the clicking of a button.
Ans : a and d.
Suppose that you want to have an object eh handle the TextEvent of a TextArea object t. How should you add eh as the event handler for t?
t.addTextListener(eh);
eh.addTextListener(t);
addTextListener(eh.t);
addTextListener(t,eh);
Ans : a.
What is the preferred way to handle an object's events in Java 2?
Override the object's handleEvent( ) method.
Add one or more event listeners to handle the events.
Have the object override its processEvent( ) methods.
Have the object override its dispatchEvent( ) methods.
Ans : b.
Which of the following are true?
A component may handle its own events by adding itself as an event listener.
A component may handle its own events by overriding its event-dispatching method.
A component may not handle oits own events.

A component may handle its own events only if it implements the handleEvent( ) method.
Ans : a and b.
APPLETS
What is an Applet? Should applets have constructors?
Ans : Applet is a dynamic and interactive program that runs inside a Web page
displayed by a Java capable browser. We don’t have the concept of Constructors in Applets.
How do we read number information from my applet’s parameters, given that Applet’s getParameter() method returns a string?
Ans : Use the parseInt() method in the Integer Class, the Float(String) constructor in the
Class Float, or the Double(String) constructor in the class Double.
How can I arrange for different applets on a web page to communicate with each other?
Ans : Name your applets inside the Applet tag and invoke AppletContext’s getApplet()
method in your applet code to obtain references to the other applets on the page.
How do I select a URL from my Applet and send the browser to that page?
Ans : Ask the applet for its applet context and invoke showDocument() on that context object.
```
Eg. URL targetURL;
String URLString
AppletContext context = getAppletContext();
try{
targetUR L = new URL(URLString);
} catch (Malformed URLException e){
// Code for recover from the exception
}
context. showDocument (targetURL);
```
Can applets on different pages communicate with each other?
Ans : No. Not Directly. The applets will exchange the information at one meeting place
either on the local file system or at remote system.
How do Applets differ from Applications?
Ans : Appln: Stand Alone
Applet: Needs no explicit installation on local m/c.
Appln: Execution starts with main() method.
Applet: Execution starts with init() method.
Appln: May or may not be a GUI
Applet: Must run within a GUI (Using AWT)
How do I determine the width and height of my application?
Ans : Use the getSize() method, which the Applet class inherits from the Component
class in the Java.awt package. The getSize() method returns the size of the applet as
a Dimension object, from which you extract separate width, height fields.
```
Eg. Dimension dim = getSize ();
int appletwidth = dim.width ();
```
8) What is AppletStub Interface?
Ans : The applet stub interface provides the means by which an applet and the browser communicate. Your code will not typically implement this interface.
It is essential to have both the .java file and the .html file of an applet in the same
directory.
True.
False.
Ans : b.
The tag contains two attributes namely _________ and _______.
Ans : Name , value.

Passing values to parameters is done in the _________ file of an applet.
Ans : .html.
12) What tags are mandatory when creating HTML to display an applet
name, height, width
code, name
codebase, height, width
d) code, height, width
Ans : d.
Applet’s getParameter( ) method can be used to get parameter values.
True.
False.
Ans : a.
What are the Applet’s Life Cycle methods? Explain them?
Ans : init( ) method - Can be called when an applet is first loaded.
start( ) method - Can be called each time an applet is started.
paint( ) method - Can be called when the applet is minimized or refreshed.

stop( ) method - Can be called when the browser moves off the applet's page.
destroy( ) method - Can be called when the browser is finished with the applet.
What are the Applet's information methods?
Ans : getAppletInfo( ) method : Returns a string describing the applet, its author ,copy right information, etc.
getParameterInfo( ) method : Returns an array of string describing the applet's parameters.
All Applets are subclasses of Applet.
True.
False.
Ans : a.
All Applets must import java.applet and java.awt.
True.
False.
Ans : a.
What are the steps involved in Applet development?
Ans : a) Edit a Java source file,
b) Compile your program and
c) Execute the appletviewer, specifying the name of your applet's source file.
Applets are executed by the console based Java run-time interpreter.
True.
False.
Ans : b.
Which classes and interfaces does Applet class consist?
Ans : Applet class consists of a single class, the Applet class and three interfaces: AppletContext, AppletStub and AudioClip.
What is the sequence for calling the methods by AWT for applets?
Ans : When an applet begins, the AWT calls the following methods, in this sequence.
init( )
start( )
paint( )
When an applet is terminated, the following sequence of method cals takes place :
stop( )
destroy( )
Which method is used to output a string to an applet?
Ans : drawString ( ) method.
Every color is created from an RGB value.
True.
False
Ans : a.

AWT : WINDOWS, GRAPHICS AND FONTS
How would you set the color of a graphics context called g to cyan?
g.setColor(Color.cyan);
g.setCurrentColor(cyan);
g.setColor("Color.cyan");
g.setColor("cyan');
g.setColor(new Color(cyan));
Ans : a.
The code below draws a line. What color is the line?
g.setColor(Color.red.green.yellow.red.cyan);
g.drawLine(0, 0, 100,100);
Red
Green
Yellow
Cyan
Black
Ans : d.
What does the following code draw?
g.setColor(Color.black);
g.drawLine(10, 10, 10, 50);
g.setColor(Color.RED);
g.drawRect(100, 100, 150, 150);
A red vertical line that is 40 pixels long and a red square with sides of 150 pixels
A black vertical line that is 40 pixels long and a red square with sides of 150 pixels
A black vertical line that is 50 pixels long and a red square with sides of 150 pixels
A red vertical line that is 50 pixels long and a red square with sides of 150 pixels

A black vertical line that is 40 pixels long and a red square with sides of 100 pixel
Ans : b.
Which of the statements below are true?
A polyline is always filled.
b) A polyline can not be filled.
c) A polygon is always filled.
d) A polygon is always closed
e) A polygon may be filled or not filled
Ans : b, d and e.
What code would you use to construct a 24-point bold serif font?
new Font(Font.SERIF, 24,Font.BOLD);
new Font("SERIF", 24, BOLD");
new Font("BOLD ", 24,Font.SERIF);
new Font("SERIF", Font.BOLD,24);
new Font(Font.SERIF, "BOLD", 24);
Ans : d.
What does the following paint( ) method draw?
Public void paint(Graphics g) {
g.drawString("question #6",10,0);
}
The string "question #6", with its top-left corner at 10,0
A little squiggle coming down from the top of the component, a little way in from the left edge
Ans : b.

What does the following paint( ) method draw?
Public void paint(Graphics g) {
g.drawString("question #6",10,0);
}
A circle at (100, 100) with radius of 44
A circle at (100, 44) with radius of 100
A circle at (100, 44) with radius of 44
The code does not compile
Ans : d.
8)What is relationship between the Canvas class and the Graphics class?
Ans : A Canvas object provides access to a Graphics object via its paint( ) method.
What are the Component subclasses that support painting.
Ans : The Canvas, Frame, Panel and Applet classes support painting.
What is the difference between the paint( ) and repaint( ) method?
Ans : The paint( ) method supports painting via a Graphics object. The repaint( ) method is used to cause paint( ) to be invoked by the AWT painting method.
What is the difference between the Font and FontMetrics classes?
Ans : The FontMetrics class is used to define implementation-specific properties, such as ascent and descent, of a Font object.
Which of the following are passed as an argument to the paint( ) method?
A Canvas object
A Graphics object
An Image object
A paint object
Ans : b.
Which of the following methods are invoked by the AWT to support paint and repaint operations?
paint( )
repaint( )
draw( )
redraw( )
Ans : a.
Which of the following classes have a paint( ) method?
Canvas
Image
Frame
Graphics
Ans : a and c.
Which of the following are methods of the Graphics class?
drawRect( )
drawImage( )
drawPoint( )

drawString( )
Ans : a, b and d.
Which Font attributes are available through the FontMetrics class?
ascent
leading
case
height
Ans : a, b and d.
Which of the following are true?
The AWT automatically causes a window to be repainted when a portion of a window has been minimized and then maximized.
The AWT automatically causes a window to be repainted when a portion of a window has been covered and then uncovered.
The AWT automatically causes a window to be repainted when application data is changed.
The AWT does not support repainting operations.
Ans : a and b.
Which method is used to size a graphics object to fit the current size of the window?
Ans : getSize( ) method.
What are the methods to be used to set foreground and background colors?
Ans : setForeground( ) and setBackground( ) methods.
19) You have created a simple Frame and overridden the paint method as follows

```
public void paint(Graphics g){
g.drawString("Dolly",50,10);
}
```

What will be the result when you attempt to compile and run the program?
The string "Dolly" will be displayed at the centre of the frame
b) An error at compilation complaining at the signature of the paint method
c) The lower part of the word Dolly will be seen at the top of the form, with the top hidden.
d) The string "Dolly" will be shown at the bottom of the form
Ans : c.
20) Where g is a graphics instance what will the following code draw on the screen.
g.fillArc(45,90,50,50,90,180);
a) An arc bounded by a box of height 45, width 90 with a centre point of 50,50, starting at an angle of 90 degrees traversing through 180 degrees counter clockwise.
b) An arc bounded by a box of height 50, width 50, with a centre point of 45,90 starting at an angle of 90 degrees traversing through 180 degrees clockwise.
c) An arc bounded by a box of height 50, width 50, with a top left at coordinates of 45, 90, starting at 90 degrees and traversing through 180 degrees counter clockwise.
d) An arc starting at 45 degrees, traversing through 90 degrees clockwise bounded by a box of height 50, width 50 with a centre point of 90, 180.
Ans : c.
21) Given the following code

```
import java.awt.*;
public class SetF extends Frame{
public static void main(String argv[]){
SetF s = new SetF();
s.setSize(300,200);
s.setVisible(true);
}
}
```

How could you set the frame surface color to pink
a)s.setBackground(Color.pink);
b)s.setColor(PINK);
c)s.Background(pink);
d)s.color=Color.pink
Ans : a.

AWT: CONTROLS, LAYOUT MANAGERS AND MENUS
What is meant by Controls and what are different types of controls?
Ans : Controls are componenets that allow a user to interact with your application.
The AWT supports the following types of controls:
Labels
Push buttons
Check boxes
Choice lists

Lists
Scroll bars
Text components
These controls are subclasses of Component.
You want to construct a text area that is 80 character-widths wide and 10 character-heights tall. What code do you use?
new TextArea(80, 10)
new TextArea(10, 80)
Ans: b.
A text field has a variable-width font. It is constructed by calling new
TextField("iiiii"). What happens if you change the contents of the text field to
"wwwww"? (Bear in mind that is one of the narrowest characters, and w is one of the widest.)
The text field becomes wider.
The text field becomes narrower.
The text field stays the same width; to see the entire contents you will have to scroll by using the ß and à keys.
The text field stays the same width; to see the entire contents you will have to scroll by using the text field's horizontal scroll bar.
Ans : c.
The CheckboxGroup class is a subclass of the Component class.
True
False
Ans : b.
5) What are the immediate super classes of the following classes?
a) Container class
b) MenuComponent class
c) Dialog class
d) Applet class
e) Menu class
Ans : a) Container - Component
b) MenuComponent - Object
c) Dialog - Window
d) Applet - Panel
e) Menu - MenuItem
6) What are the SubClass of Textcomponent Class?
Ans : TextField and TextArea
7) Which method of the component class is used to set the position and the size of a component?
Ans : setBounds()
8) Which TextComponent method is used to set a TextComponent to the read-only state?
Ans : setEditable()
9) How can the Checkbox class be used to create a radio button?
Ans : By associating Checkbox objects with a CheckboxGroup.
10) What Checkbox method allows you to tell if a Checkbox is checked?
Ans : getState()
11) Which Component method is used to access a component's immediate Container?
getVisible()
getImmediate
getParent()
getContainer
Ans : c.
12) What methods are used to get and set the text label displayed by a Button object?
Ans : getLabel( ) and setLabel( )
13) What is the difference between a Choice and a List?
Ans : A Choice is displayed in a compact form that requires you to pull it down to see the list of available choices. Only one item may be selected from a Choice.
A List may be displayed in such a way that several List items are visible. A List supports the selection of one or more List items.
14) Which Container method is used to cause a container to be laid out and redisplayed?
Ans : validate( )
15) What is the difference between a Scollbar and a Scrollpane?
Ans : A Scrollbar is a Component, but not a Container.
A Scrollpane is a Container and handles its own events and performs its own
scrolling.
16) Which Component subclass is used for drawing and painting?
Ans : Canvas.
17) Which of the following are direct or indirect subclasses of Component?
Button
Label
CheckboxMenuItem
Toolbar

Frame
Ans : a, b and e.
18) Which of the following are direct or indirect subclasses of Container?
Frame
TextArea
MenuBar
FileDialog
Applet
Ans : a,d and e.
19) Which method is used to set the text of a Label object?
setText( )
setLabel( )
setTextLabel( )
setLabelText( )
Ans : a.
20) Which constructor creates a TextArea with 10 rows and 20 columns?
new TextArea(10, 20)
new TextArea(20, 10)
new TextArea(new Rows(10), new columns(20))
new TextArea(200)
Ans : a.
(Usage is TextArea(rows, columns)
21) Which of the following creates a List with 5 visible items and multiple selection enabled?
new List(5, true)
new List(true, 5)
new List(5, false)
new List(false,5)
Ans : a.
[Usage is List(rows, multipleMode)]
22) Which are true about the Container class?
The validate( ) method is used to cause a Container to be laid out and redisplayed.
The add( ) method is used to add a Component to a Container.
The getBorder( ) method returns information about a Container's insets.
The getComponent( ) method is used to access a Component that is contained in a Container.
Ans : a, b and d.
23) Suppose a Panel is added to a Frame and a Button is added to the Panel. If the Frame's font is set to 12-point TimesRoman, the Panel's font is set to 10-point TimesRoman, and the Button's font is not set, what font will be used to dispaly the Button's label?
12-point TimesRoman
11-point TimesRoman
10-point TimesRoman
9-point TimesRoman
Ans : c.
A Frame's background color is set to Color.Yellow, and a Button's background color is to Color.Blue. Suppose the Button is added to a Panel, which is added to the Frame. What background color will be used with the Panel?
Colr.Yellow
Color.Blue
Color.Green
Color.White
Ans : a.
25) Which method will cause a Frame to be displayed?
show( )
setVisible( )
display( )
displayFrame( )
Ans : a and b.
26) All the componenet classes and container classes are derived from _________ class.
Ans : Object.
27) Which method of the container class can be used to add components to a Panel.
Ans : add ( ) method.
28) What are the subclasses of the Container class?
Ans : The Container class has three major subclasses. They are :
Window
Panel
ScrollPane
29) The Choice component allows multiple selection.

True.
False.
Ans : b.
30) The List component does not generate any events.
True.
False.
Ans : b.
31) Which components are used to get text input from the user.
Ans : TextField and TextArea.
32) Which object is needed to group Checkboxes to make them exclusive?
Ans : CheckboxGroup.
33) Which of the following components allow multiple selections?
Non-exclusive Checkboxes.
Radio buttons.
Choice.
List.
Ans : a and d.
34) What are the types of Checkboxes and what is the difference between them?
Ans : Java supports two types of Checkboxes. They are : Exclusive and Non-exclusive.
In case of exclusive Checkboxes, only one among a group of items can be selected at a time. I f an item from the group is selected, the checkbox currently checked is deselected and the new selection is highlighted. The exclusive Checkboxes are also called as Radio buttons.
The non-exclusive checkboxes are not grouped together and each one can be selected independent of the other.
35) What is a Layout Manager and what are the different Layout Managers available in java.awt and what is the default Layout manager for the panal and the panal subclasses?
Ans: A layout Manager is an object that is used to organize components in a container.
The different layouts available in java.awt are :
FlowLayout, BorderLayout, CardLayout, GridLayout and GridBag Layout.
The default Layout Manager of Panal and Panal sub classes is FlowLayout".
36) Can I exert control over the size and placement of components in my interface?
Ans : Yes.
myPanal.setLayout(null);
myPanal.setbounds(20,20,200,200);
37) Can I add the same component to more than one container?
Ans : No. Adding a component to a container automatically removes it from any previous parent(container).
38) How do I specify where a window is to be placed?
Ans : Use setBounds, setSize, or setLocation methods to implement this.
setBounds(int x, int y, int width, int height)
setBounds(Rectangle r)
setSize(int width, int height)
setSize(Dimension d)
setLocation(int x, int y)
setLocation(Point p)

39) How can we create a borderless window?
Ans : Create an instance of the Window class, give it a size, and show it on the screen.
eg. Frame aFrame = ...
Window aWindow = new Window(aFrame);
aWindow.setLayout(new FlowLayout());
aWindow.add(new Button("Press Me"));
aWindow.getBounds(50,50,200,200);
aWindow.show();

40) Can I create a non-resizable windows? If so, how?
Ans: Yes. By using setResizable() method in class Frame.
41) What is the default Layout Manager for the Window and Window subclasses (Frame,Dialog)?
Ans : BorderLayout().
42) How are the elements of different layouts organized?
Ans : FlowLayout : The elements of a FlowLayout are organized in a top to bottom, left to right fashion.
BorderLayout : The elements of a BorderLayout are organized at the
borders (North, South, East and West) and the center of a
container.
CardLayout : The elements of a CardLayout are stacked, one on top of the other, like a deck of cards.
GridLayout : The elements of a GridLayout are of equal size and are laid out using the square of a grid.
GridBagLayout : The elements of a GridBagLayout are organized according to a grid.However, the elements are of different sizes and may occupy

more than one row or column of the grid. In addition, the rows and columns may have different sizes.
43) Which containers use a BorderLayout as their default layout?
Ans : The Window, Frame and Dialog classes use a BorderLayout as their default layout.

44) Which containers use a FlowLayout as their default layout?
Ans : The Panel and the Applet classes use the FlowLayout as their default layout.
45) What is the preferred size of a component?
Ans : The preferred size of a component size that will allow the component to display normally.
46) Which method is method to set the layout of a container?
startLayout( )
initLayout( )
layoutContainer( )
setLayout( )
Ans : d.
47) Which method returns the preferred size of a component?
getPreferredSize( )
getPreferred( )
getRequiredSize( )
getLayout( )
Ans : a.

48) Which layout should you use to organize the components of a container in a
tabular form?
CardLayout
BorederLayout
FlowLayout
GridLayout
Ans : d.
An application has a frame that uses a Border layout manager. Why is it probably not a good idea to put a vertical scroll bar at North in the frame?
The scroll bar's height would be its preferred height, which is not likely to be enough.
The scroll bar's width would be the entire width of the frame, which would be much wider than necessary.
Both a and b.
Neither a nor b. There is no problem with the layout as described.
Ans : c.
What is the default layouts for a applet, a frame and a panel?
Ans : For an applet and a panel, Flow layout is the default layout, whereas Border layout is default layout for a frame.
If a frame uses a Grid layout manager and does not contain any panels, then all the components within the frame are the same width and height.
True
False.
Ans : a.
If a frame uses its default layout manager and does not contain any panels, then all the components within the frame are the same width and height.
True
False.
Ans : b.
With a Border layout manager, the component at Center gets all the space that is left over, after the components at North and South have been considered.
True
False
Ans : b.
An Applet has its Layout Manager set to the default of FlowLayout. What code would be the correct to change to another Layout Manager?
setLayoutManager(new GridLayout());
setLayout(new GridLayout(2,2));
c) setGridLayout(2,2,))
d setBorderLayout();
Ans : b.
55) How do you indicate where a component will be positioned using Flowlayout?
a) North, South,East,West
b) Assign a row/column grid reference

c) Pass a X/Y percentage parameter to the add method
d) Do nothing, the FlowLayout will position the component
Ans :d.

56) How do you change the current layout manager for a container?
a) Use the setLayout method
b) Once created you cannot change the current layout manager of a component
c) Use the setLayoutManager method
d) Use the updateLayout method
Ans :a.
57)When using the GridBagLayout manager, each new component requires a new instance of the GridBagConstraints class. Is this statement true or false?
a) true
b) false
Ans : b.
58) Which of the following statements are true?
a)The default layout manager for an Applet is FlowLayout
b) The default layout manager for an application is FlowLayout
c) A layout manager must be assigned to an Applet before the setSize method is called
d) The FlowLayout manager attempts to honor the preferred size of any components
Ans : a and d.
59) Which method does display the messages whenever there is an item selection or deselection of the CheckboxMenuItem menu?
Ans : itemStateChanged method.
60) Which is a dual state menu item?
Ans : CheckboxMenuItem.
61) Which method can be used to enable/diable a checkbox menu item?
Ans : setState(boolean).
Which of the following may a menu contain?
A separator
A check box
A menu
A button
A panel
Ans : a and c.
Which of the following may contain a menu bar?
A panel
A frame
An applet
A menu bar
A menu
Ans : b
64) What is the difference between a MenuItem and a CheckboxMenuItem?
Ans : The CheckboxMenuItem class extends the MenuItem class to support a menu item
that may be checked or unchecked.
65) Which of the following are true?
A Dialog can have a MenuBar.
MenuItem extends Menu.
A MenuItem can be added to a Menu.
A Menu can be added to a Menu.
Ans : c and d.

Which colour is used to indicate instance methods in the standard "javadoc" format documentation:
1) blue
2) red
3) purple
4) orange
Answer : 2
explain
In JDK 1.1 the variabels, methods and constructors are colour coded to simplifytheir identification.
endExplain
What is the correct ordering for the import, class and package declarations when found in a single file?
1) package, import, class

2) class, import, package
3) import, package, class
4) package, class, import
Answer : 1
explain
This is my explanation for question 2
endExplain
Which methods can be legally applied to a string object?
(Multiple)
1) equals(String)
2) equals(Object)
3) trim()
4) round()
5) toString()
Answer : 1,2,3,5
What is the parameter specification for the public static void main method?
(multiple)
1) String args []
2) String [] args
3) Strings args []
4) String args
Answer : 1,2
What does the zeroth element of the string array passed to the public static void main method contain?
(multiple)
1) The name of the program
2) The number of arguments
3) The first argument if one is present
Answer : 3
Which of the following are Java keywords?
(multiple)
1) goto
2) malloc
3) extends
4) FALSE
Answer : 3
What will be the result of compiling the following code:

```
public class Test {
public static void main (String args []) {
int age;
age = age + 1;
System.out.println("The age is " + age);
}
}
```

1) Compiles and runs with no output
2) Compiles and runs printing out The age is 1
3) Compiles but generates a runtime error
4) Does not compile
5) Compiles but generates a compile time error
Answer : 4
Which of these is the correct format to use to create the literal char value a?
(multiple)
1) 'a'
2) "a"
3) new Character(a)
4) \000a
Answer : 1
What is the legal range of a byte integral type?
1) 0 - 65, 535
2) (-128) - 127
3) (-32,768) - 32,767
4) (-256) - 255
Answer : 2
Which of the following is illegal:
1) int i = 32;
2) float f = 45.0;
3) double d = 45.0;

Answer 2

What will be the result of compiling the following code:

```
public class Test {
static int age;
public static void main (String args []) {
age = age + 1;
System.out.println("The age is " + age);
}
}
```

1) Compiles and runs with no output
2) Compiles and runs printing out The age is 1
3) Compiles but generates a runtime error
4) Does not compile
5) Compiles but generates a compile time error

Answer : 2

Which of the following are correct?
(multiple)

1) 128 >> 1 gives 64
2) 128 >>> 1 gives 64
3) 128 >> 1 gives -64
4) 128 >>> 1 gives -64

Answer : 1

Which of the following return true?
(multiple)

1) "john" == new String("john")
2) "john".equals("john")
3) "john" = "john"
4) "john".equals(new Button("john"))

Answer : 2

Which of the following do not lead to a runtime error?
(multiple)

1) "john" + " was " + " here"
2) "john" + 3
3) 3 + 5
4) 5 + 5.5

answer 1,2,3,4

Which of the following are so called "short circuit" logical operators?
(multiple)

1) &
2) ||
3) &&
4) |

Answer : 2,3

Which of the following are acceptable?
(multiple)

1) Object o = new Button("A");
2) Boolean flag = true;
3) Panel p = new Frame();
4) Frame f = new Panel();
5) Panel p = new Applet();

Answer : 1,5

What is the result of compiling and running the following code:

```
public class Test {
static int total = 10;
public static void main (String args []) {
new Test();
}
public Test () {
System.out.println("In test");
System.out.println(this);
int temp = this.total;
if (temp > 5) {
System.out.println(temp);
}
}
}
```

(multiple)
1) The class will not compile
2) The compiler reports and error at line 2
3) The compiler reports an error at line 9
4) The value 10 is one of the elements printed to the standard output
5) The class compiles but generates a runtime error
Answer : 4
Which of the following is correct:
1) String temp [] = new String {"j" "a" "z"};
2) String temp [] = { "j " " b" "c"};
3) String temp = {"a", "b", "c"};
4) String temp [] = {"a", "b", "c"};
Answer 4
What is the correct declaration of an abstract method that is intended to be public:
1) public abstract void add();
2) public abstract void add() {}
3) public abstract add();
4) public virtual add();
Answer : 1
Under what situations do you obtain a default constructor?
1) When you define any class
2) When the class has no other constructors
3) When you define at least one constructor
Answer : 2
Which of the following can be used to define a constructor for this class, given the following code:
```
public class Test {
...
}
```
1) public void Test() {...}
2) public Test() {...}
3) public static Test() {...}
4) public static void Test() {...}
Answer : 2
Which of the following are acceptable to the Java compiler:
(multiple)
1) if (2 == 3) System.out.println("Hi");
2) if (2 = 3) System.out.println("Hi");
3) if (true) System.out.println("Hi");
4) if (2 != 3) System.out.println("Hi");
5) if (aString.equals("hello")) System.out.println("Hi");
Answer : 1,3,4,5
Assuming a method contains code which may raise an Exception (but not a RuntimeException), what is the correct way for a method to indicate that it expects the caller to handle that exception:
1) throw Exception
2) throws Exception
3) new Exception
4) Don't need to specify anything
Answer : 2
What is the result of executing the following code, using the parameters 4 and 0:
```
public void divide(int a, int b) {
try {
int c = a / b;
} catch (Exception e) {
System.out.print("Exception ");
} finally {
System.out.println("Finally");
}
```
1) Prints out: Exception Finally
2) Prints out: Finally
3) Prints out: Exception
4) No output
Answer : 1
Which of the following is a legal return type of a method overloading the following method:
```
public void add(int a) {...}
```
1) void
2) int

3) Can be anything
Answer : 3
Which of the following statements is correct for a method which is overriding the following method:
public void add(int a) {...}
1) the overriding method must return void
2) the overriding method must return int
3) the overriding method can return whatever it likes
Answer : 1
Given the following classes defined in separate files, what will be the effect of compiling and running this class Test?

```
class Vehicle {
public void drive() {
System.out.println("Vehicle: drive");
}
}
class Car extends Vehicle {
public void drive() {
System.out.println("Car: drive");
}
}
public class Test {
public static void main (String args []) {
Vehicle v;
Car c;
v = new Vehicle();
c = new Car();
v.drive();
c.drive();
v = c;
v.drive();
}
}
```

1) Generates a Compiler error on the statement v= c;
2) Generates runtime error on the statement v= c;
3) Prints out:
Vehicle: drive
Car: drive
Car: drive
4) Prints out:
Vehicle: drive
Car: drive
Vehicle: drive
Answer : 3
Where in a constructor, can you place a call to a constructor defined in the super class?
1) Anywhere
2) The first statement in the constructor
3) The last statement in the constructor
4) You can't call super in a constructor
Answer : 2
Which variables can an inner class access from the class which encapsulates it?
(multiple)
1) All static variables
2) All final variables
3) All instance variables
4) Only final instance variables
5) Only final static variables
Answer : 1,2,3
What class must an inner class extend:
1) The top level class
2) The Object class
3) Any class or interface
4) It must extend an interface
Answer 3
In the following code, which is the earliest statement, where the object originally held in e, may be garbage collected:

```
1. public class Test {
2. public static void main (String args []) {
3. Employee e = new Employee("Bob", 48);
```

4. e.calculatePay();
5. System.out.println(e.printDetails());
6. e = null;
7. e = new Employee("Denise", 36);
8. e.calculatePay();
9. System.out.println(e.printDetails());
10. }
11. }
1) Line 10
2) Line 11
3) Line 7
4) Line 8
5) Never
Answer : 3
What is the name of the interface that can be used to define a class that can execute within its own thread?
1) Runnable
2) Run
3) Threadable
4) Thread
5) Executable
Answer : 1
What is the name of the method used to schedule a thread for execution?
1) init();
2) start();
3) run();
4) resume();
5) sleep();
Answer : 2
Which methods may cause a thread to stop executing?
(multiple)
1) sleep();
2) stop();
3) yield();
4) wait();
5) notify();
6) notifyAll()
7) synchronized()
Answer : 1,2,3,4
Which of the following would create a text field able to display 10 characters (assuming a fixed size font) displaying the initial string "hello":
1) new TextField("hello", 10);
2) new TextField("hello");
3) new textField(10);
4) new TextField();
Answer : 1
Which of the following methods are defined on the Graphics class:
(multiple)
1) drawLine(int, int, int, int)
2) drawImage(Image, int, int, ImageObserver)
3) drawString(String, int, int)
4) add(Component);
5) setVisible(boolean);
6) setLayout(Object);
Answer : 1,2,3
Which of the following layout managers honours the preferred size of a component:
(multiple)
1) CardLayout
2) FlowLayout
3) BorderLayout
4) GridLayout
Answer : 2
Given the following code what is the effect of a being 5:
public class Test {
public void add(int a) {
loop: for (int i = 1; i < 3; i++){
for (int j = 1; j < 3; j++) {

```
if (a == 5) {
break loop;
}
System.out.println(i * j);
}
}
}
}
```

1) Generate a runtime error
2) Throw an ArrayIndexOutOfBound***ception
3) Print the values: 1, 2, 2, 4
4) Produces no output
Answer : 4
What is the effect of issuing a wait() method on an object
1) If a notify() method has already been sent to that object then it has no effect
2) The object issuing the call to wait() will halt until another object sends a notify() or notifyAll() method
3) An exception will be raised
4) The object issuing the call to wait() will be automatically synchronized with any other objects using the receiving object.
Answer : 2
The layout of a container can be altered using which of the following methods:
(multiple)
1) setLayout(aLayoutManager);
2) addLayout(aLayoutManager);
3) layout(aLayoutManager);
4) setLayoutManager(aLayoutManager);
Answer : 1
Using a FlowLayout manager, which is the correct way to add elements to a container:
1) add(component);
2) add("Center", component);
3) add(x, y, component);
4) set(component);
Answer : 1
Given that a Button can generate an ActionEvent which listener would you expect to have to implement, in a class which would handle this event?
1) FocusListener
2) ComponentListener
3) WindowListener
4) ActionListener
5) ItemListener
Answer : 4
Which of the following, are valid return types, for listener methods:
1) boolean
2) the type of event handled
3) void
4) Component
Answer : 3
Assuming we have a class which implements the ActionListener interface, which method should be used to register this with a Button?
1) addListener(*);
2) addActionListener(*);
3) addButtonListener(*);
4) setListener(*);
Answer : 2
In order to cause the paint(Graphics) method to execute, which of the following is the most appropriate method to call:
1) paint()
2) repaint()
3) paint(Graphics)
4) update(Graphics)
5) None - you should never cause paint(Graphics) to execute
Answer : 2
Which of the following illustrates the correct way to pass a parameter into an applet:
1) tags?
1. CODEBASE
2. ALT
3. NAME
4. CLASS

5. JAVAC
6. HORIZONTALSPACE
7. VERTICALSPACE
8. WIDTH
9. PARAM
10. JAR
(multiple)
1) line 1, 2, 3
2) line 2, 5, 6, 7
3) line 3, 4, 5
4) line 8, 9, 10
5) line 8, 9
Answer : 1,5
Which of the following is a legal way to construct a RandomAccessFile:
1) RandomAccessFile("data", "r");
2) RandomAccessFile("r", "data");
3) RandomAccessFile("data", "read");
4) RandomAccessFile("read", "data");
Answer : 1
Carefully examine the following code, When will the string "Hi there" be printed?

```
public class StaticTest {
static {
System.out.println("Hi there");
}
public void print() {
System.out.println("Hello");
}
public static void main(String args []) {
StaticTest st1 = new StaticTest();
st1.print();
StaticTest st2 = new StaticTest();
st2.print();
}
}
```

1) Never.
2) Each time a new instance is created.
3) Once when the class is first loaded into the Java virtual machine.
4) Only when the static method is called explicitly.
Answer : 3
What is the result of the following program:

```
public class Test {
public static void main (String args []) {
boolean a = false;
if (a = true)
System.out.println("Hello");
else
System.out.println("Goodbye");
}
}
```

1) Program produces no output but terminates correctly.
2) Program does not terminate.
3) Prints out "Hello"
4) Prints out "Goodbye"
Answer : 3
Examine the following code, it includes an inner class, what is the result:

```
public final class Test4 {
class Inner {
void test() {
if (Test4.this.flag); {
sample();
}
}
}
private boolean flag = true;
public void sample() {
System.out.println("Sample");
}
public Test4() {
(new Inner()).test();
}
public static void main(String args []) {
new Test4();
}
}
```

1) Prints out "Sample"
2) Program produces no output but terminates correctly.
3) Program does not terminate.
4) The program will not compile
Answer : 1

Carefully examine the following class:

```
public class Test5 {
public static void main (String args []) {
/* This is the start of a comment
if (true) {
Test5 = new test5();
System.out.println("Done the test");
}
/* This is another comment */
System.out.println ("The end");
}
}
```

1) Prints out "Done the test" and nothing else.
2) Program produces no output but terminates correctly.
3) Program does not terminate.
4) The program will not compile.
5) The program generates a runtime exception.
6) The program prints out "The end" and nothing else.
7) The program prints out "Done the test" and "The end"
Answer : 6

What is the result of compiling and running the following applet:

```
import java.applet.Applet;
import java.awt.*;
public class Sample extends Applet {
private String text = "Hello World";
public void init() {
add(new Label(text));
}
public Sample (String string) {
text = string;
}
}
```

It is accessed form the following HTML page:

1) Prints "Hello World".
2) Generates a runtime error.
3) Does nothing.
4) Generates a compile time error.
Answer : 2

What is the effect of compiling and (if possible) running this class:

```
public class Calc {
public static void main (String args []) {
int total = 0;
for (int i = 0, j = 10; total > 30; ++i, --j) {
System.out.println(" i = " + i + " : j = " + j);
total += (i + j);
}
System.out.println("Total " + total);
}
}
```

1) Produce a runtime error
2) Produce a compile time error
3) Print out "Total 0"

4) Generate the following as output:

```
i = 0 : j = 10
i = 1 : j = 9
i = 2 : j = 8
Total 30
```

Answer : 3

Utility Package

1) What is the Vector class?
ANSWER : The Vector class provides the capability to implement a growable array of objects.
2) What is the Set interface?
ANSWER : The Set interface provides methods for accessing the elements of a finite mathematical set.Sets do not allow duplicate elements.
3) What is Dictionary class?
ANSWER : The Dictionary class is the abstarct super class of Hashtable and Properties class.Dictionary provides the abstarct functions used to store and retrieve objects by key-value.This class allows any object to be used as a key or value.
4) What is the Hashtable class?
ANSWER : The Hashtable class implements a hash table data structure. A hash table indexes and stores objects in a dictionary using hash codes as the objects' keys. Hash codes are integer values that identify objects.
5) What is the Properties class?
Answer : The properties class is a subclass of Hashtable that can be read from or written to a stream.It also provides the capability to specify a set of default values to be used if a specified key is not found in the table. We have two methods load() and save().
6) What changes are needed to make the following prg to compile?

```
import java.util.*;
class Ques{
public static void main (String args[]) {
String s1 = "abc";
String s2 = "def";
Vector v = new Vector();
v.add(s1);
v.add(s2);
String s3 = v.elementAt(0) + v.elementAt(1);
System.out.println(s3);
}
}
```

ANSWER : Declare Ques as public B) Cast v.elementAt(0) to a String
C) Cast v.elementAt(1) to an Object. D) Import java.lang
ANSWER : B) Cast v.elementAt(0) to a String

8) What is the output of the prg.

```
import java.util.*;
class Ques{
public static void main (String args[]) {
String s1 = "abc";
String s2 = "def";
Stack stack = new Stack();
stack.push(s1);
stack.push(s2);
try{
String s3 = (String) stack.pop() + (String) stack.pop() ;
System.out.println(s3);
}catch (EmptyStackException ex){}
}
}
```

ANSWER : abcdef B) defabc C) abcabc D) defdef
ANSWER : B) defabc
9) Which of the following may have duplicate elements?
ANSWER : Collection B) List C) Map D) Set
ANSWER : A and B Neither a Map nor a Set may have duplicate elements.
10) Can null value be added to a List?
ANSWER : Yes.A Null value may be added to any List.
11) What is the output of the following prg.

```
import java.util.*;
class Ques{
public static void main (String args[]) {
HashSet set = new HashSet();
String s1 = "abc";
String s2 = "def";
String s3 = "";
set.add(s1);
set.add(s2);
set.add(s1);
set.add(s2);
Iterator i = set.iterator();
while(i.hasNext())
{
s3 += (String) i.next();
}
System.out.println(s3);
}
}
```

A) abcdefabcdef B) defabcdefabc C) fedcbafedcba D) defabc
ANSWER : D) defabc. Sets may not have duplicate elements.
12) Which of the following java.util classes support internationalization?
A) Locale B) ResourceBundle C) Country D) Language
ANSWER : A and B . Country and Language are not java.util classes.
13) What is the ResourceBundle?
The ResourceBundle class also supports internationalization.
ResourceBundle subclasses are used to store locale-specific resources that can be loaded by a program to tailor the program's appearence to the paticular locale in which it is being run. Resource Bundles provide the capability to isolate a program's locale-specific resources in a standard and modular manner.
14) How are Observer Interface and Observable class, in java.util package, used?
ANSWER : Objects that subclass the Observable class maintain a list of Observers. When an Observable object is updated it invokes the update() method of each of its observers to notify the observers that it has changed state. The Observer interface is implemented by objects that observe Observable objects.
15) Which java.util classes and interfaces support event handling?
ANSWER : The EventObject class and the EventListener interface support event processing.
16) Does java provide standard iterator functions for inspecting a collection of objects?
ANSWER : The Enumeration interface in the java.util package provides a framework for stepping once through a collection of objects. We have two methods in that interface.

```
public interface Enumeration {
boolean hasMoreElements();
Object nextElement();
}
```

17) The Math.random method is too limited for my needs- How can I generate random numbers more flexibly?
ANSWER : The random method in Math class provide quick, convienient access to random numbers, but more power and flexibility use the Random class in the java.util package.
double doubleval = Math.random();
The Random class provide methods returning float, int, double, and long values.

```
nextFloat() // type float; 0.0 <= value < 1.0
nextDouble() // type double; 0.0 <= value < 1.0
nextInt() // type int; Integer.MIN_VALUE <= value <= Integer.MAX_VALUE
nextLong() // type long; Long.MIN_VALUE <= value <= Long.MAX_VALUE
nextGaussian() // type double; has Gaussian("normal") distribution with mean 0.0 and standard deviation 1.0)
Eg. Random r = new Random();
float floatval = r.nextFloat();
```

18) How can we get all public methods of an object dynamically?
ANSWER : By using getMethods(). It return an array of method objects corresponding to the public methods of this class.
getFields() returns an array of Filed objects corresponding to the public Fields(variables) of this class.
getConstructors() returns an array of constructor objects corresponding to the public constructors of this class.

JDBC
1) What are the steps involved in establishing a connection?
ANSWER : This involves two steps: (1) loading the driver and (2) making the connection.

2) How can you load the drivers?
ANSWER : Loading the driver or drivers you want to use is very simple and involves just one line of code. If, for example, you want to use the JDBC-ODBC Bridge driver, the following code will load it:
Eg.
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Your driver documentation will give you the class name to use. For instance, if the class name is jdbc.DriverXYZ , you would load the driver with the following line of code:
Eg.
Class.forName("jdbc.DriverXYZ");
3) What Class.forName will do while loading drivers?
ANSWER : It is used to create an instance of a driver and register it with the DriverManager.
When you have loaded a driver, it is available for making a connection with a DBMS.
4) How can you make the connection?
ANSWER : In establishing a connection is to have the appropriate driver connect to the DBMS. The following line of code illustrates the general idea:
Eg.
String url = "jdbc:odbc:Fred";
Connection con = DriverManager.getConnection(url, "Fernanda", "J8");
5) How can you create JDBC statements?
ANSWER : A Statement object is what sends your SQL statement to the DBMS. You simply create a Statement object and then execute it, supplying the appropriate execute method with the SQL statement you want to send. For a SELECT statement, the method to use is executeQuery. For statements that create or modify tables, the method to use is executeUpdate.
Eg.
It takes an instance of an active connection to create a Statement object. In the following example, we use our Connection object con to create the Statement object stmt :
Statement stmt = con.createStatement();
6) How can you retrieve data from the ResultSet?
ANSWER : Step 1.
JDBC returns results in a ResultSet object, so we need to declare an instance of the class ResultSet to hold our results. The following code demonstrates declaring the ResultSet object rs.
Eg.
ResultSet rs = stmt.executeQuery("SELECT COF_NAME, PRICE FROM COFFEES");
Step2.
String s = rs.getString("COF_NAME");
The method getString is invoked on the ResultSet object rs , so getString will retrieve (get) the value stored in the column COF_NAME in the current row of rs
7) What are the different types of Statements?
ANSWER : 1.Statement (use createStatement method) 2. Prepared Statement (Use prepareStatement method) and 3. Callable Statement (Use prepareCall)
8) How can you use PreparedStatement?
ANSWER : This special type of statement is derived from the more general class, Statement.If you want to execute a Statement object many times, it will normally reduce execution time to use a PreparedStatement object instead.
The advantage to this is that in most cases, this SQL statement will be sent to the DBMS right away, where it will be compiled. As a result, the PreparedStatement object contains not just an SQL statement, but an SQL statement that has been precompiled. This means that when the PreparedStatement is executed, the DBMS can just run the PreparedStatement 's SQL statement without having to compile it first.
Eg.
PreparedStatement updateSales = con.prepareStatement("UPDATE COFFEES SET SALES = ? WHERE COF_NAME LIKE ?");
9) What setAutoCommit does?
ANSWER : When a connection is created, it is in auto-commit mode. This means that each individual SQL statement is treated as a transaction and will be automatically committed right after it is executed. The way to allow two or more statements to be grouped into a transaction is to disable auto-commit mode
Eg.
con.setAutoCommit(false);
Once auto-commit mode is disabled, no SQL statements will be committed until you call the method commit explicitly.
Eg.
con.setAutoCommit(false);
PreparedStatement updateSales = con.prepareStatement(
"UPDATE COFFEES SET SALES = ? WHERE COF_NAME LIKE ?");
updateSales.setInt(1, 50);
updateSales.setString(2, "Colombian");
updateSales.executeUpdate();
PreparedStatement updateTotal = con.prepareStatement("UPDATE COFFEES SET TOTAL = TOTAL + ? WHERE COF_NAME LIKE ?");
updateTotal.setInt(1, 50);
updateTotal.setString(2, "Colombian");

```
updateTotal.executeUpdate();
con.commit();
con.setAutoCommit(true);
```

10) How to call a Strored Procedure from JDBC?

ANSWER : The first step is to create a CallableStatement object. As with Statement an and PreparedStatement objects, this is done with an open Connection
object. A CallableStatement object contains a call to a stored procedure;

Eg.

```
CallableStatement cs = con.prepareCall("{call SHOW_SUPPLIERS}");
ResultSet rs = cs.executeQuery();
```

11) How to Retrieve Warnings?

ANSWER : SQLWarning objects are a subclass of SQLException that deal with database access warnings. Warnings do not stop the execution of an application, as exceptions do; they simply alert the user that something did not happen as planned.
A warning can be reported on a Connection object, a Statement object (including PreparedStatement and CallableStatement objects), or a ResultSet object. Each of these classes has a getWarnings method, which you must invoke in order to see the first warning reported on the calling object

Eg.

```
SQLWarning warning = stmt.getWarnings();
if (warning != null) {
System.out.println("\n---Warning---\n");
while (warning != null) {
System.out.println("Message: " + warning.getMessage());
System.out.println("SQLState: " + warning.getSQLState());
System.out.print("Vendor error code: ");
System.out.println(warning.getErrorCode());
System.out.println("");
warning = warning.getNextWarning();
}
}
```

12) How can you Move the Cursor in Scrollable Result Sets ?

ANSWER : One of the new features in the JDBC 2.0 API is the ability to move a result set's cursor backward as well as forward. There are also methods that let you move the cursor to a particular row and check the position of the cursor.

Eg.

```
Statement stmt = con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_READ_ONLY);
ResultSet srs = stmt.executeQuery("SELECT COF_NAME, PRICE FROM COFFEES");
```

The first argument is one of three constants added to the ResultSet API to indicate the type of a ResultSet object: TYPE_FORWARD_ONLY, TYPE_SCROLL_INSENSITIVE , and TYPE_SCROLL_SENSITIVE .
The second argument is one of two ResultSet constants for specifying whether a result set is read-only or updatable: CONCUR_READ_ONLY and CONCUR_UPDATABLE . The point to remember here is that if you specify a type, you must also specify whether it is read-only or updatable. Also, you must specify the type first, and because both parameters are of type int , the compiler will not complain if you switch the order.
Specifying the constant TYPE_FORWARD_ONLY creates a nonscrollable result set, that is, one in which the cursor moves only forward. If you do not specify any constants for the type and updatability of a ResultSet object, you will automatically get one that is TYPE_FORWARD_ONLY and CONCUR_READ_ONLY

13) What's the difference between TYPE_SCROLL_INSENSITIVE , and TYPE_SCROLL_SENSITIVE?

ANSWER : You will get a scrollable ResultSet object if you specify one of these ResultSet constants.The difference between the two has to do with whether a result set reflects changes that are made to it while it is open and whether certain methods can be called to detect these changes. Generally speaking, a result set that is TYPE_SCROLL_INSENSITIVE does not reflect changes made while it is still open and one that is TYPE_SCROLL_SENSITIVE does. All three types of result sets will make changes visible if they are closed and then reopened

Eg.

```
Statement stmt = con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE, ResultSet.CONCUR_READ_ONLY);
ResultSet srs = stmt.executeQuery("SELECT COF_NAME, PRICE FROM COFFEES");
srs.afterLast();
while (srs.previous()) {
String name = srs.getString("COF_NAME");
float price = srs.getFloat("PRICE");
System.out.println(name + " " + price);
}
```

14) How to Make Updates to Updatable Result Sets?

ANSWER : Another new feature in the JDBC 2.0 API is the ability to update rows in a result set using methods in the Java programming language rather than having to send an SQL command. But before you can take advantage of this capability, you need to create a ResultSet object that is updatable. In order to do this, you supply the ResultSet constant CONCUR_UPDATABLE to the createStatement method.

Eg.

```
Connection con = DriverManager.getConnection("jdbc:mySubprotocol:mySubName");
Statement stmt = con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE);
ResultSet uprs = stmt.executeQuery("SELECT COF_NAME, PRICE FROM COFFEES");
```

Networking Concepts
1) The API doesn't list any constructors for InetAddress- How do I create an InetAddress instance?
ANSWER : In case of InetAddress the three methods getLocalHost, getByName, getByAllName can be used to create instances.
E.g.

```
InetAddress add1;
InetAddress add2;
try{
add1 = InetAddress.getByName("java.sun.com");
add2 = InetAddress.getByName("199.22.22.22");
}catch(UnknownHostException e){}
```

2) Is it possible to get the Local host IP?
ANSWER : Yes. Use InetAddress's getLocalHost method.
3) What's the Factory Method?
ANSWER : Factory methods are merely a convention whereby static methods in a class return an instance of that class. The InetAddress class has no visible constructors. To create an InetAddress object, you have to use one of the available factory methods. In InetAddress the three methods getLocalHost, getByName, getByAllName can be used to create instances of InetAddress.
4) What’s the difference between TCP and UDP?
ANSWER : These two protocols differ in the way they carry out the action of communicating. A TCP protocol establishes a two way connection between a pair of computers, while the UDP protocol is a one-way message sender. The common analogy is that TCP is like making a phone call and carrying on a two-way communication, while UDP is like mailing a letter.
5) What is the Proxy Server?
ANSWER : A proxy server speaks the client side of a protocol to another server. This is often required when clients have certain restrictions on which servers they can connect to. And when several users are hitting a popular web site, a proxy server can get the contents of the web server's popular pages once, saving expensive internetwork transfers while providing faster access to those pages to the clients.
Also, we can get multiple connections for a single server.
6) What are the seven layers of OSI model?
ANSWER : Application, Presentation, Session, Transport, Network, DataLink, Physical Layer.
What Transport Layer does?
ANSWER : It ensures that the mail gets to its destination. If a packet fails to get its destination, it handles the process of notifying the sender and requesting that another packet be sent.
8) What is DHCP?
ANSWER : Dynamic Host Configuration Protocol, a piece of the TCP/IP protocol suite that handles the automatic assignment of IP addresses to clients.
9) What is SMTP?
ANSWER : Simple Mail Transmission Protocol, the TCP/IP Standard for Internet mails. SMTP exchanges mail between servers; contrast this with POP, which transmits mail between a server and a client.
10) In OSI N/w architecture, the dialogue control and token management are responsibilities of...
Answer : Network b) Session c) Application d) DataLink
ANSWER : b) Session Layer.
11) In OSI N/W Architecture, the routing is performed by ______
Answer : Network b) Session c) Application d) DataLink
ANSWER : Answer : Network Layer.
Networking

What is the difference between URL instance and URLConnection instance?
ANSWER : A URL instance represents the location of a resource, and a URLConnection instance represents a link for accessing or communicating with the resource at the location.
2) How do I make a connection to URL?
ANSWER : You obtain a URL instance and then invoke openConnection on it.
URLConnection is an abstract class, which means you can't directly create instances of it using a constructor. We have to invoke openConnection method on a URL instance, to get the right kind of connection for your URL.

```
Eg. URL url;
URLConnection connection;
try{ url = new URL("...");
conection = url.openConnection();
}catch (MalFormedURLException e) { }
```

3) What Is a Socket?
A socket is one end-point of a two-way communication link between two programs running on the network. A socket is bound to

a port number so that the TCP layer can identify the application that data is destined to be sent.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--which implement the client side of the connection and the server side of the connection, respectively.
What information is needed to create a TCP Socket?
ANSWER : The Local System's IP Address and Port Number.
And the Remote System's IPAddress and Port Number.
5) What are the two important TCP Socket classes?
ANSWER : Socket and ServerSocket.
ServerSocket is used for normal two-way socket communication. Socket class allows us to read and write through the sockets. getInputStream() and getOutputStream() are the two methods available in Socket class.
When MalformedURLException and UnknownHostException throws?
ANSWER : When the specified URL is not connected then the URL throw MalformedURLException and If InetAddress' methods getByName and getLocalHost are unabletoresolve the host name they throwan UnknownHostException.
Servlets

1) What is the servlet?
ANSWER : Servlets are modules that extend request/response-oriented servers, such as Java-enabled web servers. For example, a servlet might be responsible for taking data in an HTML order-entry form and applying the business logic used to update a company's order database.
Servlets are to servers what applets are to browsers. Unlike applets, however, servlets have no graphical user interface.
2) Whats the advantages using servlets than using CGI?
ANSWER : Servlets provide a way to generate dynamic documents that is both easier to write and faster to run. Servlets also address the problem of doing server-side programming with platform-specific APIs: they are developed with the Java Servlet API, a standard Java extension.
3) What are the uses of Servlets?
ANSWER : A servlet can handle multiple requests concurrently, and can synchronize requests. This allows servlets to support systems such as on-line conferencing.
Servlets can forward requests to other servers and servlets.Thus servlets can be used to balance load among several servers that mirror the same content, and to partition a single logical service over several servers, according to task type or organizational boundaries.
4) Which pakage provides interfaces and classes for writing servlets?
ANSWER : javax
5) Whats the Servlet Interfcae?
ANSWER : The central abstraction in the Servlet API is the Servlet interface. All servlets implement this interface, either directly or, more commonly, by extending a class that implements it such as HttpServlet.
Servlets-->Generic Servlet-->HttpServlet-->MyServlet.
The Servlet interface declares, but does not implement, methods that manage the servlet and its communications with clients.
Servlet writers provide some or all of these methods when developing a servlet.
6) When a servlet accepts a call from a client, it receives two objects- What are they?
ANSWER : ServeltRequest: Which encapsulates the communication from the client to the server.
ServletResponse: Whcih encapsulates the communication from the servlet back to the client.
ServletRequest and ServletResponse are interfaces defined by the javax.servlet package.
7) What information that the ServletRequest interface allows the servlet access to?
ANSWER : Information such as the names of the parameters passed in by the client, the protocol (scheme) being used by the client, and the names of the remote host that made the request and the server that received it.
The input stream, ServletInputStream.Servlets use the input stream to get data from clients that use application protocols such as the HTTP POST and PUT methods.
8) What information that the ServletResponse interface gives the servlet methods for replying to the client?
ANSWER : It Allows the servlet to set the content length and MIME type of the reply.
Provides an output stream, ServletOutputStream and a Writer through which the servlet can send the reply data.
9) What is the servlet Lifecycle?
ANSWER : Each servlet has the same life cycle:
A server loads and initializes the servlet (init())
The servlet handles zero or more client requests (service())
The server removes the servlet (destroy())
(some servers do this step only when they shut down)
10) How HTTP Servlet handles client requests?
ANSWER : An HTTP Servlet handles client requests through its service method. The service method supports standard HTTP client requests by dispatching each request to a method designed to handle that request. 1

Encapsulation :
Encapsulation is the mechanism that binds together code and the data it manipulates and keeps both safe from outside interference and misuse.
Inheritance:
Inheritance is the process by which one object acquires the properties of another object.
Polymorphism:

Polymorphism is a feature that allows one interface to be used for a general class of actions. The specific action is determined by the exact nature of actions.
Code Blocks:
Two or more statements which is allowed to be grouped into blocks of code is otherwise called as Code Blocks.This is done by enclosing the statements between opening and closing curly braces.
Floating-point numbers:
Floating-point numbers which is also known as real numbers, are used when evaluating expressions that require fractional precision.
Unicode:
Unicode defines a fully international character set that can represent all of the characters found in all human languages. It is a unification of dozens of character sets, such as Latin, Greek, Arabic and many more.
Booleans:
Java has a simple type called boolean, for logical values. It can have only on of two possible values, true or false.
Casting:
A cast is simply an explicit type conversion. To create a conversion between two incompatible types, you must use a cast.
Arrays:
An array is a group of like-typed variables that are referred to by a common name. Arrays offer a convenient means of grouping related information. Arrays of any type can be created and may have one or more dimension.
Relational Operators:
The relational operators determine the relationship that one operand has to the other. They determine the equality and ordering.
11.Short-Circuit Logical Operators:
The secondary versions of the Boolean AND and OR operators are known as short-circuit logical operators. It is represented by || and &&..
12. Switch:
The switch statement is Java’s multiway branch statement. It provides an easy way to dispatch execution to different parts of your code based on the value of an experession.
13. Jump Statements:
Jump statements are the statements which transfer control to another part of your program. Java Supports three jump statements: break, continue, and return.
14. Instance Variables:
The data, or variable, defined within a class are called instance variable.

1. What releases of Java technology are currently available? What do they contain?
The Java programming language is currently shipping from Sun Microsystems, Inc. as the Java Development Kit (JDKTM). All Sun releases of the JDK software are available from the JDK software home page (http://java.sun.com/products/jdk/).
Each release of the Java Development Kit (JDK) contains:
Java Compiler
Java Virtual Machine*
Java Class Libraries
Java AppletViewer
Java Debugger and other tools
Documentation (in a separate download bundle)
To run Java 1.0 applets, use Netscape Navigator 3.x or other browsers that support Java applets. To run Java 1.1.x applets, use HotJavaTM 1.x or Netscape Navigator 4.x or other browsers that support the newest version of the Java API.
2. What are the security problems I've heard about JavaScript technology scripts?
JavaScript technology is a scripting language used with Netscape Navigator. There have been reports of privacy problems with JavaScript technology, and Netscape is committed to addressing those concerns. JavaScript technology cannot be used to invoke Java applets. The privacy problems reported with JavaScript technology are not present in Java applets.
3. Why developers should not write programs that call 'sun' packages
Java Software supports into the future only classes in java.* packages, not sun.* packages. In general, API in sun.* is subject to change at any time without notice. For more details, see the article Why Developers Should Not Write Programs That Call 'sun' Packages.
4. Where did the Java name come from? What does it stand for?
The name was chosen during one of several brainstorming sessions held by the Java software team. We were aiming to come up with a name that evoked the essence of the technology -- liveliness, animation, speed, interactivity, and more. "Java" was chosen from among many, many suggestions. The name is not an acronym, but rather a reminder of that hot, aromatic stuff that many programmers like to drink lots of.

Core Java interview questions

1. **Can there be an abstract class with no abstract methods in it?** - Yes
2. **Can an Interface be final?** - No

3. **Can an Interface have an inner class?** - Yes.

```
public interface abc
{
	static int i=0; void dd();
	class a1
	{
		a1()
		{
			int j;
			System.out.println("inside");
		};
		public static void main(String a1[])
		{
			System.out.println("in interfia");
		}
	}
}
```

4. **Can we define private and protected modifiers for variables in interfaces?** - No
5. **What is Externalizable?** - Externalizable is an Interface that extends Serializable Interface. And sends data into Streams in Compressed Format. It has two methods, writeExternal(ObjectOuput out) and readExternal(ObjectInput in)
6. **What modifiers are allowed for methods in an Interface?** - Only public and abstract modifiers are allowed for methods in interfaces.
7. **What is a local, member and a class variable?** - Variables declared within a method are “local” variables. Variables declared within the class i.e not within any methods are “member” variables (global variables). Variables declared within the class i.e not within any methods and are defined as “static” are class variables
8. **What are the different identifier states of a Thread?** - The different identifiers of a Thread are: R - Running or runnable thread, S - Suspended thread, CW - Thread waiting on a condition variable, MW - Thread waiting on a monitor lock, MS - Thread suspended waiting on a monitor lock
9. **What are some alternatives to inheritance?** - Delegation is an alternative to inheritance. Delegation means that you include an instance of another class as an instance variable, and forward messages to the instance. It is often safer than inheritance because it forces you to think about each message you forward, because the instance is of a known class, rather than a new class, and because it doesn’t force you to accept all the methods of the super class: you can provide only the methods that really make sense. On the other hand, it makes you write more code, and it is harder to re-use (because it is not a subclass).
10. **Why isn’t there operator overloading?** - Because C++ has proven by example that operator overloading makes code almost impossible to maintain. In fact there very nearly wasn’t even method overloading in Java, but it was thought that this was too useful for some very basic methods like print(). Note that some of the classes like DataOutputStream have unoverloaded methods like writeInt() and writeByte().
11. **What does it mean that a method or field is “static"?** - Static variables and methods are instantiated only once per class. In other words they are class variables, not instance variables. If you change the value of a static variable in a particular object, the value of that variable changes for all instances of that class. Static methods can be referenced with the name of the class rather than the name of a particular object of the class (though that works too). That’s how library methods like System.out.println() work. out is a static field in the java.lang.System class.
12. **How do I convert a numeric IP address like 192.18.97.39 into a hostname like java.sun.com?**

```
	String hostname =
InetAddress.getByName("192.18.97.39").getHostName();
```

13. **Difference between JRE/JVM/JDK?**
14. **Why do threads block on I/O? -** Threads block on i/o (that is enters the waiting state) so that other threads may execute while the I/O operation is performed.
15. **What is synchronization and why is it important? -** With respect to multithreading, synchronization is the capability to control the access of multiple threads to shared resources. Without synchronization, it is possible for one thread to modify a shared object while another thread is in the process of using or updating that object's value. This often leads to significant errors.
16. **Is null a keyword? -** The null value is not a keyword.
17. **Which characters may be used as the second character of an identifier,but not as the first character of an identifier? -** The digits 0 through 9 may not be used as the first character of an identifier but they may be used after the first character of an identifier.
18. **What modifiers may be used with an inner class that is a member of an outer class? -** A (non-local) inner class may be declared as public, protected, private, static, final, or abstract.
19. **How many bits are used to represent Unicode, ASCII, UTF-16, and UTF-8 characters? -** Unicode requires 16 bits and ASCII require 7 bits. Although the ASCII character set uses only 7 bits, it is usually represented as 8 bits. UTF-8 represents characters using 8, 16, and 18 bit patterns. UTF-16 uses 16-bit and larger bit patterns.
20. **What are wrapped classes? -** Wrapped classes are classes that allow primitive types to be accessed as objects.
21. **What restrictions are placed on the location of a package statement within a source code file? -** A package statement must appear as the first line in a source code file (excluding blank lines and comments).
22. **What is the difference between preemptive scheduling and time slicing? -** Under preemptive scheduling, the highest priority task executes until it enters the waiting or dead states or a higher priority task comes into existence. Under time slicing, a task executes for a predefined slice of time and then reenters the pool of ready tasks. The scheduler then determines which task should execute next, based on priority and other factors.
23. **What is a native method? -** A native method is a method that is implemented in a language other than Java.
24. **What are order of precedence and associativity, and how are they used? -** Order of precedence determines the order in which operators are evaluated in expressions. Associatity determines whether an expression is evaluated left-to-right or right-to-left
25. **What is the catch or declare rule for method declarations? -** If a checked exception may be thrown within the body of a method, the method must either catch the exception or declare it in its throws clause.
26. **Can an anonymous class be declared as implementing an interface and extending a class? -** An anonymous class may implement an interface or extend a superclass, but may not be declared to do both.
27. **What is the range of the char type? -** The range of the char type is 0 to 2^16 - 1.

Java interview questions

1. **What is garbage collection? What is the process that is responsible for doing that in java? -** Reclaiming the unused memory by the invalid objects. Garbage collector is responsible for this process
2. **What kind of thread is the Garbage collector thread? -** It is a daemon thread.
3. **What is a daemon thread? -** These are the threads which can run without user intervention. The JVM can exit when there are daemon thread by killing them abruptly.
4. **How will you invoke any external process in Java? -** Runtime.getRuntime().exec(….)
5. **What is the finalize method do? -** Before the invalid objects get garbage collected, the JVM give the user a chance to clean up some resources before it got garbage collected.
6. **What is mutable object and immutable object? -** If a object value is changeable then we can call it as Mutable object. (Ex., StringBuffer, …) If you are not allowed to change the value of an object, it is immutable object. (Ex., String, Integer, Float, …)

7. **What is the basic difference between string and stringbuffer object?** - String is an immutable object. StringBuffer is a mutable object.
8. **What is the purpose of Void class?** - The Void class is an uninstantiable placeholder class to hold a reference to the Class object representing the primitive Java type void.
9. **What is reflection?** - Reflection allows programmatic access to information about the fields, methods and constructors of loaded classes, and the use reflected fields, methods, and constructors to operate on their underlying counterparts on objects, within security restrictions.
10. **What is the base class for Error and Exception?** - Throwable
11. **What is the byte range?** -128 to 127
12. **What is the implementation of destroy method in java.. is it native or java code?** - This method is not implemented.
13. **What is a package?** - To group set of classes into a single unit is known as packaging. Packages provides wide namespace ability.
14. **What are the approaches that you will follow for making a program very efficient?** - By avoiding too much of static methods avoiding the excessive and unnecessary use of synchronized methods Selection of related classes based on the application (meaning synchronized classes for multiuser and non-synchronized classes for single user) Usage of appropriate design patterns Using cache methodologies for remote invocations Avoiding creation of variables within a loop and lot more.
15. **What is a DatabaseMetaData?** - Comprehensive information about the database as a whole.
16. **What is Locale?** - A Locale object represents a specific geographical, political, or cultural region
17. **How will you load a specific locale?** - Using ResourceBundle.getBundle(…);
18. **What is JIT and its use?** - Really, just a very fast compiler… In this incarnation, pretty much a one-pass compiler – no offline computations. So you can’t look at the whole method, rank the expressions according to which ones are re-used the most, and then generate code. In theory terms, it’s an on-line problem.
19. **Is JVM a compiler or an interpreter?** - Interpreter
20. **When you think about optimization, what is the best way to findout the time/memory consuming process?** - Using profiler
21. **What is the purpose of assert keyword used in JDK1.4.x?** - In order to validate certain expressions. It effectively replaces the if block and automatically throws the AssertionError on failure. This keyword should be used for the critical arguments. Meaning, without that the method does nothing.
22. **How will you get the platform dependent values like line separator, path separator, etc., ? -** Using Sytem.getProperty(…) (line.separator, path.separator, …)
23. **What is skeleton and stub? what is the purpose of those?** - Stub is a client side representation of the server, which takes care of communicating with the remote server. Skeleton is the server side representation. But that is no more in use… it is deprecated long before in JDK.
24. **What is the final keyword denotes?** - final keyword denotes that it is the final implementation for that method or variable or class. You can’t override that method/variable/class any more.
25. **What is the significance of ListIterator?** - You can iterate back and forth.
26. **What is the major difference between LinkedList and ArrayList?** - LinkedList are meant for sequential accessing. ArrayList are meant for random accessing.
27. **What is nested class?** - If all the methods of a inner class is static then it is a nested class.
28. **What is inner class?** - If the methods of the inner class can only be accessed via the instance of the inner class, then it is called inner class.
29. **What is composition?** - Holding the reference of the other class within some other class is known as composition.
30. **What is aggregation?** - It is a special type of composition. If you expose all the methods of a composite class and route the method call to the composite method through its reference, then it is called aggregation.
31. **What are the methods in Object?** - clone, equals, wait, finalize, getClass, hashCode, notify, notifyAll, toString
32. **Can you instantiate the Math class? -** You can’t instantiate the math class. All the methods in this class are static. And the constructor is not public.

33. **What is singleton?** - It is one of the design pattern. This falls in the creational pattern of the design pattern. There will be only one instance for that entire JVM. You can achieve this by having the private constructor in the class. For eg., public class Singleton { private static final Singleton s = new Singleton(); private Singleton() { } public static Singleton getInstance() { return s; } // all non static methods … }
34. **What is DriverManager?** - The basic service to manage set of JDBC drivers.
35. **What is Class.forName() does and how it is useful?** - It loads the class into the ClassLoader. It returns the Class. Using that you can get the instance (“class-instance".newInstance()).
36. **Inq** adds a question: Expain the reason for each keyword of

```
public static void main(String args[])
```

Java interview questions

1. **What is the Collections API?** - The Collections API is a set of classes and interfaces that support operations on collections of objects
2. **What is the List interface?** - The List interface provides support for ordered collections of objects.
3. **What is the Vector class?** - The Vector class provides the capability to implement a growable array of objects
4. **What is an Iterator interface?** - The Iterator interface is used to step through the elements of a Collection
5. **Which java.util classes and interfaces support event handling?** - The EventObject class and the EventListener interface support event processing
6. **What is the GregorianCalendar class?** - The GregorianCalendar provides support for traditional Western calendars
7. **What is the Locale class?** - The Locale class is used to tailor program output to the conventions of a particular geographic, political, or cultural region
8. **What is the SimpleTimeZone class?** - The SimpleTimeZone class provides support for a Gregorian calendar
9. **What is the Map interface?** - The Map interface replaces the JDK 1.1 Dictionary class and is used associate keys with values
10. **What is the highest-level event class of the event-delegation model?** - The java.util.EventObject class is the highest-level class in the event-delegation class hierarchy
11. **What is the Collection interface?** - The Collection interface provides support for the implementation of a mathematical bag - an unordered collection of objects that may contain duplicates
12. **What is the Set interface?** - The Set interface provides methods for accessing the elements of a finite mathematical set. Sets do not allow duplicate elements
13. **What is the purpose of the enableEvents() method?** - The enableEvents() method is used to enable an event for a particular object. Normally, an event is enabled when a listener is added to an object for a particular event. The enableEvents() method is used by objects that handle events by overriding their event-dispatch methods.
14. **What is the ResourceBundle class?** - The ResourceBundle class is used to store locale-specific resources that can be loaded by a program to tailor the program’s appearance to the particular locale in which it is being run.
15. **What is the difference between yielding and sleeping?** - When a task invokes its yield() method, it returns to the ready state. When a task invokes its sleep() method, it returns to the waiting state.
16. **When a thread blocks on I/O, what state does it enter?** - A thread enters the waiting state when it blocks on I/O.
17. **When a thread is created and started, what is its initial state?** - A thread is in the ready state after it has been created and started.
18. **What invokes a thread’s run() method?** - After a thread is started, via its start() method or that of the Thread class, the JVM invokes the thread’s run() method when the thread is initially executed.

19. **What method is invoked to cause an object to begin executing as a separate thread?** - The start() method of the Thread class is invoked to cause an object to begin executing as a separate thread.
20. **What is the purpose of the wait(), notify(), and notifyAll() methods?** - The wait(),notify(), and notifyAll() methods are used to provide an efficient way for threads to wait for a shared resource. When a thread executes an object’s wait() method, it enters the waiting state. It only enters the ready state after another thread invokes the object’s notify() or notifyAll() methods.
21. **What are the high-level thread states?** - The high-level thread states are ready, running, waiting, and dead
22. **What happens when a thread cannot acquire a lock on an object?** - If a thread attempts to execute a synchronized method or synchronized statement and is unable to acquire an object’s lock, it enters the waiting state until the lock becomes available.
23. **How does multithreading take place on a computer with a single CPU?** - The operating system’s task scheduler allocates execution time to multiple tasks. By quickly switching between executing tasks, it creates the impression that tasks execute sequentially.
24. **What happens when you invoke a thread’s interrupt method while it is sleeping or waiting?** - When a task’s interrupt() method is executed, the task enters the ready state. The next time the task enters the running state, an InterruptedException is thrown.
25. **What state is a thread in when it is executing?** - An executing thread is in the running state
26. **What are three ways in which a thread can enter the waiting state?** - A thread can enter the waiting state by invoking its sleep() method, by blocking on I/O, by unsuccessfully attempting to acquire an object’s lock, or by invoking an object’s wait() method. It can also enter the waiting state by invoking its (deprecated) suspend() method.
27. **What method must be implemented by all threads?** - All tasks must implement the run() method, whether they are a subclass of Thread or implement the Runnable interface.
28. **What are the two basic ways in which classes that can be run as threads may be defined?** - A thread class may be declared as a subclass of Thread, or it may implement the Runnable interface.
29. **How can you store international / Unicode characters into a cookie?** - One way is, before storing the cookie URLEncode it. URLEnocder.encoder(str); And use URLDecoder.decode(str) when you get the stored cookie.

Java interview questions

1. **What are synchronized methods and synchronized statements?** Synchronized methods are methods that are used to control access to an object. For example, a thread only executes a synchronized method after it has acquired the lock for the method’s object or class. Synchronized statements are similar to synchronized methods. A synchronized statement can only be executed after a thread has acquired the lock for the object or class referenced in the synchronized statement.
2. **What are different ways in which a thread can enter the waiting state?** A thread can enter the waiting state by invoking its sleep() method, blocking on I/O, unsuccessfully attempting to acquire an object’s lock, or invoking an object’s wait() method. It can also enter the waiting state by invoking its (deprecated) suspend() method.
3. **Can a lock be acquired on a class?** Yes, a lock can be acquired on a class. This lock is acquired on the class’s Class object.
4. **What’s new with the stop(), suspend() and resume() methods in new JDK 1.2?** The stop(), suspend() and resume() methods have been deprecated in JDK 1.2.
5. **What is the preferred size of a component?** The preferred size of a component is the minimum component size that will allow the component to display normally.
6. **What method is used to specify a container’s layout?** The setLayout() method is used to specify a container’s layout. For example, setLayout(new FlowLayout()); will be set the layout as FlowLayout.
7. **Which containers use a FlowLayout as their default layout?** The Panel and Applet classes use the FlowLayout as their default layout.
8. **What state does a thread enter when it terminates its processing?** When a thread terminates its processing, it enters the dead state.

9. **What is the Collections API?** The Collections API is a set of classes and interfaces that support operations on collections of objects. One example of class in Collections API is Vector and Set and List are examples of interfaces in Collections API.
10. **What is the List interface?** The List interface provides support for ordered collections of objects. It may or may not allow duplicate elements but the elements must be ordered.
11. **How does Java handle integer overflows and underflows?** It uses those low order bytes of the result that can fit into the size of the type allowed by the operation.
12. **What is the Vector class?** The Vector class provides the capability to implement a growable array of objects. The main visible advantage of this class is programmer needn't to worry about the number of elements in the Vector.
13. **What modifiers may be used with an inner class that is a member of an outer class?** A (non-local) inner class may be declared as public, protected, private, static, final, or abstract.
14. **If a method is declared as protected, where may the method be accessed?** A protected method may only be accessed by classes or interfaces of the same package or by subclasses of the class in which it is declared.
15. **What is an Iterator interface?** The Iterator interface is used to step through the elements of a Collection.
16. **How many bits are used to represent Unicode, ASCII, UTF-16, and UTF-8 characters?** Unicode requires 16 bits, ASCII require 7 bits (although the ASCII character set uses only 7 bits, it is usually represented as 8 bits), UTF-8 represents characters using 8, 16, and 18 bit patterns, UTF-16 uses 16-bit and larger bit patterns
17. **What is the difference between yielding and sleeping?** Yielding means a thread returning to a ready state either from waiting, running or after creation, where as sleeping refers a thread going to a waiting state from running state. With reference to Java, when a task invokes its yield() method, it returns to the ready state and when a task invokes its sleep() method, it returns to the waiting state
18. **What are wrapper classes?** Wrapper classes are classes that allow primitive types to be accessed as objects. For example, Integer, Double. These classes contain many methods which can be used to manipulate basic data types
19. **Does garbage collection guarantee that a program will not run out of memory?** No, it doesn't. It is possible for programs to use up memory resources faster than they are garbage collected. It is also possible for programs to create objects that are not subject to garbage collection. The main purpose of Garbage Collector is recover the memory from the objects which are no longer required when more memory is needed.
20. **Name Component subclasses that support painting?** The following classes support painting: Canvas, Frame, Panel, and Applet.
21. **What is a native method?** A native method is a method that is implemented in a language other than Java. For example, one method may be written in C and can be called in Java.
22. **How can you write a loop indefinitely?**

```
for(;;) //for loop
while(true); //always true
```

23. **Can an anonymous class be declared as implementing an interface and extending a class?** An anonymous class may implement an interface or extend a superclass, but may not be declared to do both.
24. **What is the purpose of finalization?** The purpose of finalization is to give an unreachable object the opportunity to perform any cleanup processing before the object is garbage collected. For example, closing a opened file, closing a opened database Connection.
25. **What invokes a thread's run() method?** After a thread is started, via its start() method or that of the Thread class, the JVM invokes the thread's run() method when the thread is initially executed.
26. **What is the GregorianCalendar class?** The GregorianCalendar provides support for traditional Western calendars.
27. **What is the SimpleTimeZone class?** The SimpleTimeZone class provides support for a Gregorian calendar.
28. **What is the Properties class?** The properties class is a subclass of Hashtable that can be read from or written to a stream. It also provides the capability to specify a set of default values to be used.

29. **What is the purpose of the Runtime class?** The purpose of the Runtime class is to provide access to the Java runtime system.
30. **What is the purpose of the System class?** The purpose of the System class is to provide access to system resources.
31. **What is the purpose of the finally clause of a try-catch-finally statement?** The finally clause is used to provide the capability to execute code no matter whether or not an exception is thrown or caught. For example,

```
try
{
//some statements
}
catch
{
// statements when exception is cought
}
finally
{
//statements executed whether exception occurs or not
}
```

32. **What is the Locale class?** The Locale class is used to tailor program output to the conventions of a particular geographic, political, or cultural region.
33. **What must a class do to implement an interface?** It must provide all of the methods in the interface and identify the interface in its implements clause.

Java interview questions

1. **What is a class?** A class is a blueprint, or prototype, that defines the variables and the methods common to all objects of a certain kind.
2. **What is a object?** An object is a software bundle of variables and related methods.An instance of a class depicting the state and behavior at that particular time in real world.
3. **What is a method?** Encapsulation of a functionality which can be called to perform specific tasks.
4. **What is encapsulation? Explain with an example.** Encapsulation is the term given to the process of hiding the implementation details of the object. Once an object is encapsulated, its implementation details are not immediately accessible any more. Instead they are packaged and are only indirectly accessible via the interface of the object
5. **What is inheritance? Explain with an example.** Inheritance in object oriented programming means that a class of objects can inherit properties and methods from another class of objects.
6. **What is polymorphism? Explain with an example.** In object-oriented programming, polymorphism refers to a programming language’s ability to process objects differently depending on their data type or class. More specifically, it is the ability to redefine methods for derived classes. For example, given a base class shape, polymorphism enables the programmer to define different area methods for any number of derived classes, such as circles, rectangles and triangles. No matter what shape an object is, applying the area method to it will return the correct results. Polymorphism is considered to be a requirement of any true object-oriented programming language
7. **Is multiple inheritance allowed in Java?** No, multiple inheritance is not allowed in Java.
8. **What is interpreter and compiler?** Java interpreter converts the high level language code into a intermediate form in Java called as bytecode, and then executes it, where as a compiler converts the high level language code to machine language making it very hardware specific
9. **What is JVM?** The Java interpreter along with the runtime environment required to run the Java application in called as Java virtual machine(JVM)
10. **What are the different types of modifiers?** There are access modifiers and there are other identifiers. Access modifiers are public, protected and private. Other are final and static.

11. **What are the access modifiers in Java?** There are 3 access modifiers. Public, protected and private, and the default one if no identifier is specified is called friendly, but programmer cannot specify the friendly identifier explicitly.
12. **What is a wrapper class?** They are classes that wrap a primitive data type so it can be used as a object
13. **What is a static variable and static method? What's the difference between two?** The modifier static can be used with a variable and method. When declared as static variable, there is only one variable no matter how instances are created, this variable is initialized when the class is loaded. Static method do not need a class to be instantiated to be called, also a non static method cannot be called from static method.
14. **What is garbage collection?** Garbage Collection is a thread that runs to reclaim the memory by destroying the objects that cannot be referenced anymore.
15. **What is abstract class?** Abstract class is a class that needs to be extended and its methods implemented, aclass has to be declared abstract if it has one or more abstract methods.
16. **What is meant by final class, methods and variables?** This modifier can be applied to class method and variable. When declared as final class the class cannot be extended. When declared as final variable, its value cannot be changed if is primitive value, if it is a reference to the object it will always refer to the same object, internal attributes of the object can be changed.
17. **What is interface?** Interface is a contact that can be implemented by a class, it has method that need implementation.
18. **What is method overloading?** Overloading is declaring multiple method with the same name, but with different argument list.
19. **What is method overriding?** Overriding has same method name, identical arguments used in subclass.
20. **What is singleton class?** Singleton class means that any given time only one instance of the class is present, in one JVM.
21. **What is the difference between an array and a vector?** Number of elements in an array are fixed at the construction time, whereas the number of elements in vector can grow dynamically.
22. **What is a constructor?** In Java, the class designer can guarantee initialization of every object by providing a special method called a constructor. If a class has a constructor, Java automatically calls that constructor when an object is created, before users can even get their hands on it. So initialization is guaranteed.
23. **What is casting?** Conversion of one type of data to another when appropriate. Casting makes explicitly converting of data.
24. **What is the difference between final, finally and finalize?** The modifier final is used on class variable and methods to specify certain behaviour explained above. And finally is used as one of the loop in the try catch blocks, It is used to hold code that needs to be executed whether or not the exception occurs in the try catch block. Java provides a method called finalize( ) that can be defined in the class. When the garbage collector is ready to release the storage ed for your object, it will first call finalize( ), and only on the next garbage-collection pass will it reclaim the objects memory. So finalize( ), gives you the ability to perform some important cleanup at the time of garbage collection.
25. **What is are packages?** A package is a collection of related classes and interfaces providing access protection and namespace management.
26. **What is a super class and how can you call a super class?** When a class is extended that is derived from another class there is a relationship is created, the parent class is referred to as the super class by the derived class that is the child. The derived class can make a call to the super class using the keyword super. If used in the constructor of the derived class it has to be the first statement.
27. **What is meant by a Thread?** Thread is defined as an instantiated parallel process of a given program.
28. **What is multi-threading?** Multi-threading as the name suggest is the scenario where more than one threads are running.
29. **What are two ways of creating a thread? Which is the best way and why?** Two ways of creating threads are, one can extend from the Java.lang.Thread and can implement the rum method or the run method of a different class can be called which implements the interface Runnable, and the then

implement the run() method. The latter one is mostly used as first due to Java rule of only one class inheritance, with implementing the Runnable interface that problem is sorted out.

30. **What is deadlock?** Deadlock is a situation when two threads are waiting on each other to release a resource. Each thread waiting for a resource which is held by the other waiting thread. In Java, this resource is usually the object lock obtained by the synchronized keyword.
31. **What are the three types of priority?** MAX_PRIORITY which is 10, MIN_PRIORITY which is 1, NORM_PRIORITY which is 5.
32. **What is the use of synchronizations?** Every object has a lock, when a synchronized keyword is used on a piece of code the, lock must be obtained by the thread first to execute that code, other threads will not be allowed to execute that piece of code till this lock is released.

Java software engineering interview questions

Question 1: What is the three tier model?
Answer: It is the presentation, logic, backend

Question 2: Why do we have index table in the database?
Answer: Because the index table contain the information of the other tables. It will be faster if we access the index table to find out what the other contain.

Question 3: Give an example of using JDBC access the database.
Answer:

```
try
{
Class.forName("register the driver");
Connection con = DriverManager.getConnection("url of db", "username","password");
Statement state = con.createStatement();
state.executeUpdate("create table testing(firstname varchar(20), lastname varchar(20))");
state.executeQuery("insert into testing values('phu','huynh')");
state.close();
con.close();
}
catch(Exception e)
{
System.out.println(e);
}
```

Question 4: What is the different of an Applet and a Java Application
Answer: The applet doesn't have the main function

Question 5: How do we pass a reference parameter to a function in Java?
Answer: Even though Java doesn't accept reference parameter, but we can pass in the object for the parameter of the function.
For example in C++, we can do this:

```
void changeValue(int& a)
{
a++;
}
void main()
{
int b=2;
changeValue(b);
}
```

however in Java, we cannot do the same thing. So we can pass the the int value into Integer object, and we pass this object into the the function. And this function will change the object.

Advanced enterprise Java interview questions

1) What is the purpose of garbage collection in Java, and when is it used?

The purpose of garbage collection is to identify and discard objects that are no longer needed by a program so that their resources can be reclaimed and reused. A Java object is subject to garbage collection when it becomes unreachable to the program in which it is used.

2) Describe synchronization in respect to multithreading.

With respect to multithreading, synchronization is the capability to control the access of multiple threads to shared resources. Without synchonization, it is possible for one thread to modify a shared variable while another thread is in the process of using or updating same shared variable. This usually leads to significant errors.

3) How is JavaBeans differ from Enterprise JavaBeans?

The JavaBeans architecture is meant to provide a format for general-purpose components. On the other hand, the Enterprise JavaBeans architecture provides a format for highly specialized business logic components.

4) In what ways do design patterns help build better software?

Design patterns helps software developers to reuse successful designs and architectures. It helps them to choose design alternatives that make a system reusable and avoid alternatives that compromise reusability through proven techniques as design patterns.

5) Describe 3-Tier Architecture in enterprise application development.

In 3-tier architecture, an application is broken up into 3 separate logical layers, each with a well-defined set of interfaces. The presentation layer typically consists of a graphical user interfaces. The business layer consists of the application or business logic, and the data layer contains the data that is needed for the application.

Java Language Questions

1. What is a platform?

   A platform is the hardware or software environment in which a program runs. Most platforms can be described as a combination of the operating system and hardware, like Windows 2000/XP, Linux, Solaris, and MacOS.

2. What is the main difference between Java platform and other platforms?

   The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

   The Java platform has two components:

   1. The Java Virtual Machine (Java VM)
   2. The Java Application Programming Interface (Java API)

3. What is the Java Virtual Machine?

The Java Virtual Machine is a software that can be ported onto various hardware-based platforms.

4. What is the Java API?

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets.

5. What is the package?

The package is a Java namespace or part of Java libraries. The Java API is grouped into libraries of related classes and interfaces; these libraries are known as packages.

6. What is native code?

The native code is code that after you compile it, the compiled code runs on a specific hardware platform.

7. Is Java code slower than native code?

Not really. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time bytecode compilers can bring performance close to that of native code without threatening portability.

8. What is the serialization?

The serialization is a kind of mechanism that makes a class or a bean persistence by having its properties or fields and state information saved and restored to and from storage.

9. How to make a class or a bean serializable?

By implementing either the java.io.Serializable interface, or the java.io.Externalizable interface. As long as one class in a class's inheritance hierarchy implements Serializable or Externalizable, that class is serializable.

10. How many methods in the Serializable interface?

There is no method in the Serializable interface. The Serializable interface acts as a marker, telling the object serialization tools that your class is serializable.

11. How many methods in the Externalizable interface?

There are two methods in the Externalizable interface. You have to implement these two methods in order to make your class externalizable. These two methods are readExternal() and writeExternal().

12. What is the difference between Serializalble and Externalizable interface?

When you use Serializable interface, your class is serialized automatically by default. But you can override writeObject() and readObject() two methods to control more complex object serailization process. When you use Externalizable interface, you have a complete control over your class's serialization process.

13. What is a transient variable?

A transient variable is a variable that may not be serialized. If you don't want some field to be serialized, you can mark that field transient or static.

14. Which containers use a border layout as their default layout?

The Window, Frame and Dialog classes use a border layout as their default layout.

15. How are Observer and Observable used?

Objects that subclass the Observable class maintain a list of observers. When an Observable object is updated it invokes the update() method of each of its observers to notify the observers that it has changed state. The Observer interface is implemented by objects that observe Observable objects.

16. What is synchronization and why is it important?

With respect to multithreading, synchronization is the capability to control the access of multiple threads to shared resources. Without synchronization, it is possible for one thread to modify a shared object while another thread is in the process of using or updating that object's value. This often causes dirty data and leads to significant errors.

17. What are synchronized methods and synchronized statements?

Synchronized methods are methods that are used to control access to an object. A thread only executes a synchronized method after it has acquired the lock for the method's object or class. Synchronized statements are similar to synchronized methods. A synchronized statement can only be executed after a thread has acquired the lock for the object or class referenced in the synchronized statement.

18. What are three ways in which a thread can enter the waiting state?

A thread can enter the waiting state by invoking its sleep() method, by blocking on I/O, by unsuccessfully attempting to acquire an object's lock, or by invoking an object's wait() method. It can also enter the waiting state by invoking its (deprecated) suspend() method.

19. Can a lock be acquired on a class?

Yes, a lock can be acquired on a class. This lock is acquired on the class's Class object.

20. What's new with the stop(), suspend() and resume() methods in JDK 1.2?

The stop(), suspend() and resume() methods have been deprecated in JDK 1.2.

21. What is the preferred size of a component?

The preferred size of a component is the minimum component size that will allow the component to display normally.

22. What method is used to specify a container's layout?

The setLayout() method is used to specify a container's layout.

23. Which containers use a FlowLayout as their default layout?

The Panel and Applet classes use the FlowLayout as their default layout.

24. What is thread?

A thread is an independent path of execution in a system.

25. What is multithreading?

Multithreading means various threads that run in a system.

26. How does multithreading take place on a computer with a single CPU?

The operating system's task scheduler allocates execution time to multiple tasks. By quickly switching between executing tasks, it creates the impression that tasks execute sequentially.

27. How to create multithread in a program?

You have two ways to do so. First, making your class "extends" *Thread* class. Second, making your class "implements" *Runnable* interface. Put jobs in a run() method and call start() method to start the thread.

28. Can Java object be locked down for exclusive use by a given thread?

Yes. You can lock an object by putting it in a "synchronized" block. The locked object is inaccessible to any thread other than the one that explicitly claimed it.

29. Can each Java object keep track of all the threads that want to exclusively access to it?

Yes.

30. What state does a thread enter when it terminates its processing?

When a thread terminates its processing, it enters the dead state.

31. What invokes a thread's run() method?

After a thread is started, via its start() method of the Thread class, the JVM invokes the thread's run() method when the thread is initially executed.

32. What is the purpose of the wait(), notify(), and notifyAll() methods?

The wait(),notify(), and notifyAll() methods are used to provide an efficient way for threads to communicate each other.

33. What are the high-level thread states?

The high-level thread states are ready, running, waiting, and dead.

34. What is the Collections API?

The Collections API is a set of classes and interfaces that support operations on collections of objects.

35. What is the List interface?

The List interface provides support for ordered collections of objects.

36. How does Java handle integer overflows and underflows?

It uses those low order bytes of the result that can fit into the size of the type allowed by the operation.

37. What is the Vector class?

The Vector class provides the capability to implement a growable array of objects

38. What modifiers may be used with an inner class that is a member of an outer class?

A (non-local) inner class may be declared as public, protected, private, static, final, or abstract.

39. If a method is declared as protected, where may the method be accessed?

A protected method may only be accessed by classes or interfaces of the same package or by subclasses of the class in which it is declared.

40. What is an Iterator interface?

The Iterator interface is used to step through the elements of a Collection.

41. How many bits are used to represent Unicode, ASCII, UTF-16, and UTF-8 characters?

Unicode requires 16 bits and ASCII require 7 bits. Although the ASCII character set uses only 7 bits, it is usually represented as 8 bits. UTF-8 represents characters using 8, 16, and 18 bit patterns. UTF-16 uses 16-bit and larger bit patterns.

42. What is the difference between yielding and sleeping?

When a task invokes its yield() method, it returns to the ready state. When a task invokes its sleep() method, it returns to the waiting state.

43. Is sizeof a keyword?

The sizeof operator is not a keyword in Java.

44. What are wrapped classes?

Wrapped classes are classes that allow primitive types to be accessed as objects.

45. Does garbage collection guarantee that a program will not run out of memory?

No, it doesn't. It is possible for programs to use up memory resources faster than they are garbage collected. It is also possible for programs to create objects that are not subject to garbage collection

46. What is the difference between preemptive scheduling and time slicing?

Under preemptive scheduling, the highest priority task executes until it enters the waiting or dead states or a higher priority task comes into existence. Under time slicing, a task executes for a predefined slice of time and then reenters the pool of ready tasks. The scheduler then determines which task should execute next, based on priority and other factors.

47. Name Component subclasses that support painting.

The Canvas, Frame, Panel, and Applet classes support painting.

48. What is a native method?

A native method is a method that is implemented in a language other than Java.

49. How can you write a loop indefinitely?

for(;;)--for loop; while(true)--always true, etc.

50. Can an anonymous class be declared as implementing an interface and extending a class?

An anonymous class may implement an interface or extend a superclass, but may not be declared to do both.

51. What is the purpose of finalization?

The purpose of finalization is to give an unreachable object the opportunity to perform any cleanup processing before the object is garbage collected.

52. Which class is the superclass for every class.

Object

53. What is the difference between the Boolean & operator and the && operator?

If an expression involving the Boolean & operator is evaluated, both operands are evaluated. Then the & operator is applied to the operand. When an expression involving the && operator is evaluated, the first operand is evaluated. If the first operand returns a value of true then the second operand is evaluated. The && operator is then applied to the first and second operands. If the first operand evaluates to false, the evaluation of the second operand is skipped. Operator & has no chance to skip both sides evaluation and && operator does. If asked why, give details as above.

54. What is the GregorianCalendar class?

The GregorianCalendar provides support for traditional Western calendars.

55. What is the SimpleTimeZone class?

The SimpleTimeZone class provides support for a Gregorian calendar.

56. Which Container method is used to cause a container to be laid out and redisplayed?

validate()

---

57. What is the Properties class?

The properties class is a subclass of Hashtable that can be read from or written to a stream. It also provides the capability to specify a set of default values to be used.

---

58. What is the purpose of the Runtime class?

The purpose of the Runtime class is to provide access to the Java runtime system.

---

59. What is the purpose of the System class?

The purpose of the System class is to provide access to system resources.

---

60. What is the purpose of the finally clause of a try-catch-finally statement?

The finally clause is used to provide the capability to execute code no matter whether or not an exception is thrown or caught.

---

61. What is the Locale class?

The Locale class is used to tailor program output to the conventions of a particular geographic, political, or cultural region.

---

62. What must a class do to implement an interface?

It must provide all of the methods in the interface and identify the interface in its implements clause.

---

63. What is an abstract method?

An abstract method is a method whose implementation is deferred to a subclass. Or, a method that has no implementation (an interface of a method).

---

64. What is a static method?

A static method is a method that belongs to the class rather than any object of the class and doesn't apply to an object or even require that any objects of the class have been instantiated.

---

65. What is a protected method?

A protected method is a method that can be accessed by any method in its package and inherited by any subclass of its class.

66. What is the difference between a static and a non-static inner class?

A non-static inner class may have object instances that are associated with instances of the class's outer class. A static inner class does not have any object instances.

67. What is an object's lock and which object's have locks?

An object's lock is a mechanism that is used by multiple threads to obtain synchronized access to the object. A thread may execute a synchronized method of an object only after it has acquired the object's lock. All objects and classes have locks. A class's lock is acquired on the class's Class object.

68. When can an object reference be cast to an interface reference?

An object reference be cast to an interface reference when the object implements the referenced interface.

69. What is the difference between a Window and a Frame?

The Frame class extends Window to define a main application window that can have a menu bar.

70. What do heavy weight components mean?

Heavy weight components like Abstract Window Toolkit (AWT), depend on the local windowing toolkit. For example, java.awt.Button is a heavy weight component, when it is running on the Java platform for Unix platform, it maps to a real Motif button. In this relationship, the Motif button is called the peer to the java.awt.Button. If you create two Buttons, two peers and hence two Motif Buttons are also created. The Java platform communicates with the Motif Buttons using the Java Native Interface. For each and every component added to the application, there is an additional overhead tied to the local windowing system, which is why these components are called heavy weight.

71. Which package has light weight components?

javax.Swing package. All components in Swing, except JApplet, JDialog, JFrame and JWindow are lightweight components.

72. What are peerless components?

The peerless components are called light weight components.

73. What is the difference between the Font and FontMetrics classes?

The FontMetrics class is used to define implementation-specific properties, such as ascent and descent, of a Font object.

74. What happens when a thread cannot acquire a lock on an object?

If a thread attempts to execute a synchronized method or synchronized statement and is unable to acquire an object's lock, it enters the waiting state until the lock becomes available.

75. What is the difference between the Reader/Writer class hierarchy and the InputStream/OutputStream class hierarchy?

The Reader/Writer class hierarchy is character-oriented, and the InputStream/OutputStream class hierarchy is byte-oriented.

76. What classes of exceptions may be caught by a catch clause?

A catch clause can catch any exception that may be assigned to the Throwable type. This includes the Error and Exception types.

77. What is the difference between throw and throws keywords?

The *throw* keyword denotes a statement that causes an exception to be initiated. It takes the Exception object to be thrown as argument. The exception will be caught by an immediately encompassing try-catch construction or propagated further up the calling hierarchy.

The *throws* keyword is a modifier of a method that designates that exceptions may come out of the mehtod, either by virtue of the method throwing the exception itself or because it fails to catch such exceptions that a method it calls may throw.

78. If a class is declared without any access modifiers, where may the class be accessed?

A class that is declared without any access modifiers is said to have package or friendly access. This means that the class can only be accessed by other classes and interfaces that are defined within the same package.

79. What is the Map interface?

The Map interface replaces the JDK 1.1 Dictionary class and is used associate keys with values.

80. Does a class inherit the constructors of its superclass?

A class does not inherit constructors from any of its superclasses.

81. Name primitive Java types.

The primitive types are byte, char, short, int, long, float, double, and boolean.

82. Which class should you use to obtain design information about an object?

The Class class is used to obtain information about an object's design.

83. How can a GUI component handle its own events?

A component can handle its own events by implementing the required event-listener interface and adding itself as its own event listener.

84. How are the elements of a GridBagLayout organized?

The elements of a GridBagLayout are organized according to a grid. However, the elements are of different sizes and may occupy more than one row or column of the grid. In addition, the rows and columns may have different sizes.

85. What advantage do Java's layout managers provide over traditional windowing systems?

Java uses layout managers to lay out components in a consistent manner across all windowing platforms. Since Java's layout managers aren't tied to absolute sizing and positioning, they are able to accommodate platform-specific differences among windowing systems.

86. What are the problems faced by Java programmers who don't use layout managers?

Without layout managers, Java programmers are faced with determining how their GUI will be displayed across multiple windowing systems and finding a common sizing and positioning that will work within the constraints imposed by each windowing system.

87. What is the difference between static and non-static variables?

A static variable is associated with the class as a whole rather than with specific instances of a class. Non-static variables take on unique values with each object instance.

88. What is the difference between the paint() and repaint() methods?

The paint() method supports painting via a Graphics object. The repaint() method is used to cause paint() to be invoked by the AWT painting thread.

89. What is the purpose of the File class?

The File class is used to create objects that provide access to the files and directories of a local file system.

90. What restrictions are placed on method overloading?

Two methods may not have the same name and argument list but different return types.

91. What restrictions are placed on method overriding?

Overridden methods must have the same name, argument list, and return type. The overriding method may not limit the access of the method it overrides. The overriding method may not throw any exceptions that may not be thrown by the overridden method.

92. What is casting?

There are two types of casting, casting between primitive numeric types and casting between object references. Casting between numeric types is used to convert larger values, such as double values, to smaller values, such as byte values. Casting between object references is used to refer to an object by a compatible class, interface, or array type reference.

93. Name Container classes.

Window, Frame, Dialog, FileDialog, Panel, Applet, or ScrollPane

94. What class allows you to read objects directly from a stream?

The ObjectInputStream class supports the reading of objects from input streams.

95. How are this() and super() used with constructors?

this() is used to invoke a constructor of the same class. super() is used to invoke a superclass constructor.

96. How is it possible for two String objects with identical values not to be equal under the == operator?

The == operator compares two objects to determine if they are the same object in memory. It is possible for two String objects to have the same value, but located indifferent areas of memory.

97. What an I/O filter?

An I/O filter is an object that reads from one stream and writes to another, usually altering the data in some way as it is passed from one stream to another.

98. What is the Set interface?

The Set interface provides methods for accessing the elements of a finite mathematical set. Sets do not allow duplicate elements.

99. What is the List interface?

The List interface provides support for ordered collections of objects.

100. What is the purpose of the enableEvents() method?

The enableEvents() method is used to enable an event for a particular object. Normally, an event is enabled when a listener is added to an object for a particular event. The enableEvents() method is used by objects that handle events by overriding their event-dispatch methods.

101. What is the difference between the File and RandomAccessFile classes?

The File class encapsulates the files and directories of the local file system. The RandomAccessFile class provides the methods needed to directly access data contained in any part of a file.

102. What interface must an object implement before it can be written to a stream as an object?

An object must implement the Serializable or Externalizable interface before it can be written to a stream as an object.

103. What is the ResourceBundle class?

The ResourceBundle class is used to store locale-specific resources that can be loaded by a program to tailor the program's appearance to the particular locale in which it is being run.

104. What is the difference between a Scrollbar and a ScrollPane?

A Scrollbar is a Component, but not a Container. A ScrollPane is a Container. A ScrollPane handles its own events and performs its own scrolling.

105. What is a Java package and how is it used?

A Java package is a naming context for classes and interfaces. A package is used to create a separate name space for groups of classes and interfaces. Packages are also used to organize related classes and interfaces into a single API unit and to control accessibility to these classes and interfaces.

106. What are the Object and Class classes used for?

The Object class is the highest-level class in the Java class hierarchy. The Class class is used to represent the classes and interfaces that are loaded by a Java program.

107. What is Serialization and deserialization?

Serialization is the process of writing the state of an object to a byte stream.
Deserialization is the process of restoring these objects.

108. what is tunnelling?

Tunnelling is a route to somewhere. For example, RMI tunnelling is a way to make RMI application get through firewall. In CS world, tunnelling means a way to transfer data.

109. Does the code in finally block get executed if there is an exception and a return statement in a catch block?

If an exception occurs and there is a return statement in catch block, the finally block is still executed. The finally block will not be executed when the System.exit(1) statement is executed earlier or the system shut down earlier or the memory is used up earlier before the thread goes to finally block.

110. How you restrict a user to cut and paste from the html page?

Using javaScript to lock keyboard keys. It is one of solutions.

111. Is Java a super set of JavaScript?

No. They are completely different. Some syntax may be similar.

112. What is a Container in a GUI?

A Container contains and arranges other components (including other containers) through the use of layout managers, which use specific layout policies to determine where components should go as a function of the size of the container.

113. How the object oriented approach helps us keep complexity of software development under control?

We can discuss such issue from the following aspects:

- Objects allow procedures to be encapsulated with their data to reduce potential interference.
- Inheritance allows well-tested procedures to be reused and enables changes to make once and have effect in all relevant places.
- The well-defined separations of interface and implementation allows constraints to be imposed on inheriting classes while still allowing the flexibility of overriding and overloading.

114. What is polymorphism?

Polymorphism allows methods to be written that needn't be concerned about the specifics of the objects they will be applied to. That is, the method can be specified at a higher level of abstraction and can be counted on to work even on objects of yet unconceived classes.

115. What is design by contract?

The design by contract specifies the obligations of a method to any other methods that may use its services and also theirs to it. For example, the preconditions specify what the method required to be true when the method is called. Hence making sure that preconditions are. Similarly, postconditions specify what must be true when the method is finished, thus the called method has the responsibility of satisfying the post conditions.

In Java, the exception handling facilities support the use of design by contract, especially in the case of checked exceptions. The assert keyword can be used to make such contracts.

116. What are use cases?

A use case describes a situation that a program might encounter and what behavior the program should exhibit in that circumstance. It is part of the analysis of a program. The collection of use cases should, ideally, anticipate all the standard circumstances and many of the extraordinary circumstances possible so that the program will be robust.

117. What is the difference between interface and abstract class?

- interface contains methods that must be abstract; abstract class may contain concrete methods.
- interface contains variables that must be static and final; abstract class may contain non-final and final variables.
- members in an interface are public by default, abstract class may contain non-public members.
- interface is used to "implements"; whereas abstract class is used to "extends".
- interface can be used to achieve multiple inheritance; abstract class can be used as a single inheritance.
- interface can "extends" another interface, abstract class can "extends" another class and "implements" multiple interfaces.
- interface is absolutely abstract; abstract class can be invoked if a main() exists.
- interface is more flexible than abstract class because one class can only "extends" one super class, but "implements" multiple interfaces.
- If given a choice, use interface instead of abstract class.

Basic Java interview questions

1. **What is a Marker Interface?** - An interface with no methods. Example: Serializable, Remote, Cloneable
2. **What interface do you implement to do the sorting?** - Comparable
3. **What is the eligibility for a object to get cloned?** - It must implement the Cloneable interface
4. **What is the purpose of abstract class?** - It is not an instantiable class. It provides the concrete implementation for some/all the methods. So that they can reuse the concrete functionality by inheriting the abstract class.
5. **What is the difference between interface and abstract class?** - Abstract class defined with methods. Interface will declare only the methods. Abstract classes are very much useful when there is a some functionality across various classes. Interfaces are well suited for the classes which varies in functionality but with the same method signatures.
6. **What do you mean by RMI and how it is useful?** - RMI is a remote method invocation. Using RMI, you can work with remote object. The function calls are as though you are invoking a local

variable. So it gives you a impression that you are working really with a object that resides within your own JVM though it is somewhere.

7. **What is the protocol used by RMI?** - RMI-IIOP
8. **What is a hashCode?** - hash code value for this object which is unique for every object.
9. **What is a thread?** - Thread is a block of code which can execute concurrently with other threads in the JVM.
10. **What is the algorithm used in Thread scheduling?** - Fixed priority scheduling.
11. **What is hash-collision in Hashtable and how it is handled in Java?** - Two different keys with the same hash value. Two different entries will be kept in a single hash bucket to avoid the collision.
12. **What are the different driver types available in JDBC?** - 1. A JDBC-ODBC bridge 2. A native-API partly Java technology-enabled driver 3. A net-protocol fully Java technology-enabled driver 4. A native-protocol fully Java technology-enabled driver For more information: Driver Description
13. **Is JDBC-ODBC bridge multi-threaded?** - No
14. **Does the JDBC-ODBC Bridge support multiple concurrent open statements per connection?** - No
15. **What is the use of serializable?** - To persist the state of an object into any perminant storage device.
16. **What is the use of transient?** - It is an indicator to the JVM that those variables should not be persisted. It is the users responsibility to initialize the value when read back from the storage.
17. **What are the different level lockings using the synchronization keyword?** - Class level lock Object level lock Method level lock Block level lock
18. **What is the use of preparedstatement?** - Preparedstatements are precompiled statements. It is mainly used to speed up the process of inserting/updating/deleting especially when there is a bulk processing.
19. **What is callable statement? Tell me the way to get the callable statement?** - Callablestatements are used to invoke the stored procedures. You can obtain the callablestatement from Connection using the following methods prepareCall(String sql) prepareCall(String sql, int resultSetType, int resultSetConcurrency)
20. **In a statement, I am executing a batch. What is the result of the execution?** - It returns the int array. The array contains the affected row count in the corresponding index of the SQL.
21. **Can a abstract method have the static qualifier?** - No
22. **What are the different types of qualifier and what is the default qualifier?** - public, protected, private, package (default)
23. **What is the super class of Hashtable?** - Dictionary
24. **What is a lightweight component?** - Lightweight components are the one which doesn't go with the native call to obtain the graphical units. They share their parent component graphical units to render them. Example, Swing components
25. **What is a heavyweight component?** - For every paint call, there will be a native call to get the graphical units. Example, AWT.
26. **What is an applet?** - Applet is a program which can get downloaded into a client environment and start executing there.
27. **What do you mean by a Classloader?** - Classloader is the one which loads the classes into the JVM.
28. **What are the implicit packages that need not get imported into a class file?** - java.lang
29. **What is the difference between lightweight and heavyweight component?** - Lightweight components reuses its parents graphical units. Heavyweight components goes with the native graphical unit for every component. Lightweight components are faster than the heavyweight components.
30. **What are the ways in which you can instantiate a thread?** - Using Thread class By implementing the Runnable interface and giving that handle to the Thread class.
31. **What are the states of a thread?** - 1. New 2. Runnable 3. Not Runnable 4. Dead
32. **What is a socket?** - A socket is an endpoint for communication between two machines.
33. **How will you establish the connection between the servlet and an applet?** - Using the URL, I will create the connection URL. Then by openConnection method of the URL, I will establish the connection, through which I can be able to exchange data.

34. **What are the threads will start, when you start the java program?** - Finalizer, Main, Reference Handler, Signal Dispatcher