QUESTION: 1

Given the code fragment:

```
public class ForTest {
public static void main(String[] args) { int[] array = {1, 2, 3};
for ( foo ) {
}
}
```

Which three code fragments, when replaced individually for foo, enables the program to compile?

A. int i : array
B. int i = 0; i < 1;
C. ; ;
D. ; i < 1; i++
E. i = 0; i<1;

Answer: A, B, C

QUESTION: 2

Given

```
public class ComputeSum { public int x;
public int y; public int sum;
public ComputeSum (int nx, int ny) { x = nx; y =ny;
updateSum();
}
public void setX(int nx) { x = nx; updateSum();} public void setY(int ny) { x = ny;
updateSum();} void updateSum() { sum = x + y;}
}
```

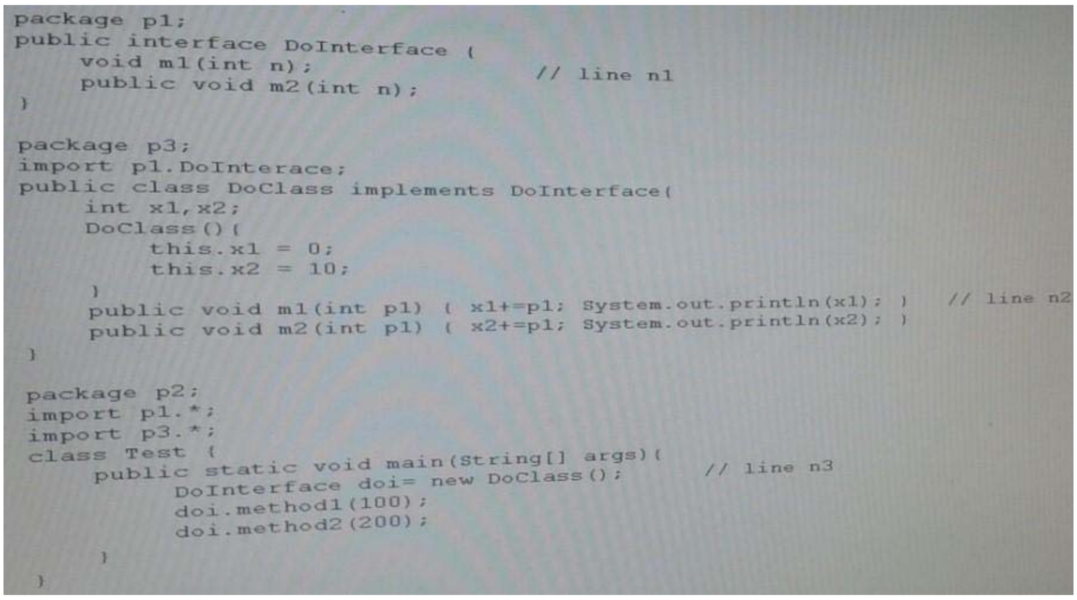
This class needs to protect an invariant on the sum field.
Which three members must have the private access modifier to ensure that this invariant is maintained?

A. The x field
B. The y field
C. The sum field
D. The ComputerSum ( ) constructor
E. The setX ( ) method
F. The setY ( ) method

Answer: C, E, F

Explanation:
The sum field and the two methods (setX and SetY) that updates the sum field.

QUESTION: 3
Given:



```
package p1;
public interface DoInterface {
    void m1(int n);                              // line n1
    public void m2(int n);
}

package p3;
import p1.DoInterace;
public class DoClass implements DoInterface{
    int x1,x2;
    DoClass(){
        this.x1 = 0;
        this.x2 = 10;
    }
    public void m1(int p1) { x1+=p1; System.out.println(x1); }    // line n2
    public void m2(int p1) { x2+=p1; System.out.println(x2); }
}

package p2;
import p1.*;
import p3.*;
class Test {
    public static void main(String[] args){
        DoInterface doi= new DoClass();          // line n3
        doi.method1(100);
        doi.method2(200);
    }
}
```

What is the result?

A. 100210
B. Compilation fails due to an error in line n1
C. Compilation fails due to an error at line n2
D. Compilation fails due to an error at line n3

Answer: C

QUESTION: 4
Which two statements are true?

A. An abstract class can implement an interface.
B. An abstract class can be extended by an interface.
C. An interface CANNOT be extended by another interface.
D. An interface can be extended by an abstract class.
E. An abstract class can be extended by a concrete class.
F. An abstract class CANNOT be extended by an abstract class.

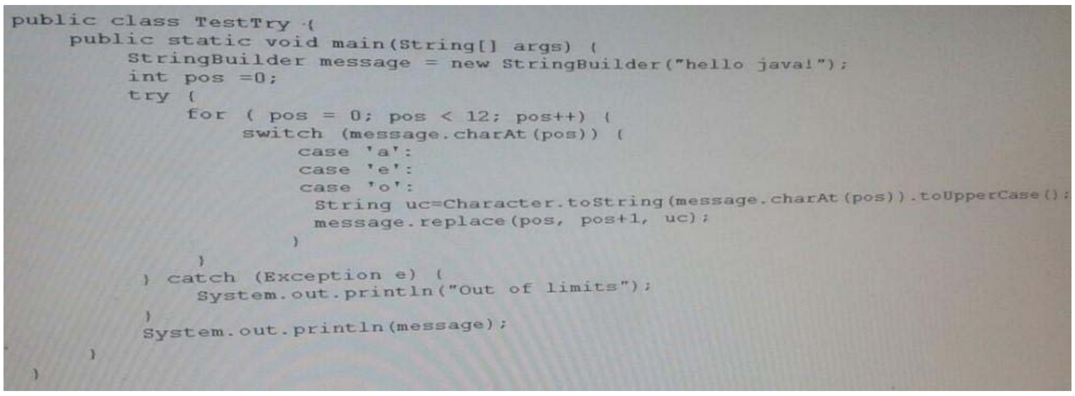
Answer: A, E

Explanation:
http://docs.oracle.com/javase/tutorial/java/IandI/abstract.html

QUESTION: 5
Given:
abstract class A1 {
public abstract void m1();
public void m2() { System.out.println("Green"); }
}
abstract class A2 extends A1 { public abstract void m3();
public void m1() { System.out.println("Cyan"); } public void m2()
{ System.out.println("Blue"); }
}
public class A3 extends A2 {
public void m1() { System.out.println("Yellow"); } public void m2()
{ System.out.println("Pink"); } public void m3() { System.out.println("Red"); }
public static void main(String[] args) {
A2 tp = new A3(); tp.m1();
tp.m2();
tp.m3();
}
}
What is the result?

A. Yellow Pink Red
B. Cyan Blue Red C.
Cyan Green Red D.
Compilation Fails

Answer: A

QUESTION: 6
Given:

```
public class TestTry {
    public static void main(String[] args) {
        StringBuilder message = new StringBuilder("hello java!");
        int pos =0;
        try {
            for ( pos = 0; pos < 12; pos++) {
                switch (message.charAt(pos)) {
                    case 'a':
                    case 'e':
                    case 'o':
                     String uc=Character.toString(message.charAt(pos)).toUpperCase();
                     message.replace(pos, pos+1, uc);
                    }
            }
        } catch (Exception e) {
           System.out.println("Out of limits");
        }
        System.out.println(message);
    }
}
```

What is the result?

A. hEllOjAvA!
B. Hello java!
C. Out of limits hEllOjAvA!
D. Out of limits

Answer: C

QUESTION: 7
Given the code fragment:
int [][] array2d = new int[2][3]; System.out.println("Loading the data."); for (int x = 0; x < array2d.length; x++) { for (int y = 0; y < array2d[0].length; y++) { System.out.println(" x = " + x); System.out.println(" y = " + y);
// insert load statement here.
}
}
System.out.println("Modify the data. "); for (int x = 0; x < array2d.length; x++) { for (int y = 0; y < array2d[0].length; y++) { System.out.println(" x = " + x); System.out.println(" y = " + y);
// insert modify statement here.
}
}
Which pair of load and modify statement should be inserted in the code? The load statement should set the array's x row and y column value to the sum of x and y The modify statement should modify the array's x row and y column value by multiplying it by 2

A. Load statement: array2d(x, y) = x + y;
Modify statement: array2d(x, y) = array2d(x, y) * 2
B. Load statement: array2d[x y] = x + y;
Modify statement: array2d[x y] = array2d[x y] * 2
C. Load statement: array2d[x, y] = x + y;
Modify statement: array2d[x, y] = array2d[x, y] * 2
D. Load statement: array2d[x][y] = x + y;
Modify statement: array2d[x][y] = array2d[x][y] * 2
E. Load statement: array2d[[x][y]] = x + y;
Modify statement: array2d[[x][y]] = array2d[[x][y]] * 2

Answer: D

QUESTION: 8
public class ForTest {
public static void main(String[] args) { int[] arrar = {1,2,3};
for (foo) {
}
}

}
Which three are valid replacements for foo so that the program will compiled and run?

A. int i: array
B. int i = 0; i < 1; i++
C. ;;
D. ; i < 1; i++
E. ; i < 1;

Answer: A, B, C

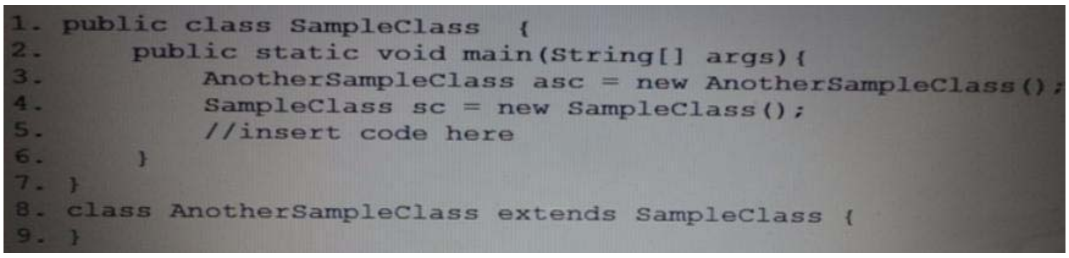
QUESTION: 9
You are writing a method that is declared not to return a value. Which two are permitted in the method body?

A. omission of the return statement
B. return null;
C. return void;
D. return;

Answer: A, D

Explanation:
Any method declared void doesn't return a value. It does not need to contain a return statement, but it may do so. In such a case, a return statement can be used to branch out of a control flow block and exit the method and is simply used like this:return;

QUESTION: 10
Given:



```
1. public class SampleClass {
2.     public static void main(String[] args){
3.         AnotherSampleClass asc = new AnotherSampleClass();
4.         SampleClass sc = new SampleClass();
5.         //insert code here
6.     }
7. }
8. class AnotherSampleClass extends SampleClass {
9. }
```

Which statement, when inserted into line 5, is valid change?

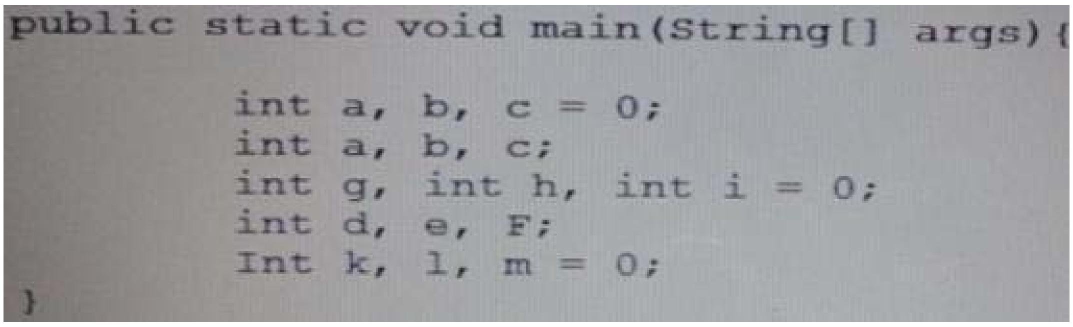
A. asc = sc;
B. sc = asc;

C. asc = (object) sc;
D. asc = sc.clone ()

Answer: B

Explanation:
Works fine.

QUESTION: 11
Given:



```
public static void main(String[] args){
        int a, b, c = 0;
        int a, b, c;
        int g, int h, int i = 0;
        int d, e, F;
        Int k, l, m = 0;
}
```

Which two declarations will compile?

A. int a, b, c = 0;
B. int a, b, c;
C. int g, int h, int i = 0;
D. int d, e, F;
E. int k, l, m; = 0;

Answer: A, D

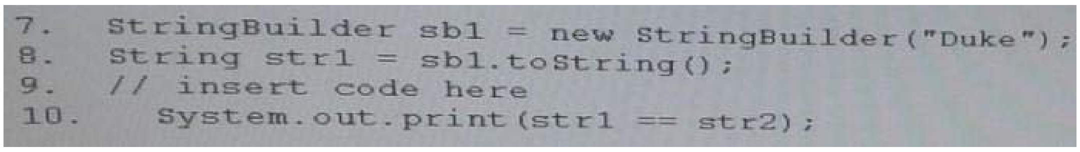
QUESTION: 12
Given the code fragment:
public class Test {
static String[][] arr =new String[3][]; private static void doPrint() {
//insert code here
}
public static void main(String[] args) { String[] class1 = {"A","B","C"};
String[] class2 = {"L","M","N","O"}; String[] class3 = {"I","J"};
arr[0] = class1; arr[1] = class2; arr[2] = class3; Test.doPrint();
}
}
Which code fragment, when inserted at line //insert code here, enables the code to print COJ?

A. int i = 0;
for (String[] sub: arr) { int j = sub.length -1; for (String str: sub)
{ System.out.println(str[j]); i++;
}
}
B. private static void doPrint() { for (int i = 0;i < arr.length;i++) { int j = arr[i].length-1;
System.out.print(arr[i][j]);
}
}
C. int i = 0;
for (String[] sub: arr[][]) { int j = sub.length; System.out.print(arr[i][j]); i++;
}
D. for (int i = 0;i < arr.length-1;i++) { int j = arr[i].length-1; System.out.print(arr[i][j]);
i++;
}

Answer: B

Explanation:
Incorrect:
not A: The following line causes a compile error:
System.out.println(str[j]); Not C: Compile erro line: for (String[] sub: arr[][]) not D: Output: C

QUESTION: 13
Given:



```
7.   StringBuilder sb1 = new StringBuilder("Duke");
8.   String str1 = sb1.toString();
9.   // insert code here
10.     System.out.print(str1 == str2);
```

Which code fragment, when inserted at line 9, enables the code to print true?

A. String str2 = str1;
B. String str2 = new string (str1);
C. String str2 = sb1.toString();
D. String str2 = “Duke”;

Answer: B

QUESTION: 14
Given:

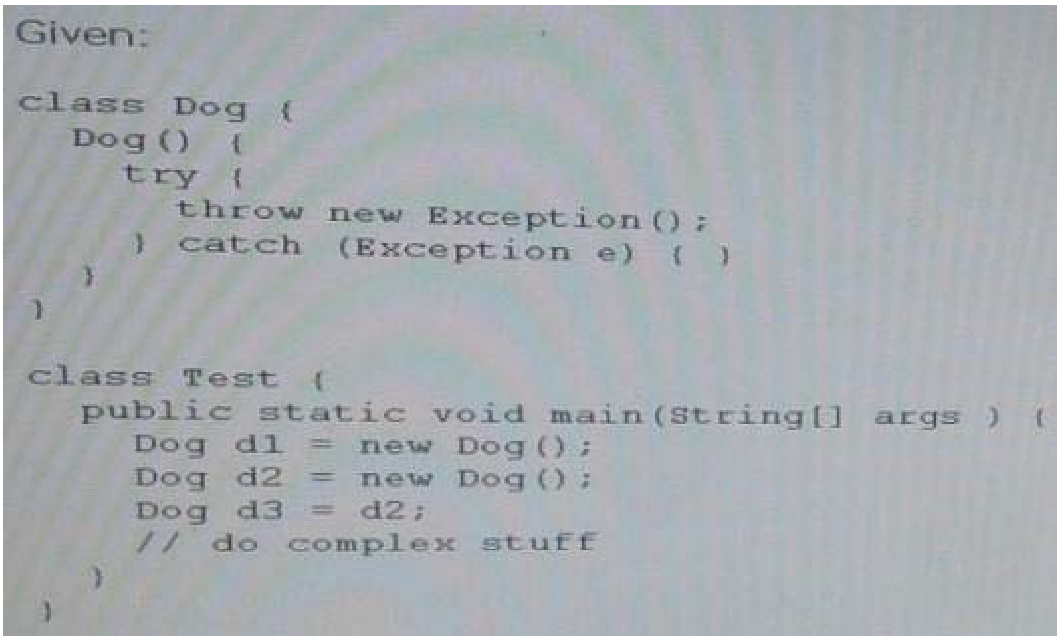
import java.util.*; public class Ref {
public static void main(String[] args) {
StringBuilder s1 = new StringBuilder("Hello Java!"); String s2 = s1.toString();
List<String> lst = new ArrayList<String>();
lst.add(s2); System.out.println(s1.getClass()); System.out.println(s2.getClass());
System.out.println(lst.getClass());
}
}
What is the result?

A. class java.lang.String class java.lang.String class java.util.ArrayList
B. class java.lang.Object class java.lang. Object class java.util.Collection
C. class java.lang.StringBuilder class java.lang.String
class java.util.ArrayList
D. class java.lang.StringBuilder class java.lang.String
class java.util.List

Answer: C

Explanation:
class java.lang.StringBuilder class java.lang.String
class java.util.ArrayList

QUESTION: 15
Given:



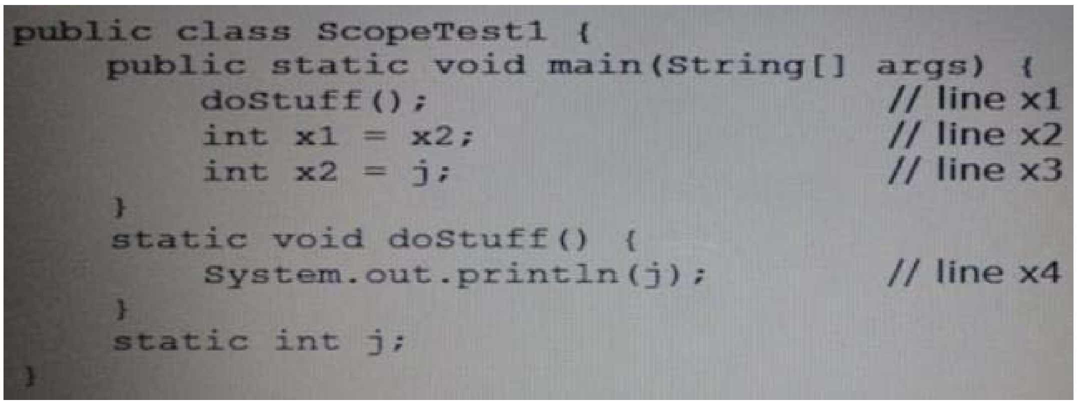
```
Given:

class Dog {
   Dog() {
      try {
         throw new Exception();
      } catch (Exception e) { }
   }
}

class Test {
   public static void main(String[] args ) {
      Dog d1 = new Dog();
      Dog d2 = new Dog();
      Dog d3 = d2;
      // do complex stuff
   }
}
```

How many objects have been created when the line / / do complex stuff is reached?

A. Two
B. Three
C. Four
D. Six

Answer: C

QUESTION: 16
Given:



```
public class ScopeTest1 {
    public static void main(String[] args) {
        doStuff();                          // line x1
        int x1 = x2;                        // line x2
        int x2 = j;                         // line x3
    }
    static void doStuff() {
        System.out.println(j);              // line x4
    }
    static int j;
}
```

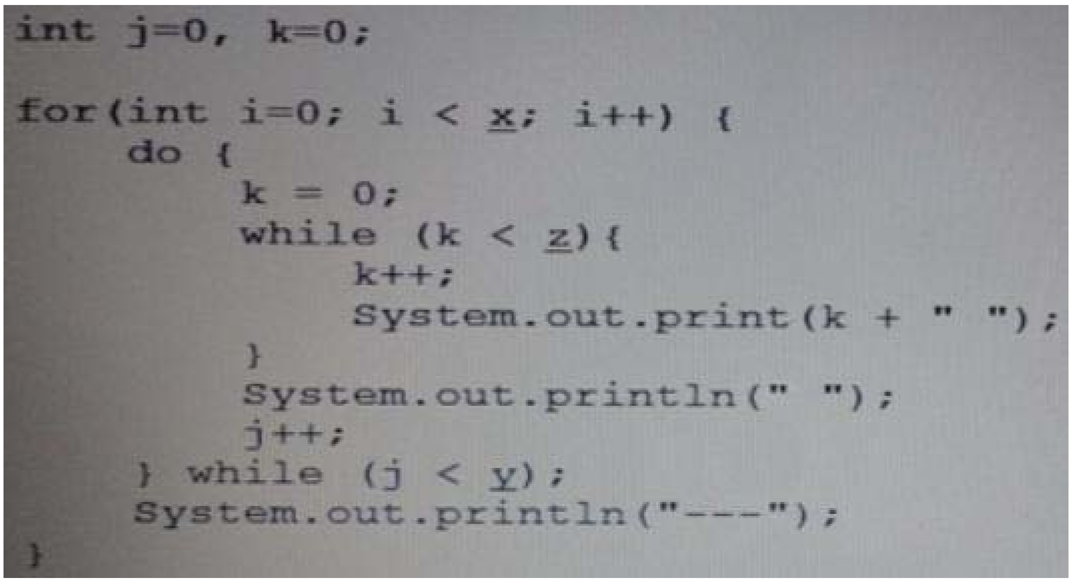
Which line causes a compilation error?

A. line x1
B. line x2
C. line x3
D. line x4

Answer: B

Explanation:
The variable x2 is used before it has been declared.

QUESTION: 17
Given the code fragment:



```
int j=0, k=0;

for(int i=0; i < x; i++) {
        do {
                k = 0;
                while (k < z){
                        k++;
                        System.out.print(k + " ");
                }
                System.out.println(" ");
                j++;
        } while (j < y);
        System.out.println("---");
}
```

What values of x, y, z will produce the following result? 1 2 3 4

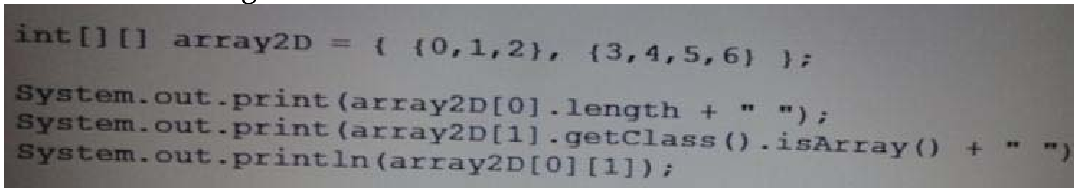
```
1 2 3 4
1 2 3 4
----
1 2 3 4
----
```

A. X = 4, Y = 3, Z = 2
B. X = 3, Y = 2, Z = 3
C. X = 2, Y = 3, Z = 3
D. X = 4, Y = 2, Z = 3
E. X = 2, Y = 3, Z = 4

Answer: E

Explanation:
Z is for the innermost loop. Should print 1 2 3 4. So Z must be 4.
Y is for the middle loop. Should print three lines of 1 2 3 4. So Y must be set 3. X is for the outmost loop. Should print 2 lines of. So X should be 2.

QUESTION: 18
Given the code fragment:



```
int[][] array2D = { {0,1,2}, {3,4,5,6} };

System.out.print(array2D[0].length + " ");
System.out.print(array2D[1].getClass().isArray() + " ")
System.out.println(array2D[0][1]);
```

What is the result?

A. 3 false 1
B. 2 true 3
C. 2 false 3
D. 3 true 1
E. 3 false 3
F. 2 true 1
G. 2 false 1

Answer: D

Explanation:
The length of the element with index 0, {0, 1, 2}, is 3. Output: 3
The element with index 1, {3, 4, 5, 6}, is of type array. Output: true
The element with index 0, {0, 1, 2} has the element with index 1: 1. Output: 1

QUESTION: 19
Which usage represents a valid way of compiling java source file with the name "Main"?

A. javac Main.java
B. java Main.class
C. java Main.java
D. javac Main
E. java Main

Answer: A

Explanation:
The compiler is invoked by the javac command. When compiling a Java class, you must include the file name, which houses the main classes including the Java extension. So to run Main.java file we have to use command in option A. TO execute Java program we can use Java command but can't use it for compiling.
https://docs.oracle.com/javase/tutorial/getStarted/application/index.html

QUESTION: 20
Given the classes:
* AssertionError
* ArithmeticException
* ArrayIndexOutofBoundsException
* FileNotFoundException
* IllegalArgumentException
* IOError
* IOException
* NumberFormatException
* SQLException

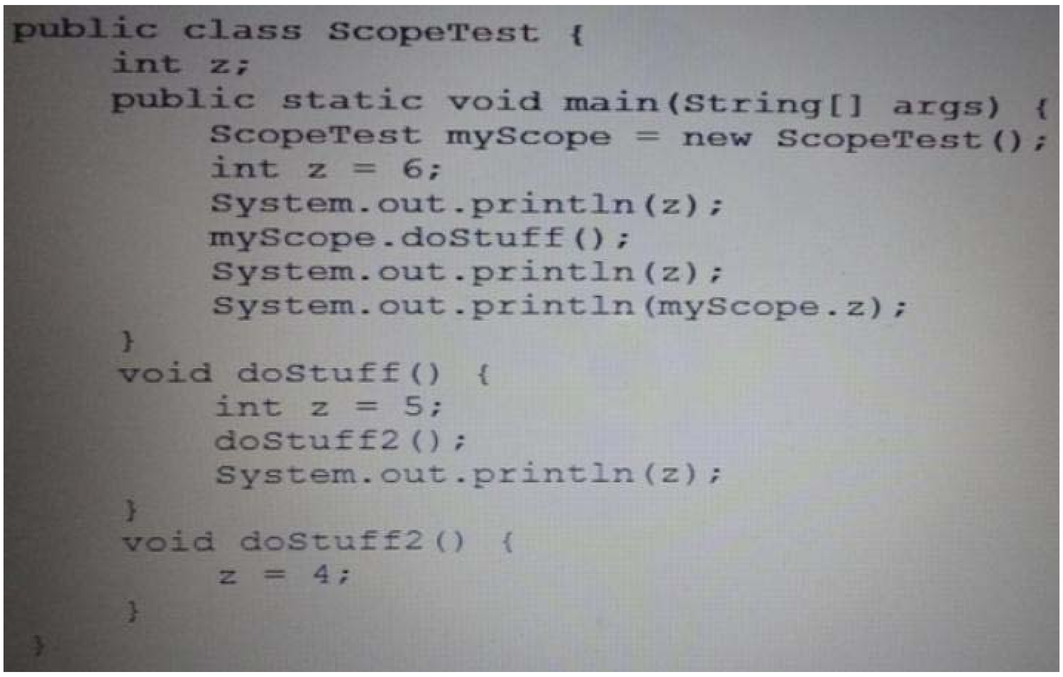
Which option lists only those classes that belong to the unchecked exception category?

A. AssertionError, ArrayIndexOutOfBoundsException, ArithmeticException
B. AssertionError, IOError, IOException
C. ArithmeticException, FileNotFoundException, NumberFormatException
D. FileNotFoundException, IOException, SQLException
E. ArrayIndexOutOfBoundException, IllegalArgumentException, FileNotFoundException

Answer: A

Explanation:
Not B: IOError and IOException are both checked errors. Not C, not D, not E: FileNotFoundException is a checked error.
Note:
Checked exceptions:
* represent invalid conditions in areas outside the immediate control of the program (invalid user input, database problems, network outages, absent files)
* are subclasses of Exception
* a method is obliged to establish a policy for all checked exceptions thrown by its implementation (either pass the checked exception further up the stack, or handle it somehow)
Note:
Unchecked exceptions:
* represent defects in the program (bugs) - often invalid arguments passed to a non-private method. To quote from The Java Programming Language, by Gosling, Arnold, and Holmes: "Unchecked runtime exceptions represent conditions that, generally speaking, reflect errors in your program's logic and cannot be reasonably recovered from at run time."
* are subclasses of RuntimeException, and are usually implemented using IllegalArgumentException, NullPointerException, or IllegalStateException
* method is not obliged to establish a policy for the unchecked exceptions thrown by its implementation (and they almost always do not do so)

QUESTION: 21
Given:



```
public class ScopeTest {
    int z;
    public static void main(String[] args) {
        ScopeTest myScope = new ScopeTest();
        int z = 6;
        System.out.println(z);
        myScope.doStuff();
        System.out.println(z);
        System.out.println(myScope.z);
    }
    void doStuff() {
        int z = 5;
        doStuff2();
        System.out.println(z);
    }
    void doStuff2() {
        z = 4;
    }
}
```

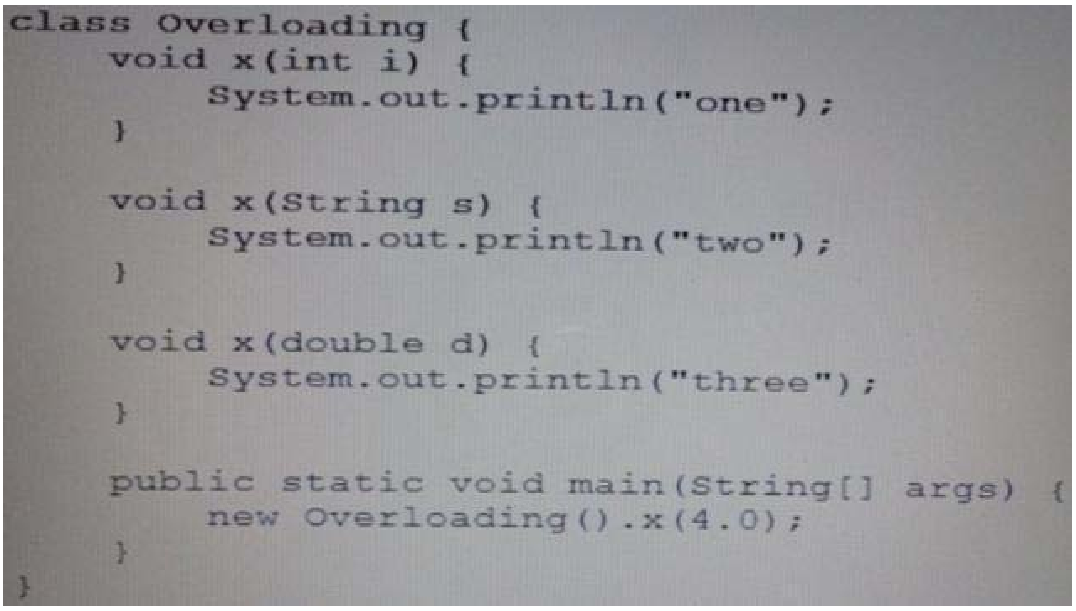
What is the result?

A. 6 564
B. 6 554
C. 6 566
D. 6 565

Answer: A

Explanation:
Within main z is assigned 6. z is printed. Output: 6
Within doStuff z is assigned 5.DoStuff2 locally sets z to 4 (but MyScope.z is set to 4), but in Dostuff z is still 5. z is printed. Output: 5
Again z is printed within main (with local z set to 6). Output: 6
Finally MyScope.z is printed. MyScope.z has been set to 4 within doStuff2(). Output: 4

QUESTION: 22
Given:



```
class Overloading {
    void x(int i) {
        System.out.println("one");
    }

    void x(String s) {
        System.out.println("two");
    }

    void x(double d) {
        System.out.println("three");
    }

    public static void main(String[] args) {
        new Overloading().x(4.0);
    }
}
```

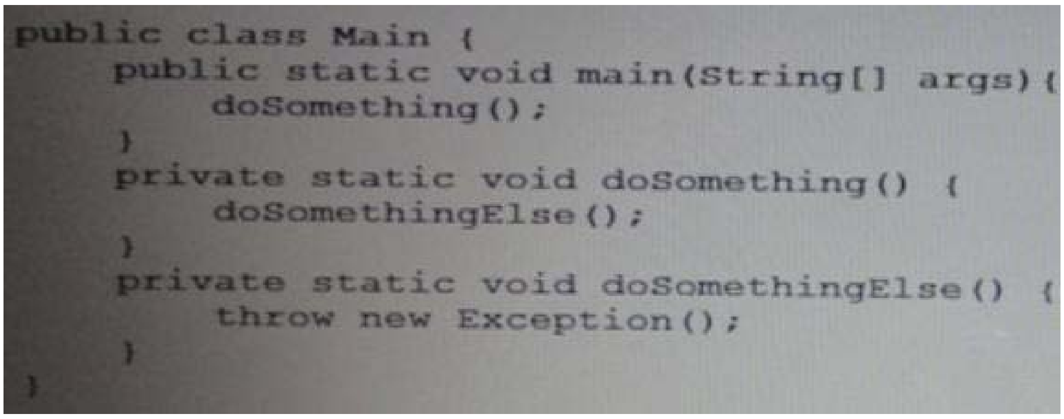
What is the result?

A. One B.
Two C.
Three
D. Compilation fails

Answer: C

Explanation:
In this scenario the overloading method is called with a double/float value, 4.0. This makes the third overload method to run.
Note:
The Java programming language supports overloading methods, and Java can distinguish between methods with different method signatures. This means that methods within a class can have the same name if they have different parameter lists. Overloaded methods are differentiated by the number and the type of the arguments passed into the method.

QUESTION: 23
Given:



```
public class Main {
    public static void main(String[] args){
        doSomething();
    }
    private static void doSomething() {
        doSomethingElse();
    }
    private static void doSomethingElse() {
        throw new Exception();
    }
}
```

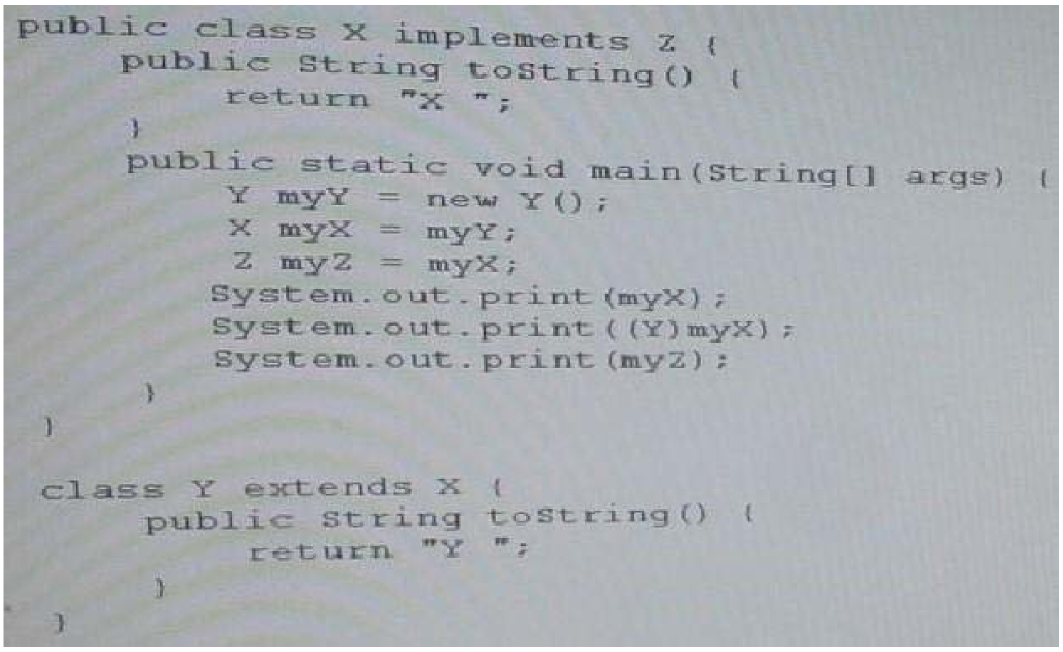
Which approach ensures that the class can be compiled and run?

A. Put the throw new Exception() statement in the try block of try – catch
B. Put the doSomethingElse() method in the try block of a try – catch
C. Put the doSomething() method in the try block of a try – catch
D. Put the doSomething() method and the doSomethingElse() method in the try block of a try – catch

Answer: A

Explanation:
We need to catch the exception in the doSomethingElse() method. Such as:
private static void doSomeThingElse() { try {
throw new Exception();} catch (Exception e)
{}
}
Note: One alternative, but not an option here, is the declare the exception in doSomeThingElse and catch it in the doSomeThing method.

QUESTION: 24
Given:

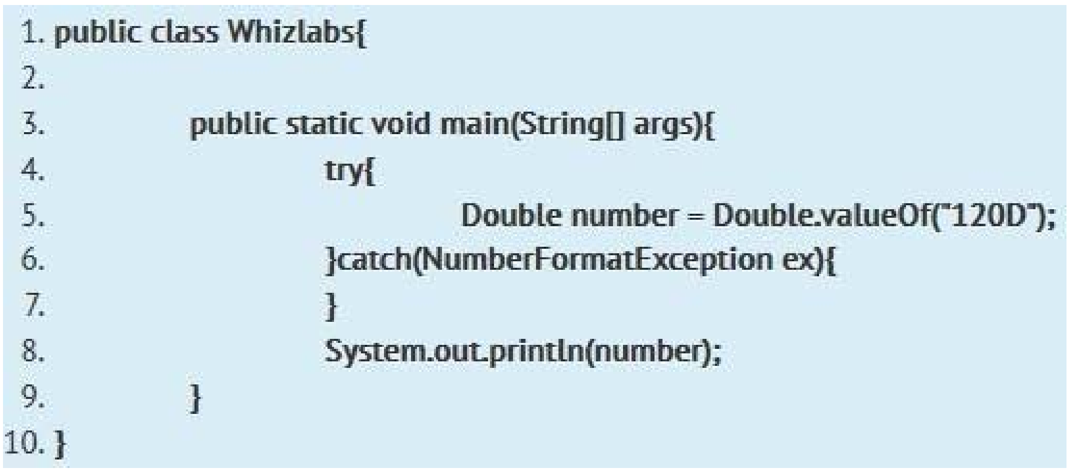


```
public class X implements Z {
    public String toString() {
        return "X ";
    }
    public static void main(String[] args) {
        Y myY = new Y();
        X myX = myY;
        Z myZ = myX;
        System.out.print(myX);
        System.out.print((Y)myX);
        System.out.print(myZ);
    }
}

class Y extends X {
    public String toString() {
        return "Y ";
    }
}
```

A. X XX
B. X Y X
C. Y Y X
D. Y YY

Answer: D

QUESTION: 25
Given:



```
1. public class Whizlabs{
2.
3.         public static void main(String[] args){
4.                 try{
5.                         Double number = Double.valueOf("120D");
6.                 }catch(NumberFormatException ex){
7.                 }
8.                 System.out.println(number);
9.         }
10. }
```

What is the result?

A. 120

B. 120D
C. A NumberFormatException will be thrown.
D. Compilation fails due to error at line 5.
E. Compilation tails due to error at line 8.

Answer: E

Explanation:
At line 5, we have created a wrapper object of double by passing 120D, which is convertible to a Double, so there won't be any exception there. But if you check carefully, you can see the variable number is declared inside try block, so the scope of the variable number is limited to that block, so trying to access it outside causes a compile time error. httpsy/docs.oracle.com/javase/tutorial/iava/nutsandbolts/variables.html

QUESTION: 26
Given the code fragment:

```
public class Test {
public static void main(String[] args) { boolean isChecked = false;
int arry[] = {1,3,5,7,8,9};
int index = arry.length; while ( <code1> ) {
if (arry[index-1] % 2 ==0) { isChecked = true;
}
<code2>
}
System.out.print(arry(index]+", "+isChecked));
}
}
```

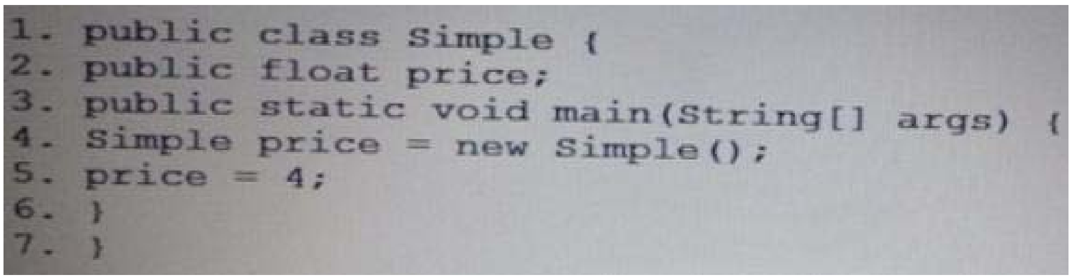
Which set of changes enable the code to print 1, true?

A. Replacing <code1> with index > 0 and replacing <code2> with index--;
B. Replacing <code1> with index > 0 and replacing <code2> with --index;
C. Replacing <code1> with index > 5 and replacing <code2> with --index ;
D. Replacing <code1> with index and replacing <code2> with --index ;

Answer: A

Explanation:
Note: Code in B (code2 is --index;). also works fine.

QUESTION: 27
Given the following code:



```
1. public class Simple {
2. public float price;
3. public static void main(String[] args) {
4. Simple price = new Simple();
5. price = 4;
6. }
7. }
```

What will make this code compile and run?

A. Change line 2 to the following: Public int price
B. Change line 4 to the following: int price = new simple ();
C. Change line 4 to the following: Float price = new simple ();
D. Change line 5 to the following: Price = 4f;
E. Change line 5 to the following: price.price = 4; F.
Change line 5 to the following: Price = (float) 4:
G. Change line 5 to the following: Price = (Simple) 4;
H. The code compiles and runs properly; no changes are necessary

Answer: E

Explanation:
price.price =4; is correct, not price=4;
The attribute price of the instance must be set, not the instance itself.

QUESTION: 28
Given the code fragment:
Int [] [] array = {{0}, {0, 1}, {0, 2, 4}, {0, 3, 6, 9}, {0, 4, 8, 12, 16}};
Systemout.printIn(array [4] [1]);
System.out.printIn (array) [1] [4]); What is the result?

A. 4 Null
B. Null 4
C. An IllegalArgumentException is thrown at run time
D. 4An ArrayIndexOutOfBoundException is thrown at run time

Answer: D

Explanation:
The first println statement, System.out.println(array [4][1]);, works fine. It selects the element/array with index 4, {0, 4, 8, 12, 16}, and from this array it selects the element with index 1, 4. Output: 4 The second println statement, System.out.println(array) [1][4]);, fails. It selects the array/element with index 1, {0, 1}, and from this array it try to select the element with index 4. This causes an exception. Output: 4

Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 4

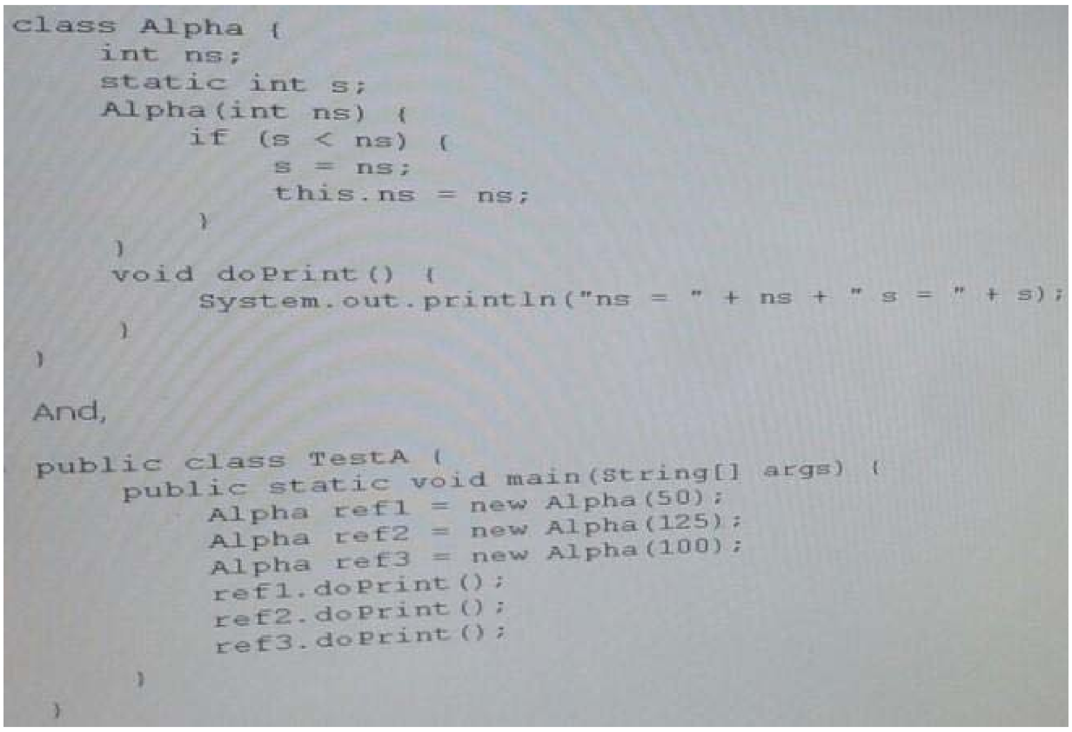
QUESTION: 29

Given: Given:
public class SuperTest {
public static void main(String[] args) { statement1
statement2 statement3
}
}
class Shape { public Shape()
{ System.out.println("Shape:
constructor");
}
public void foo() { System.out.println("Shape: foo");
}
}
class Square extends Shape { public Square() {
super();
}
public Square(String label) { System.out.println("Square: constructor");
}
public void foo() { super.foo();
}
public void foo(String label) { System.out.println("Square: foo");
}
}
}
}
What should statement1, statement2, and statement3, be respectively, in order to produce the result? Shape: constructor Square: foo Shape: foo

A. Square square = new Square ("bar"); square.foo ("bar");
square.foo();
B. Square square = new Square ("bar"); square.foo ("bar");
square.foo ("bar");
C. Square square = new Square (); square.foo ();
square.foo(bar);
D. Square square = new Square (); square.foo ();
square.foo("bar");
E. Square square = new Square (); square.foo ();
square.foo ();
F. Square square = new Square(); square.foo("bar");
square.foo();

Answer: F

QUESTION: 30
Given:



```
class Alpha {
    int ns;
    static int s;
    Alpha(int ns) {
        if (s < ns) {
            s = ns;
            this.ns = ns;
        }
    }
    void doPrint() {
        System.out.println("ns = " + ns + " s = " + s);
    }
}
```

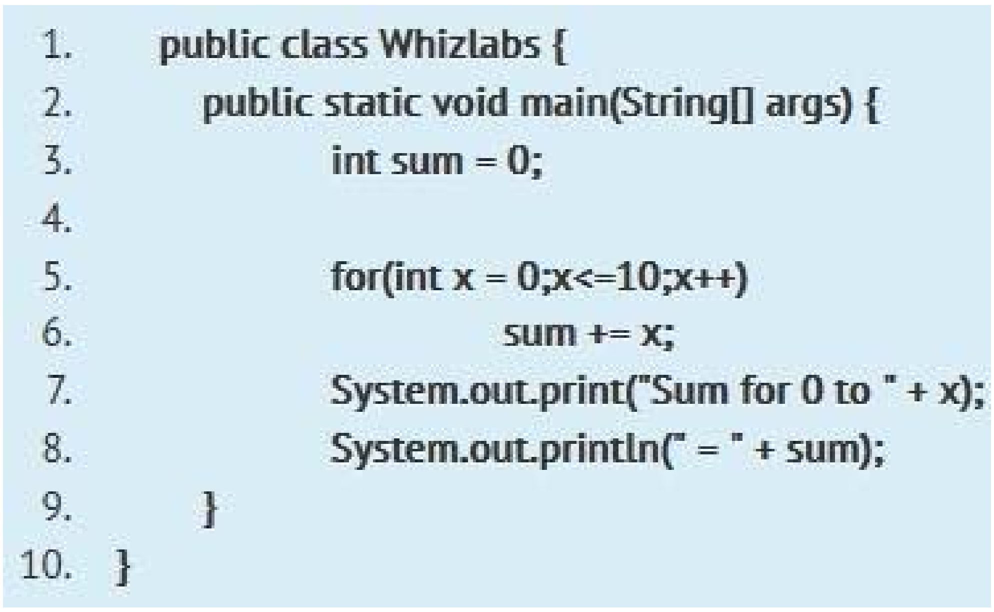
And,

```
public class TestA {
    public static void main(String[] args) {
        Alpha ref1 = new Alpha(50);
        Alpha ref2 = new Alpha(125);
        Alpha ref3 = new Alpha(100);
        ref1.doPrint();
        ref2.doPrint();
        ref3.doPrint();
    }
}
```

A. ns = 50 S = 125 ns = 125 S = 125
ns = 100 S = 125
B. ns = 50 S = 125 ns = 125 S = 125
ns = 0 S = 125
C. ns = 50 S = 50 ns = 125 S = 125
ns = 100 S = 100 D. ns = 50 S = 50 ns = 125 S = 125
ns = 0 S = 125

Answer: B

QUESTION: 31
Given:



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

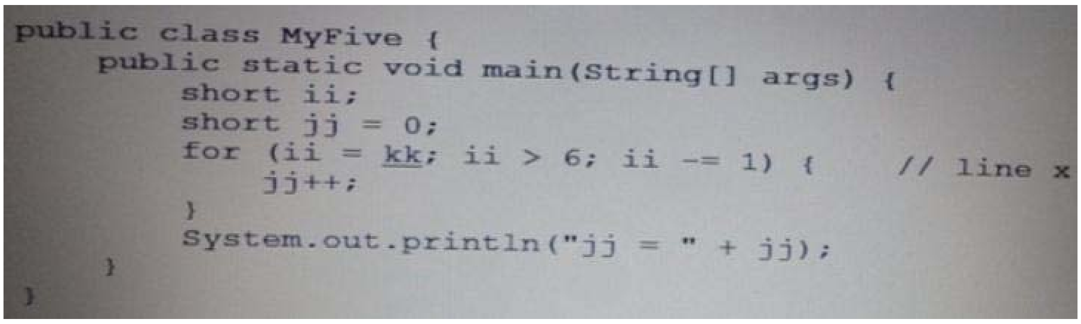
Which is true?

A. Sum for 0 to 0 = 55
B. Sum for 0 to 10 = 55
C. Compilation fails due to error on line 6.
D. Compilation fails due to error on line 7.
E. An Exception is thrown at the runtime.

Answer: D

Explanation:
Loop variables scope limited to that enclosing loop. So in this case, the scope of the loop variable x declared at line 5, limited to that for loop. Trying to access that variable at line 7, which is out of scope of the variable x, causes a compile time error. So compilation fails due to error at line 7. Hence option D is correct.
Options A and B are incorrect, since code fails to compile.
Reference: httpsy/docs.oracle.com/javase/tutorial/java/nutsandbolts/variables.html

QUESTION: 32
Give:



```
public class MyFive {
    public static void main(String[] args) {
        short ii;
        short jj = 0;
        for (ii = kk; ii > 6; ii -= 1) {      // line x
            jj++;
        }
        System.out.println("jj = " + jj);
    }
}
```

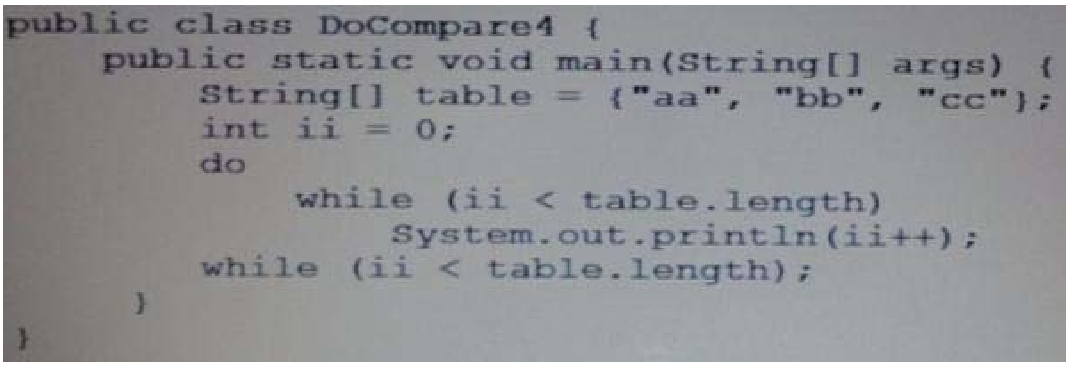
What value should replace kk in line x to cause jj = 5 to be output?

A. -1
B. 1
C. 5
D. 8
E. 11

Answer: E

Explanation:
We need to get jj to 5. It is initially set to 0. So we need to go through the for loop 5 times. The for loops ends when ii > 6 and ii decreases for every loop. So we need to initially set ii to 11. We set kk to 11.

QUESTION: 33
Given:



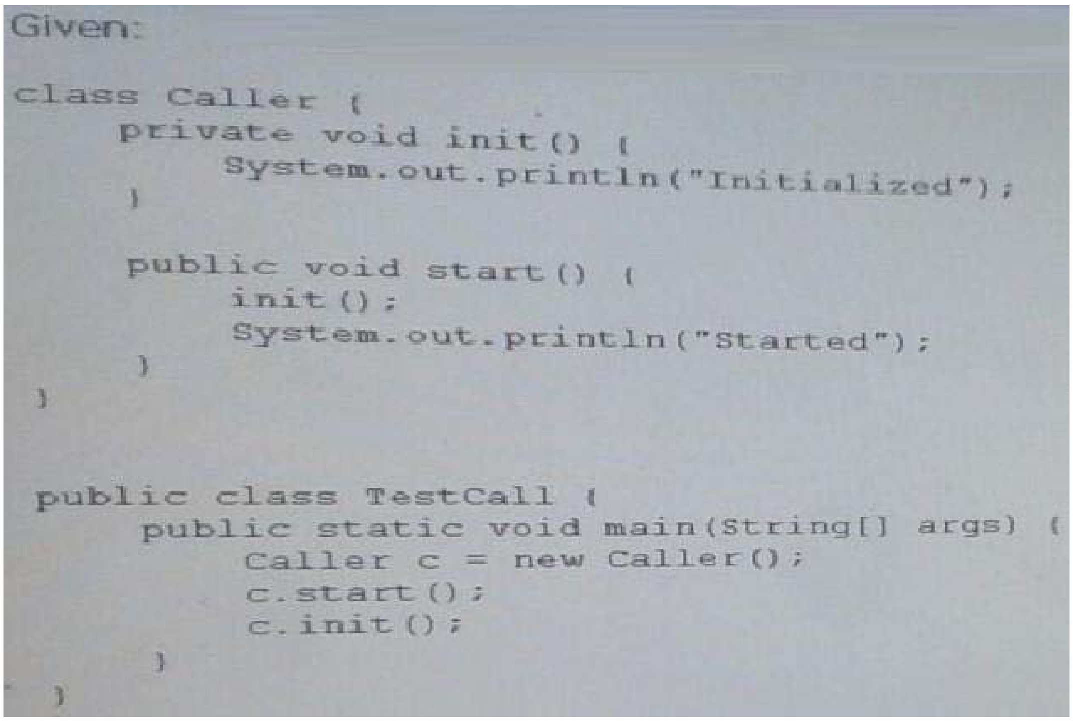
```
public class DoCompare4 {
    public static void main(String[] args) {
        String[] table = {"aa", "bb", "cc"};
        int ii = 0;
        do
            while (ii < table.length)
                System.out.println(ii++);
        while (ii < table.length);
    }
}
```

What is the result?

A. 0
B. 0 12
C. 0 12012012
D. Compilation fails

Answer: B

Explanation:
table.length is 3. So the do-while loop will run 3 times with ii=0, ii=1 and ii=2. The second while statement will break the do-loop when ii = 3.
Note: The Java programming language provides a do-while statement, which can be expressed as follows:
do { statement(s)
} while (expression);

QUESTION: 34
Given:

Given:

```
class Caller {
    private void init() {
        System.out.println("Initialized");
    }

    public void start() {
        init();
        System.out.println("Started");
    }
}

public class TestCall {
    public static void main(String[] args) {
        Caller c = new Caller();
        c.start();
        c.init();
    }
}
```

What is the result?

A. Initialized Started
B. Initialized Started Initialized
C. Compilation fails
D. An exception is thrown at runtime

Answer: B

QUESTION: 35
Given:
public class X { static int i;

```
int j;
public static void main(String[] args) { X x1 = new X();
X x2 = new X(); x1.i = 3;
x1.j  =  4;
x2.i  =  5;
x2.j = 6;
System.out.println( x1.i + " "+
x1.j  +  "  "+
x2.i  +  "  "+
x2.j);
}
}
```
What is the result?

A. 3 4 5 6
B. 3 4 3 6
C. 5 4 5 6
D. 3 6 4 6

Answer: C

QUESTION: 36
Which code fragment cause a compilation error?

A. flat flt = 100F;
B. float flt = (float) 1_11.00;
C. float flt = 100;
D. double y1 = 203.22; floatflt = y1
E. int y2 = 100; floatflt = (float) y2;

Answer: B

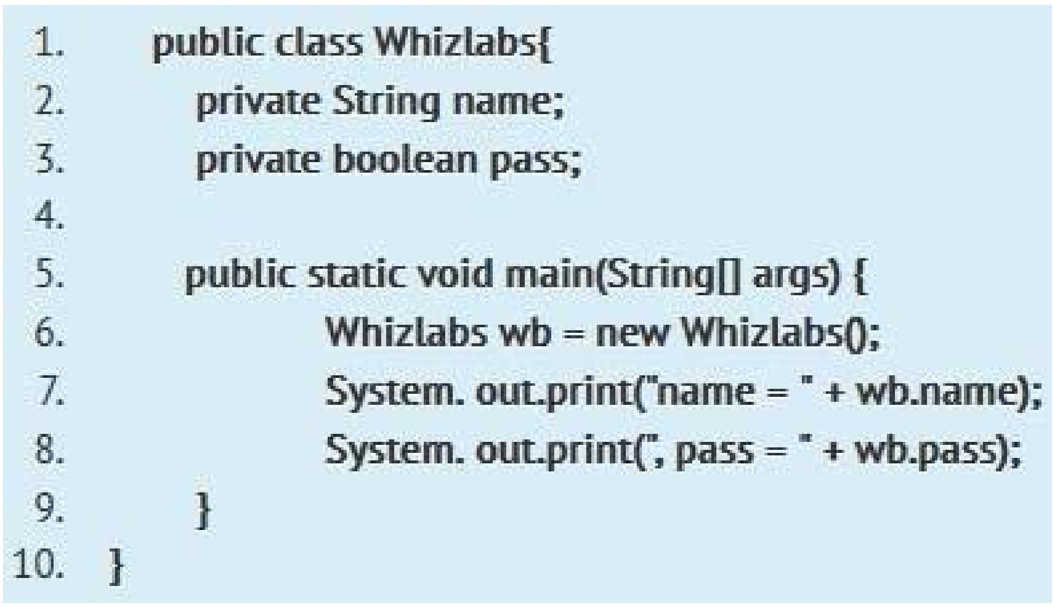
QUESTION: 37
Given:
```
public class Test {
public static void main(String[] args) { int ax = 10, az = 30;
int aw = 1, ay = 1; try {
aw = ax % 2; ay = az / aw;
} catch (ArithmeticException e1) { System.out.println("Invalid Divisor");
} catch (Exception e2) { aw = 1;
System.out.println("Divisor Changed");
}
ay = az /aw; // Line 14 System.out.println("Succesful Division " + ay);
```

}
}
What is the result?

A. Invalid Divisor Divisor Changed Successful Division 30
B. Invalid Divisor Successful Division 30
C. Invalid Divisor Exception in thread "main" java.lang.ArithmeticException: / by zero at test.Teagle.main(Teagle.java:14)
D. Invalid Divisor Exception in thread "main" java.lang.ArithmeticException: / by zero at test.Teagle.main(Teagle.java:14) Successful Division 1

Answer: C

QUESTION: 38
Given:



```
public class Whizlabs{
   private String name;
   private boolean pass;

  public static void main(String[] args) {
          Whizlabs wb = new Whizlabs();
          System. out.print("name = " + wb.name);
          System. out.print(", pass = " + wb.pass);
   }
}
```

What would be the output, if it is executed as a program?

A. name =, pass =
B. name = null, pass = null
C. name = null, pass = false
D. name = null pass = true E.
Compile error.

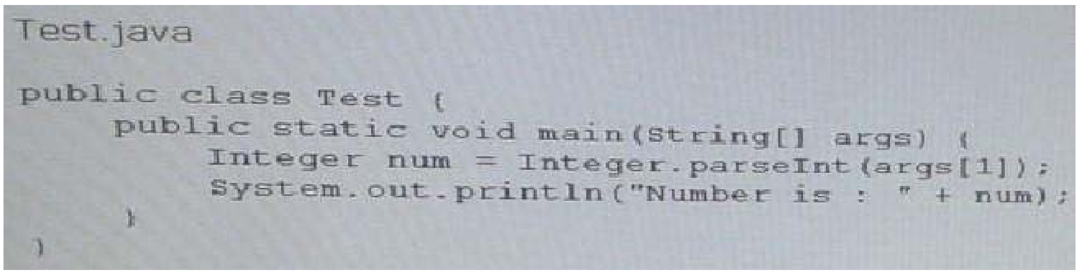
Answer: C

Explanation:

Both name and pass variables are instance variables, and we haven't given them any values, so they take their default values. For Boolean default value is false and for string which is not a primitive type default is null So at line 7, null will printed as the value of the variable name, and at line 8 false will be printed. Hence Option C is correct. As explained above options A, B and D are incorrect. Code compiles fine so option E is incorrect.

Reference:
https://docs.oracle.com/javaseAutorial/java/javaOOAariables.html

QUESTION: 39
Given:



Test.java

```
public class Test {
    public static void main(String[] args) {
        Integer num = Integer.parseInt(args[1]);
        System.out.println("Number is : " + num);
    }
}
```

And the commands: Javac Test.java Java Test 12345 What is the result?

A. Number us : 12345
B. A NullPointerException is thrown at runtime
C. A NumberFormatException is thrown at runtime
D. AnArrayIndexOutOfBoundException is thrown at runtime.

Answer: A

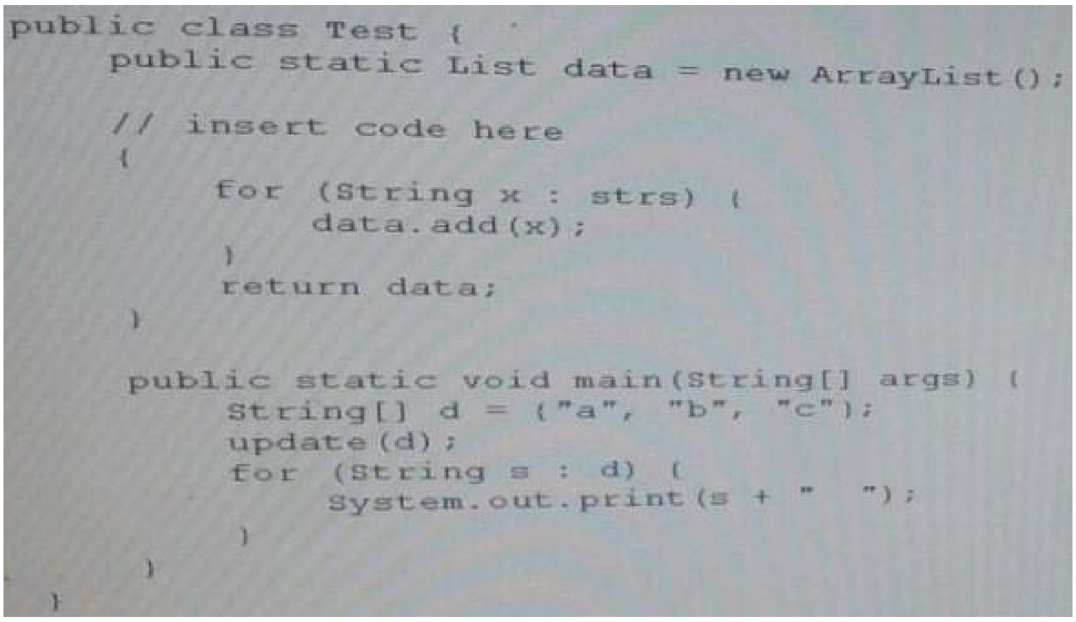
QUESTION: 40
Given:
interface Pet { }
class Dog implements Pet { } public class Beagle extends Dog{ }
Which three are valid?

A. Pet a = new Dog();
B. Pet b = new Pet();
C. Dog f = new Pet();
D. Dog d = new Beagle();
E. Pet e = new Beagle(); F.
Beagle c = new Dog();

Answer: A, D, E

Explanation:
Incorrect:
Not B, not C: Pet is abstact, cannot be instantiated. Not F: incompatible type. Required Beagle, found Dog.

QUESTION: 41
Given the code fragment:



```
public class Test {
    public static List data = new ArrayList();
    // insert code here
    {
        for (String x : strs) {
            data.add(x);
        }
        return data;
    }

    public static void main(String[] args) {
        String[] d = {"a", "b", "c"};
        update(d);
        for (String s : d) {
            System.out.print(s + "  ");
        }
    }
}
```

Which code fragment, when inserted at // insert code here, enables the code to compile and and print a b c?

A. List update (String[] strs)
B. Static ArrayListupdate(String [] strs)
C. Static List update (String [] strs)
D. Static void update (String[] strs)
E. ArrayList static update(String [] strs)

Answer: E

QUESTION: 42
Given:
public class ColorTest {
public static void main(String[] args) {
String[] colors = {"red", "blue","green","yellow","maroon","cyan"}; int count = 0;
for (String c : colors) { if (count >= 4) {break;}
else { continue;
}

```
if (c.length() >= 4) { colors[count] = c.substring(0,3);
}
count++;
}
System.out.println(colors[count]);
}
}
```

What is the result?

A. Yellow
B. Maroon
C. Compilation fails
D. A StringIndexOutOfBoundsException is thrown at runtime.

Answer: C

Explanation:
The line, if (c.length() >= 4) {, is never reached. This causes a compilation error.
Note: The continue statement skips the current iteration of a for, while , or do-while loop. An unlabeled break statement terminates the innermost switch, for, while, or do-while statement, but a labeled break terminates an outer statement.

QUESTION: 43
Which two actions will improve the encapsulation of a class?

A. Changing the access modifier of a field from public to private
B. Removing the public modifier from a class declaration
C. Changing the return type of a method to void
D. Returning a copy of the contents of an array or ArrayList instead of a direct reference

Answer: A, D

Reference:
http://www.tutorialspoint.com/java/java_access_modifiers.htm

QUESTION: 44

```
public class Two {
public static void main(String[] args) { try {
doStuff(); system.out.println("1");
}
catch { system.out.println("2");
}}
public static void do Stuff() {
```

if (Math.random() > 0.5) throw new RunTimeException(); doMoreStuff();
System.out.println("3 ");
}
public static void doMoreStuff() { System.out.println("4");
}
}
Which two are possible outputs?

A. 2
B. 4 3
1
C. 1
D. 1 2

Answer: A, B

Explanation:
A: Output is 2 if Math.random() is greater than 0.5. B: If Math.random() returns a value less equal to 0.5, the code won't throw an exception, it will continue with the doMore() method which will println "4" after which the program will continue with the doStuff() method and will println "3", after that we will be back in main() and the program will print "1".

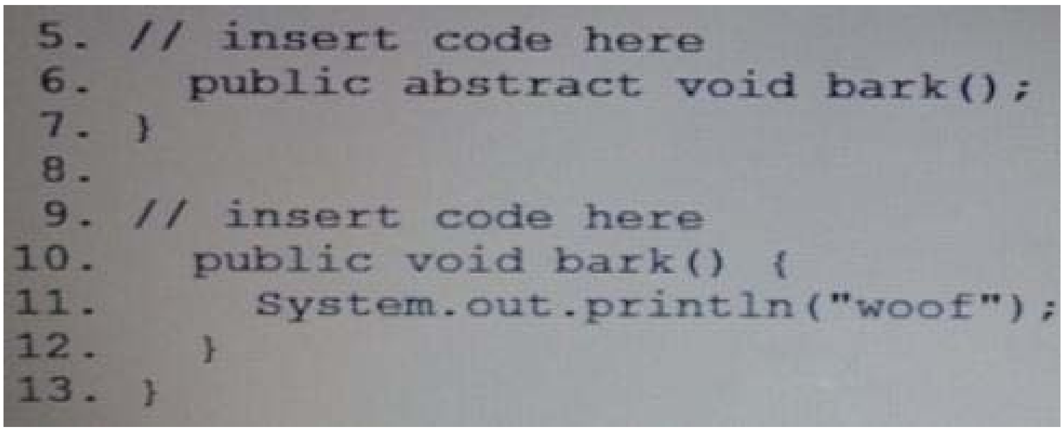
QUESTION: 45
Given:
public class Test {
public static void main(String[] args) { int day = 1;
switch (day) {
case "7": System.out.print("Uranus");
case "6": System.out.print("Saturn");
case "1": System.out.print("Mercury");
case "2": System.out.print("Venus");
case "3": System.out.print("Earth");
case "4": System.out.print("Mars");
case "5": System.out.print("Jupiter");
}
}
}
Which two modifications, made independently, enable the code to compile and run?

A. Adding a break statement after each print statement
B. Adding a default section within the switch code-block
C. Changing the string literals in each case label to integer
D. Changing the type of the variable day to String
E. Arranging the case labels in ascending order

Answer: A, C

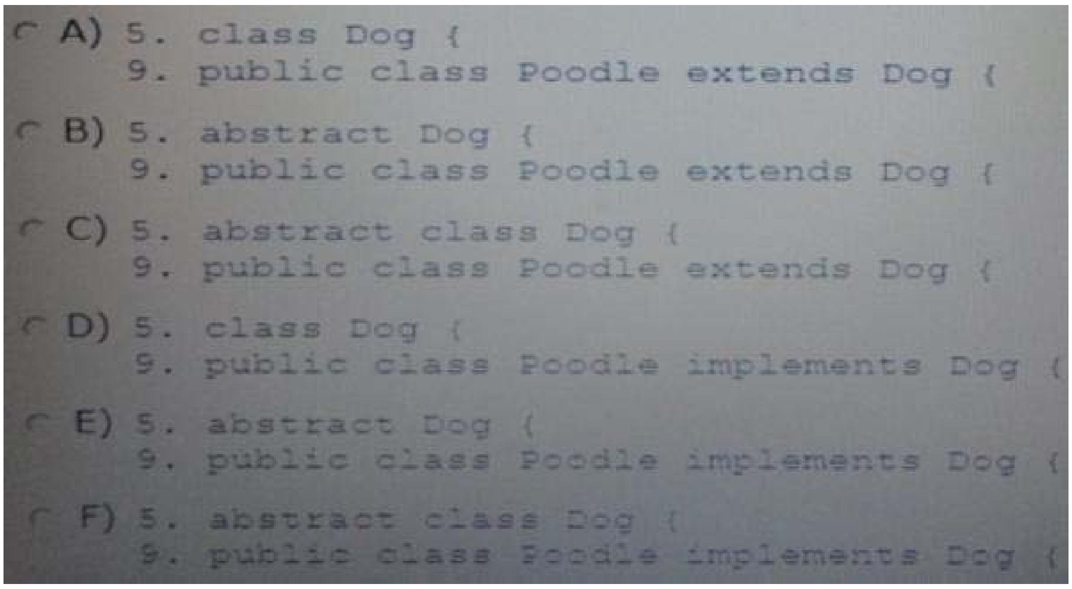
Explanation:
The following will work fine:
public class Test {
public static void main(String[] args) { int day = 1;
switch (day) {
case 7: System.out.print("Uranus"); break; case 6: System.out.print("Saturn"); break; case 1: System.out.print("Mercury"); break; case 2: System.out.print("Venus"); break; case 3: System.out.print("Earth"); break; case 4: System.out.print("Mars"); break; case 5: System.out.print("Jupiter"); break;
}
}
}

QUESTION: 46
Given:



```
// insert code here
    public abstract void bark();
}

// insert code here
    public void bark() {
        System.out.println("woof");
    }
}
```

What code should be inserted?



```
A) 5. class Dog {
   9. public class Poodle extends Dog {

B) 5. abstract Dog {
   9. public class Poodle extends Dog {

C) 5. abstract class Dog {
   9. public class Poodle extends Dog {

D) 5. class Dog {
   9. public class Poodle implements Dog {

E) 5. abstract Dog {
   9. public class Poodle implements Dog {

F) 5. abstract class Dog {
   9. public class Poodle implements Dog {
```

A. Option A
B. Option B
C. Option C
D. Option D
E. Option E
F. Option F

Answer: C

Explanation:
Dog should be an abstract class. The correct syntax for this is: abstract class Dog { Poodle should extend Dog (not implement).

QUESTION: 47
View the Exhibit.
public class Hat { public int ID =0;
public String name = "hat";
public String size = "One Size Fit All"; public String color="";
public String getName() { return name; } public void setName(String name) { this.name = name;
}
}
Given
public class TestHat {
public static void main(String[] args) { Hat blackCowboyHat = new Hat();
}
}
Which statement sets the name of the Hat instance?

A. blackCowboyHat.setName = "Cowboy Hat";
B. setName("Cowboy Hat");
C. Hat.setName("Cowboy Hat");
D. blackCowboyHat.setName("Cowboy Hat");

Answer: D

QUESTION: 48
Given:
public class Test1 {
static void doubling (Integer ref, int pv) { ref =20;
pv = 20;
}
public static void main(String[] args) { Integer iObj = new Integer(10);
int iVar = 10; doubling(iObj++, iVar++);
System.out.println(iObj+ ", "+iVar); What is the result?

A. 11, 11
B. 10, 10
C. 21, 11
D. 20, 20
E. 11, 12

Answer: A

Explanation:
The code doubling(iObj++, iVar++); increases both variables from to 10 to 11.

QUESTION: 49
Which three are advantages of the Java exception mechanism?

A. Improves the program structure because the error handling code is separated from the normal program function
B. Provides a set of standard exceptions that covers all the possible errors
C. Improves the program structure because the programmer can choose where to handle exceptions
D. Improves the program structure because exceptions must be handled in the method in which they occurred
E. allows the creation of new exceptions that are tailored to the particular program being

Answer: A, C, E

Explanation:
A: The error handling is separated from the normal program logic. C: You have some choice where to handle the exceptions.
E: You can create your own exceptions.

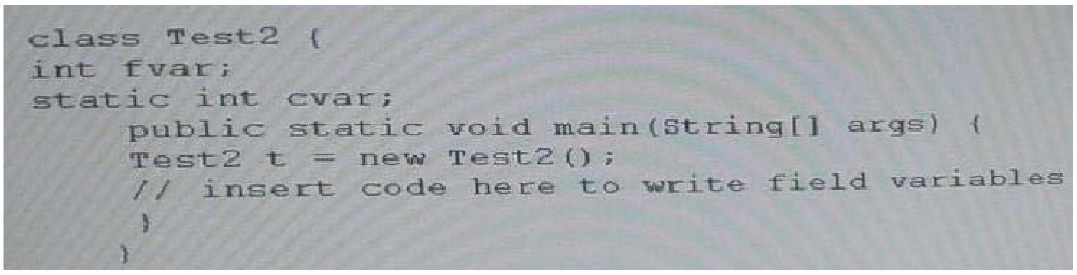
QUESTION: 50
Identify two benefits of using ArrayList over array in software development.

A. reduces memory footprint
B. implements the Collection API
C. is multi.thread safe
D. dynamically resizes based on the number of elements in the list

Answer: A, D

Explanation:
ArrayList supports dynamic arrays that can grow as needed. In Java, standard arrays are of a fixed length. After arrays are created, they cannot grow or shrink, which means that you must know in advance how many elements an array will hold. But, sometimes, you may not know until run time precisely how large of an array you need. To handle this situation, the collections framework defines ArrayList. In essence, an ArrayList is a variable-length array of object references. That is, an ArrayList can dynamically increase or decrease in size. Array lists are created with an initial size. When this size is exceeded, the collection is automatically enlarged. When objects are removed, the array may be shrunk.

QUESTION: 51
Given the code fragment



```
class Test2 {
 int fvar;
 static int cvar;
     public static void main(String[] args) {
     Test2 t = new Test2();
     // insert code here to write field variables
      }
    }
```

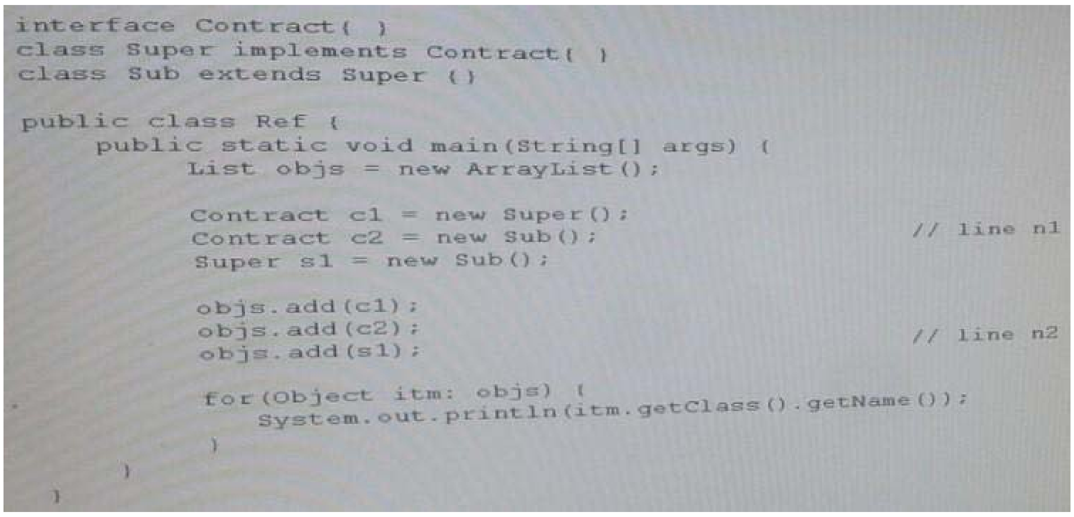
Which code fragments, inserted independently, enable the code compile?

A. t.fvar = 200;
B. cvar = 400;
C. fvar = 200; cvar = 400;
D. this.fvar = 200; this.cvar = 400;
E. t.fvar = 200; Test2.cvar = 400;

F. this.fvar = 200; Test2.cvar = 400;

Answer: B

QUESTION: 52
Given the code fragments:



```
interface Contract{ }
class Super implements Contract{ }
class Sub extends Super {}

public class Ref {
     public static void main(String[] args) {
          List objs = new ArrayList();

          Contract c1 = new Super();
          Contract c2 = new Sub();                     // line n1
          Super s1 = new Sub();

          objs.add(c1);
          objs.add(c2);                                 // line n2
          objs.add(s1);

          for(Object itm: objs) {
               System.out.println(itm.getClass().getName());
          }
     }
}
```

What is the result?

A. Super Sub Sub
B. Contract Contract Super
C. Compilation fails at line n1
D. Compilation fails at line n2

Answer: D

QUESTION: 53
Given:
public class TestLoop1 {
public static void main(String[] args) { int a = 0, z=10;
while (a < z) { a++;
--z;
}
System.out.print(a + " : " + z);
}
}
What is the result?

A. 5 : 5
B. 6 : 4
C. 6 : 5
D. 5 : 4

Answer: A

QUESTION: 54
Given the code fragment:
// insert code here arr[0] = new int[3]; arr[0][0] = 1;
arr[0][1] = 2;
arr[0][2] = 3;
arr[1] = new int[4]; arr[1][0] = 10;
arr[1][1] = 20;
arr[1][2] = 30;
arr[1][3] = 40;
Which two statements, when inserted independently at line // insert code here, enable the code to compile?

A. int [] [] arr = null;
B. int [] [] arr = new int [2];
C. int [] [] arr = new int [2] [ ];
D. int [] [] arr = new int [] [4]; E.
int [] [] arr = new int [2] [0]; F.
int [] [] arr = new int [0] [4];

Answer: C, E

QUESTION: 55
Give:
Public Class Test {}
Which two packages are automatically imported into the java source file by the java compiler?

A. Java.lang
B. Java.awt
C. Java.util
D. Javax.net
E. Java.*
F. The package with no name

Answer: A, F

Explanation:
For convenience, the Java compiler automatically imports three entire packages for each source file: (1) the package with no name, (2) the java.lang package, and (3) the current package (the package for the current file).
Note:Packages in the Java language itself begin with java. or javax.

QUESTION: 56
Which is a valid abstract class?

A. public abstract class Car { protected void accelerate();}
B. public interface Car {protected abstract void accelerate();}
C. public abstract class Car { protected final void accelerate();}
D. public abstract class Car { protected abstract void accelerate();}
E. public abstract class Car { protected abstract void accelerate() {
//more car can do}}

Answer: D

QUESTION: 57
Given the code fragment:
public static void main(String[] args) { int iArray[] = {65, 68, 69};
iArray[2] = iArray[0]; iArray[0] = iArray[1]; iArray[1] = iArray[2];
for (int element : iArray) { System.out.print(element + " ");}

A. 68, 65, 69
B. 68, 65, 65
C. 65, 68, 65
D. 65, 68, 69
E. Compilation fails

Answer: B

QUESTION: 58
Which three statements are true about the structure of a Java class?

A. A class can have only one private constructor.
B. A method can have the same name as a field.
C. A class can have overloaded static methods.
D. A public class must have a main method.
E. The methods are mandatory components of a class.

F. The fields need not be initialized before use.

Answer: A, B, C

Explanation:
A: Private constructors prevent a class from being explicitly instantiated by its callers.
If the programmer does not provide a constructor for a class, then the system will always provide a default, public no-argument constructor. To disable this default constructor, simply add a private no-argument constructor to the class. This private constructor may be empty.
B: The following works fine: int cake() {
int cake=0; return (1);}
C: We can overload static method in Java. In terms of method overloading static method are just like normal methods and in order to overload static method you need to provide another static method with same name but different method signature.
Incorrect:
Not D: Only a public class in an application need to have a main method. Not E:
Example:
class A
{public string something; public int a;}
Q: What do you call classes without methods? Most of the time: An anti pattern.
Why? Because it faciliates procedural programming with "Operator" classes and data structures. You separate data and behaviour which isn't exactly good OOP.
Often times: A DTO (Data Transfer Object)
Read only datastructures meant to exchange data, derived from a business/domain object. Sometimes: Just data structure.
Well sometimes, you just gotta have those structures to hold data that is just plain and simple and has no operations on it.
Not F: Fields need to be initialtized. If not the code will not compile. Example:
Uncompilable source code - variable x might not have been initialized

QUESTION: 59
Which two items can legally be contained within a java class declaration?

A. An import statement
B. A field declaration
C. A package declaration
D. A method declaration

Answer: B, D

Reference:
http://docs.oracle.com/javase/tutorial/java/javaOO/methods.html

QUESTION: 60

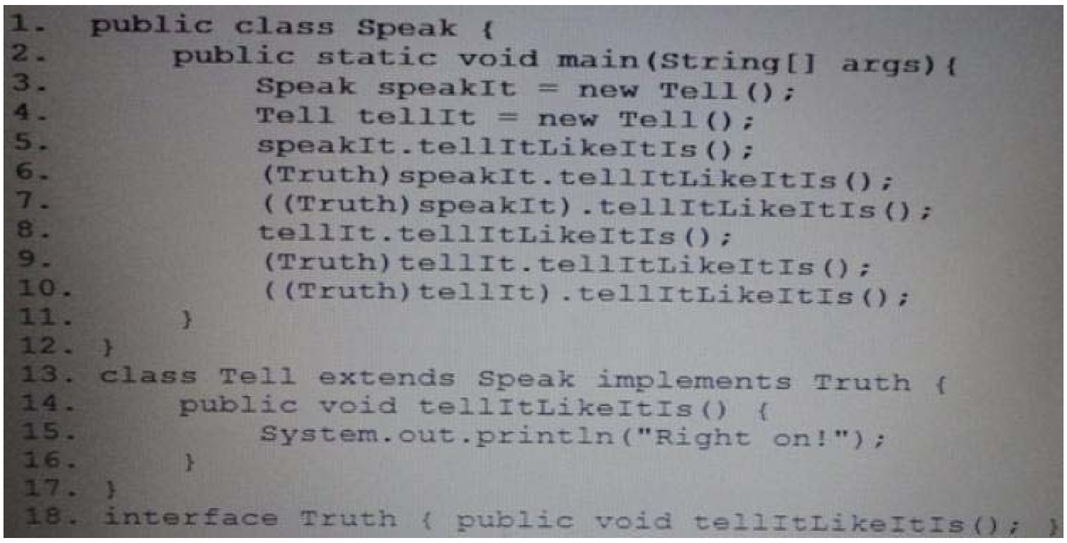
Which two may precede the word ‘class’ in a class declaration?

A. local
B. public
C. static
D. volatile
E. synchronized

Answer: B, C

Explanation:
B: A class can be declared as public or private.
C: You can declare two kinds of classes: top-level classes and inner classes.
You define an inner class within a top-level class. Depending on how it is defined, an inner class can be one of the following four types: Anonymous, Local, Member and Nested top-level. A nested top-level class is a member classes with a static modifier. A nested top-level class is just like any other top-level class except that it is declared within another class or interface. Nested top-level classes are typically used as a convenient way to group related classes without creating a new package. The following is an example: public class Main { static class Killer {

QUESTION: 61
Given:



```
1.  public class Speak {
2.      public static void main(String[] args){
3.          Speak speakIt = new Tell();
4.          Tell tellIt = new Tell();
5.          speakIt.tellItLikeItIs();
6.          (Truth)speakIt.tellItLikeItIs();
7.          ((Truth)speakIt).tellItLikeItIs();
8.          tellIt.tellItLikeItIs();
9.          (Truth)tellIt.tellItLikeItIs();
10.         ((Truth)tellIt).tellItLikeItIs();
11.     }
12. }
13. class Tell extends Speak implements Truth {
14.     public void tellItLikeItIs() {
15.         System.out.println("Right on!");
16.     }
17. }
18. interface Truth { public void tellItLikeItIs(); }
```

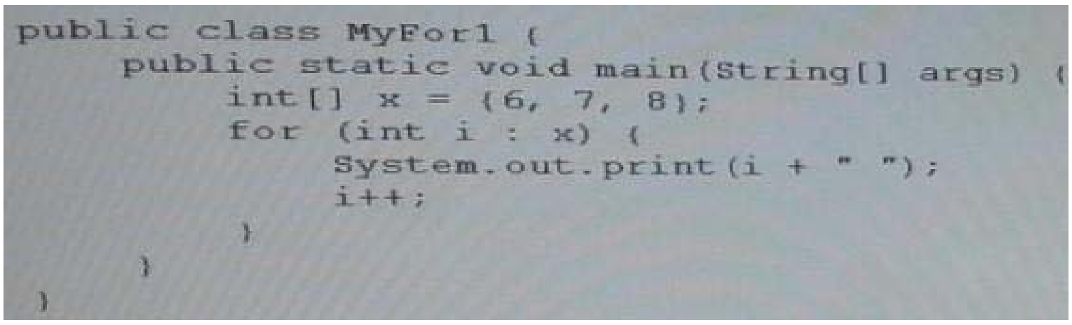
Which three lines will compile and output “right on!”?

A. Line 5
B. Line 6
C. Line 7
D. Line 8

E. Line 9
F. Line 10

Answer: C, D, F

QUESTION: 62
Given:



```
public class MyFor1 {
    public static void main(String[] args) {
        int[] x = {6, 7, 8};
        for (int i : x) {
            System.out.print(i + " ");
            i++;
        }
    }
}
```

What is the result?

A. 6 7 8
B. 7 8 9
C. 0 1 2
D. 6 8 10
E. Compilation fails

Answer: A

QUESTION: 63
Given the code fragment:
1. ArrayList<Integer> list = new ArrayList<>(1); 2. list.add(1001);
3. list.add(1002);
4. System.out.println(list.get(list.size())); What is the result?

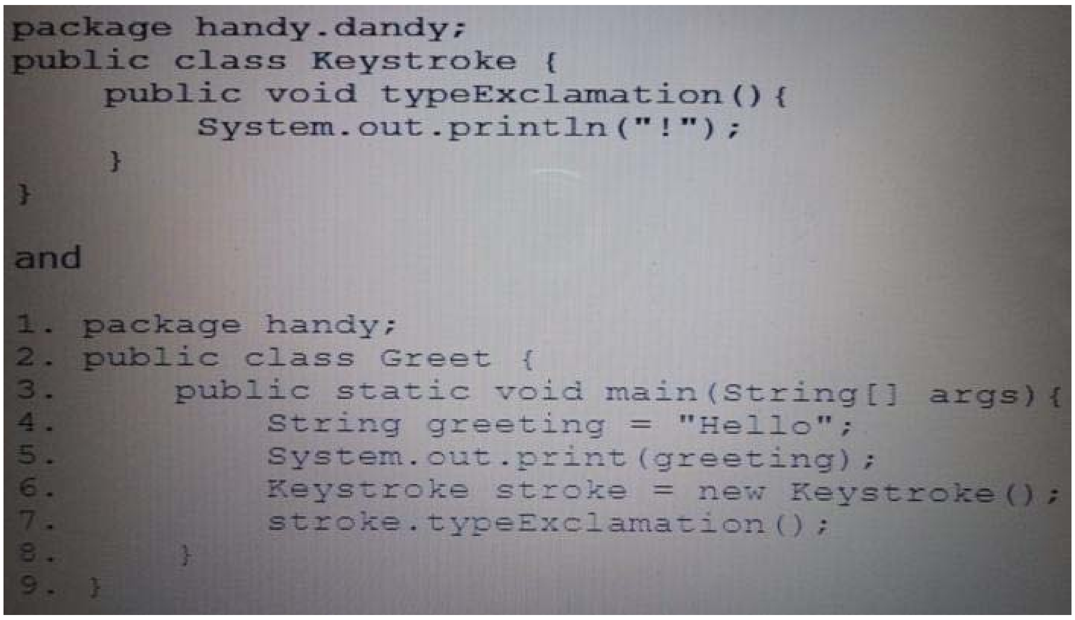
A. Compilation fails due to an error on line 1.
B. An exception is thrown at run time due to error on line 3
C. An exception is thrown at run time due to error on line 4
D. 1002

Answer: C

Explanation:
The code compiles fine. At runtime an IndexOutOfBoundsException is thrown when the

second list item is added.

QUESTION: 64
Given:



```
package handy.dandy;
public class Keystroke {
    public void typeExclamation(){
        System.out.println("!");
    }
}
```

and

```
1. package handy;
2. public class Greet {
3.     public static void main(String[] args){
4.         String greeting = "Hello";
5.         System.out.print(greeting);
6.         Keystroke stroke = new Keystroke();
7.         stroke.typeExclamation();
8.     }
9. }
```

What three modifications, made independently, made to class greet, enable the code to compile and run?

A. line 6 replaced with handy.dandy.keystroke stroke = new KeyStroke ( );
B. line 6 replaced with handy.*.KeyStroke = new KeyStroke ( );
C. line 6 replaced with handy.dandy.KeyStroke Stroke = new handy.dandy.KeyStroke();
D. import handy.*; added before line 1
E. import handy.dandy.*; added after line 1
F. import handy.dandy,KeyStroke; added after line 1
G. import handy.dandy.KeyStroke.typeException(); added before line 1

Answer: C, E, F

Explanation:
Three separate solutions:
C: the full class path to the method must be stated (when we have not imported the package)
D: We can import the hold dandy class F: we can import the specific method

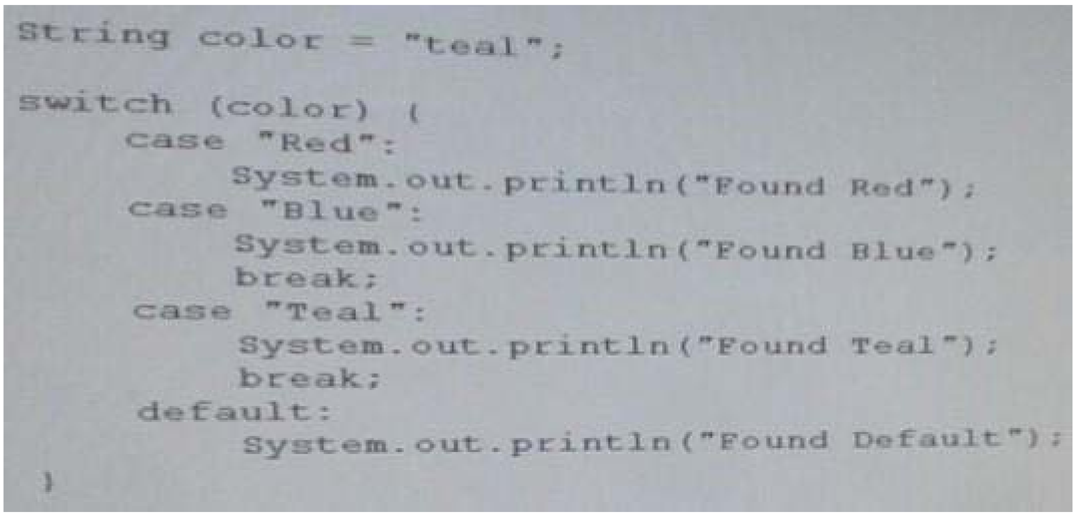
QUESTION: 65
Which statement is/are true?
I. Default constructor only contains "super();" call.
II. We can't use any access modifier with a constructor.
III. A constructor should not have a return type.

A. Only I.
B. Only II.
C. Only I and II.
D. Only I and III.
E. AIL

Answer: D

Explanation:
Statement I is correct as the default constructor only contains super0 call Statement II is incorrect as we can use any access modifier with a constructor. Statement III is correct as constructor can't have return type, even void. So option D is correct.
httpsy/docs.oracle.com/javase/tutorial/iava/javaOO/construaors.html

QUESTION: 66
Given the code fragment:



```
String color = "teal";

switch (color) {
     case "Red":
          System.out.println("Found Red");
     case "Blue":
          System.out.println("Found Blue");
          break;
     case "Teal":
          System.out.println("Found Teal");
          break;
     default:
          System.out.println("Found Default");
}
```

What is the result?

A. Found Red Found Default
B. Found Teal
C. Found Red Found Blue Found Teal
D. Found Red Found Blue Found Teal Found Default
E. Found Default

Answer: B

QUESTION: 67
Given the code fragment:
StringBuilder sb = new StringBuilder ( ) ; Sb.append (“world”);
Which code fragment prints Hello World?

A. sb.insert(0,"Hello "); System.out.println(sb); B.
sb.append(0,"Hello "); System.out.println(sb); C.
sb.add(0,"Hello "); System.out.println(sb);
D. sb.set(0,"Hello "); System.out.println(sb);D

Answer: A

Explanation:
The java.lang.StringBuilder.insert(int offset, char c) method inserts the string representation of the char argument into this sequence. The second argument is inserted into the contents of this sequence at the position indicated by offset. The length of this sequence increases by one.The offset argument must be greater than or equal to 0, and less than or equal to the length of this sequence.

Reference: Java.lang.StringBuilder.insert() Method

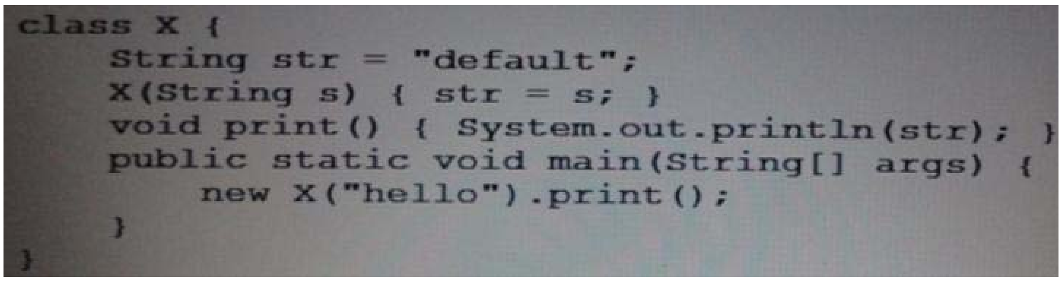
QUESTION: 68
Given the code fragment:
String h1 = "Bob";
String h2 = new String ("Bob");
What is the best way to test that the values of h1 and h2 are the same?

A. if (h1 = = h2)
B. if (h1.equals(h2))
C. if (h1 = = h2)
D. if (h1.same(h2))

Answer: B

Explanation:
The equals method compares values for equality.

QUESTION: 69
Given:



```
class X {
     String str = "default";
     X(String s) { str = s; }
     void print() { System.out.println(str); }
     public static void main(String[] args) {
          new X("hello").print();
     }
}
```

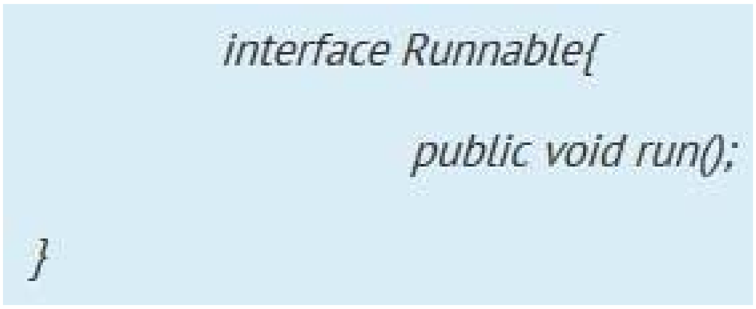
What is the result?

A. Hello
B. Default
C. Compilation fails
D. The program prints nothing
E. An exception is thrown at run time

Answer: A

Explanation:
The program compiles fine. The program runs fine. The output is: hello

QUESTION: 70
Consider following interface.



```
interface Runnable{
          public void run();
}
```

Which of the following will create instance of Runnable type?

A. Runnable run = 0 -> {System.out.println("Run");}
B. Runnable run = 0 -> System.outprintlnfRun");
C. Runnable run = 0 > System.outprintlnfRun");
D. Runnable run = > System.ouLprintlnfRun"}; E.
None of the above.

Answer: A

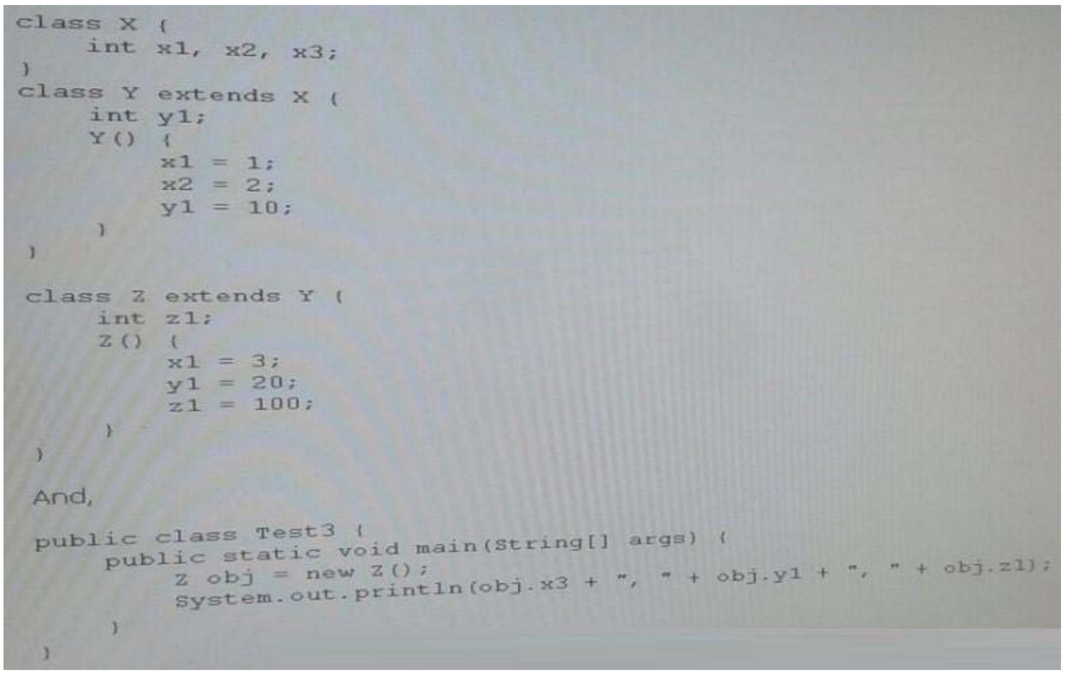
Explanation:
Option A is the correct answer.
To create we have used following method with LocalDate class; public static LocalDate of(intyear, int month, intdayOfMonth)
Here we need to remember that month is not zero based so if you pass 1 for month, then month will be January.
Then we have used period object of 1 day and add to date object which makes current date to next day, so final output is 2015-03-27. Hence option A is correct.

QUESTION: 71
Given:
public class Painting { private String type;
public String getType() { return type;}
public void setType(String type) { this.type = type;}
public static void main(String[] args) {
Painting obj1 = new Painting(); Painting obj2 = new Painting(); obj1.setType(null); obj2.setType("Fresco");
System.out.print(obj1.getType() + " : " + obj2.getType());}}
What is the result?

A. : Fresco
B. null : Fresco
C. Fresco : Fresco
D. A NullPointerException is thrown at runtime

Answer: B

QUESTION: 72
Given:



```
class X {
    int x1, x2, x3;
}
class Y extends X {
    int y1;
    Y() {
        x1 = 1;
        x2 = 2;
        y1 = 10;
    }
}

class Z extends Y {
    int z1;
    Z() {
        x1 = 3;
        y1 = 20;
        z1 = 100;
    }
}
```

And,

```
public class Test3 {
    public static void main(String[] args) {
        Z obj = new Z();
        System.out.println(obj.x3 + ", " + obj.y1 + ", " + obj.z1);
    }
}
```

Which constructor initializes the variable x3?

A. Only the default constructor of class X
B. Only the no-argument constructor of class Y
C. Only the no-argument constructor of class Z
D. Only the default constructor of object class

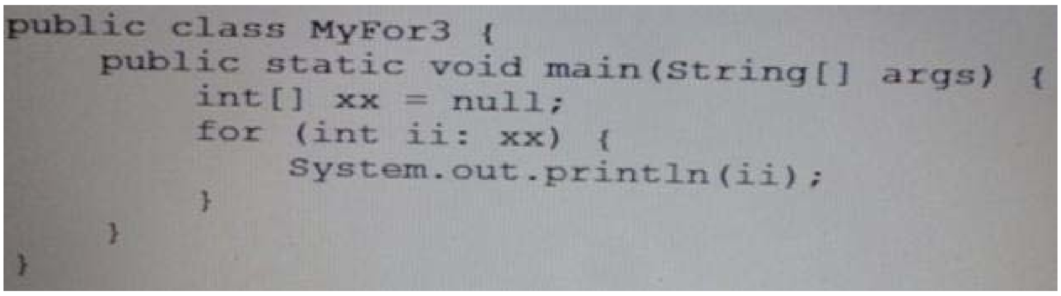
Answer: C

QUESTION: 73
boolean log3 = (5.0 != 6.0) && (4 != 5);
boolean log4 = (4 != 4) || (4 == 4); System.out.println("log3:"+ log3 + \nlog4" + log4);
What is the result?

A. log3:false log4:true
B. log3:true log4:true C.
log3:true log4:false D.
log3:false log4:false

Answer: B

QUESTION: 74
Given:



```
public class MyFor3 {
    public static void main(String[] args) {
        int[] xx = null;
        for (int ii: xx) {
            System.out.println(ii);
        }
    }
}
```

What is the result?

A. null
B. compilation fails
C. Java.lang.NullPointerException
D. 0

Answer: A

Explanation:
An array variable (here xx) can very well have the null value.
Note:
Null is the reserved constant used in Java to represent a void reference i.e a pointer to nothing. Internally it is just a binary 0, but in the high level Java language, it is a magic constant, quite distinct from zero, that internally could have any representation.

QUESTION: 75
Given:
public class Test { static boolean bVar;
public static void main(String[] args) { boolean bVar1 = true;
int count =8; do {
System.out.println("Hello Java! " +count); if (count >= 7) {
bVar1 = false;
}
} while (bVar != bVar1 && count > 4);
count -= 2;
}
}
What is the result?

A. Hello Java! 8 Hello Java! 6 Hello Java! 4
B. Hello Java! 8 Hello Java! 6
C. Hello Java! 8
D. Compilation fails

Answer: C

Explanation:
Hello Java! 8

QUESTION: 76
Given:
```
public class TestLoop {
public static void main(String[] args) { int array[] = {0, 1, 2, 3, 4};
int key = 3;
for (int pos = 0; pos < array.length; ++pos) { if (array[pos] == key) {
break;
}
}
System.out.print("Found " + key + "at " + pos);
}
}
```
What is the result?

A. Found 3 at 2
B. Found 3 at 3
C. Compilation fails
D. An exception is thrown at runtime

Answer: C

Explanation:
The following line does not compile: System.out.print("Found " + key + "at " + pos);
The variable pos is undefined at this line, as its scope is only valid in the for loop. Any variables created inside of a loop are LOCAL TO THE LOOP.

QUESTION: 77
A method doSomething () that has no exception handling code is modified to trail a method that throws a checked exception. Which two modifications, made independently, will allow the program to compile?

A. Catch the exception in the method doSomething().
B. Declare the exception to be thrown in the doSomething() method signature.
C. Cast the exception to a RunTimeException in the doSomething() method. D.
Catch the exception in the method that calls doSomething().

Answer: A, B

Explanation:
Valid Java programming language code must honor the Catch or Specify Requirement. This means that code that might throw certain exceptions must be enclosed by either of the following:
* A try statement that catches the exception. The try must provide a handler for the exception, as described in Catching and Handling Exceptions.
* A method that specifies that it can throw the exception. The method must provide a throws clause that lists the exception, as described in Specifying the Exceptions Thrown by a Method. Code that fails to honor the Catch or Specify Requirement will not compile.

QUESTION: 78
Given:
class X {}
class Y { Y ( ) { } }
class Z { Z (int i) { } }
Which class has a default constructor?

A. X only
B. Y only
C. Z only
D. X and Y
E. Y and Z
F. X and Z
G. X, Y and Z

Answer: A

QUESTION: 79
The protected modifier on a Field declaration within a public class means that the field___________.

A. Cannot be modified
B. Can be read but not written from outside the class
C. Can be read and written from this class and its subclasses only within the same package
D. Can be read and written from this class and its subclasses defined in any package

Answer: D

Reference:
http://beginnersbook.com/2013/05/java-access-modifiers/

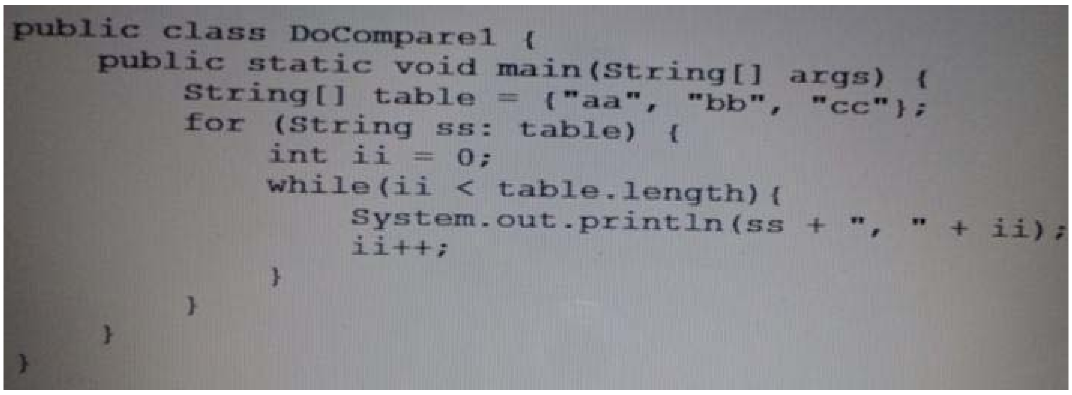
QUESTION: 80

Given:
public class DoBreak1 {
public static void main(String[] args) { String[] table = {"aa", "bb", "cc", "dd"}; for (String ss: table) {
if ("bb".equals(ss)) { continue;
}
System.out.println(ss); if ("cc".equals(ss)) { break;
}
}
}
}
What is the result?

A. aa cc
B. aa bb cc
C. cc dd
D. cc
E. Compilation fails.

Answer: A

QUESTION: 81
Given:



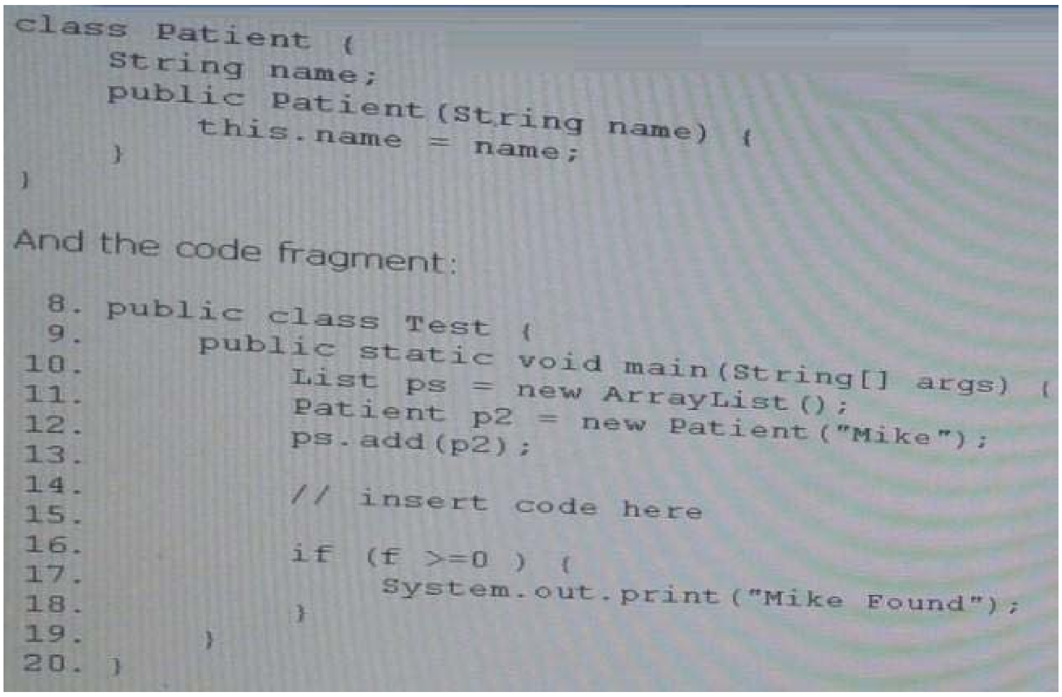
```
public class DoCompare1 {
    public static void main(String[] args) {
        String[] table = {"aa", "bb", "cc"};
        for (String ss: table) {
            int ii = 0;
            while(ii < table.length){
                System.out.println(ss + ", " + ii);
                ii++;
            }
        }
    }
}
```

How many times is 2 printed as a part of the output?

A. Zero
B. Once
C. Twice
D. Thrice
E. Compilation fails.

Answer: D

QUESTION: 82
Given:



```
class Patient {
    String name;
    public Patient(String name) {
        this.name = name;
    }
}
```

And the code fragment:

```
 8. public class Test {
 9.     public static void main(String[] args) {
10.         List ps = new ArrayList();
11.         Patient p2 = new Patient("Mike");
12.         ps.add(p2);
13.
14.         // insert code here
15.
16.         if (f >=0 ) {
17.             System.out.print("Mike Found");
18.         }
19.     }
20. }
```

Which code fragment, when inserted at line 14, enables the code to print Mike Found?

A. int f = ps.indexOf {new patient (“Mike”)};
B. int f = ps.indexOf (patient(“Mike”));
C. patient p = new Patient (“Mike”); int f = pas.indexOf(P)
D. int f = ps.indexOf(p2);

Answer: C

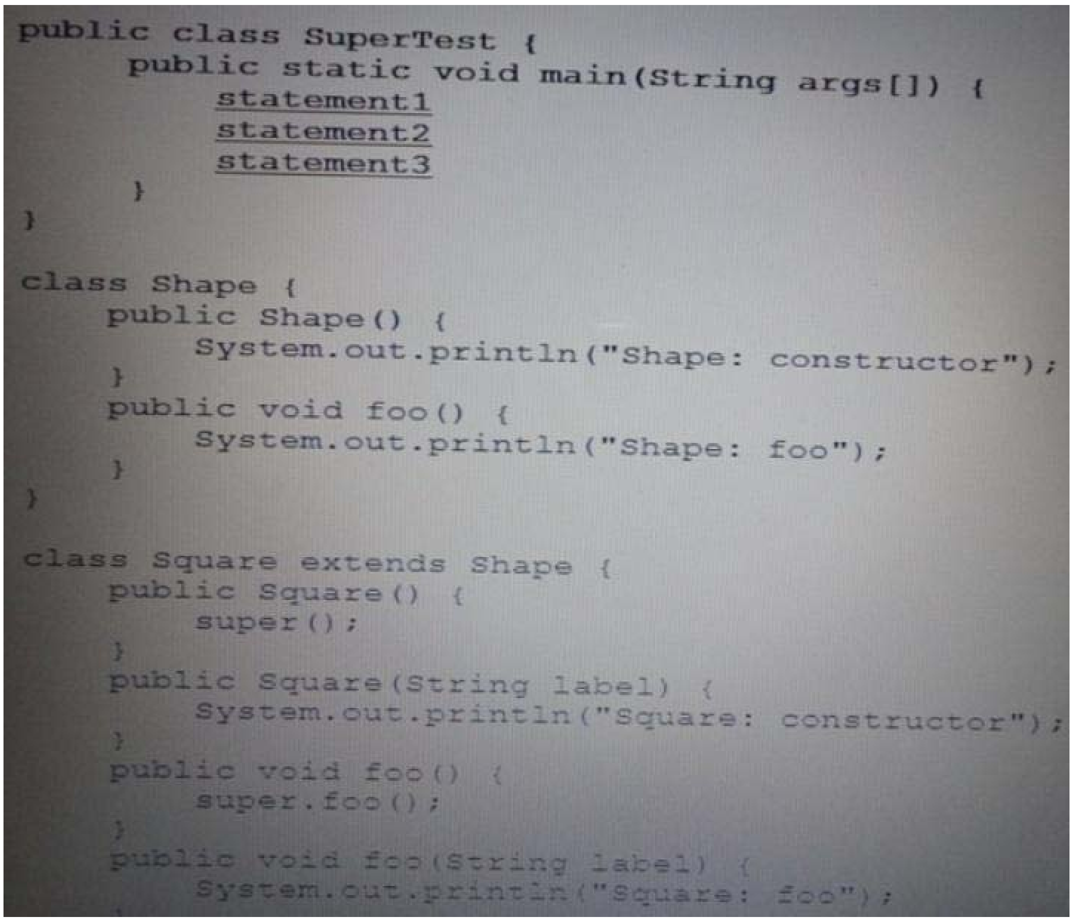
QUESTION: 83
Given:
public class TestOperator {
public static void main(String[] args) { int result = 30 -12 / (2*5)+1; System.out.print("Result = " + result);
}
}
What is the result?

A. Result = 2
B. Result = 3

C. Result = 28
D. Result = 29
E. Result = 30

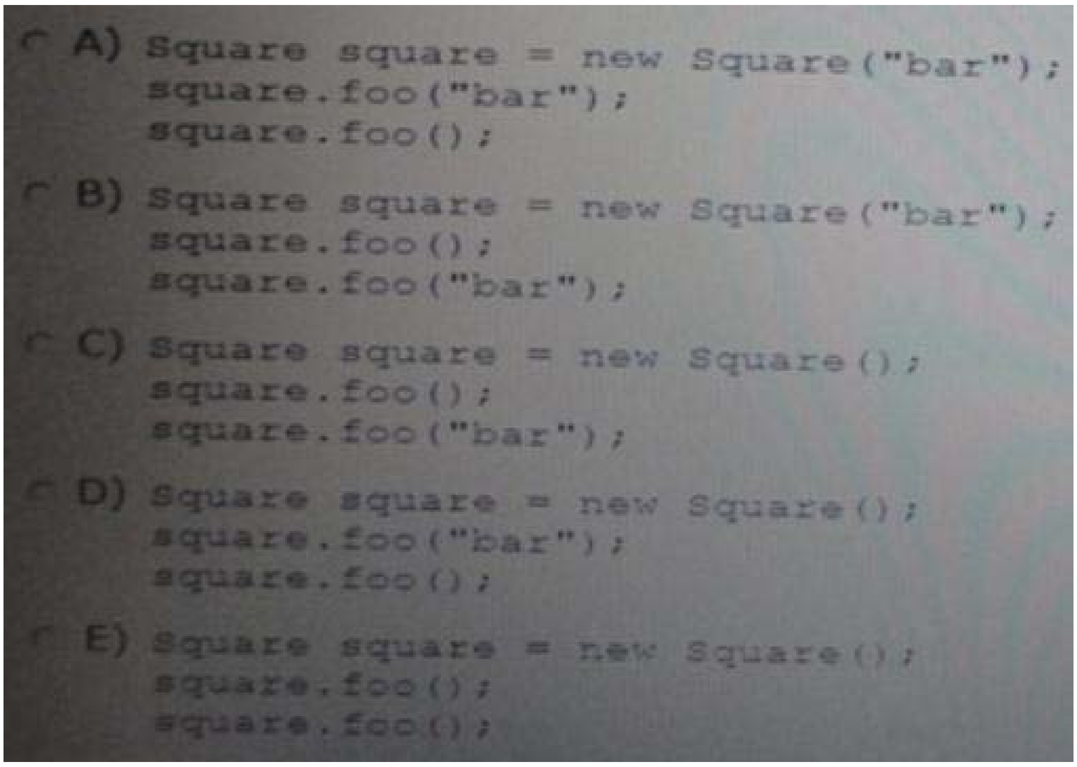
Answer: E

QUESTION: 84
Given:



```
public class SuperTest {
    public static void main(String args[]) {
        statement1
        statement2
        statement3
    }
}

class Shape {
    public Shape() {
        System.out.println("Shape: constructor");
    }
    public void foo() {
        System.out.println("Shape: foo");
    }
}

class Square extends Shape {
    public Square() {
        super();
    }
    public Square(String label) {
        System.out.println("Square: constructor");
    }
    public void foo() {
        super.foo();
    }
    public void foo(String label) {
        System.out.println("Square: foo");
```

What should statement1, statement2, and statement3, be respectively, in order to produce the result? Shape: constructor Square: foo Shape: foo

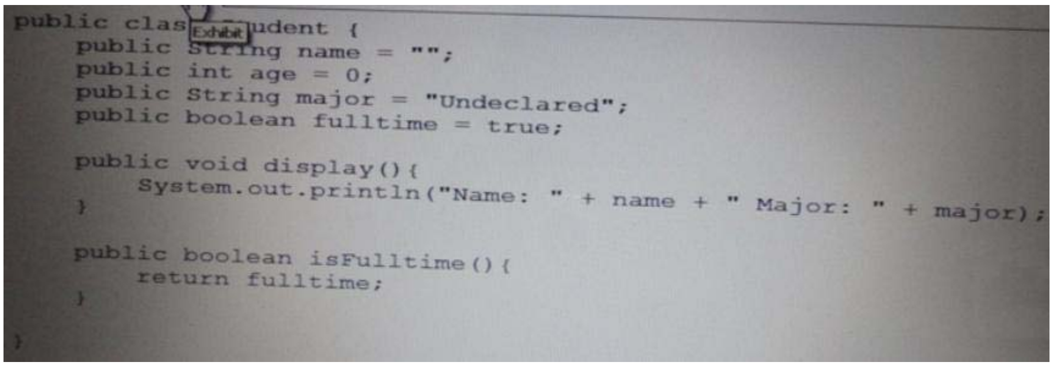


```
A) Square square = new Square("bar");
   square.foo("bar");
   square.foo();

B) Square square = new Square("bar");
   square.foo();
   square.foo("bar");

C) Square square = new Square();
   square.foo();
   square.foo("bar");

D) Square square = new Square();
   square.foo("bar");
   square.foo();

E) Square square = new Square();
   square.foo();
   square.foo();
```

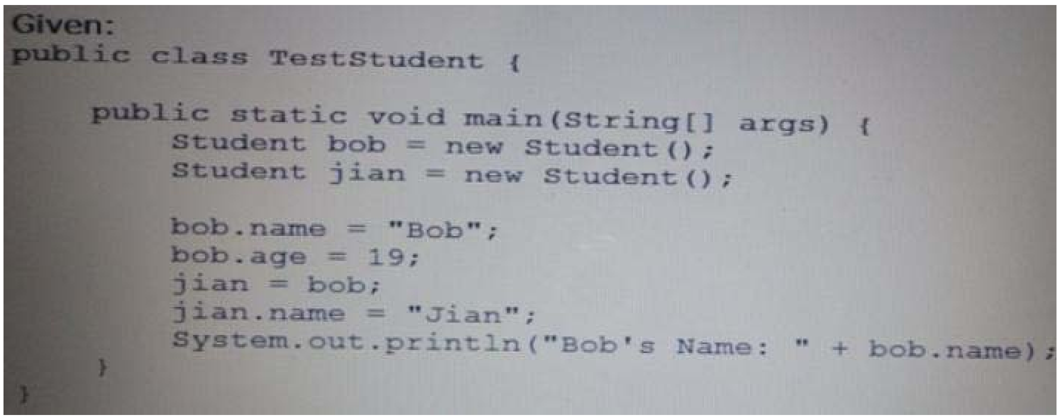
A. Option A
B. Option B
C. Option C
D. Option D
E. Option E

Answer: D

QUESTION: 85
View the exhibit:



```
public class Student {
    public String name = "";
    public int age = 0;
    public String major = "Undeclared";
    public boolean fulltime = true;

    public void display(){
        System.out.println("Name: " + name + " Major: " + major);
    }

    public boolean isFulltime(){
        return fulltime;
    }
}
```



```
Given:
public class TestStudent {

    public static void main(String[] args) {
        Student bob = new Student();
        Student jian = new Student();

        bob.name = "Bob";
        bob.age = 19;
        jian = bob;
        jian.name = "Jian";
        System.out.println("Bob's Name: " + bob.name);
    }
}
```

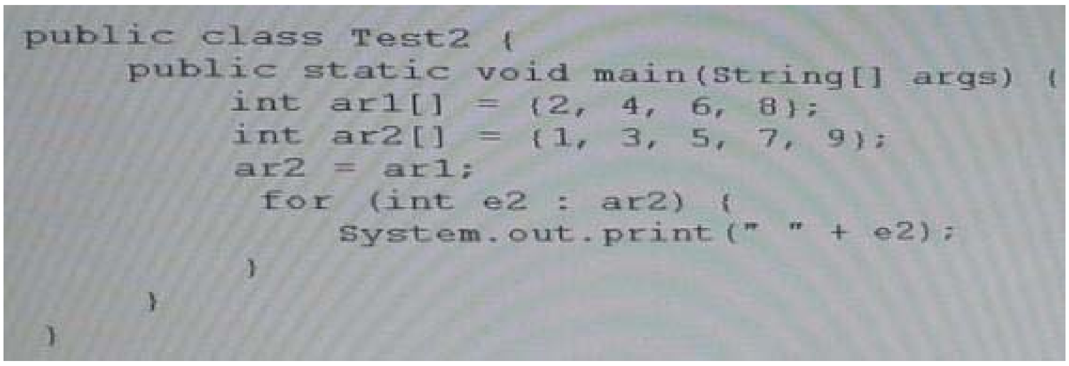
What is the result when this program is executed?

A. Bob's Name: Bob
B. Bob's Name: Jian
C. Nothing prints
D. Bob's name

Answer: B

Explanation:
After the statement jian = bob; the jian will reference the same object as bob.

QUESTION: 86
Given:



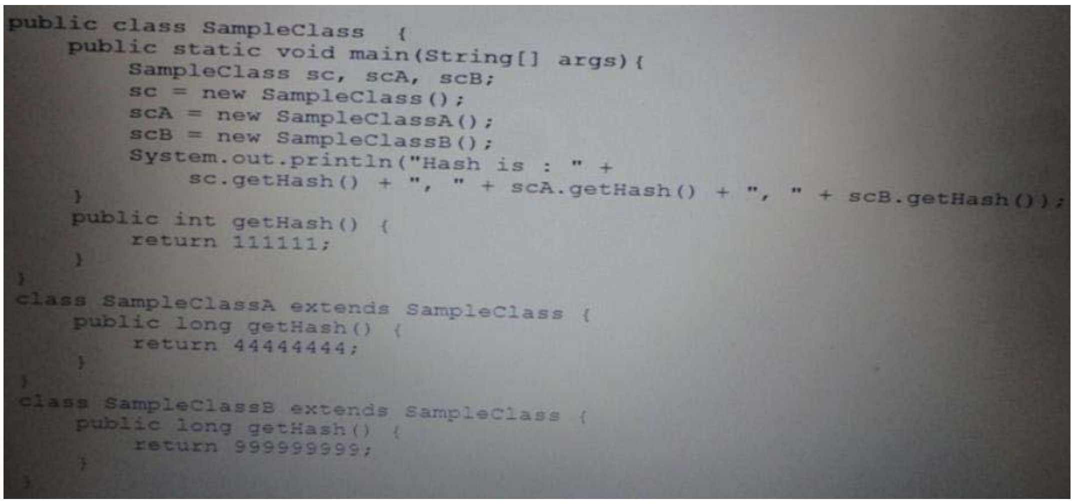
```
public class Test2 {
    public static void main(String[] args) {
        int ar1[] = {2, 4, 6, 8};
        int ar2[] = {1, 3, 5, 7, 9};
        ar2 = ar1;
         for (int e2 : ar2) {
            System.out.print(" " + e2);
        }
    }
}
```

What is the result?

A. 2 4 6 8
B. 2 4 6 8 9
C. 1 3 5 7
D. 1 3 5 7 9

Answer: D

QUESTION: 87
Given:



```
public class SampleClass   {
    public static void main(String[] args){
        SampleClass sc, scA, scB;
        sc = new SampleClass();
        scA = new SampleClassA();
        scB = new SampleClassB();
        System.out.println("Hash is : " +
            sc.getHash() + ", " + scA.getHash() + ", " + scB.getHash());
    }
    public int getHash() {
        return 111111;
    }
}
class SampleClassA extends SampleClass {
    public long getHash() {
        return 44444444;
    }
}
class SampleClassB extends SampleClass {
    public long getHash() {
        return 999999999;
    }
}
```

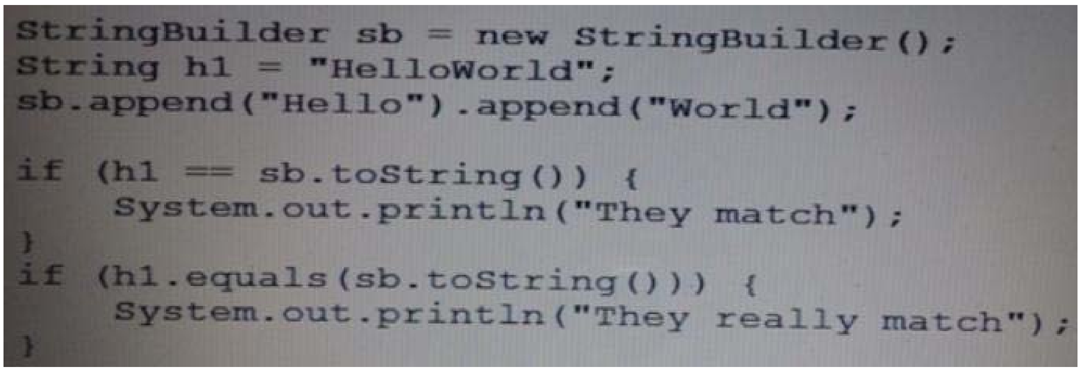
What is the result? A.

Compilation fails
B. An exception is thrown at runtime
C. There is no result because this is not correct way to determine the hash code
D. Hash is: 111111, 44444444, 999999999

Answer: A

Explanation:
The compilation fails as SampleClassA and SampleClassB cannot override SampleClass because the return type of SampleClass is int, while the return type of SampleClassA and SampleClassB is long.
Note: If all three classes had the same return type the output would be: Hash is : 111111, 44444444, 999999999

QUESTION: 88
Given a code fragment:



```
StringBuilder sb = new StringBuilder();
String h1 = "HelloWorld";
sb.append("Hello").append("World");

if (h1 == sb.toString()) {
    System.out.println("They match");
}
if (h1.equals(sb.toString())) {
    System.out.println("They really match");
}
```

What is the result?

A. They match They real match
B. They really match
C. They match
D. Nothing is printed to the screen

Answer: B

QUESTION: 89
Which three are bad practices?

A. Checking for ArrayIndexoutofBoundsException when iterating through an array to determine when all elements have been visited
B. Checking for Error and. If necessary, restarting the program to ensure that users are unaware problems
C. Checking for FileNotFoundException to inform a user that a filename entered is not valid
D. Checking for ArrayIndexoutofBoundsException and ensuring that the program can recover if one occur
E. Checking for an IOException and ensuring that the program can recover if one occurs

Answer: A, B, D

QUESTION: 90
Given the code fragment:
String[] cartoons = {"tom","jerry","micky","tom"}; int counter =0;
if ("tom".equals(cartoons[0])) { counter++;
} else if ("tom".equals(cartoons[1])) { counter++;
} else if ("tom".equals(cartoons[2])) { counter++;
} else if ("tom".equals(cartoons[3])) { counter++;
}
System.out.print(counter); What is the result?

A. 1
B. 2
C. 4
D. 0

Answer: A

Explanation:
Counter++ will be executed only once because of the else if constructs.

QUESTION: 91
Which statement is true about the default constructor of a top-level class?

A. It can take arguments.
B. It has private access modifier in its declaration.
C. It can be overloaded.
D. The default constructor of a subclass always invokes the no-argument constructor of its superclass.

Answer: D

Explanation:
In both Java and C#, a "default constructor" refers to a nullary constructor that is automatically generated by the compiler if no constructors have been defined for the class. The default constructor is also empty, meaning that it does nothing. A programmer- defined constructor that takes no parameters is also called a default constructor.

QUESTION: 92
Given the fragment:
String[][] arra = new String[3][]; arra[0] = new String[]{"rose", "lily"};
arra[1] = new String[]{"apple", "berry","cherry","grapes"};
arra[0] = new String[]{"beans", "carrot","potato"};
// insert code fragment here
Which code fragment when inserted at line '// insert code fragment here', enables the code to successfully change arra elements to uppercase?

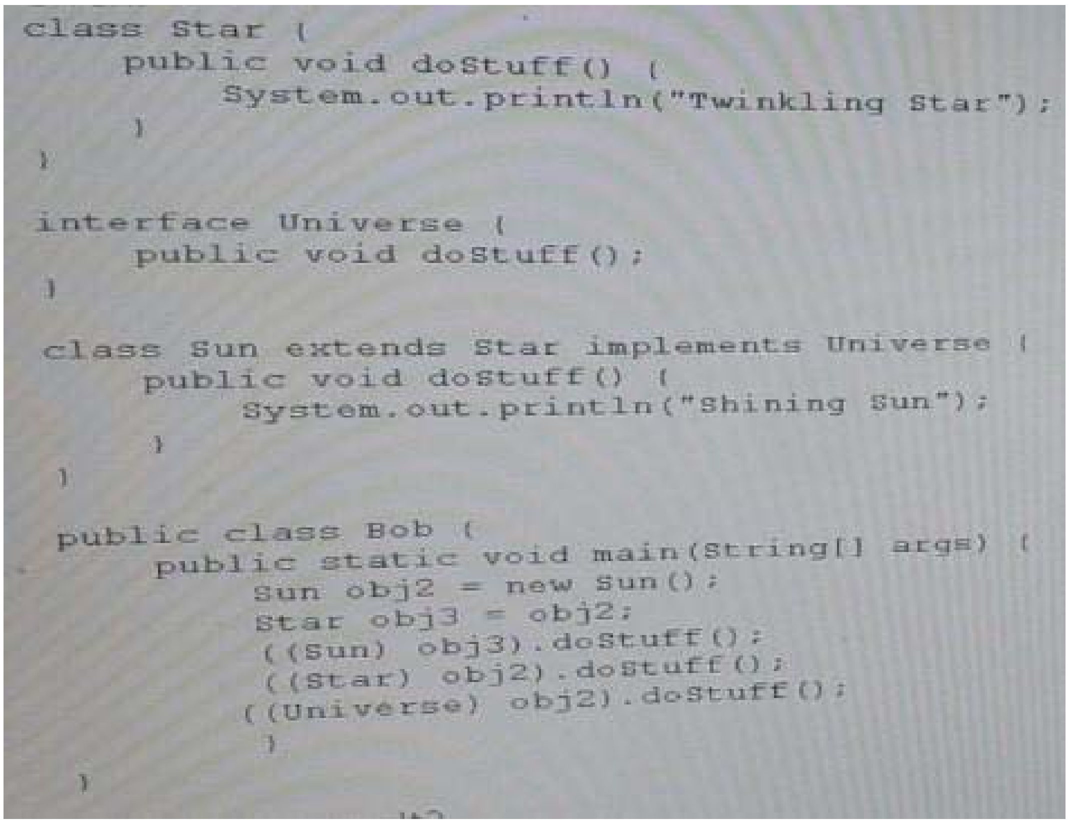
A. String[][] arra = new String[3][]; arra[0] = new String[]{"rose", "lily"};
arra[1] = new String[]{"apple", "berry","cherry","grapes"};
arra[0] = new String[]{"beans", "carrot","potato"}; for (int i = 0; i < arra.length; i++) {
for (int j=0; j < arra[i].length; j++) { arra[i][j] = arra[i][j].toUpperCase();
}

}
B. for (int i = 0; i < 3; i++) { for (int j=0; j < 4; j++) {
arra[i][j] = arra[i][j].toUpperCase();
}
}
C. for (String a[]:arra[][]) { for (String x:a[])
{ D. toUpperCase();
}
}
E. for (int i:arra.length) { for (String x:arra) { arra[i].toUpperCase();
}
}

Answer: C

Explanation:
Incorrect:
not A: arra.length is 3, but the subarrays have 2, 3 and 4 elements. Index will be out of bound.
not B: The subarrys are of different lengths. Index will be out of bound. not D: Compile error.

QUESTION: 93
Given:



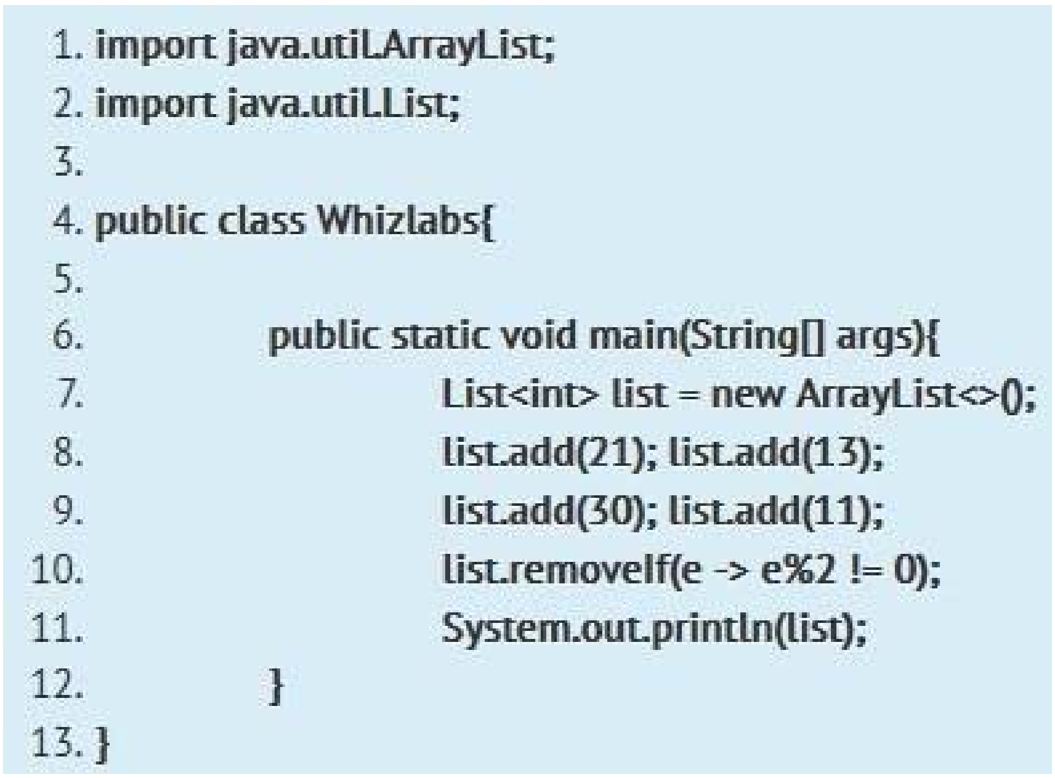
```
class Star {
    public void doStuff() {
        System.out.println("Twinkling Star");
    }
}

interface Universe {
    public void doStuff();
}

class Sun extends Star implements Universe {
    public void doStuff() {
        System.out.println("Shining Sun");
    }
}

public class Bob {
    public static void main(String[] args) {
        Sun obj2 = new Sun();
        Star obj3 = obj2;
        ((Sun) obj3).doStuff();
        ((Star) obj2).doStuff();
        ((Universe) obj2).doStuff();
    }
}
```

What is the result?

A. Shining Sun Shining Sun Shining Sun
B. Shining Sun Twinkling Star Shining Sun
C. Compilation fails
D. A ClassCastException is thrown at runtime

Answer: D

QUESTION: 94
Given:



```
1. import java.util.ArrayList;
2. import java.util.List;
3.
4. public class Whizlabs{
5.
6.              public static void main(String[] args){
7.                           List<int> list = new ArrayList<>();
8.                           list.add(21); list.add(13);
9.                           list.add(30); list.add(11);
10.                          list.removeIf(e -> e%2 != 0);
11.                          System.out.println(list);
12.             }
13. }
```

What is the output?

A. [21, 13, 11]
B. [30]
C. []
D. Compilation fails due to error at line 7
E. Compilation tails due to error at line 10

Answer: D

Explanation:
Option D is the correct answer. Code fails to compile as we can't use primitive for collections type, so in this code trying to use int at line 7, causes a compile error. We should have use wrapper. Integer there. So option D is correct.
https://docs.oracle.eom/javase/8/docs/api/java/util/ArrayList.html

QUESTION: 95
Given the code fragment: List colors = new ArrayList();
colors.add("green"); colors.add("red");
colors.add("blue"); colors.add("yellow"); colors.remove(2); colors.add(3,"cyan");
System.out.print(colors);
What is the result?

A. [green, red, yellow, cyan]
B. [green, blue, yellow, cyan]
C. [green, red, cyan, yellow]
D. Am IndexOutOfBoundsException is thrown at runtime

Answer: A

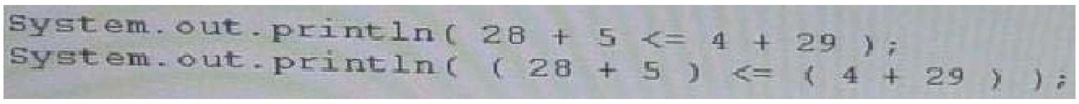
Explanation:
First the list [green, red, blue, yellow] is build. The blue element is removed:
[green, red, yellow] Finally the element cyan is added at then end of the list (index 3). [green, red, yellow, cyan]

QUESTION: 96
Given the code fragment?
public class Test {
public static void main(String[] args) { Test t = new Test();
int[] arr = new int[10]; arr = t.subArray(arr,0,2);
}
// insert code here
}
Which method can be inserted at line // insert code here to enable the code to compile?

A. public int[] subArray(int[] src, int start, int end) { return src;
}
B. public int subArray(int src, int start, int end) { return src;
}
C. public int[] subArray(int src, int start, int end) { return src;
}
D. public int subArray(int[] src, int start, int end) { return src;
}

Answer: A

QUESTION: 97
Given the code fragment:



```
System.out.println( 28 + 5 <= 4 + 29 );
System.out.println( ( 28 + 5 ) <= ( 4 + 29 ) );
```

What is the result?

A. 28false29 true
B. 285 < 429 true
C. true true
D. compilation fails

Answer: C

QUESTION: 98
The catch clause argument is always of type .

A. Exception
B. Exception but NOT including RuntimeException
C. Throwable
D. RuntimeException
E. CheckedException
F. Error

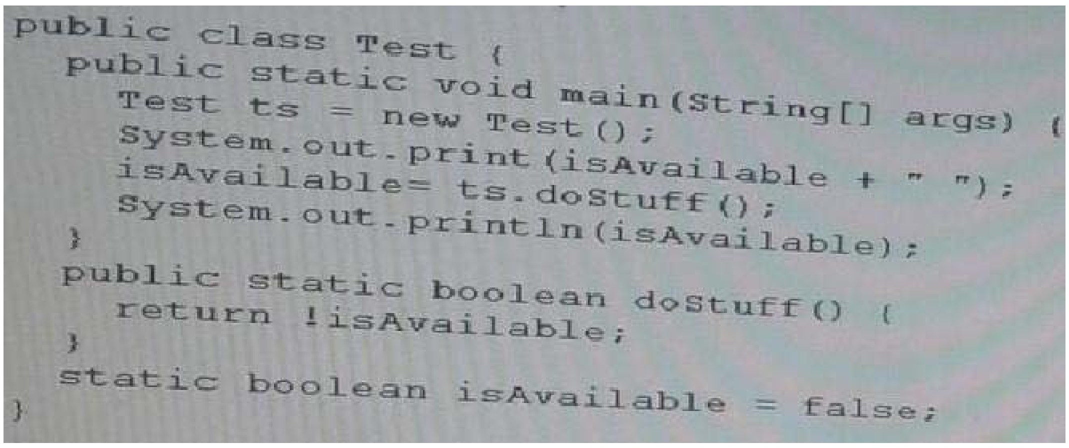
Answer: C

Explanation:
Because all exceptions in Java are the sub-class ofjava.lang.Exception class, you can have a single catch block that catches an exception of type Exception only. Hence the compiler is fooled into thinking that this block can handle any exception.
See the following example:

```
try
{
// ...
}
catch(Exception ex)
{
// Exception handling code for ANY exception
```

}
You can also use the java.lang.Throwable class here, since Throwable is the parent class for the application-specific Exception classes. However, this is discouraged in Java programming circles. This is because Throwable happens to also be the parent class for the non-application specific Error classes which are not meant to be handled explicitly as they are catered for by the JVM itself.
Note: The Throwable class is the superclass of all errors and exceptions in the Java language. Only objects that are instances of this class (or one of its subclasses) are thrown by the Java Virtual Machine or can be thrown by the Java throw statement.
A throwable contains a snapshot of the execution stack of its thread at the time it was created. It can also contain a message string that gives more information about the error.

QUESTION: 99
Given:



```
public class Test {
    public static void main(String[] args) {
        Test ts = new Test();
        System.out.print(isAvailable + " ");
        isAvailable= ts.doStuff();
        System.out.println(isAvailable);
    }
    public static boolean doStuff() {
        return !isAvailable;
    }
    static boolean isAvailable = false;
}
```

What is the result?

A. true true
B. true false
C. false true
D. false false
E. Compilation fails

Answer: E

QUESTION: 100
Given:
public class MainMethod { void main()
{ System.out.println("one");
}
static void main(String args) { System.out.println("two");
}

```
public static void main(String[] args) { System.out.println("three");
}
oid mina(Object[] args) { System.out.println("four");
}
}
```

What is printed out when the program is excuted?

A. one
B. two
C. three
D. four

Answer: C