Core Java - Dumps

Sample Code

```
public class TestOverload {
  public static void print(Float a, double b) {
     System.out.println("1"); }
  public static void print(double a, double b) {
     System.out.println("2"); }
  public static void print(float a, int b) {
     System.out.println("3"); }
  public static void print(float a, Double b) {
     System.out.println("4"); }
  public static void print(double a, int b) {
     System.out.println("5"); }

  public static void main(String args[]) {
     TestOverload.print(2,3.0);
  }
}
```

Which one of the following is the output from the main(String args[]) method in the TestOverload class defined in the sample code above?

1

2

3

4

5

Ans: 2

Sample Code

```
import java.io.*;

public class TestScope{
   public static void main(String args[]) {
       int i=0,k=0;
       for (i=1;i<20;i++) {                // Line A
           int j=i*2;                      // Line B
           k=j*k;
           System.out.println(j);
       }
       for (int m=1;m<10;m++) {            // Line C
           System.out.println(m);          // Line D
       }
       System.out.println("Value is "+m); // Line E
   }
}
```

Question

Which line in the sample code above contains the compile-time error?

Line A

Line B

Line C

Line D

Line E

Ans: Line E

Sample Code

a. public void foo() {..}
b. void foo() {..}
c. protected void foo() {..}
d. private void foo() {..}
e. public final void foo() {..}

Question

Referring to the sample code above, which of the methods can be overridden via inheritance?

a and c

a and e

a, b, and c

a, b, and d

c, d, and e

Ans: a,b, and c

Sample Code

```
EnhancedForStatement:
    for ( Type Identifier : Expression )
        Statement
```

Question

As defined in the sample code above, what interface must the instance variable obtained from Expression implement in order to use it with the "enhanced for loop" construct?

Map

Collection

Iterable

Enumeration

Queue

Ans: Collection

Question

What can an abstract class do that an interface CANNOT?

Define methods with no implementation.

Be defined with the protected keyword.

Include inner classes.

Define constants.

Define static methods.

Ans: Define methods with no implementation.

Scenario

You use an ArrayList as the implementation for a List collection.

Question

Referring to the scenario above, what happens when you add an element that exceeds the ArrayList's capacity?

This type of error cannot be trapped by the code.

The ArrayList expands automatically to fit the addition.

The call throws an ArrayIndexOutOfBounds exception.

The virtual machine terminates the application.

The add() call returns with a value of -1 rather than the index.

Ans: The ArrayList expands automatically to fit the addition.

Casting a short to which one of the following may result in a loss of information?

double

float

int

byte

long

Ans: byte

Which one of the following command lines do you use in order to extract the contents of jar file tools.jar?

tar fx tools.jar

jar xf tools.jar

java java.util.zip.Jar -e tools.jar

jar evf tools.jar

unjar tools.jar

Ans: jar xf tools.jar

```
MessageFormat format = new MessageFormat(
   "First {2} Second {1}");
String[] input = {"one", "two", "three"};
System.out.println(format.format(input));
```

Question

When the sample code above is executed, what is printed out?

First Second

First two Second three

First one Second two

First three Second two

First two Second one

Ans: First three Second two

If you want to maintain a FIFO (First in First Out) data structure where added elements are NOT available until some time period has expired, which Collection implementation do you use?

PriorityQueue

PriorityStack

DelayStack

Ans: DelayQueue

DelayQueue

LinkedStackQueue

String
doubleStr="4.6";
double doubleVal;

Question

Using the definitions of doubleStr and doubleVal as in the sample code above, what statement do you use to convert doubleStr to a value that can be stored in doubleVal?

Double.parseDouble(doubleStr);

val(doubleStr);

String.toDouble(doubleStr);

Double.fromString(doubleStr);

new Double(doubleStr).double();

Ans: Double.parseDouble(doubleStr);

Sample Code

```
public class TestPassing {
  private int value;

  public static void add(int a, int b) {a=a+b;}
  public void subtract(TestPassing tp, int b) {
     tp.value=tp.value-b;
  }
  public void setup(int value) {this.value=5;}
  public void run() {
     value=2;
     setup(value);
     add(value,5);
     subtract(this,3);
     System.out.println(value);
  }
}
```

Question

Referring to the sample code above, what is the output from "new TestPassing().run()"?

-1

2

4

5

7

Ans: 2

Sample Code

```
int a=5,b=7;
int c= a+=2*3+b--;
```

Question What is the value of c after executing the sample code above?

12

18

23

27

28

Ans: 18

Which one of the following is a requirement of a JavaBean component?

Implement the java.beans.Bean interface.

Be serializable.

Provide a BeanContext object.

Implement the java.beans.Customizer interface.

Extend the java.beans.Bean class.

Ans: Be serializable.

How is a static inner class declared?

Inside another class and cannot be instantiated by any class other than the one within which it is embedded

Inside another class and cannot be instantiated

As an immutable utility class

Next to another class in the same source file

Inside another class, but it has no reference to the instance fields of the parent class

Ans: Inside another class, but it has no reference to the instance fields of the parent class

Q: In Java, what does Collator do?

It implements a sorting routine.

It formats messages based on locale.

It abstracts a list interface.

It allows keys to be created for hashcodes for a locale.

It allows more flexible comparison of strings for a given locale.

Ans: It allows more flexible comparison of strings for a given locale.

Sample Code

```
String s = "Hello, World";
s.replace('o', 'u');
s.toUpperCase();
System.out.println(s.substring(7));
```

After manipulating the string in the sample code above, what is printed?

WURLD

Ans: World

WORLD

world

Wurld

Code

```
class A extends Thread {
 public void run() {
   while(true) {
      System.out.println("Hello, World!");
   }
 }
 public static void main(String[] s) {
   A a = new A();
   a.setDaemon(true);
   a.start();
 }

}
```

What is the result of executing the sample code above?

A java.lang.StackOverflowException is thrown.

A java.lang.OutOfMemoryError is thrown.

The system exits immediately with no output or Hello, World! printed a few times.

The CPU is used up completely by the infinite loop.
Ans: The system exits immediately with no output or Hello, World! printed a few times.

Sample Code

```
System.out.format("%10.3f%n", Math.PI);
```

With the value of PI at 3.141592653589793, what is printed out with the line in the sample code above?

000003.142

3.141

3.142

3.14159265

3.142

Ans: 3.142

Question Which one of the following do you put at the end of a printf format statement to generate a platform-independent newline?

\n\t

%n

\r

\n\r

\r\n

Ans: %n

Sample Code CLASSPATH=app3.jar;app1.jar;\lib\Sys.class

Question Given the classpath in the sample code above, where is the first user defined place that the JVM searches for "com.foo.App1"?

%JAVA_HOME%\lib\core.jar

app3.jar

%JAVA_HOME%\lib\lib.jar

app1.jar

\lib\Sys.class
Ans: app3.jar

Sample Code long millis = System.currentTimeMillis();

Referring to the sample code above, what is the time base for the number of milliseconds reported?

Since midnight on October 15, 1582 (the start of the Gregorian calendar)

Since midnight on January 1, 1800

Since midnight on January 1, 1900

Since midnight on January 1, 1970

Since the last time the machine rebooted
Ans: Since midnight on January 1, 1970

Sample Code

```
double d;
try {
  d = 0 / 0.0;
} catch (ArithmeticException ae) {
  d = -1;
} catch (NumberFormatException nfe) {
  d = -2;
 }
System.out.println(d);
```

Question In the sample code above, what is the value of d when the results are printed out?

-1

-2

-Infinity

Infinity

NaN
Ans: NaN

Q:public static void main(String[] args) {

```
int i = 5;
if (i++ == 5 || false) {
        System.out.println(i);

  i += i;
        System.out.println(i);

}
System.out.println(i);
```

Ans : 12

Sample Code

```
class Counter {
  int count = 0;
  int increment() {
     int n=count;
     count = n + 1;
     return n;
  }
}
```

Question Referring to the above sample code, how do you make increment() thread-safe?

Declare increment() synchronized.

Implement the java.lang.SingleThreaded interface.

Declare increment() as static final.

Modify increment() to stop all other threads.

Declare Counter synchronized.

Ans:
Declare increment() synchronized.

List

a. lib.jar
b. the standard Java class libraries
c. the 'current directory'
d. c:\libs\ant.jar
e. c:\libs\javacore.jar

Question Referring to the list above, given the classpath "c:\libs\ant.jar;c:\libs\javacore.jar;." (on a Windows platform), in what order does the JVM search for classes?

a, d, e, and c

b, a, d, e, and c

b, d, and c

b, d, e, and c

d, e, and c

Ans: b, d, e, and c

The search order for J2SDK, using the example code above, is:

1. The bootstrap classpath, which is in the sun.boot.class.path property,
2. The extension directories, which is in the java.ext.dirs property,
3. The default system classpath,
4. The current working directory,
5. The myclasses.zip file that is located in the "root" (/) file system,
6. The classes directory in the Product directory in the "root" (/) file system.

Sample Code java com.brainbench.TestCommandLine -p Parameter1 Parameter2 Parameter3 Parameter4

Question Given that the user enters the command line shown in the sample code above, how many elements are contained in the array that is passed to public static void main(String args[]) in TestCommandLine?

Three

Four

Five

Six

Seven

Ans : Five

Sample Code

```
public double SquareRoot(double value)
   throws ArithmeticException {
 if (value >= 0) return Math.sqrt(value);
 else throw new ArithmeticException();
}

public double func(int x) {
  double y = (double) x;
  y *= -9.0;
  try {
    y = SquareRoot( y );
  } catch(ArithmeticException e) {
    y /= 3;
  } finally { y += 10; }
  return y;
}
```

Question Referring to the sample code above, what value is returned when you invoke method func(9)?

-27

-17

9

NaN

Ans: -17

Question Which one of the following statements returns the IP address of www.brainbench.com?

InetAddress ip=InetAddress.getByName("www.brainbench.com");

InetAddress ip=Socket.DNSLookup("www.brainbench.com");

InetAddress ip=System.getHostByName("www.brainbench.com");

InetAddress ip=new InetAddress("www.brainbench.com");

String ip=InetAddress.DNSLookup("www.brainbench.com");

Ans : InetAddress ip=InetAddress.getByName("www.brainbench.com");

Q:

When you call the deleteOnExit() method for a File, at what point does the file get deleted?

When all the threads in the current thread group end

When the current method ends

When the virtual machine exits

When you close the file

When the current thread ends

Ans; When the virtual machine exits

File.deleteOnExit Method

Deletes the file or folder when the object is no longer needed and the system reclaims the object through garbage collection.

Package: java.io

Q:

public final synchronized void foo()

Which one of the following statements is true regarding the method signature in the sample code above?

This is the same as declaring the method private.

Only one synchronized method can be invoked at a time for the entire class.

All final variables referenced in the method can be modified by only one thread at a time.

The method cannot be overridden and is callable by only one thread at a time on an object instance.

Methods that are synchronized cannot be final.

Ans: The method cannot be overridden and is callable by only one thread at a

time on an object instance.

Sample Code

```
class SuperClass {
  public void printIt() {System.out.println("SuperClass");}
  public void printIt(boolean print) {
     if (print) {
        System.out.println("Super-part 2");
     } else {
        printIt();
     }
  }
}

class SubClass extends SuperClass {
  public void printIt() {System.out.println("SubClass");}
}

public class TestSub {
  public static void main(String args[]) {
     SuperClass sc=new SubClass();
     sc.printIt();
     sc.printIt(false);
  }
}
```

Question

What is the output from running the command java TestSub after compiling the sample code above?

SubClass
SuperClass

SubClass
SubClass

SuperClass
SuperClass

SuperClass
SubClass

SubClass
Super-part 2

Ans: SubClass
SubClass

Sample Code

```
public double SquareRoot(double value)
   throws ArithmeticException {
 if (value >= 0) return Math.sqrt(value);
 else throw new ArithmeticException();
}

public double func(int x) {
  double y = (double) x;
  try {
    y = SquareRoot( y );
  } catch(ArithmeticException e) {
     y = 0;
  } finally {
     --y;
  }
  return y;
}
```

Question

Referring to the sample code above, what value is returned when you invoke method func(4)?

-2.0

-1.0

0

1.0

2.0

Ans : 1.0

Question

In a text file, how do you count the number of lines?

Subclass FilterInputStream and count the number of \n characters that go by.

Ask a LineNumberInputStream after reading the whole file.

Ask a LineNumberReader before reading the whole file.

Ask a LineNumberReader after reading the whole file.

Read the file a line at a time via a BufferedOutputStream.

Ans: Ask a LineNumberReader after reading the whole file.

Question

How does Sun recommend that Java package namespaces be used in order to guarantee unique namespaces even when installing third party packages?

Use the name of your company as the whole package name.

Use the head developer's name in the package names.

Reverse a registered domain name.

Register the package name at java.sun.com.

Use the product's name.
Ans :

Sample Code

```
class Foo {
   List list;

   // ... other code

   public int size() {
       return list.size();
   }

}
```

Question

Referring to the sample code above, how does the javadoc for the size() method describe the return value?

@return Size of list

@returns int Size of list

@return(int) Size of list

@return int {desc:Size of list}

@returned Size of list

Ans : @return Size of list

Q:
What does a classpath do?

It exports Java code remotely.

It defines a search path for loading class files from disk.

It describes the security behavior of java.security.

It specifies which Java files must be compiled.

It protects the multiple JVMs from interfering with each other.

Ans : It defines a search path for loading class files from disk.

Q:

What java.util class method do you use to sort an array of beans, where the method can specify the order of sorting?

Arrays.sort(beans, comparator);

Collection.sort(beans, comparator);

Algorithms.quickSort(beans, DESCENDING);

ArrayList.sort(beans, ASCENDING);

ArrayList.sort(beans, comparator);

Ans: Arrays.sort(beans, comparator);

Sample Code

```
public class Sample {
 String name;
 private Sample(String name) {
  this.name = name;
 }
 @Override
 public int hashCode() {
  return name.hashCode();
 }
 @deprecated
 public static String getCompanyName() {
  return "Brainbench";
 }
 public String getName() {
  return name;
 }
}
```

Question

Why does the sample code above cause a compilation error?

The class does not have a public no-argument constructor.

The hash code method is spelled hashcode, not hashCode.

The annotation for deprecated methods is @Deprecated, not @deprecated.

The class does not have any public constructors.

The annotation for overriding methods is @override, not @Override.

Ans : **The annotation for deprecated methods is @Deprecated, not @deprecated.**

Q: When using the Sun development environment, at which one of the following times does JIT compilation happen?

At runtime, when the system pauses for garbage collection

At runtime, before the code is first executed

At runtime, when a class is loaded

At runtime, during the bytecode verification phase

At compilation-time with the javac compiler

Ans : At runtime, before the code is first executed

Sample Code

```
synchronized void move() throws Exception, java.io.IOException {

  Additional code here

}
```

Question

Referring to the above sample code, how do you make a method in the same class invoke move()?

Enclose the call inside a try block followed by a catch(Exception e) block.

Enclose the call inside a try block.

Follow the call with a catch(java.lang.Exception e) block.

Enclose the call inside a catch(Exception e) block.

Enclose the call inside a try block followed by a finally block.

Ans :
Enclose the call inside a try block followed by a catch(Exception e) block

Sample Code

```
RoundingMode mode = RoundingMode.???;
BigDecimal bigl = new BigDecimal(-11);
BigDecimal big2 = new BigDecimal(2);
System.out.println(bigl.divide(big2, mode));
```

Question

In the sample code above, which enumerated type of RoundingMode produces a rounding behavior resulting in -5 when the input number is
-5.5 (-11/2)?

UNNECESSARY

CEILING

HALF_EVEN

UP

HALF_UP

Ans : CEILING

Sample Code

```
public static void main(String[] args) {
  list.add(args);
  list.add(args);
  list.add(args);
  for (String[] strings: list) {
     for (String string: strings) {
        System.out.print(string);
     }
     System.out.println();
  }
}
```

Question

In the sample code above, how do you declare the variable list so the code compiles, and compiles with NO warnings?

List<String[]> list = new ArrayList<String[]>();

List list = new ArrayList();

List<String[]> list = new ArrayList();

List list = new ArrayList<String[]>();

List<String> list = new ArrayList<String>();
Ans : List<String[]> list = new ArrayList<String[]>();

Sample Code

```
Connection con=DriverManager.getConnection
("jdbc:somedb:employees");
Statement st=con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,

                        ResultSet.CONCUR_UPDATABLE);
ResultSet rs=st.executeQuery("Select emp_no,vacation_days "
                            + "from employees;");
```

Question

After the ResultSet in the sample code above is created, what happens if another user changes a record in the employees table?

A security exception is thrown if another user tries to access the data that was changed.

The updated data appears in the result set as soon as it is written to the database.

The result set is read only.

The next attempt to access the result set throws a ConcurrentUpdateException.

The other user's code is blocked until this result set is closed.

Ans : The updated data appears in the result set as soon as it is written to the database.

Question

Which one of the following expressions results in a positive value for x?

int x = -1; x = x >>> 32;

int x = -1; x = x << 16;

int x = -1; x = x >> 5;

byte x = -1; x =(byte) (x >>> 5);

int x = -1; x = x >>> 5;

Ans: int x = -1; x = x >>> 5;

Sample Code

```
public class Test {
 public static void main(String args[]) {
   int i = args.length;
   switch (i) {
     case 0: System.out.println("Zero");
     case 1: System.out.println("One");
     case 2: System.out.println("Two");
     case 3: System.out.println("Three");
     default: System.out.println("Default");
   }
 }
}
```

Question

Where do you add @SuppressWarnings("fallthrough") to the sample code above in order for the following command to NOT generate any warnings?

javac -Xlint:fallthrough Test.java

Within the javadoc for the class

Within the javadoc for the main method

Before each case clause that falls through

Above the main method declaration

Immediately above the switch statement
Ans: Above the main method declaration

Sample Code

```
ByteArrayOutputStream baos = new ByteArrayOutputStream();
ObjectOutputStream out = new ObjectOutputStream(baos);
out.writeObject(new StringBuffer("Hello"));
byte bArray[]= baos.toByteArray();
```

Question

Referring to the above sample code, what does the array named bArray contain after execution?

Unicode values for each character in Hello

A serialized version of a StringBuffer object string Hello

A reference to ByteOutputStream b

The status of ByteArrayOutputStream b (0x20 open, 0x21 closed)

A hash code created from StringBuffer

Ans: A serialized version of a StringBuffer object string Hello

Question

What does the values() method of an enumeration return?

A LinkedList containing the values of the type in declaration order

An ArrayList containing the values of the type in declaration order

An array containing the values of the type in declaration order

A TreeMap, where each key-value pair represents the ordered position of the enumeration and its value

A Set containing the values of the type
Ans: An array containing the values of the type in declaration order

a. There are no mutator, or setter methods.
b. The class is declared final to prevent subclassing.
c. The instance fields have private accessibility.

Question

Which of the features in the above list are found in an immutable class such as the String class?

a

a and b

a and c

b and c

a, b, and c
Ans : a and b

Question

What is the purpose of the transient keyword?

It prevents the variable from being serialized.

It prevents the variable from being accessed beyond the scope of the defining file.

It allows the variable to be accessed across multiple threads.

It prevents the variable from being accessed across multiple threads.

It allows the variable to be accessed globally.

Ans : It prevents the variable from being serialized.

Question

How do you create a fixed size thread pool of ten threads?

ExecutorService threadpool =
  Thread.createThreadPool(10);

ThreadPool threadpool =
  Executors.newFixedThreadPool(10);

ThreadPool threadpool =
 new ThreadPool(10);

ExecutorService threadpool =
  Executors.newFixedThreadPool(10);

Executor threadpool =
  Executors.newThreadPool(10);

Ans : ThreadPool threadpool =
new ThreadPool(10);

Sample Code

```
public class Test {
 public static void main(String[] args) {
   foo("foo");
   foo(new String("foo"));
   foo("bar");
 }

 private static void foo(String arg) {
   if (arg == "foo") {
     arg = "1";
   } else if (arg.equals("foo")) {
     arg = "2";
   } else if (arg.equals("baR")) {
     arg = "3";
   } else {
     arg = "4";
   }

   System.out.print(arg);
 }
}
```

What does the sample code above print?

113

114

124

223

224

Ans : 124

Sample Code

```
<APPLET CODE=MyApplet.class>
 <PARAM NAME=Parameter VALUE=12>
</APPLET>
```

Question

Referring to the above sample code, how do you get the value of the parameter in the applet?

String param = Applet.getParameter("Parameter");

String param = System.getParameter("Parameter");

String param = getApplet().getParam("Parameter");

String param = getParameter("Parameter");

int param = getParameterAsInt("Parameter");

Ans: String param = getParameter("Parameter");

Question

When subclassing the Thread class, which one of the following is a LIMITATION on the subclass?

It must declare the class final.

It must implement the ThreadPolicy interface.

It must catch the ThreadDeath exception.

It is forced to declare all of its methods static.

It cannot subclass any other class.

Ans: It cannot subclass any other class.

Sample Code

```
1   public class Outside {
2
3      private int a;
4
5      public void foo(int b) {
6         int c = a;
7         int d = b;
8         class Inside {
9            public Inside() {
10              // What can I see here?
11           }
12        }
13        Inside e = new Inside();
14     }
15
16  }
```

Question

Which variables are visible at line 10 in the sample code above?

a

a and b

c and d

a, c, and d

a, b, c, and d

Ans : a

Question Which one of the following command line arguments do you use to increase the maximum Java heap size?

-Xmx

-mx

-Xms

-Ms

-heap

Ans : -Xmx

java -Xms<initial heap size> -Xmx<maximum heap size>

Defaults are:

java -Xms32m -Xmx128m

Code

```
public void printIt(String txt) {
    Pattern wordBreakPattern =
        Pattern.compile("[\\s]");
    String words[] = wordBreakPattern.split(txt)
    for (String word: words) {
         System.out.println(word);
    }
}
```

Question Referring to the sample code above, what is the result when you invoke the following statement?
printIt("Hello\nWorld\t!");

The program outputs the following:
Hello
World

The program outputs the following:
HelloWorld!

The program throws java.util.NoSuchElementException.

The program outputs the following:
Hello
World
!

The program outputs the following:
Hello
World!

Ans : The program outputs the following:
Hello
World
!

Sample Code

```
public synchronized int doIt(int z) {
   int x=5, count=0;
   while(otherMethod()) {
       try {
         wait();
       } catch(InterruptedException e) {
       }
   }
   for (int i=0;i<z;i++)
     count += x;
   int result = count - x;
   notifyAll();
   return result;
}
```

Question

Using the sample code above, what happens when you invoke notifyAll() in method doIt() do?

It signals waiting threads that they can run if they get the object's monitor.

It does nothing unless the method is overridden.

It changes all class methods to synchronized until release() is invoked.

It informs other threads that this thread has completed doIt().

It calls the notifyAll() method as defined in this class and does whatever is defined there.

Ans: It signals waiting threads that they can run if they get the object's monitor.

Sample Code

```
/**
* The (horizontal/vertical) distances of point (width/height)
*/
public int width, height;
```

Question

Referring to the sample code above, what comments are in the generated javadoc for the variables width and height?

Choice 1 ○

width gets The vertical distances of point height
height gets The horizontal distances of point width

Choice 2 ○

width gets The horizontal distances of point width
height gets The vertical distances of point height

Choice 3 ○

width gets nothing
height gets
  The (horizontal/vertical) distances of point (width/height)

Choice 4 ◉

width gets
  The (horizontal/vertical) distances of point (width/height)
height gets
  The (horizontal/vertical) distances of point (width/height)

Choice 5 ○

width gets
  The (horizontal/vertical) distances of point (width/height)
height gets nothing

Option 4

Scenario

A file named brainbench.txt exists in the directory c:\tests\study

File file = new File(".." + File.separator + "study"
  + File.separator + "brainbench.txt");

Question

Working with the file described in the scenario above, what are the results of calls to getParent() and getCanonicalPath() on the associated File object?

getParent() returns ..\study
getCanonicalPath returns
c:\tests\study\brainbench.txt

getParent() returns ..\study
getCanonicalPath returns
c:\tests\study\..\study\brainbench.txt

getParent() returns c:\tests\study
getCanonicalPath returns
..\study\brainbench.txt

getParent() returns ..\study
getCanonicalPath returns
..\study\brainbench.txt

getParent() returns c:\tests\study
getCanonicalPath returns
c:\tests\study\..\study\brainbench.txt

**Ans: getParent() returns ..\study
getCanonicalPath returns
c:\tests\study\brainbench.txt**

Sample Code

```
class StopWatch {
 java.util.Calendar call,cal2;

 public void start() {
   call = java.util.Calendar.getInstance();
 }
 public void stop() {
   cal2 = java.util.Calendar.getInstance();
 }
 public long getDifference() {
   long diff = 0;
   try {
     diff = ____________________; // in seconds
   } catch (Exception e) {
     diff = -1;
   }
   return diff;
 }
}
```

Question

Referring to the sample code above, in the getDifference() method, to which one of the following do you set the local variable diff so that the number of seconds elapsed is returned?

(long) (cal2 - cal1)

(cal2 - cal1) / System.currentTimeMillis()

(**cal2.getTime().getTime() - cal1.getTime().getTime()) / 1000**

cal2.get(java.util.Calendar.SECOND) - cal1.get(java.util.Calendar.SECOND)

cal2.getTimeInMillis() - cal1.getTimeInMillis()

Ans: (cal2.getTime().getTime() - cal1.getTime().getTime()) / 1000

Sample Code

```
class X {
 public int i=5;
}
class Y extends X {
 public int i=10;
}
public class Test {
 public static void main(String[] args) {
   X x = new Y();
   System.out.println("x.i = " + x.i );
 }
}
```

Question

Given the sample code above, what is the result when the code is compiled and run?

The code does not compile because class Y has no constructor.

The code does not compile because class Y cannot redefine member variable i, which is already declared in its super class X.

"x.i = 10" is printed to standard out.

"x.i = 5" is printed to standard out.

The code does compile, but it does result in a ClassCastException at runtime; you cannot assign an instance of Y to a variable of type X.

Ans: "x.i = 5" is printed to standard out.

Sample Code

```
public interface Foo <E>{
   void bar(E e);
   Iterator<E> iterator();
}
```

Question

For the interface declared in the sample code above, how do you refer to E?

As a parameterized generic type

As a type parameter

As an invocation parameter

As a generic type

As an actual type argument

Ans: As a parameterized generic type

Scenario

You want to define a method that is given an array and a collection to copy into.

Question

In the scenario above, how do you define the method signature to avoid copying the wrong type of object into the collection?

<T> void copy(T[] array, Collection<T> col)

<T> void copy(T[] array, Collection col<T>)

void copy(T array[], Collection col<T>)

void copy(Object[] a, Collection c)

void copy(T[] array, Collection<T> col)

Ans: <T> void copy(T[] array, Collection<T> col)

Question

What happens when you add a primitive data type into a collection?

The primitive data type is added as is.

The data type is automatically boxed into the appropriate wrapper class and then added.

A compilation error occurs.

The data type is converted into a Double and then added.

The data type is converted to a String and then added.
Ans: option 2-- The data type is automatically boxed into the appropriate wrapper class and then added.

Question

Which one of the following lines of code do you use to return the HTML text of the root Web page at www.brainbench.com?

Object content=new URL("http://www.brainbench.com").readObject();

String content=new FileInputStream("http://www.brainbench.com").read();

HTMLDocument doc=HTMLDocument.create("http://www.brainbench.com");

String content=URL.get("http://www.brainbench.com").getContent();

Object content=new URL("http://www.brainbench.com").getContent();
Ans: Object content=new URL("http://www.brainbench.com").getContent();

Sample Code

```
1. import java.util.*;
2. public class Test {
3.   // calcCache will reference a large HashMap that is
4.   // only needed for the duration of doComputation().
5.   private Map calcCache = null;
6.   public void doWork() {
7.     init();
8.     doComputation();
9.     // Insert code fragment here
10.  }
11.  private void init() {
12.    // initialize calcCache code here ...
13.  }
14.  private void doComputation() {
15.    // computation code here ...
16.  }
17.}
```

Question

What code fragment do you add to the sample code above at Line 9 in order to allow the memory held by the large HashMap to be reclaimed by the system garbage collector?

free calcCache;

System.getRuntime().dealloc(calcCache);

calcCache = null;

System.free(calcCache);

dealloc(calcCache);

Ans: calcCache=null;

Question

Which one of the following classes do you subclass in order to package several locale-specific versions of a set of greetings?

java.text.CollationKey

java.util.ResourceBundle

java.util.TimeZone

java.text.RuleBasedCollator

java.lang.Cloneable
Ans: java.util.ResourceBundle

Sample Code

```
class A {}
class B extends A {}
class C extends A implements Runnable { // .. }
class D extends C {}
class E extends C {}

A Aobj=new A();
B Bobj=new B();
C Cobj=new C();
D Dobj=new D();
E Eobj=new E();
```

Question

Given the classes defined in the sample code above, which one of the following cast expressions fails?

Runnable r = (Runnable) Dobj;

A a = (A) Bobj;

A a = (A) ((Runnable) Cobj);

D d = (D) Eobj;

A a = (A) Eobj;

Ans :

D d = (D) Eobj;

Question

What is the difference between a Reader class and an InputStream class?

A Reader buffers its input in memory; an InputStream reads its input directly from a device.

A Reader can use files or pipes; an InputStream can only read from files.

A Reader is guaranteed not to deadlock when multiple threads are reading from the same file; an InputStream is not.

A Reader works with random-access files; an InputStream works only with sequential files.

An instance of a Reader class reads characters; an instance of an InputStream class reads bytes.

Ans: An instance of a Reader class reads characters; an instance of an InputStream class reads bytes.

Question

What is the purpose of acquiring an object's monitor by using a synchronized(object) {} construct before accessing the object's resources?

To ensure that objects accessed on remote machines have their dates expressed in the local time zone

To establish scheduling priority over other threads so that the bytecode interpreter gives more processor time to the synchronized thread

To prevent multiple threads from accessing the same resource at the same time, which could leave the resource in an undefined state

To prevent flicker on the user's screen during updates that cover more than 80 percent of the display area

To prevent a deadlock or race situation from occurring between multiple threads during event handling

Ans: To prevent multiple threads from accessing the same resource at the same time, which could leave the resource in an undefined state

Sample Code

```
Float f=new Float(3.1);
Integer i=new Integer(1);
long l=2;

System.out.println("Result is "+l+f+i);
```

Question What is the output from the sample code above?

Result is 4.12

Result is 5.11

Result is 6.1

Result is 23.11

Result

Ans: Result is 23.11

Question Which one of the following statements is true for a non-abstract class that implements the Iterator interface?

It contains implementations of the hasNext() and next() methods.

It works with the StringTokenizer class.

It has implementations of the nextElement() and hasMoreElements() methods.

It is an implementation of the Collection interface.

It is used to store associative arrays.

Ans: It contains implementations of the hasNext() and next() methods.

The Iterator Interface also has a remove() method

Sample Code

```
public class ThreadUnsafe {
  private static int commonCounter=0;

  public int getCount() {return commonCounter;}
  public void addCount(int val) { commonCounter+=val;}
  public void subtractCount(int val)
     { commonCounter-=val; }
}
```

Question How do you make the class defined in the sample code above thread-safe?

Store the ThreadUnsafe objects in a collection created using Collections.synchronizedCollection().

Modify getCount(), addCount(), and subtractCount() to synchronize on ThreadUnsafe.class

Have ThreadUnsafe implement the java.servlet.SingleThreadModel interface.

Wrap the ThreadUnsafe object within a class that declares all methods as synchronized.

Change the definition of commonCounter to protected static int.
Ans:

Modify getCount(), addCount(), and subtractCount() to synchronize on ThreadUnsafe.class

Sample Code

```
big_loop: for(int i = 0; i < 3; i++) {
 try {
   for(int j = 0; j < 3; j++) {
      if (i==j) continue;
      else if (i>j) continue big_loop;
      System.out.print("A ");
   }
 } finally {
   System.out.print("B ");
 }
 System.out.print("C ");
}
```

Question

What is the output from the sample code above?

A A A B C A A A B C A A A B C

A A A B C B C

A A B B C A C A

A A B C B B

A B C A B C A B C

Ans: **A A B C B B**

Question

If no Retention annotation is present on an annotation type declaration, what does the retention policy default to?

Runtime only

Class only

Source and class level

Source only

Source, class, and runtime level

Ans: **Class only**

Indicates how long annotations with the annotated type are to be retained. If no Retention annotation is present on an annotation type declaration, the retention policy defaults to RetentionPolicy.CLASS.

Sample Code

```
int Array1[]={3,6,2,9,5,8};
int Array2[]=Array1;
int Array3[]=Array2;

Array1[2]=2;
Array2[3]=5;
Array3[4]=7;
Array2[4]=Array3[4];
```

Question

After the sample code above is executed, what is the value of Array1[4]?

2

3

5

7

8

ANS: 7

Sample Code

```
import java.util.*;
public class Utils {
 public void printStrings (List stringList) {
   if (stringList != null) {
     Iterator li = stringList.iterator();
     while (li.hasNext()) {
        System.out.println( (String) li.next());
     }
   }
 }
}
```

Question

Which one of the following statements explains the assumed risk of a potential run-time error in the sample code above?

The object referenced by stringList may or may not implement the iterator() method.

System.out may have been redirected to a file rather than the terminal.

You do not know that stringList contains only String objects, so the cast could fail.

You need to check the size of the List using its size() method before accessing it so you do not read past the end of the list.

Since printStrings returns void, you have no way to return a failure code.

Ans: You do not know that stringList contains only String objects, so the cast could fail.

Sample Code

```
public class Classless {
  int last;

 // more code here
}
```

Question

Which access modifier, if any, do you add to the last variable in the sample code above so that subclasses in other packages can access it, but non-subclasses in other packages CANNOT?

static

private

<no keyword>

public

protected

ANS: protected

```
class A {
   int i=0;
   public A() { i=8; }
   public static void main(String args[]) {
        int i = 0;
        A h = new A();
        while (h.i <= 10) h.doIt();
   }
   public static void doIt() {
        i++;
        System.out.println("Hello");
   }
}
```

Question

What does the sample code above do?

It prints "Hello" two times.

It prints "Hello" three times.

It prints "Hello" 11 times.

It does not compile because variable i has been declared twice.

It does not compile because doIt() cannot reference non-static variable i.

ANS: It does not compile because doIt() cannot reference non-static variable i.

Question

How do you define the enumeration representing the suits in a standard playing card deck?

public enum Suit {
  CLUBS, DIAMONDS, HEARTS, SPADES

```
}
```

```
public class Suit {
  enum CLUBS;
  enum DIAMONDS;
  enum HEARTS;
  enum SPADES;
}
```

```
public class Suit extends Enum {
  CLUBS, DIAMONDS, HEARTS, SPADES
}
```

```
public enum Suit {
 int CLUBS = 0;
 int DIAMONDS = 1;
 int HEARTS = 2;
 int SPADES = 3;
}
```

```
public enumeration Suit {
  CLUBS, DIAMONDS, HEARTS, SPADES
}
```

ANS: **OPTION 1.**

```
public enum Suit {
  CLUBS, DIAMONDS, HEARTS, SPADES
}
```

Code

```
void shifter(int[] array, int arrayLength) {
  System.out.println("Go");
  synchronized (array) {
     for(int i=1;i<arrayLength;i++) {
        array[i-1] = array[i];
     }
  }
}
```

Question Referring to the above sample code, what does the synchronized code block do?

It forces the virtual machine to treat the array as a static object.

It performs the same as declaring shifter() as synchronized.

It forces the thread to obtain the monitor for the array before continuing.

It sets all methods of the object to synchronized while the code block executes.

It ensures that any threads created in the code block begin execution in the array object.

ANS: It forces the thread to obtain the monitor for the array before continuing.

Sample Code

```
SecurityManager sm = System.getSecurityManager();
if (sm != null) {
  sm.checkRead("filename.txt");
}
```

Question When an untrusted application executes the sample code above, what happens if the SecurityManager object is set up to deny this resource?

The checkRead() method throws a java.security.SecurityException

The checkRead() method returns false.

The SecurityManager terminates the thread attempting to read the file.

The SecurityManager stops the read method when it is invoked.

The SecurityManager causes the Virtual Machine to exit

ANS: The checkRead() method throws a java.security.SecurityException

Q:

Where is the data representing the internal state of a Java object held?

Choice 1 Method arguments

Choice 2 Member variables

Choice 3 Packages

Choice 4 Methods

Choice 5 Exceptions

ANS: Choice 2 Member variables

Q: Where does the JVM store objects?

Choice 1 In the heap

Choice 2 In the threads local memory space

Choice 3 In the methods local memory space

Choice 4 In the method stack

Choice 5 In the global stack

ANS: Choice 1 In the heap

Q: When you use the javadoc utility, how do you format comments?

Choice 1 // comment //

Choice 2 <!-- comment -->

Choice 3 /* comment */

Choice 4 /** comment */

Choice 5 @doc{{ comment }}

ANS: Choice 4 / comment */**

Q: When creating a JavaBean component, how do you make its properties available to users?

Choice 1 Implement support for ActionEvent listeners for the fields.

Choice 2 Declare the Bean's fields public.

Choice 3 Include public accessor methods of the form getVar() and setVar(val).

Choice 4 Provide a custom BeanInfo class that declares the properties public.

Choice 5 Declare an array of type java.lang.reflect.Field that contains the property names that should be available.

ANS: Include public accessor methods of the form getVar() and setVar(val).

Q: What is the purpose of using a thread pool?

Choice 1 To avoid recreating Runnable objects when they complete their task

Choice 2 To ensure newly created threads all share the same priority, group, and daemon status

Choice 3 To swap tasks among executing threads

Choice 4 To schedule tasks based upon a non-standard priority

Choice 5 To avoid recreating Thread objects when they complete their task

ANS: To ensure newly created threads all share the same priority, group, and daemon status

Sample Code

```
public class Test {
   public static void main(String[] args){
     StringBuffer[] messages = new StringBuffer[5];
     messages[0].append("Hello, World!");
     System.out.println("First message is " + messages[0]);
   }
}
```

Question What is the output when you run the above sample code?

First message is Hello, World!

An ArrayIndexOutOfBounds is thrown.

The code does not compile.

A NullPointerException is thrown.

First message is null.

ANS: A NullPointerException is thrown.

Question Which one of the following code fragments returns a string representation of j, where j is an int?

new String(j)

new Integer(j).toString()

str(j)

(String) j

((Object) j).toString()

ANS: new Integer(j).toString()

List

a. final
b. abstract
c. virtual
d. static

Question

Which of the keywords in the list above are often used with methods that can be overridden?

a

b

c

a, b, and d

a, b, c, and d

ANS: b

Question

How do you create and start a new thread of execution?

Create a new Runnable object with a Thread target and call the run method.

Extend the Thread class and call the run method.

Implement the Thread interface and call the start method.

Implement the Runnable interface and call the run method.

Create a new Thread object with a Runnable target and call the start method.

ANS: Create a new Thread object with a Runnable target and call the start method.

Question

What is the purpose of using a thread pool?

To avoid recreating Thread objects when they complete their task

To avoid recreating Runnable objects when they complete their task

To swap tasks among executing threads

To ensure newly created threads all share the same priority, group, and daemon status

To schedule tasks based upon a non-standard priority

ANS: To ensure newly created threads all share the same priority, group, and daemon status

Scenario

Class "Foo" resides in package "com.foo" and contains methods with the following declarations:

a. final String getName()
b. protected final String getDescription()
c. private String getCode()
d. synchronized String getFilename()
e. String getID()

Class "Bar" extends "Foo" and resides in package "com.foo.bar"

Question Referring to the scenario above, which one of the following methods of Foo is accessible to class Bar?

Method a

Method b

Method c

Method d

Method e

ANS: Method b

Sample Code

```
public class Outer {

  private static class Inner implements Runnable {
     private int a;
     Inner() { a=1;}

     // Additional code not shown

  }

  public static Runnable createInner() {
     return new Inner();
  }
}
```

Question Given the class Outer defined in the sample code above, which one of the following code fragments obtains a reference to a new Inner object from another class?

Outer.Inner.new();

new Outer.new Inner();

Outer.new Inner();

Outer.createInner();

new Outer.Inner();

ANS: new Outer.Inner();

Question How do you determine the size of an array of ints passed to a method?

Use the array's getLength() method.

Traverse the array until an exception is thrown.

Use the array's length field.

Call the array's getSize() method.

Use the sizeof() operator.

Ans: Use the array's length field.

Sample Code

```
public class TestOrder {
   static int deck[]=new int[25];

   static {
       for (int j=0;j<deck.length;j++) deck[j]=j;
       System.out.println("Finished creating the deck");
   }


   public static void main(String args[]) {
       System.out.println("Creating an Object");
       TestOrder to=new TestOrder();
       System.out.println("Finished creating");
   }
}
```

Question

What is the output when you invoke the class in the sample code above from the command line?

Creating an Object
Finished creating
Finished creating the deck

Finished creating the deck
Finished creating
Creating an Object

Finished creating
Creating an Object
Finished creating the deck

Creating an Object
Finished creating the deck
Finished creating

Finished creating the deck
Creating an Object
Finished creating

ANS: OPTION 5 (Static block runs first when class gets loaded)

Sample Code

```
class Classl {
 static void fix(String s) {
   String t = s;
   t = t.trim();
   t = t.replace(' ', '_');
   s = t;
 }

 public static void main(String args[]) {
   String x = "> This is a test <";
   fix(x);
   System.out.println(x);
 }
}
```

Question

What is the output of the sample code above?

>Thisisatest<

>This_is_a_test<

> This is a test <

>This is a test<

>_This_is_a_test_<

ANS: OPTION 3

Sample Code

```
DocumentBuilderFactory factory =
  DocumentBuilderFactory.newInstance();
DocumentBuilder builder = factory.newDocumentBuilder();
Document doc = builder.newDocument();
Node node1 = doc.createElement("node1");
Node node2 = doc.createElement("node2");
Node node3 = doc.createTextNode("Hello, World");
doc.appendChild(node2);
node1.appendChild(node3);
node3 = doc.createElement("node4");
node2.appendChild(node1);
node1 = doc.createTextNode("Good-bye");
node2.appendChild(node3);
node3.appendChild(node1);
```

Question

Which one of the following is the XML tree for the Document created with the sample code above?

Choice 1

<node1>
<node2>Hello, World</node2>
<node4>Good-bye</node4>
</node1>

Choice 2

<node1>
<node4>Good-bye</node4>
<node2>Hello, World</node2>
</node1>

Choice 3

<node2>
<node4>Good-bye</node4>
<node1>Hello, World</node1>
</node2>

Choice 4

<node2>
<node1>Hello, World</node1>
<node4>Good-bye</node4>
</node2>

Choice 5

<node1>
<node2>Hello, World</node2>
<node3>Good-bye</node3>
</node1>

Sample Code

```
public class SaveObject{
 public static void main( String[] args )
  throws java.io.IOException {
   Employee smith = new Employee
("012345","Smith","James","Payroll");
   FileOutputStream fos = new FileOutputStream("smith.dat");
   ObjectOutputStream oos = new ObjectOutputStream(fos);
   oos.writeObject(smith);
   fos.close();
 }
}
```

Question

The sample code above does NOT work properly if which one of the following is true?

If the file smith.dat already exists

If the class Employee does not implement Serializable

If the class Employee implements Externalizable

If the file smith.dat does not exist

If the class Employee is declared as transient

Ans: If the class Employee does not implement Serializable

Question

Which one of the following classes do you use to store a set of key-value pairs that are sorted by the key and are NOT inserted randomly?

Hashtable

HashMap

TreeSet

LinkedList

TreeMap

ANS: TreeMap

Question

Renaming classes and variables to protect your Java code from being reverse engineered and easily read by someone else is called which one of the following?

Commenting

Serializing

Obfuscating

Encrypting

Decompiling

ANS: Obfuscating

Question Which one of the following code segments listens for a socket connection?

Socket socket = (new ServerSocket(8080)).accept();

Socket socket = new Socket();

Socket socket = SocketImpl.listen(8080);

(new MulticastSocket(8080)).joinGroup();

Socket socket = HttpURLConnection.open(8080);

ANS: Socket socket = (new ServerSocket(8080)).accept();

Question What is a difference between the String and StringBuffer classes?

All String objects must be initialized with constant values at compile time; StringBuffers can be modified.

The StringBuffer class provides methods to retrieve a substring; String does not.

A StringBuffer's contents can be modified; the contents of a String are immutable.

StringBuffer supports Unicode characters; String only supports ASCII.

String cannot be passed to the print() method of PrintWriter interface; StringBuffer can be passed.

ANS: A StringBuffer's contents can be modified; the contents of a String are immutable.

Sample Code
```
List myList = new ArrayList();

// your code here
```

Question Referring to the sample code above, which one of the following do you use to make the List thread safe?

List syncList = List.synchronizedList(myList);

List syncList = Collections.synchronizedList(myList);

List syncList = myList.getSynchronizedList();

myList.setSynchronized(true);

List syncList = new SynchronizedList(myList);

ANS: List syncList = Collections.synchronizedList(myList);

Sample Code
```
Integer i = 3;
i = i + 1;
Integer j = i;
j = i + j;
```

Question How many objects are created as a result of the statements in the sample code above?

0

1

3

4

5

ANS: 3

Question What does method overriding mean in Java?

Method access may be restricted via special keywords.

Multiple methods with the same name but differing argument types may exist on the same object.

Methods with the exact same signature may exist in an object hierarchy.

Reflection may be used to redefine which method is called.

Methods with the exact same signature on the same object may exist

ANS: Methods with the exact same signature may exist in an object hierarchy.

Sample Code

```
// The following code defines the ThreadSample class.

public class ThreadSample implements Runnable {
  public static void main(String args[]) {
     System.out.println("This is the ThreadSample object");
  }

  public void run() {
     while (true) {
        try {
          Thread.sleep(100);
          System.out.println("Thread ran.");
        } catch (Exception e) {}
     }
  }
}
```

Question Referring to the sample code above, how do you start an instance of ThreadSample in a new thread?

new Thread(new ThreadSample()).start();

new ThreadSample().main({""});

Thread.new(ThreadSample);

Thread.start(new ThreadSample());

new ThreadSample().run();

ANS: new Thread(new ThreadSample()).start();

Question When you parse an XML file like a tree, which one of the following types of parsers do you use?

Interface Definition Language (IDL)

Document Object Model (DOM)

Simple API for XML (SAX)

Streaming API for XML (StAX)

Schema

ANS: Document Object Model (DOM)

Sample Code

```
for (int i=0;i<=22;) {
   if (i<= 10) {
      int j= 2 + i;
      i++;
      // /* let us know */ //
      System.out.println("i: " + i + " j: " + j);
   }
}
```

Question What is the problem with the sample code above?

The comment line is not formatted correctly.

You cannot print integer values without converting them to strings.

The loop never terminates.

Variable j is referenced outside its scope.

You cannot declare variables inside a for-loop.

ANS: This code runs perfectly fine in Eclipse – there is no problem

Question Where does the Java virtual machine keep space for local method variables?

The shared memory queue

On the stack

With hotspot

In the heap

In the reference pool

ANS: On the stack

Question Which one of the following statements creates and initializes an array that holds integer primitives?

int[] a = Integer.toArray(1,2,3,4,5);

int[5] a = new int[] {1,2,3,4,5};

int[] a = new int(1,2,3,4,5);

int[] a = int[] {1,2,3,4,5};

int[] a = {1,2,3,4,5};

ANS: int[] a = {1,2,3,4,5};

Sample Code

```
int count=0, i=0;
do {
  count += i;
  i++;
  if (count > 5) break;
} while (i<=4);
System.out.println("count: " + count);
```

Question

After execution, what is the output of the above sample code?

0

4

5

6

10

ANS: 6

Sample Code

```
// Initialize the winterMonths array to contain
//   "December","January","February"

     // Line A
```

Question

Which one of the following substitutions for Line A initializes winterMonths as shown in the sample code above?

String winterMonths[]="December"+"January"+"February";

String winterMonths[]={Calendar.December..Calendar.February};

String winterMonths[]={"December","January","February"};

String winterMonths[]=new String(){"December","January","February"};

String winterMonths={"December";"January";"February";};

ANS: String winterMonths[]={"December","January","February"};

Question

Why do you provide accessor methods in a class rather than making the class's fields publicly accessible?

They can be made accessible to classes in other packages. Public fields are only usable from the same package.

They improve performance by avoiding a context switch in the bytecode interpreter. Accessing a public field requires loading the object's data pointer.

They can prevent memory overwrites by performing bounds checks on arrays. Public fields are not checked for bounds overrun.

They are automatically available to other tasks using the RMI subsystem. Public fields cannot be remotely accessed.

They can ensure data integrity by validating the proposed value. Any value could be written to a public field.

ANS: They can ensure data integrity by validating the proposed value. Any value could be written to a public field.

Facts

- Scrollable is an interface.
- Writable is an interface.
- TextScreen is an abstract class.
- VideoScreen is a class that extends TextScreen and implements both Scrollable and Writable.

Question

In reference to the above facts, which one of the following statements is always true?

A VideoScreen object cannot be passed to a method that expects a Scrollable.

A Scrollable object can be cast to produce a VideoScreen.

A VideoScreen object cannot be passed to a method that expects a Writable.

A VideoScreen object cannot be passed to a method that expects a TextScreen.

A VideoScreen can be cast as a Scrollable.

ANS: A Scrollable object can be cast to produce a VideoScreen.

Sample Code

```
int f = 2;
int g = 5;
int h;

h = 3+f/g+2;
```

Question

What is the value of h after you execute the sample code above?

2

5

5.2

5.4

6

ANS: 5

Sample Code

```
public class TestEx {
   static class Ex1 extends Exception {}
   static class Ex2 extends Ex1 {}
   static class Ex3 extends Exception {}

   static void method1() throws Ex1,Ex2,Ex3 {
       throw  new Ex2(); }

   public static void main(String args[]) {
       try {
           method1();
       } catch (Ex3 e) {
          System.out.print("C");
       } catch (Ex2 e) {
          System.out.print("B");
       } catch (Ex1 e) {
          System.out.print("A");
       } catch (Exception e) {
          System.out.print("D");
       } finally {
          System.out.println("F");
       }
   }
}
```

Question What is the output of the sample code above?

AF

ABCDF

BF

CF

DF

ANS: BF

Sample Code

```
File f = ...
DocumentBuilderFactory factory =
  DocumentBuilderFactory.newInstance();
DocumentBuilder builder =
  factory.newDocumentBuilder();
document = builder.parse(f);

// Get the first <brainbench> element in the DOM
// ???

// Use a Transformer for output
TransformerFactory tFactory =
  TransformerFactory.newInstance();
Transformer transformer = tFactory.newTransformer();
DOMSource source = new DOMSource(node);
StreamResult result = new StreamResult(System.out);
transformer.transform(source, result);
```

Question

Which one of the following do you add to the sample code above in order to get the first brainbench node from the DOM tree before dumping its subtree?

```
Node node = document.getElementByTagName("brainbench");
```

```
NodeList list =
  document.getElementsByTagName("brainbench");
Node node = list.item(0);
```

```
NodeList list =
  document.getElementsByTagName("brainbench");
Node node = list.item(1);
```

```
ElementList list =
  document.getElementsByTagName("brainbench");
Node node = list.item(0);
```

```
ElementList list =
  document.getElementsByTagName("brainbench");
Node node = list.item(1);
```

ANS:

```
NodeList list =
  document.getElementsByTagName("brainbench");
Node node = list.item(0);
```

Sample Code

```
class Animal{ void speak(){ System.out.println("speak"); } }
class Dog extends Animal{ void speak(){ System.out.println
("woof!"); } }
class Cat extends Animal{ void speak(){ System.out.println
("meow!"); } }
public class AnimalTest{
 public static void main( String[] args ){
   Animal[] animals = new Animal[3];
   animals[0] = new Animal();
   animals[1] = new Cat();
   animals[2] = new Dog();
   for( int i=0; i < animals.length; i++ )
     animals[i].speak();
 }
}
```

Question

Given the sample code above, what is printed when AnimalTest is run?

speak
speak
speak

speak
meow!
woof!

speak
speak
meow!
speak
woof!

The code does not compile because you cannot assign a Cat or a Dog to a variable of type Animal.

The code does not compile because the Cat and Dog classes cannot redefine the speak() method.

ANS:

speak
meow!
woof!