

**QUESTION: 1**

Given the code fragment  
`int var1 = -5;  
int var2 = var1--; int var3 = 0;  
if (var2 < 0) { var3 = var2++;  
} else {  
var3 = --var2;  
}  
System.out.println(var3);`  
What is the result?

- A. - 6
- B. - 4
- C. - 5
- D. 5
- E. 4
- F. Compilation fails

**Answer: C**

**QUESTION: 2**

Given:  
`public class FieldInit { char c;  
boolean b; float f;  
void printAll() { System.out.println("c = " + c); System.out.println("c = " + b);  
System.out.println("c = " + f);  
}  
public static void main(String[] args) { FieldInit f = new FieldInit();  
f.printAll();  
}  
}`  
What is the result?

- A. c = null b = false  
f = 0.0F
- B. c = 0 b = false f = 0.0f
- C. c = null b = true  
f = 0.0
- D. c =  
b = false f = 0.0

**Answer: D**

**QUESTION: 3**

Given:

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

What is the reference type of myZ and what is the type of the object it references?

- A. Reference type is Z; object type is Z.
- B. Reference type is Y; object type is Y.
- C. Reference type is Z; object type is Y.
- D. Reference type is X; object type is Z.

**Answer:** C

**QUESTION:** 4

1. class StaticMethods {
2. static void one() {
3. two();
4. StaticMethods.two();
5. three();
6. StaticMethods.four(); 7. }
8. static void two() { }
9. void three() {
10. one();
11. StaticMethods.two();
12. four();
13. StaticMethods.four();
14. }
15. void four() { }
16. }

Which three lines are illegal?

- A. line 3
- B. line 4
- C. line 5

- D. line 6
- E. line 10
- F. line 11
- G. line 12
- H. line 13

**Answer:** C, D, H

**QUESTION: 5**

Given:

```
class Test {
    int sum = 0;
    public void doCheck(int number) {
        if (number % 2 == 0) {
            break;
        } else {
            for (int i = 0; i < number; i++) {
                sum += i;
            }
        }
    }
    public static void main(String[] args) {
        Test obj = new Test();
        System.out.println("Red " + obj.sum);
        obj.doCheck(2);
        System.out.println("Orange " + obj.sum);
        obj.doCheck(3);
        System.out.println("Green " + obj.sum);
    }
}
```

What is the result?

- A. Red 0 Orange 0 Green 3
- B. Red 0 Orange 0 Green 6
- C. Red 0 Orange 1
- D. Green 4
- E. Compilation fails

**Answer:** E

**QUESTION: 6**

View the exhibit.

```

class MissingInfoException extends Exception { }
class AgeOutOfRangeException extends Exception { }

class Candidate {
    String name;
    int age;
    Candidate(String name, int age) throws Exception {
        if (name == null) {
            throw new MissingInfoException();
        } else if (age <= 10 || age >= 150) {
            throw new AgeOutOfRangeException();
        } else {
            this.name = name;
            this.age = age;
        }
    }
    public String toString() {
        return name + " age: " + age;
    }
}

```

Given the code fragment:

```

4. public class Test {
5.     public static void main(String[] args) {
6.         Candidate c = new Candidate("James", 20);
7.         Candidate c1 = new Candidate("Williams", 32);
8.         System.out.println(c);
9.         System.out.println(c1);
10.    }
11. }

```

Which change enables the code to print the following? James age: 20 Williams age: 32

- A. Replacing line 5 with public static void main (String [] args) throws MissingInfoException, AgeOutOfRangeException {
- B. Replacing line 5 with public static void main (String [] args) throws.Exception {
- C. Enclosing line 6 and line 7 within a try block and adding: catch(Exception e1) { //code goes here}  
catch (missingInfoException e2) { //code goes here} catch (AgeOutOfRangeException e3) { //code goes here}
- D. Enclosing line 6 and line 7 within a try block and adding: catch (missingInfoException e2) { //code goes here}  
catch (AgeOutOfRangeException e3) { //code goes here}

**Answer:** C

### QUESTION: 7

Given the code fragment:

```

int a = 0; a++;
System.out.println(a++); System.out.println(a);

```

What is the result?

- A.12
- B.01
- C.11
- D.22

**Answer:** A

**Explanation:**

The first println prints variable a with value 1 and then increases the variable to 2.

**QUESTION: 8**

Given:

```
public class Test {  
    public static void main(String[] args) { int arr[] = new int[4];  
        arr[0] = 1;  
        arr[1] = 2;  
        arr[2] = 4;  
        arr[3] = 5; int sum = 0; try {  
            for (int pos = 0; pos <= 4; pos++) { sum = sum + arr[pos];  
            }  
        } catch (Exception e) { System.out.println("Invalid index");  
        }  
        System.out.println(sum);  
    }  
}
```

What is the result?

- A. 12
- B. Invalid Index 12
- C. Invalid Index
- D. Compilation fails

**Answer:** B

**Explanation:**

The loop ( for (int pos = 0; pos <= 4; pos++) { }, it should be pos <= 3, causes an exception, which is caught. Then the correct sum is printed.

**QUESTION: 9**

```
int [] array = {1,2,3,4,5}; for (int i: array) {  
    if ( i < 2) { keyword1 ;
```

```
}  
System.out.println(i); if ( i == 3) { keyword2 ;  
}}
```

What should keyword1 and keyword2 be respectively, in order to produce output 2345?

- A. continue, break
- B. break, break
- C. break, continue
- D. continue, continue

**Answer:** D

**QUESTION:** 10

Given:

```
class X {  
    static void m(int i) {  
        i += 7;  
    }  
    public static void main(String[] args) {  
        int j = 12;  
        m(j);  
        System.out.println(j);  
    }  
}
```

What is the result?

- A. 7
- B. 12
- C. 19
- D. Compilation fails
- E. An exception is thrown at run time

**Answer:** B

**QUESTION:** 11

Given:

```
package p1;  
public interface DoInterface { void method1(int n1); // line n1  
}  
package p3;  
import p1.DoInterface;
```

```

public class DoClass implements DoInterface { public DoClass(int p1) { }
public void method1(int p1) { } // line n2 private void method2(int p1) { } // line n3
}
public class Test {
public static void main(String[] args) { DoInterface doi= new DoClass(100); // line n4
doi.method1(100);
doi.method2(100);
}
}

```

Which change will enable the code to compile?

- A. Adding the public modifier to the declaration of method1 at line n1
- B. Removing the public modifier from the definition of method1 at line n2
- C. Changing the private modifier on the declaration of method 2 public at line n3
- D. Changing the line n4 DoClass doi = new DoClass ( );

**Answer:** C

**Explanation:**

Private members (both fields and methods) are only accessible inside the class they are declared or inside inner classes. private keyword is one of four access modifier provided by Java and its a most restrictive among all four e.g. public, default(package), protected and private.

[Read more: http://javarevisited.blogspot.com/2012/03/private-in-java-why-should-you-always.html#ixzz3Sh3mOc4D](http://javarevisited.blogspot.com/2012/03/private-in-java-why-should-you-always.html#ixzz3Sh3mOc4D)

**QUESTION:** 12

Consider

Integer number = Integer.valueOf("808.1"); Which is true about the above statement?

- A. The value of the variable number will be 808.1
- B. The value of the variable number will be 808
- C. The value of the variable number will be 0.
- D. A NumberFormatException will be throw.
- E. It will not compile.

**Answer:** D

**Explanation:**

The Integer class valueOf() returns an Integer from given string. But we need to pass string which has correct format for integer otherwise it will throw a NumberFormatException. In this case we have passed string which is not an integer value (since what we passed is fractional number), so option D is correct.

**QUESTION: 13**

Given the code format:

```
class DBConfiguration {
    String user;
    String password;
}

And:

4. public class DBHandler {
5.     DBConfiguration configureDB(String uname, String password) {
6.         // insert code here
7.     }
8.     public static void main(String[] args) {
9.         DBHandler r = new DBHandler();
10.        DBConfiguration dbConf = r.configureDB("manager", "manager");
11.    }
12. }
```

Which code fragment must be inserted at line 6 to enable the code to compile?

- A. DBConfiguration f; return f;
- B. Return DBConfiguration;
- C. Return new DBConfiguration;
- D. Return 0;

**Answer: B**

**QUESTION: 14**

Given:

```
class Base {
// insert code here
}

public class Derived extends Base{ public static void main(String[] args) { Derived obj =
new Derived(); obj.setNum(3);
System.out.println("Square = " + obj.getNum() * obj.getNum());
}
```

Which two options, when inserted independently inside class Base, ensure that the class is being properly encapsulated and allow the program to execute and print the square of the number?

- A. private int num; public int getNum() { return num; } public void setNum(int num) { this.num = num; }
- B. public int num; protected public int getNum() { return num; } protected public void setNum(int num) { this.num = num; }
- C. private int num; public int getNum() { return num; } private void setNum(int num) { this.num = num; }
- D. protected int num; public int getNum() { return num; } public void setNum(int num) { this.num = num; }



E. `protected int num; private int getNum() { return num; } public void setNum(int num) { this.num = num; }`

**Answer:** A, D

**Explanation:**

Incorrect:

Not B: illegal combination of modifiers: protected and public not C: setNum method cannot be private.

not E: getNum method cannot be private.

**QUESTION: 15**

Given the code fragment:

```
class Student {
    String name;
    int age;
}

And,

1. public class Test {
2.     public static void main(String[] args) {
3.         Student s1 = new Student();
4.         Student s2 = new Student();
5.         Student s3 = new Student();
6.         s1 = s3;
7.         s3 = s2;
8.         s2 = null;
9.     }
10. }
```

Which statement is true?

- A. After line 8, three objects are eligible for garbage collection
- B. After line 8, two objects are eligible for garbage collection
- C. After line 8, one object is eligible for garbage collection
- D. After line 8, none of the objects are eligible for garbage collection

**Answer:** C

**QUESTION: 16**

Given:

```

public class App {
    public static void main(String[] args) {
        int i = 10;
        int j = 20;
        int k = j += i / 5;
        System.out.print(i + " : " + j + " : " + k);
    }
}

```

What is the result?

- A. 10 : 22 : 20
- B. 10 : 22 : 22
- C. 10 : 22 : 6
- D. 10 : 30 : 6

**Answer:** B

**QUESTION:** 17

Given:

```

class Sports { int num_players;
String name, ground_condition;
Sports(int np, String sname, String sground){ num_players = np;
name = sname; ground_condition = sground;
}
}

```

```

class Cricket extends Sports { int num_umpires;
int num_substitutes;

```

Which code fragment can be inserted at line //insert code here to enable the code to compile?

- A. Cricket() {  
super(11, "Cricket", "Condition OK"); num\_umpires =3; num\_substitutes=2;  
}
- B. Cricket() {  
super.ground\_condition = "Condition OK"; super.name="Cricket"; super.num\_players = 11;  
num\_umpires =3; num\_substitutes=2;  
}
- C. Cricket() { this(3,2);  
super(11, "Cricket", "Condition OK");  
}  
Cricket(int nu, ns) { this.num\_umpires =nu; this.num\_substitutes=ns;  
}
- D. Cricket() { this.num\_umpires =3; this.num\_substitutes=2;  
super(11, "Cricket", "Condition OK");  
}

**Answer:** A

**Explanation:**

Incorrect:

not C, not D: call to super must be the first statement in constructor.

**QUESTION: 18**

Given:

```
public class MyFor {  
    public static void main(String[] args) { for (int ii = 0; ii < 4; ii++) { System.out.println("ii =  
    "+ ii);  
    ii = ii +1;  
    }  
    }  
}
```

What is the result?

- A. ii = 0 ii = 2
- B. ii = 0 ii = 1 ii = 2 ii = 3
- C. ii =
- D. Compilation fails.

**Answer:** A

**QUESTION: 19**

Given the code fragment: System.out.println("Result: " + 2 + 3 + 5); System.out.println("Result: " + 2 + 3 \* 5); What is the result?

- A. Result: 10 Result: 30
- B. Result: 10 Result: 25
- C. Result: 235 Result: 215
- D. Result: 215 Result: 215
- E. Compilation fails

**Answer:** C

**Explanation:**

First line:

System.out.println("Result: " + 2 + 3 + 5); String concatenation is produced.

Second line:

System.out.println("Result: " + 2 + 3 \* 5);  
3\*5 is calculated to 15 and is appended to string 2. Result 215.  
The output is: Result: 235  
Result: 215

Note #1:

To produce an arithmetic result, the following code would have to be used:

System.out.println("Result: " + (2 + 3 + 5));  
System.out.println("Result: " + (2 + 1 \* 5)); run:  
Result: 10

Result: 7

Note #2:

If the code was as follows:

System.out.println("Result: " + 2 + 3 + 5");  
System.out.println("Result: " + 2 + 1 \* 5");  
The compilation would fail. There is an unclosed string literal, 5", on each line.

### QUESTION: 20

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<Integer> list = new ArrayList<>();
8.         list.add(21); list.add(13);
9.         list.add(30); list.add(11);
10.        list.add(2);
11.        //insert here
12.        System.out.println(list);
13.    }
14. }
```

Which inserted at line 11, will provide the following output? [21, 15, 11]

- A. list.removeIf(e -> e%2 != 0);
- B. list.removeIf(e -> e%2 == 0);

- C. `Ust.removeIf(e -> e%2 == 0);`
- D. `list.remove(e -> e%2 == 0);`
- E. None of the above.

**Answer:** C

**Explanation:**

In output we can see that only odd numbers present, so we need to remove only even numbers to get expected output. From Java SE 8, there is new method call `removeIf` which takes predicate object and remove elements which satisfies predicate condition. Predicate has functional method call `take` object and check if the given condition met or not, if met it returns true, otherwise false. Option C we have passed correct lambda expression to check whether the number is odd or even that matches to the functional method of predicate interface. Option A is incorrect as it is invalid lambda expression. Option B is incorrect as it removes all odd numbers. Option D is incorrect as there is no `remove` method that takes predicate as argument.

<https://docs.oracle.com/javase/8/docs/api/java/util/ArrayList.html>

**QUESTION: 21**

Given the class definitions:

```
class Alpha {
    public String doStuff(String msg) {
        return msg;
    }
}
class Beta extends Alpha {
    public String doStuff(String msg) {
        return msg.replace('a', 'e');
    }
}
class Gamma extends Beta {
    public String doStuff(String msg) {
        return msg.substring(2);
    }
}
```

And the code fragment of the `main()` method,

```
12. List<Alpha> strs = new ArrayList<Alpha>();
13. strs.add(new Alpha());
14. strs.add(new Beta());
15. strs.add(new Gamma());
16. for (Alpha t : strs) {
17.     System.out.println(t.doStuff("Java"));
18. }
```

What is the result?

- A. Java Java Java
- B. Java Jeve va
- C. Java Jeve ve
- D. Compilation fails

**Answer:** D

**QUESTION:** 22

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 (); bob.name = "Bob";
bob.age = 18;
bob.year = 1982;
}
}
```

What is the result?

- A. year is set to 1982.
- B. bob.year is set to 1982
- C. A runtime error is generated.
- D. A compile time error is generated.

**Answer:** D

**QUESTION:** 23

Given the code fragment:

```

public static void main(String[] args) {
    ArrayList<String> list = new ArrayList<>();

    list.add("SE");
    list.add("EE");
    list.add("ME");
    list.add("SE");
    list.add("EE");

    list.remove("SE");

    System.out.print("Values are : " + list);
}

```

What is the result?

- A. Values are : [EE, ME]
- B. Values are : [EE, EE, ME]
- C. Values are : [EE, ME, EE]
- D. Values are : [SE, EE, ME, EE]
- E. Values are : [EE, ME, SE, EE]

**Answer:** E

**QUESTION:** 24

Given:

```

public class MyClass {
    public static void main(String[] args) { while (int ii = 0; ii < 2) {
        ii++;
        System.out.println("ii = " + ii);
    }
}

```

What is the result?

- A. ii = 1 ii = 2
- B. Compilation fails
- C. The program prints nothing
- D. The program goes into an infinite loop with no output
- E. The program goes to an infinite loop outputting: ii = 1 ii = 1

**Answer:** B

**Explanation:**

The while statement is incorrect. It has the syntax of a for statement.

The while statement continually executes a block of statements while a particular condition is

true. Its syntax can be expressed as:

```
while (expression) { statement(s) }
```

The while statement evaluates expression, which must return a boolean value. If the expression evaluates to true, the while statement executes the statement(s) in the while block. The while statement continues testing the expression and executing its block until the expression evaluates to false.

### Reference:

The while and do-while Statements

### QUESTION: 25

Given:

```
class Base {  
    public static void main(String[] args) { System.out.println("Base " + args[2]);  
    }  
}  
public class Sub extends Base{  
    public static void main(String[] args) { System.out.println("Overridden " + args[1]);  
    }  
}
```

And the commands: javac Sub.java  
java Sub 10 20 30 What is the result?

- A. Base 30
- B. Overridden 20
- C. Overridden 20 Base 30
- D. Base 30 Overridden 20

**Answer: B**

### QUESTION: 26

Given:

```
public class ScopeTest { int j, int k;  
    public static void main(String[] args) { ew ScopeTest().doStuff(); }  
    void doStuff() { nt x = 5; oStuff2();  
        System.out.println("x");  
    }  
    void doStuff2() { nt y = 7;  
        ystem.out.println("y");  
        or (int z = 0; z < 5; z++) { ystem.out.println("z");  
        ystem.out.println("y");  
    }  
}
```

Which two items are fields?



- A. j
- B. k
- C. x
- D. y
- E. z

**Answer:** A, B

**QUESTION: 27**

Given the code fragment:

```
9.    int a = -10;
10.   int b = 17;
11.   int c = expression1;
12.   int d = expression2;
13.   c++;
14.   d--;
15.   System.out.print(c + ", " + d);
```

What could expression1 and expression2 be, respectively, in order to produce output -8, 16?

- A. ++a, --b
- B. ++a, b--
- C. ++a, --b
- D. ++a, b--

**Answer:** D

**QUESTION: 28**

An unchecked exception occurs in a method dosomething(). Should other code be added in the dosomething() method for it to compile and execute?

- A. The Exception must be caught
- B. The Exception must be declared to be thrown.
- C. The Exception must be caught or declared to be thrown.
- D. No other code needs to be added.

**Answer:** D

**Explanation:**

Because the Java programming language does not require methods to catch or to specify unchecked exceptions (RuntimeException, Error, and their subclasses), programmers may be tempted to write code that throws only unchecked exceptions or to make all their exception subclasses inherit from RuntimeException. Both of these shortcuts allow programmers to write code without bothering with compiler errors and without bothering to specify or to catch any exceptions. Although this may seem convenient to the programmer, it sidesteps the intent of the catch or specify requirement and can cause problems for others using your classes.

**QUESTION: 29**

Given the code fragment:

```
String[] colors = {"red", "blue", "green", "yellow", "maroon", "cyan"};
```

Which code fragment prints blue, cyan, ?

```
C A) for (String c:colors) {  
    if (c.length() != 4) {  
        continue;  
    }  
    System.out.print(c+", ");  
}  
  
C B) for (String c:colors[]) {  
    if (c.length() <= 4) {  
        continue;  
    }  
    System.out.print(c+", ");  
}  
  
C C) for (String c:String[] colors) {  
    if (c.length() >= 3) {  
        continue;  
    }  
    System.out.print(c+", ");  
}  
  
C D) for (String c:colors){  
    if (c.length() != 4) {  
        System.out.print(c+", ");  
        continue;  
    }  
}
```

- A. Option A
- B. Option B
- C. Option C
- D. Option D

**Answer:** A

**QUESTION: 30**

Given:

```
public class Series {  
    public static void main(String[] args) {  
        int arr[] = {1, 2, 3};  
  
        for (int var : arr) {  
            int i = 1;  
            while (i <= var);  
            System.out.println(i++);  
        }  
    }  
}
```

What is the result?

- A. 1 1 1
- B. 1 2 3
- C. 2 3 4
- D. Compilation fails
- E. The loop executes infinite times

**Answer:** E

**QUESTION: 31**

Given:

```
public class Case {  
    public static void main(String[] args) {  
        String product = "Pen";  
        product.toLowerCase();  
        product.concat(" BOX".toLowerCase());  
        System.out.print(product.substring(4, 6));  
    }  
}
```

What is the result?

- A. box
- B. nbo
- C. bo
- D. nb
- E. An exception is thrown at runtime

**Answer:** E

**QUESTION:** 32

Given the code fragment:

```
12. int row = 10;
13. for ( ; row > 0 ; ) {
14.     int col = row;
15.     while (col >= 0) {
16.         System.out.print(col + " ");
17.         col -= 2;
18.     }
19.     row = row / col;
20. }
```

What is the result?

- A. 10 8 6 4 2 0
- B. 10 8 6 4 2
- C. AnArithmeticException is thrown at runtime
- D. The program goes into an infinite loop outputting: 10 8 6 4 2 0. . .
- E. Compilation fails

**Answer:** B

**QUESTION:** 33

Which two statements are true for a two-dimensional array of primitive data type?

- A. It cannot contain elements of different types.
- B. The length of each dimension must be the same.
- C. At the declaration time, the number of elements of the array in each dimension must be specified.
- D. All methods of the class object may be invoked on the two-dimensional array.

**Answer:** C, D

**Explanation:**

<http://stackoverflow.com/questions/12806739/is-an-array-a-primitive-type-or-something-else-entirely> an-object-or-

**QUESTION: 34**

A method is declared to take three arguments. A program calls this method and passes only two arguments. What is the result?

- A. Compilation fails.
- B. The third argument is given the value null.
- C. The third argument is given the value void.
- D. The third argument is given the value zero.
- E. The third argument is given the appropriate false value for its declared type.
- F. An exception occurs when the method attempts to access the third argument.

**Answer: A**

**Explanation:**

The problem is noticed at build/compile time. At build you would receive an error message like: required: int,int,int found: int,int

**QUESTION: 35**

Given:

```
public class DoWhile1 {  
    public static void main(String[] args) {  
        int ii = 2;  
        do {  
            System.out.println(ii);  
        } while (--ii);  
    }  
}
```

What is the result?

- A. 2 1
- B. 2 1 0
- C. null
- D. an infinite loop
- E. compilation fails

**Answer: E**

**Explanation:**

The line while (--ii); will cause the compilation to fail. ii is not a boolean value.  
A correct line would be while (--ii>0);

**QUESTION: 36**

Which three statements are benefits of encapsulation?

- A. Allows a class implementation to change without changing the clients
- B. Protects confidential data from leaking out of the objects
- C. Prevents code from causing exceptions
- D. Enables the class implementation to protect its invariants
- E. Permits classes to be combined into the same package
- F. Enables multiple instances of the same class to be created safely

**Answer:** A, B, D

**QUESTION: 37**

Given:

```
public class Test2 {
    public static void doChange(int[] arr) {
        for(int pos = 0; pos < arr.length; pos++){
            arr[pos] = arr[pos] + 1;
        }
    }
    public static void main(String[] args) {
        int[] arr = {10, 20, 30};
        doChange(arr);
        for(int x: arr) {
            System.out.print(x + ", ");
        }
        doChange(arr[0], arr[1], arr[2]);
        System.out.print(arr[0] + ", " + arr[1] + ", " + arr[2]);
    }
}
```

What is the result?

- A. 11, 21, 31, 11, 21, 31
- B. 11, 21, 31, 12, 22, 32
- C. 12, 22, 32, 12, 22, 32
- D. 10, 20, 30, 10, 20, 30

**Answer:** D

**QUESTION: 38**

Which of the following can fill in the blank in this code to make it compile?

```
interface CanFly{  
  
    String type = "A";  
  
    void fly();  
  
    ____ String getType(){  
        return type;  
    }  
}
```

- A. abstract
- B. public
- C. default
- D. It will not compile with any as interfaces cannot have non abstract methods.
- E. It will compile without filling the blank.

**Answer:** C

**Explanation:**

From Java SE 8, we can use static and/or default methods in interfaces, but they should be non abstract methods. SO in this case using default in blank is completely legal. Hence option C is correct.

Option A is incorrect as given method is not abstract, so can't use abstract there. Options B and E are incorrect as we can't have non abstract method interface if they are not default or static. <https://docs.oracle.com/javase/8/tutorial/java/and/defaultmethods.html>

**QUESTION:** 39

Given:



```

class SpecialException extends Exception {
    public SpecialException(String message) {
        super(message);
        System.out.println(message);
    }
}

public class ExceptionTest {
    public static void main(String[] args) {
        try {
            doSomething();
        }
        catch (SpecialException e) {
            System.out.println(e);
        }
    }

    static void doSomething() throws SpecialException {
        int[] ages = new int[4];
        ages[4] = 17;
        doSomethingElse();
    }

    static void doSomethingElse() throws SpecialException {
        throw new SpecialException("Thrown at end of doSomething() method");
    }
}

```

What will be the output?

```

C A) SpecialException: Thrown at end of doSomething() method
C B) Error in thread "main" java.lang.ArrayIndexOutOfBoundsException
C C) Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 4
    at ExceptionTest.doSomething(ExceptionTest.java:13)
    at ExceptionTest.main(ExceptionTest.java:4)
C D) SpecialException: Thrown at end of doSomething() method
    at ExceptionTest.doSomethingElse(ExceptionTest.java:16)
    at ExceptionTest.doSomething(ExceptionTest.java:13)
    at ExceptionTest.main(ExceptionTest.java:4)

```

- A. Option A
- B. Option B
- C. Option C
- D. Option D

**Answer:** D

**QUESTION:** 40

A method is declared to take three arguments. A program calls this method and passes only two arguments. What is the results?

- A. Compilation fails.
- B. The third argument is given the value null.
- C. The third argument is given the value void.
- D. The third argument is given the value zero.
- E. The third argument is given the appropriate falsy value for its declared type.
- F. An exception occurs when the method attempts to access the third argument.

**Answer:** A



**QUESTION: 41**

Given:

```
import java.io.IOException;

public class Y {
    public static void main(String[] args) {
        try {
            doSomething();
        }
        catch (RuntimeException e) {
            System.out.println(e);
        }
    }
    static void doSomething() {
        if (Math.random() > 0.5) throw new IOException();
        throw new RuntimeException();
    }
}
```

Which two actions, used independently, will permit this class to compile?

- A. Adding throws IOException to the main() method signature
- B. Adding throws IOException to the doSomething() method signature
- C. Adding throws IOException to the main() method signature and to the doSomething() method
- D. Adding throws IOException to the doSomething() method signature and changing the catch argument to IOException
- E. Adding throws IOException to the main() method signature and changing the catch argument to IOException

**Answer:** C, D

**Explanation:**

The IOException must be caught or be declared to be thrown.

We must add a throws exception to the doSomething () method signature (static void doSomething() throws IOException).

Then we can either add the same throws IOException to the main method (public static void main (String[] args) throws IOException), or change the catch statement in main to IOException.

**QUESTION: 42**

Given the following code fragment:

```

if (value >= 0) {
    if (value != 0)
        System.out.print("the ");
    else
        System.out.print("quick ");
    if (value < 10)
        System.out.print("brown ");
    if (value > 30)
        System.out.print("fox ");
    else if (value < 50)
        System.out.print("jumps ");
    else if (value < 10)
        System.out.print("over ");
    else
        System.out.print("the ");
    if (value > 10)
        System.out.print("lazy ");
} else {
    System.out.print("dog ");
}
System.out.println( "... " );

```

What is the result if the integer value is 33?

- A. The fox jump lazy ...
- B. The fox lazy ...
- C. Quick fox over lazy ...
- D. Quick fox the ....

**Answer:** B

**Explanation:**

33 is greater than 0.  
 33 is not equal to 0. the is printed.  
 33 is greater than 30 fox is printed  
 33 is greater then 10 (the two else if are skipped)  
 lazy is printed finally ... is printed.

**QUESTION:** 43

Which statement will empty the contents of a StringBuilder variable named sb?

- A. `sb.deleteAll();`
- B. `sb.delete(0, sb.size());`
- C. `sb.delete(0, sb.length());`
- D. `sb.removeAll();`

**Answer:** C

**QUESTION:** 44

Given the code fragment:

```
System.out.println(2 + 4 * 9 - 3); //Line 21
```

```
System.out.println((2 + 4) * 9 - 3); // Line 22
```

```
System.out.println(2 + (4 * 9) - 3); // Line 23
```

```
System.out.println(2 + 4 * (9 - 3)); // Line 24
```

```
System.out.println((2 + 4 * 9) - 3); // Line 25
```

Which line of codes prints the highest number?

- A. Line 21
- B. Line 22
- C. Line 23
- D. Line 24
- E. Line 25

**Answer:** B

**Explanation:**

The following is printed: 35

51

35

26

35

**QUESTION:** 45

Given the for loop construct:

```
for ( expr1 ; expr2 ; expr3 ) { statement;  
}
```

Which two statements are true?

- A. This is not the only valid for loop construct; there exists another form of for loop constructor.
- B. The expression `expr1` is optional. it initializes the loop and is evaluated once, as the loop begin.
- C. When `expr2` evaluates to false, the loop terminates. It is evaluated only after each iteration through the loop.

D. The expression `expr3` must be present. It is evaluated after each iteration through the loop.

**Answer:** B, C

**Explanation:**

The for statement have this forms: `for (init-stmt; condition; next-stmt) { body }`

There are three clauses in the for statement.

The init-stmt statement is done before the loop is started, usually to initialize an iteration variable.

The condition expression is tested before each time the loop is done. The loop isn't executed if the boolean expression is false (the same as the while loop).

The next-stmt statement is done after the body is executed. It typically increments an iteration variable.

**QUESTION:** 46

Given:

```
public class X {  
    public static void main(String[] args) {  
        String theString = "Hello World";  
        System.out.println(theString.charAt(11));  
    }  
}
```

What is the result?

- A. The program prints nothing
- B. d
- C. A `StringIndexOutOfBoundsException` is thrown at runtime.
- D. An `ArrayIndexOutOfBoundsException` is thrown at runtime.
- E. A `NullPointerException` is thrown at runtime.

**Answer:** C

**QUESTION:** 47

Given:

```

public class CharToStr {
    public static void main(String[] args) {
        String str1 = "Java";
        char str2[] = { 'J', 'a', 'v', 'a' };
        String str3 = null;
        for (char c : str2) {
            str3 = str3 + c;
        }
        if (str1.equals(str3))
            System.out.print("Successful");
        else
            System.out.print("Unsuccessful");
    }
}

```

What is result?

- A. Successful
- B. Unsuccessful
- C. Compilation fails
- D. An exception is thrown at runtime

**Answer:** C

**QUESTION:** 48

Given:

```

class Overloading { int x(double d)
{ System.out.println("one"); return 0;
}
String x(double d) { System.out.println("two"); return null;
}
double x(double d) { System.out.println("three"); return 0.0;
}
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:** D

**QUESTION: 49**

Given the following four Java file definitions:

```
// Foo.java package facades;
public interface Foo { }
// Boo.java package facades;
public interface Boo extends Foo { }
// Woofy.java package org.domain
// line n1
public class Woofy implements Boo, Foo { }
// Test.java package.org; public class Test {
public static void main(String[] args) { Foo obj=new Woofy();
Which set modifications enable the code to compile and run?
```

- A. At line n1, Insert: import facades;At line n2, insert:import facades;import org.domain;
- B. At line n1, Insert: import facades.\*;At line n2, insert:import facades;import org.\*;
- C. At line n1, Insert: import facades.\*;At line n2, insert:import facades.Boo;import org.\*; D.
- At line n1, Insert: import facades.Foo, Boo;At line n2, insert:import org.domain.Woofy;
- E. At line n1, Insert: import facades.\*;At line n2, insert:import facades;import org.domain.Woofy;

**Answer: E**

**QUESTION: 50**

Given:

```
public class Test {
public static void main(String[] args) { try
{ String[] arr =new String[4]; arr[1] =
"Unix"; arr[2] = "Linux";
arr[3] = "Solaris"; for (String var : arr)
{ System.out.print(var + " ");
}
} catch(Exception e) { System.out.print (e.getClass());
}
}
}
```

What is the result?

- A. Unix Linux Solaris
- B. Null Unix Linux Solaris
- C. Class java.lang.Exception
- D. Class java.lang.NullPointerException

**Answer:** B

**Explanation:**

null Unix Linux Solarios

The first element, arr[0], has not been defined.

**QUESTION: 51**

Which of the following can fill in the blank in this code to make it compile?

```
public class Exam {  
  
    void method() {}  
  
}  
  
public class OCAJP extends Exam{  
    ____ void method() {}  
}
```

- A. abstract
- B. final
- C. private
- D. default
- E. int

**Answer:** C

**Explanation:**

From Java SE 8, we can use static and/or default methods in interfaces, but they should be non abstract methods. SO in this case using default in blank is completely legal. Hence option C is correct. Option A is incorrect as given method is not abstract, so can't use abstract there. Options B and E are incorrect as we can't have non abstract method interface if they are not default or static.

<https://docs.oracle.com/javase/tutorial/iava/landl/defaultmethods.html>

**QUESTION: 52**

Which of the following exception will be thrown due to the statement given here? int array[] = new int[-2];

- A. NullPointerException
- B. NegativeArraySizeException
- C. ArrayIndexOutOfBoundsException
- D. IndexOutOfBoundsException
- E. This statement does not cause any exception.

**Answer:** B

**Explanation:**

In given statement we can see that, we have passed negative value for creating int array, which results a NegativeArraySize Exception. Hence option B is correct.

Option A is incorrect as it is thrown when an application attempts to use null in a case where an object is required.

Option D is incorrect as IndexOutOfBoundsException thrown to indicate that an index of some sort (such as to an array, to a string, or to a vector) is out of range.

**REFERENCE**

<https://docs.oracle.com/javase/8/docs/api/java/lang/NegativeArraySizeException.html>

**QUESTION: 53**

Given:

```
public abstract class Shape {  
    private int x;  
    private int y;  
    public abstract void draw();  
    public void setAnchor(int x, int y) {  
        this.x = x;  
        this.y = y;  
    }  
}
```

Which two classes use the shape class correctly?



- ☐ A) `public class Circle implements Shape {  
 private int radius;  
}`
- ☐ B) `public abstract class Circle extends Shape {  
 private int radius;  
}`
- ☐ C) `public class Circle extends Shape {  
 private int radius;  
 public void draw();  
}`
- ☐ D) `public abstract class Circle implements Shape {  
 private int radius;  
 public void draw();  
}`
- ☐ E) `public class Circle extends Shape {  
 private int radius;  
 public void draw() { /* code here */ }  
}`
- ☐ F) `public abstract class Circle implements Shape {  
 private int radius;  
 public void draw() { /* code here */ }  
}`

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

**Answer:** B, E

**Explanation:**

When an abstract class is subclassed, the subclass usually provides implementations for all of the abstract methods in its parent class (E). However, if it does not, then the subclass must also be declared abstract (B).

Note: An abstract class is a class that is declared abstract—it may or may not include abstract methods. Abstract classes cannot be instantiated, but they can be subclassed.

**QUESTION:** 54

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. An exception is thrown at runtime
- D. 0

**Answer:** C

**QUESTION:** 55

Given:

```
public class X {  
    public static void main(String[] args){  
        String theString = "Hello World";  
        System.out.println(theString.charAt(11));  
    }  
}
```

What is the result?

- A. There is no output
- B. d is output
- C. A `StringIndexOutOfBoundsException` is thrown at runtime
- D. An `ArrayIndexOutOfBoundsException` is thrown at runtime
- E. A `NullPointerException` is thrown at runtime
- F. A `StringArrayIndexOutOfBoundsException` is thrown at runtime

**Answer:** C

**Explanation:**

There are only 11 characters in the string "Hello World". The code `theString.charAt(11)` retrieves the 12th character, which does not exist. A `StringIndexOutOfBoundsException` is thrown.

Exception in thread "main" java.lang.StringIndexOutOfBoundsException: String index out of range: 11

**QUESTION: 56**

Given the code fragment:

```
float x = 22.00f % 3.00f; int y = 22 % 3;
```

System.out.print(x + ", " + y); What is the result?

- A. 1.0, 1
- B. 1.0f, 1
- C. 7.33, 7
- D. Compilation fails
- E. An exception is thrown at runtime

**Answer: A**

**QUESTION: 57**

Given:

```
package p1; public class Test { static  
double dvalue; static Test ref; public  
static void main(String[] args) {  
System.out.println(ref); System.out.println(dvalue);  
}  
}
```

What is the result?

- A. p1.Test.class 0.0
- B. <the summary address referenced by ref> 0.000000
- C. Null 0.0
- D. Compilation fails
- E. A NullPointerException is thrown at runtime

**Answer: C**

**QUESTION: 58**

Given a java source file:

```

class X {
    X() { }
    private void one() { }
}

public class Y extends X {
    Y() { }
    private void two() { one(); }
    public static void main(String[] args) {
        new Y().two();
    }
}

```

What changes will make this code compile? (Select Two)

- A. Adding the public modifier to the declaration of class x
- B. Adding the protected modifier to the x() constructor
- C. Changing the private modifier on the declaration of the one() method to protected
- D. Removing the Y () constructor
- E. Removing the private modifier from the two () method

**Answer:** C, E

**Explanation:**

Using the private protected, instead of the private modifier, for the declaration of the one() method, would enable the two() method to access the one() method.

**QUESTION: 59**

Given: class Mid {  
 public int findMid(int n1, int n2) { return (n1 + n2) / 2;  
 }  
 }  
 public class Calc extends Mid { public static void main(String[] args) { int n1 = 22, n2 = 2;  
 // insert code here System.out.print(n3);  
 }  
 }

Which two code fragments, when inserted at // insert code here, enable the code to compile and print 12?

- A. Calc c = new Calc(); int n3 = c.findMid(n1,n2);
- B. int n3 = super.findMid(n1,n3);
- C. Calc c = new Mid(); int n3 = c.findMid(n1, n2);
- D. Mid m1 = new Calc(); int n3 = m1.findMid(n1, n2);
- E. int n3 = Calc.findMid(n1, n2);

**Answer:** A, D

**Explanation:**

Incorrect:

Not B: circular definition of n3.

Not C: Compilation error. line `Calc c = new Mid();` required: `Calc`  
found: `Mid`

Not E: Compilation error. line `int n3 = Calc.findMid(n1, n2);`  
non-static method `findMid(int,int)` cannot be referenced from a static context

**QUESTION:** 60

Given the code fragment:

```
interface SampleClosable {  
    public void close () throws java.io.IOException;  
}
```

Which three implementations are valid?

```
☐ A) public class Test implements SampleClosable {  
    public void close() throws java.io.IOException {  
        // do something  
    }  
}  
☐ B) public class Test implements SampleClosable {  
    public void close() throws Exception {  
        // do something  
    }  
}  
☐ C) public class Test implements SampleClosable {  
    public void close() throws java.io.FileNotFoundException {  
        // do something  
    }  
}  
☐ D) public class Test extends SampleClosable {  
    public void close() throws java.io.IOException {  
        // do something  
    }  
}  
☐ E) public class Test implements SampleClosable {  
    public void close() {  
        // do something  
    }  
}
```

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

**Answer:** A, C, E

**Explanation:**

A: Throwing the same exception is fine.

C: Using a subclass of java.io.IOException (here java.io.FileNotFoundException) is fine E:  
Not using a throw clause is fine.

**QUESTION: 61**

Given:

```
class Test {
    public static void main(String[] args) {
        int numbers[];
        numbers = new int[2];
        numbers[0] = 10;
        numbers[1] = 20;

        numbers = new int[4];
        numbers[2] = 30;
        numbers[3] = 40;
        for (int x : numbers) {
            System.out.print(" "+x);
        }
    }
}
```

What is the result?

- A. 10 20 30 40
- B. 0 0 30 40
- C. Compilation fails
- D. An exception is thrown at runtime

**Answer: A**

**QUESTION: 62**

Given the code fragment:

```
if (aVar++ < 10) {
    System.out.println(aVar + " Hello World!");
} else {
    System.out.println(aVar + " Hello Universe!");
}
```

What is the result if the integer aVar is 9?

- A. 10 Hello world!

- B. 10 Hello universe!
- C. 9 Hello world!
- D. Compilation fails.

**Answer:** A

**QUESTION:** 63

Given:

```
abstract class X {  
    public abstract void methodX();  
}  
interface Y{  
    public void methodY();  
}
```

Which two code fragments are valid?

```
☐ A) class Z extends X implements Y{  
    public void methodZ(){}  
}  
☐ B) abstract class Z extends X implements Y{  
    public void methodZ(){}  
}  
☐ C) class Z extends X implements Y{  
    public void methodX(){}  
}  
☐ D) abstract class Z extends X implements Y{  
}  
☐ E) class Z extends X implements Y{  
    public void methodY(){}  
}
```

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

**Answer:** B, C

**Explanation:**



When an abstract class is subclassed, the subclass usually provides implementations for all of the abstract methods in its parent class (C). However, if it does not, then the subclass must also be declared abstract (B).

Note: An abstract class is a class that is declared abstract—it may or may not include abstract methods. Abstract classes cannot be instantiated, but they can be subclassed.

**QUESTION: 64**

Given:

```
class Overloading {
    int x(double d) {
        System.out.println("one");
        return 0;
    }

    String x(double d) {
        System.out.println("two");
        return null;
    }

    double x(double d) {
        System.out.println("three");
        return 0.0;
    }

    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: D**

**QUESTION: 65**

Given the code fragment:



```
int b = 4;  
b--;  
System.out.println(--b);  
System.out.println(b);
```

What is the result?

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

**Answer:** A

**Explanation:** Variable b is set to 4. Variable b is decreased to 3.

Variable b is decreased to 2 and then printed. Output: 2 Variable b is printed. Output: 2

**QUESTION:** 66

```
Class StaticField { static int i = 7;  
public static void main(String[] args) { StaticFied obj = new StaticField(); obj.i++;  
StaticField.i++; obj.i++;  
System.out.println(StaticField.i + " " + obj.i);  
}  
}
```

What is the result?

- A. 10 10
- B. 8 9
- C. 9 8
- D. 7 10

**Answer:** A

**QUESTION:** 67

Which declaration initializes a boolean variable?

- A. boolean h = 1;

B. boolean k = 0;  
C. boolean m = null; D.  
boolean j = (1 < 5);

**Answer:** D

**Explanation:**

The primitive type boolean has only two possible values: true and false. Here j is set to (1 < 5), which evaluates to true.

**QUESTION:** 68

Which two are Java Exception classes?

- A. ServletException
- B. DuplicatePathException
- C. IllegalArgumentException
- D. TooManyArgumentsException

**Answer:** A, C

**QUESTION:** 69

Given:

```
1. public class Whizlabs{  
2.     public static void main(String[] args){  
3.         StringBuilder sb = new StringBuilder("1Z0");  
4.         sb.concat("-808");  
5.         System.out.println(sb);  
6.     }  
7. }
```

What is the output?

- A. 1Z0
- B. 1Z0-808
- C. An exception will be thrown.
- D. Compilation fails due to error at line 3.
- E. Compilation fails due to error at line 4.

**Answer:** E

**Explanation:**

Option E is the correct answer.

Code fails to compile because there is no method called `concert` in `StringBuilder` class. The `concert` method is in `String` class. Hence option E is correct. Here we should have used `append` method of `StringBuilder` class, in that case option B would be correct. <https://docs.oracle.com/javase/tutorial/java/data/buffers.html>

**QUESTION:** 70

What is the proper way to define a method that takes two `int` values and returns their sum as an `int` value?

- A. `int sum(int first, int second) { first + second; }`
- B. `int sum(int first, second) { return first + second; }`
- C. `sum(int first, int second) { return first + second; }`
- D. `int sum(int first, int second) { return first + second; }`
- E. `void sum (int first, int second) { return first + second; }`

**Answer:** D

**QUESTION:** 71

Given the code fragment:

```
String color = "Red";

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

What is the result?

- A. Found Red
- B. Found Red Found Blue
- C. Found Red Found Blue Found White

D. Found Red Found Blue Found White Found Default

**Answer:** B

**Explanation:**

As there is no break statement after the case "Red" statement the case Blue statement will run as well.

Note: The body of a switch statement is known as a switch block. A statement in the switch block can be labeled with one or more case or default labels. The switch statement evaluates its expression, then executes all statements that follow the matching case label.

Each break statement terminates the enclosing switch statement. Control flow continues with the first statement following the switch block. The break statements are necessary because without them, statements in switch blocks fall through: All statements after the matching case label are executed in sequence, regardless of the expression of subsequent case labels, until a break statement is encountered.

**QUESTION:** 72

Given:

```
public class MyClass {  
    public static void main(String[] args) { String s = " Java Duke ";  
    int len = s.trim().length(); System.out.print(len);  
    }  
}
```

What is the result?

- A. 8
- B. 9
- C. 11
- D. 10
- E. Compilation fails

**Answer:** B

**Explanation:**

Java - String trim() Method

This method returns a copy of the string, with leading and trailing whitespace omitted.

**QUESTION:** 73

Given:

```

Given:
class X {
    public void mX() {
        System.out.println("Xm1");
    }
}
class Y extends X {
    public void mX() {
        System.out.println("Xm2");
    }
    public void mY() {
        System.out.println("Ym");
    }
}

public class Test {
    public static void main(String[] args) {
        X xRef = new Y();
        Y yRef = (Y) xRef;
        yRef.mY();
        xRef.mX();
    }
}

```

- A. Ym Xm2
- B. Ym Xm1
- C. Compilation fails
- D. A ClassCastException is thrown at runtime

**Answer:** B

**QUESTION:** 74

Given the code fragment:

```

int[] lst = {1, 2, 3, 4, 5, 4, 3, 2, 1};
int sum = 0;
for (int frnt = 0, rear = lst.length - 1;
     frnt < 5 && rear >= 5;
     frnt++, rear--) {
    sum = sum + lst[frnt] + lst[rear];
}
System.out.print(sum);

```

What is the result?

- A. 20
- B. 25

- C. 29
- D. Compilation fails
- E. AnArrayIndexOutOfBoundsException is thrown at runtime

**Answer:** A

**QUESTION:** 75

Given:

```
1. public class Whizlabs {  
2.  
3.     public static void main(String[] args) {  
4.         String s = "A";  
5.  
6.         switch (s) {  
7.             case "a":  
8.                 System.out.print("simaple A ");  
9.             default:  
10.                System.out.print("default ");  
11.             case "A":  
12.                 System.out.print("Capital A ");  
13.         }  
14.     }  
15. }
```

What is the result?

- A. simaple A
- B. Capital A
- C. simaple A default Capital A
- D. simaple A default
- E. Compilation fails.

**Answer:** C

**Explanation:**

Here we have to use two ternary operators combined. SO first we can use to check first condition which is  $x > 10$ , as follows;

`x>10?">":` (when condition false) Now we have to use another to check if `x<10` as follows;  
`x<10?V:"="` We can combine these two by putting last ternary statement in the false position  
of first ternary statement as follows;

`x>10?">":x<10?'<':"="`

<https://docs.oracle.com/javase/tutorial/java/nutsandbolts/if.html>

**QUESTION: 76**

Given:

```
public class SampleClass {
    public static void main(String[] args) {
        AnotherSampleClass asc = new AnotherSampleClass();
        SampleClass sc = new SampleClass();
        sc = asc;
        System.out.println("sc: " + sc.getClass());
        System.out.println("asc: " + asc.getClass());
    }
}
class AnotherSampleClass extends SampleClass {
}
```

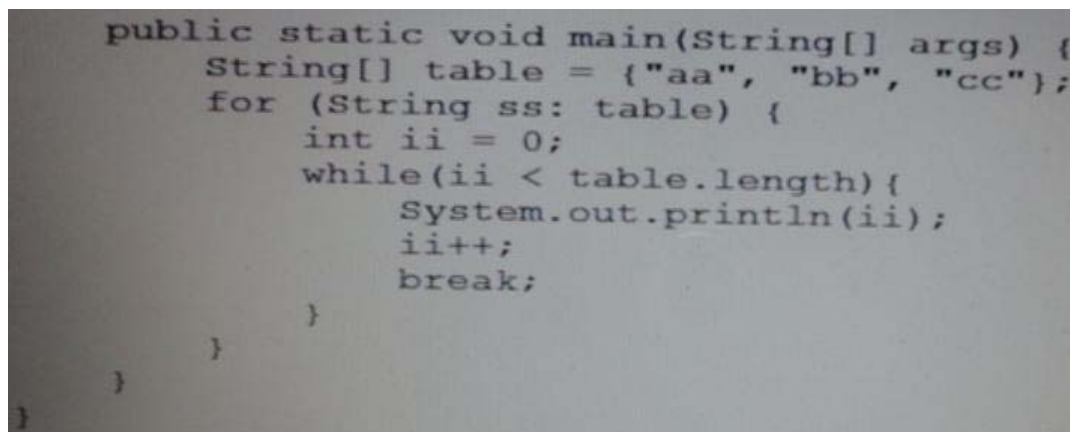
What is the result?

- A. sc: class Object asc: class AnotherSampleClass
- B. sc: class SampleClass asc: class AnotherSampleClass
- C. sc: class AnotherSampleClass asc: class SampleClass
- D. sc: class AnotherSampleClass asc: class AnotherSampleClass

**Answer: D**

**QUESTION: 77**

Given the code fragment:



```
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(ii);
            ii++;
            break;
        }
    }
}
```

How many times is 2 printed?

- A. Zero B.
- Once C.
- Twice D.
- Thrice
- E. It is not printed because compilation fails

**Answer:** B

**Explanation:**

The outer loop will run three times, one time each for the elements in table. The break statement breaks the inner loop immediately each time. 2 will be printed once only.

Note: If the line `int ii = 0;` is missing the program would not compile.

**QUESTION:** 78

Given:

```
public class App {  
    // Insert code here  
    System.out.print("Welcome to the world of Java");  
}  
}
```

Which two code fragments, when inserted independently at line `// Insert code here`, enable the program to execute and print the welcome message on the screen?

- A. `static public void main (String [] args)`
- { B. `static void main (String [] args) {`
- C. `public static void Main (String [] args)`
- { D. `public static void main (String [] args)`
- { E. `public void main (String [] args) {`

**Answer:** A, D

**Explanation:**

Incorrect:

Not B: No main class found.

Not C: Main method not found not E: Main method is not static.

**QUESTION:** 79

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;  
    }  
}
```

Which line of code initializes a student instance?



- A. Student student1;
- B. Student student1 = Student.new();
- C. Student student1 = new Student();
- D. Student student1 = Student();

**Answer:** C

**QUESTION:** 80

Which of the following will print current time?

- A. System.out.print(new LocalTime()-now0);
- B. System.out.print(new LocalTime());
- C. System.out.print(LocalTime.now());
- D. System.out.print(LocalTime.today());
- E. None of the above.

**Answer:** C

**Explanation:**

The LocalTime is an interface, so we can't use new keyword with them. So options A and C are incorrect. To get current time we can call now method on LocalTime interface. So option C is correct. Option D is incorrect as there is no method called today as in LocalTime interface <https://docs.oracle.com/javase/tutorial/datetime/iso/datetime.html>

**QUESTION:** 81

Given the code fragment:

```
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);
        objs.add(s1);                       // line n2

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

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

**Answer:** D

**QUESTION:** 82

Given:

```
public class Marklist { int num;  
public static void graceMarks(Marklist obj4) { obj4.num += 10;  
}  
public static void main(String[] args) { MarkList obj1 = new MarkList(); MarkList obj2 =  
obj1;  
MarkList obj1 = null; obj2.num = 60; graceMarks(obj2);  
}  
}
```

How many objects are created in the memory runtime?

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

**Answer:** B

**Explanation:**

obj1 and obj3.

when you do e2 = e1 you're copying object references - you're not making a copy of the object - and so the variables e1 and e2 will both point to the same object.

**QUESTION:** 83

Which code fragment is illegal?

```
C A) class Base1 {  
        abstract class Abs1 { }  
    }  
  
C B) abstract class Abs1 {  
        void doit() { }  
    }  
  
C C) class Base1 { }  
        abstract class Abs1 extends Base1 { }  
  
C D) abstract int var1 = 89;
```

- A. Option A
- B. Option B
- C. Option C
- D. Option D

**Answer:** D

**Explanation:**

The abstract keyword cannot be used to declare an int variable.

The abstract keyword is used to declare a class or method to be abstract[3]. An abstract method has no implementation; all classes containing abstract methods must themselves be abstract, although not all abstract classes have abstract methods.

**QUESTION:** 84

Given:

```
1. public abstract class Wow {  
2.     private int wow;  
3.     public Wow(int wow) {  
4.         this.wow = wow;  
5.     }  
6.     public void wow() { }  
7.     private void wowza() { }  
8. }
```

What is true about the class Wow?

- A. It compiles without error.
- B. It does not compile because an abstract class cannot have private methods.

- C. It does not compile because an abstract class cannot have instance variables.
- D. It does not compile because an abstract class must have at least one abstract method.
- E. It does not compile because an abstract class must have a constructor with no arguments.

**Answer:** A

**QUESTION:** 85

Given:

```
public class Msg {  
    public static String doMsg(char x) {  
        return "Good Day!";  
    }  
    public static String doMsg(int y) {  
        return "Good Luck!";  
    }  
    public static void main(String[] args) {  
        char x = 8;  
        int z = '8';  
        System.out.println(doMsg(x));  
        System.out.print(doMsg(z));  
    }  
}
```

What is the result?

- A. Good Day! Good Luck!
- B. Good Day! Good Day!
- C. Good Luck! Good Day!
- D. Good Luck! Good Luck!
- E. Compilation fails

**Answer:** E

**QUESTION:** 86

Given:

```
public class Main {  
    public static void main(String[] args) { try {  
        doSomething();  
    }  
    catch (SpecialException e) { System.out.println(e);  
    }  
    static void doSomething() { int [] ages = new int[4]; ages[4] = 17; doSomethingElse();  
    }  
    static void doSomethingElse() {
```

```
throw new SpecialException("Thrown at end of doSomething() method"); }  
}
```

What is the output?

- A. SpecialException: Thrown at end of doSomething() method
- B. Error in thread "main" java.lang. ArrayIndexOutOfBoundsException
- C. Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 4 at Main.doSomething(Main.java:12) at Main.main(Main.java:4)
- D. SpecialException: Thrown at end of doSomething() method at Main.doSomethingElse(Main.java:16) at Main.doSomething(Main.java:13) at Main.main(Main.java:4)

**Answer:** C

**Explanation:**

The following line causes a runtime exception (as the index is out of bounds): `ages[4] = 17;`

A runtime exception is thrown as an `ArrayIndexOutOfBoundsException`.

Note: The third kind of exception (compared to checked exceptions and errors) is the runtime exception. These are exceptional conditions that are internal to the application, and that the application usually cannot anticipate or recover from. These usually indicate programming bugs, such as logic errors or improper use of an API.

Runtime exceptions are not subject to the Catch or Specify Requirement. Runtime exceptions are those indicated by `RuntimeException` and its subclasses.

**QUESTION:** 87

Given:

```
public class Calculator {  
    public static void main(String[] args) {  
        int num = 5;  
        int sum;  
  
        do {  
            sum += num;  
        } while ((num--) > 1);  
  
        System.out.println("The sum is " + sum + ".");  
    }  
}
```

What is the result?

- A. The sum is 2
- B. The sum is 14
- C. The sum is 15
- D. The loop executes infinite times
- E. Compilation fails

**Answer:** E

**QUESTION:** 88

Given:

```
public class Series {
    private boolean flag;

    public void displaySeries() {
        int num = 2;
        while (flag) {
            if (num % 7 == 0)
                flag = false;
            System.out.print(num);
            num += 2;
        }
    }

    public static void main(String[] args) {
        new Series().displaySeries();
    }
}
```

What is the result?

- A. 2 4 6 8 10 12
- B. 2 4 6 8 10 12 14
- C. Compilation fails
- D. The program prints multiple of 2 infinite times
- E. The program prints nothing

**Answer:** B

**QUESTION:** 89

Given the code fragment:

String name = "Spot"; int age = 4;

String str = "My dog " + name + " is " + age; System.out.println(str);

And

StringBuilder sb = new StringBuilder();

Using StringBuilder, which code fragment is the best potion to build and print the following string My dog Spot is 4

- A. sb.append("My dog " + name + " is " + age); System.out.println(sb);
- B. sb.insert("My dog ").append( name + " is " + age); System.out.println(sb);

C. sb.insert("My dog ").insert( name ).insert(" is " ).insert(age); System.out.println(sb);  
D. sb.append("My dog ").append( name ).append(" is " ).append(age);  
System.out.println(sb);

**Answer:** A, D

**QUESTION:** 90

Give:

```
class Alpha {
    public String[] main = new String[2];
    Alpha(String[] main) {
        for (int ii = 0; ii < main.length; ii++) {
            this.main[ii] = main[ii] + 5;
        }
    }
    public void main() {
        System.out.print(main[0] + main[1]);
    }
}

public class Test {
    public static void main(String[] args) {
        Alpha main = new Alpha(args);
        main.main();
    }
}

And the commands:

javac Test.java
java Test 1 2
```

What is the result?

- A. 1525
- B. 13
- C. Compilation fails
- D. An exception is thrown at runtime
- E. The program fails to execute due to runtime error

**Answer:** D

**QUESTION:** 91

Given the code fragment:



```
String valid = "true";  
if (valid) System.out.println("valid");  
else      System.out.println("not valid");
```

What is the result?

- A. Valid
- B. Not valid
- C. Compilation fails
- D. An IllegalArgumentException is thrown at run time

**Answer:** C

**Explanation:**

In segment 'if (valid)' valid must be of type boolean, but it is a string. This makes the compilation fail.

**QUESTION:** 92

Consider following method

```
default void print(){  
  
}
```

Which statement is true?

- A. This method is invalid.
- B. This method can be used only in an interface.
- C. This method can return anything.
- D. This method can be used only in an interface or an abstract class.
- E. None of above.

**Answer:** B

**Explanation:**

Given method is declared as default method so we can use it only inside an interface. Hence option B is correct and option D is incorrect.

Option A is incorrect as it is valid method. Option C is incorrect as return type is void, which means we can't return anything.



**QUESTION: 93**

Given the fragment:

```
int[] array = {1,2,3,4,5};  
System.arraycopy(array, 2, array, 1, 2);  
System.out.print(array[1]);  
System.out.print(array[4]);
```

What is the result?

- A. 14
- B. 15
- C. 24
- D. 25
- E. 34
- F. 35

**Answer: F**

**Explanation:**

The two elements 3 and 4 (starting from position with index 2) are copied into position index 1 and 2 in the same array.

After the arraycopy command the array looks like:

{1, 3, 4, 4, 5};

Then element with index 1 is printed: 3

Then element with index 4 is printed: 5

Note: The System class has an arraycopy method that you can use to efficiently copy data from one array into another:

```
public static void arraycopy(Object src, int srcPos, Object dest, int destPos, int length)
```

The two Object arguments specify the array to copy from and the array to copy to. The three int arguments specify the starting position in the source array, the starting position in the destination array, and the number of array elements to copy.

**QUESTION: 94**

Which statement initializes a stringBuilder to a capacity of 128?

- A. `StringBuilder sb = new String ("128");`
- B. `StringBuilder sb = StringBuilder.setCapacity (128);`
- C. `StringBuilder sb = StringBuilder.getInstance (128);`
- D. `StringBuilder sb = new StringBuilder (128);`

**Answer: D**

**Explanation:**

StringBuilder(int capacity)

Constructs a string builder with no characters in it and an initial capacity specified by the capacity argument.

Note: An instance of a StringBuilder is a mutable sequence of characters.

The principal operations on a StringBuilder are the append and insert methods, which are overloaded so as to accept data of any type. Each effectively converts a given datum to a string and then appends or inserts the characters of that string to the string builder. The append method always adds these characters at the end of the builder; the insert method adds the characters at a specified point.

**QUESTION: 95**

Given:

```
public class Test {  
  
    static void dispResult(int[] num) {  
        try {  
            System.out.println(num[1] / (num[1] - num[2]));  
        } catch(ArithmeticException e) {  
            System.err.println("first exception");  
        }  
        System.out.println("Done");  
    }  
  
    public static void main(String[] args) {  
        try {  
            int[] arr = {100, 100};  
            dispResult(arr);  
        } catch(IllegalArgumentException e) {  
            System.err.println("second exception");  
        } catch(Exception e) {  
            System.err.println("third exception");  
        }  
    }  
}
```

What is the result?

- A. 0 Done
- B. First Exception Done
- C. Second Exception
- D. Done
- Third Exception
- E. Third Exception

**Answer: B**

**QUESTION: 96**

Given:

```

public class SampleClass {
    public static void main(String[] args){
        AnotherSampleClass asc = new AnotherSampleClass();
        SampleClass sc = new SampleClass();
        sc = asc;
        System.out.println("sc: " + sc.getClass());
        System.out.println("asc: " + asc.getClass());
    }
}
class AnotherSampleClass extends SampleClass {
}

```

What is the result?

- A. sc: class.Object  
asc: class.AnotherSampleClass
- B. sc: class.SampleClass  
asc: class.AnotherSampleClass
- C. sc: class.AnotherSampleClass asc: class.SampleClass
- D. sc: class.AnotherSampleClass asc: class.AnotherSampleClass

**Answer:** D

**Explanation:**

Note: The getClass method Returns the runtime class of an object. That Class object is the object that is locked by static synchronized methods of the represented class.

Note: Because Java handles objects and arrays by reference, classes and array types are known as reference types.

**QUESTION:** 97

Given:

```

public class Vowel {
    private char var;
    public static void main(String[] args) {
        char var1 = 'a';
        char var2 = var1;
        var2 = 'e';

        Vowel obj1 = new Vowel();
        Vowel obj2 = obj1;
        obj1.var = 'i';
        obj2.var = 'o';

        System.out.println(var1 + ", " + var2);
        System.out.print(obj1.var + ", " + obj2.var);
    }
}

```

- A. a, e i, o

- B. a, e o, o
- C. e, e I, o
- D. e, e o, o

**Answer:** B

**QUESTION: 98**

Given:

```
public class Circle {
    double radius;
    public double area;
    public Circle(double r) { radius = r; }
    public double getRadius() { return radius; }
    public void setRadius(double r) { radius = r; }
    public double getArea() { return /* ??? */; }
}

class App {
    public static void main(String[] args) {
        Circle c1 = new Circle(17.4);
        c1.area = Math.PI * c1.getRadius() * c1.getRadius();
    }
}
```

The class is poorly encapsulated. You need to change the circle class to compute and return the area instead. Which two modifications are necessary to ensure that the class is being properly encapsulated?

- A. Remove the area field.
- B. Change the getArea( ) method as follows:  
`public double getArea( ) { return Match.PI * radius * radius; }`
- C. Add the following method:  
`public double getArea( ) { area = Match.PI * radius * radius; }`
- D. Change the caccess modifier of the SerRadius ( ) method to be protected.

**Answer:** B, D

**QUESTION: 99**

Given the code fragment:

```
class Student { int rollnumber; String name;
List cources = new ArrayList();
// insert code here public String toString() {
return rollnumber + " : " + name + " : " + cources;
}
}
```

And,

```
public class Test {
```

```
public static void main(String[] args) { List cs = new ArrayList(); cs.add("Java");
cs.add("C");
Student s = new Student(123,"Fred", cs); System.out.println(s);
}
}
```

Which code fragment, when inserted at line // insert code here, enables class Test to print 123 : Fred : [Java, C]?

- A. private Student(int i, String name, List cs) {  
/\* initialization code goes here \*/  
}
- B. public void Student(int i, String name, List cs) {  
/\* initialization code goes here \*/  
}
- C. Student(int i, String name, List cs) {  
/\* initialization code goes here \*/  
}
- D. Student(int i, String name, ArrayList cs) {  
/\* initialization code goes here \*/  
}

**Answer:** C

**Explanation:**

Incorrect:

Not A: Student has private access line: Student s = new Student(123,"Fred", cs);

Not D: Cannot be applied to given types. Line: Student s = new Student(123,"Fred", cs);

**QUESTION:** 100

Given the fragment:

```
24. float var1 = (12_345.01 >= 123_45.00) ? 12_456 : 124_56.02f;
25. float var2 = var1 + 1024;
26. System.out.print(var2);
```

What is the result?

- A. 13480.0
- B. 13480.02
- C. Compilation fails
- D. An exception is thrown at runtime

**Answer:** A

**QUESTION: 101**

Given the fragments:

```
public class TestA extends Root {
    public static void main(String[] args) {
        Root r = new TestA();
        System.out.println(r.method1());    // line n1
        System.out.println(r.method2());    // line n2
    }
}

class Root {
    private static final int MAX = 20000;
    private int method1() {
        int a = 100 + MAX;                // line n3
        return a;
    }
    protected int method2() {
        int a = 200 + MAX;                // line n4
        return a;
    }
}
```

Which line causes a compilation error?

- A. Line n1
- B. Line n2
- C. Line n3
- D. Line n4

**Answer: A**

**QUESTION: 102**

```
public class StringReplace {
    public static void main(String[] args) { String message = "Hi everyone!";
    System.out.println("message = " + message.replace("e", "X")); }
}
```

What is the result?

- A. message = Hi everyone!
- B. message = Hi XvXryonX!
- C. A compile time error is produced.
- D. A runtime error is produced.
- E. message =
- F. message = Hi Xeveryone!

**Answer: B**



**QUESTION: 103**

Given:

```
1. import java.io.Error;
2.     public class TestApp {
3.         public static void main(String[] args) {
4.             TestApp t = new TestApp();
5.             try {
6.                 t.doPrint();
7.                 t.doList();
8.
9.             } catch (Exception e2) {
10.                System.out.println("Caught " + e2);
11.            }
12.        }
13.        public void doList() throws Exception {
14.            throw new Error("Error");
15.        }
16.        public void doPrint() throws Exception {
17.            throw new RuntimeException("Exception");
18.        }
19.    }
```

What is the result?

```
C A) Caught java.lang.RuntimeException: Exception
   Exception in thread "main" java.lang.Error: Error
   at TestApp.doList(TestApp.java: 14)
   at TestApp.main(TestApp.java: 6)

C B) Exception in thread "main" java.lang.Error: Error
   at TestApp.doList(TestApp.java: 14)
   at TestApp.main(TestApp.java: 6)

C C) Caught java.lang.RuntimeException: Exception
   Caught java.lang.Error: Error

C D) Caught java.lang.RuntimeException: Exception
```

- A. Option A
- B. Option B
- C. Option C
- D. Option D

**Answer: C**

**QUESTION: 104**

Given the code fragment:

```
int b = 3;
if ( !(b > 3))
{ System.out.println("square
");
```

```
}{  
System.out.println("circle ");  
}  
System.out.println("...");
```

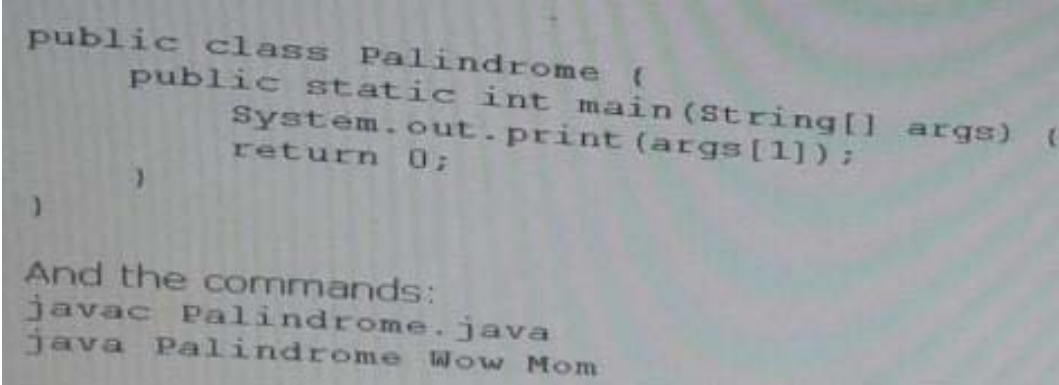
What is the result?

- A. square...
- B. circle...
- C. squarecircle...
- D. Compilation fails.

**Answer:** C

**QUESTION:** 105

Given:



```
public class Palindrome {  
    public static int main(String[] args) {  
        System.out.print(args[1]);  
        return 0;  
    }  
}
```

And the commands:  
javac Palindrome.java  
java Palindrome Wow Mom

What is the result? A.

- Compilation fails
- B. The code compiles, but does not execute.
- C. Paildrome
- D. Wow
- E. Mom

**Answer:** B

**QUESTION:** 106

Given:



```
int x = 10;

if (x > 10) {

    System.out.println(">");

} else if (x < 10) {

    System.out.println("<");

} else {

    System.out.println("=");

}
```

Which of the following is equivalent to the above code fragment?

- A. `System.out.println(x>10?">","<","=");`
- B. `System.out.println(x>10? ">"?"<":"=");`
- C. `System.out.println(x>10?">":x<10?"<":"=");`
- D. `System.out.println(x>10?">"?,"<?"=");`
- E. None of the above

**Answer:** B

**Explanation:**

Option A is incorrect as we can't use abstract with non abstract method, (here method has method body.)

Option C is incorrect as when overriding method we can't use more restrictive access modifier, so trying to use private to override default access Level method causes a compile time error.

Option D is incorrect as default methods (not methods with default access level) are allowed only in interfaces.

Option E is incorrect as method all ready has void as return type, so we can't add int there.

Option B is correct as we can use final there, since the method is non abstract  
<https://docs.oracle.com/javase/tutorial/java/landl/polymorphism.html>

**QUESTION:** 107

Given:

```
public class Equal {  
    public static void main(String[] args) { String str1 = "Java";  
    String[] str2 = {"J","a","v","a"}; String str3 = "";  
    for (String str : str2) { str3 = str3+str;  
    }  
    boolean b1 = (str1 == str3); boolean b2 = (str1.equals(str3)); System.out.print(b1+", "+b2);  
    }
```

What is the result?

- A. true, false
- B. false, true
- C. true, true
- D. false, false

**Answer:** B

**Explanation:**

== strict equality. equals compare state, not identity.

**QUESTION:** 108

Which of the following data types will allow the following code snippet to compile?

```
float i = 4;  
  
float j = 2;  
  
_____ z = i + j;
```

- A. long
- B. double
- C. int
- D. float
- E. byte

**Answer:** B, D

**Explanation:**

Option B and D are the correct answer.

Since the variables i and j are floats, resultant will be float type too. So we have to use float

or primitive type which can hold float, such a primitive type is double, it has wider range and also can hold floating point numbers, hence we can use double or float for the blank.

As explained above options B and D are correct.

long and int can't be used with floating point numbers so option A is incorrect.

Option E is incorrect as it have smaller range and also can't be used with floating point numbers. <https://docs.oracle.com/javase/tutorial/java/javaOO/variables.html>

**QUESTION: 109**

Given:

```
public class String1 {  
    public static void main(String[] args) { String s = "123";  
    if (s.length() >2)  
        s.concat("456");  
    for(int x = 0; x <3; x++) s += "x";  
    System.out.println(s);  
    }  
}
```

What is the result?

- A. 123
- B. 123xxx
- C. 123456
- D. 123456xxx
- E. Compilation fails

**Answer: B**

**Explanation:**

123xxx

The if clause is not applied. Note: Syntax of if-statement:

if ( Statement ) { }

**QUESTION: 110**

Which two are valid declarations of a two-dimensional array?

- A. int [] [] array2D;
- B. int [2] [2] array2D;
- C. int array2D [];
- D. int [] array2D [];
- E. int [] [] array2D [];

**Answer: A, D**

**Explanation:**

int[][] array2D; is the standard convention to declare a 2-dimensional integer array. int[] array2D[]; works as well, but it is not recommended.

**QUESTION: 111**

Given:

```
class Cake { int model; String flavor; Cake() { model = 0;
flavor = "Unknown";
}
}
public class Test {
public static void main(String[] args) { Cake c = new Cake();
bake1(c);
System.out.println(c.model + " " + c.flavor); bake2(c);
System.out.println(c.model + " " + c.flavor);
}
public static Cake bake1(Cake c) { c.flavor = "Strawberry";
c.model = 1200; return c;
}
public static void bake2(Cake c) { c.flavor = "Chocolate";
c.model = 1230; return;
}
}
```

What is the result?

- A. 0 unknown 0 unknown
- B. 1200 Strawberry 1200 Strawberry
- C. 1200 Strawberry 1230 Chocolate
- D. Compilation fails

**Answer: C**

**Explanation:**

1200 Strawberry  
1230 Chocolate

**QUESTION: 112**

Given:

```

1. public class TestLoop {
2.     public static void main(String[] args) {
3.         float myarray[] = {10.20f, 20.30f, 30.40f, 50.60f};
4.         int index = 0;
5.         boolean isFound = false;
6.         float key = 30.40f;
7.         // insert code here
8.         System.out.println(isFound);
9.     }
10. }

```

Which code fragment, when inserted at line 7, enables the code print true?

```

C A) while (key == myarray[index++]) {
    isFound = true;
}

C B) while (index <= 4) {
    if (key == myarray[index]) {
        index++;
        isFound = true;
        break;
    }
}

C C) while (index++ < 5) {
    if (key == myarray[index]) {
        isFound = true;
    }
}

C D) while (index < 5) {
    if (key == myarray[index]) {
        isFound = true;
        break;
    }
    index++;
}

```

- A. Option A
- B. Option B
- C. Option C
- D. Option D

**Answer:** A

**QUESTION:** 113

Given:

```

public class Main {
    public static void main(String[] args) throws Exception {
        doSomething();
    }
    private static void doSomething() throws Exception {
        System.out.println("Before if clause");
        if (Math.random() > 0.5) {
            throw new Exception();
        }
        System.out.println("After if clause");
    }
}

```

Which two are possible outputs?

- ☐ A) Before if clause  
Exception in thread "main" java.lang.Exception  
at Main.doSomething(Main.java:8)  
at Main.main(Main.java:3)
- ☐ B) Before if clause  
Exception in thread "main" java.lang.Exception  
at Main.doSomething(Main.java:8)  
at Main.main(Main.java:3)  
After if clause
- ☐ C) Exception in thread "main" java.lang.Exception  
at Main.doSomething(Main.java:8)  
at Main.main(Main.java:3)
- ☐ D) Before if clause  
After if clause

- A. Option A
- B. Option B
- C. Option C
- D. Option D

**Answer:** A, D

**Explanation:**

The first println statement, System.out.println("Before if clause");, will always run.

If Math.Random() > 0.5 then there is an exception. The exception message is displayed and the program terminates.

If Math.Random() > 0.5 is false, then the second println statement runs as well.

**QUESTION:** 114

Which two statements correctly describe checked exception?

- A. These are exceptional conditions that a well-written application should anticipate and

recover from.

B. These are exceptional conditions that are external to the application, and that the application usually cannot anticipate or recover from.

C. These are exceptional conditions that are internal to the application, and that the application usually cannot anticipate or recover from.

D. Every class that is a subclass of RuntimeException and Error is categorized as checked exception.

E. Every class that is a subclass of Exception, excluding RuntimeException and its subclasses, is categorized as checked exception.

**Answer:** B, D

**Explanation:**

Checked exceptions:

\* (B) 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

It's somewhat confusing, but note as well that RuntimeException (unchecked) is itself a subclass of Exception (checked).

\* 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)

**Reference:**

Checked versus unchecked exceptions

**QUESTION:** 115

Which two are valid instantiations and initializations of a multi dimensional array?

```
☐ A) int[][] array2D = { {0,1,2,4}, {5,6} };
☐ B) int[][] array2D = new int[][2];
    array2D[0][0] = 1;
    array2D[0][1] = 2;
    array2D[1][0] = 3;
    array2D[1][1] = 4;
☐ C) int[][][] array3D = { {0,1}, {2,3}, {4,5} };
☐ D) int[] array = {0,1};
    int[][][] array3D = new int[2][2][2];
    array3D[0][0] = array;
    array3D[0][1] = array;
    array3D[1][0] = array;
    array3D[1][1] = array;
☐ E) int[][] array2D = { 0,1 };
```

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

**Answer:** B, D

**Explanation:**

In the Java programming language, a multidimensional array is simply an array whose components are themselves arrays.

**QUESTION:** 116

Which two are valid array declaration?

- A. Object array[];
- B. Boolean array[3];
- C. int[] array;
- D. Float[2] array;

**Answer:** A, C

**QUESTION:** 117

Given:

```
public class Access {
    private int x = 0;
    private int y = 0;

    public static void main(String[] args) {
        Access accApp = new Access();
        accApp.printThis(1, 2);
        accApp.printThat(3, 4);
    }

    public void printThis(int x, int y) {
        x = x;
        y = y;
        System.out.println("x: " + this.x + " y: " + this.y);
    }

    public void printThat(int x, int y) {
        this.x = x;
        this.y = y;
        System.out.println("x: " + this.x + " y: " + this.y);
    }
}
```

What is the result?



- A. x: 1 y: 2
- B. 3 y: 4
- C. x: 0 y: 0
- D. 3 y: 4
- E. x: 1 y: 2
- F. 0 y: 0
- G. x: 0 y: 0
- H. 0 y: 0

**Answer:** C

**QUESTION:** 118

Given the code fragment:

```
for (int ii = 0; ii < 3; ii++) { int count = 0;
for (int jj = 3; jj > 0; jj--) {
if (ii == jj) {
++count; break;
}
}
System.out.print(count); continue;
}
```

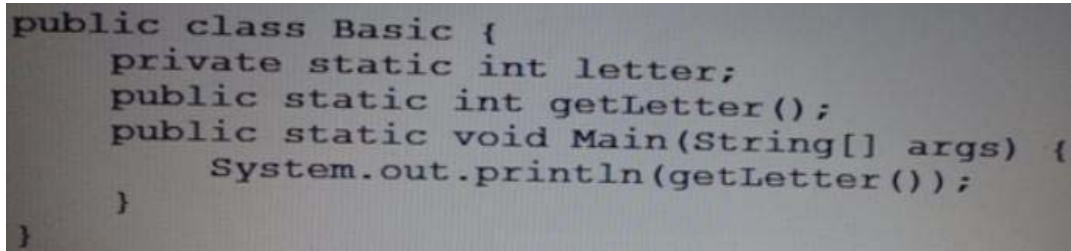
What is the result?

- A. 011
- B. 012
- C. 123
- D. 000

**Answer:** A

**QUESTION:** 119

Given:



```
public class Basic {
    private static int letter;
    public static int getLetter();
    public static void Main(String[] args) {
        System.out.println(getLetter());
    }
}
```

Why will the code not compile?

- A. A static field cannot be private.
- B. The getLetter method has no body.
- C. There is no setLetter method.
- D. The letter field is uninitialized.
- E. It contains a method named Main instead of ma

**Answer:** B

**Explanation:**

The getLetter() method needs a body public static int getLetter() { }; .

**QUESTION:** 120

Given:

```
String message1 = "Wham bam!";
String message2 = new String("Wham bam!");

if (message1 == message2)
    System.out.println("They match");

if (message1.equals(message2))
    System.out.println("They really match");
```

What is the result?

- A. They match They really match
- B. They really match
- C. They match
- D. Nothing Prints
- E. They really match They really match

**Answer:** B

**Explanation:**

The strings are not the same objects so the == comparison fails. See note #1 below. As the value of the strings are the same equals is true. The equals method compares values for equality.

Note: #1 ==

Compares references, not values. The use of == with object references is generally limited to the following:

Comparing to see if a reference is null.

Comparing two enum values. This works because there is only one object for each enum constant.

You want to know if two references are to the same object.

**QUESTION: 121**

Given:

```
class Jump {  
    static String args[] = {"lazy", "lion", "is", "always"};  
    public static void main(String[] args) {  
        System.out.println(  
            args[1] + " " + args[2] + " " + args[3] + " jumping");  
    }  
}
```

And the commands: Javac Jump.java Java Jump crazy elephant is always What is the result?

- A. Lazy lion is jumping
- B. Lion is always jumping
- C. Crazy elephant is jumping
- D. Elephant is always jumping
- E. Compilation fails

**Answer: B**

**QUESTION: 122**

Which three are valid types for switch?

- A. int
- B. float
- C. double
- D. integer
- E. String
- F. Float

**Answer: A, D, E**

**Explanation:**

A switch works with the byte, short, char, and int primitive data types. It also works with enumerated types the String class, and a few special classes that wrap certain primitive types: Character, Byte, Short, and Integer.

**QUESTION: 123**

Given:

```

public class Circle {
    double radius;
    public double area;
    public Circle(double r) { radius = r; }
    public double getRadius() { return radius; }
    public void setRadius(double r) { radius = r; }
    public double getArea() { return /* ??? */; }
}

class App {
    public static void main(String[] args) {
        Circle c1 = new Circle(17.4);
        c1.area = Math.PI * c1.getRadius() * c1.getRadius();
    }
}

```

This class is poorly encapsulated. You need to change the circle class to compute and return the area instead. What three modifications are necessary to ensure that the class is being properly encapsulated?

- A. Change the access modifier of the setradius () method to private
- B. Change the getArea () method public double getArea () { return area; }
- C. When the radius is set in the Circle constructor and the setRadius () method, recomputed the area and store it into the area field
- D. Change the getRadius () method: public double getRadius () { area = Math.PI \* radius \* radius; return radius; }

**Answer:** B, C, D

**QUESTION:** 124

Which two statements are true for a two-dimensional array?

- A. It is implemented as an array of the specified element type.
- B. Using a row by column convention, each row of a two-dimensional array must be of the same size.
- C. At declaration time, the number of elements of the array in each dimension must be specified.
- D. All methods of the class Object may be invoked on the two-dimensional array.

**Answer:** A, D

**QUESTION:** 125

Which two will compile, and can be run successfully using the command: Java fred1 hello walls

```

☐ A) class fred1 {
    public static void main(String args) {
        System.out.println(args[1]);
    }
}

☐ B) class fred1 {
    public static void main(String[] args) {
        System.out.println(args[2]);
    }
}

☐ C) class fred1 {
    public static void main(String[] args) {
        System.out.println(args);
    }
}

☐ D) class fred1 {
    public static void main(String[] args) {
        System.out.println(args[1]);
    }
}

```

- A. Option A
- B. Option B
- C. Option C
- D. Option D

**Answer:** C, D

**Explanation:**

Throws java.lang.ArrayIndexOutOfBoundsException: 2 at  
certquestions.Fred1.main(Fred1.java:3)

C. Prints out: [Ljava.lang.String;@39341183

D. Prints out: walls

**QUESTION:** 126

Given:

```
public class TestField { int x;
```

```
int y;
```

```
public void doStuff(int x, int y) { this.x = x;
```

```
y = this.y;
```

```
}
```

```
public void display() { System.out.print(x + " " + y + " : ");
```

```
}
```

```
public static void main(String[] args) { TestField m1 = new TestField(); m1.x = 100;
m1.y = 200;
```

```
TestField m2 = new TestField(); m2.doStuff(m1.x, m1.y); m1.display();
```

```
m2.display();  
}  
}
```

What is the result?

- A. 100 200 : 100 200
- B. 100 0 : 100 0 :
- C. 100 200 : 100 0 :
- D. 100 0 : 100 200 :

**Answer:** C

**QUESTION:** 127

Which of the following can fill in the blank in this code to make it compile? (Select 2 options.)

```
1.    public void method() ____ Exception {  
2.    _____ Exception();  
3.    }
```

- A. On line 1, fill in throws
- B. On line 1, fill in throws new
- C. On line 2, fill in throw new
- D. On line 2, fill in throws
- E. On line 2, fill in throws new

**Answer:** A, C

**Explanation:**

Option A and C are the correct answer.

In a method declaration, the keyword throws is used. So here at line 1 we have to use option A.

To actually throw an exception, the keyword throw is used and a new exception is created, so at line 2 we have to use throw and new keywords, which is option C. Finally it will look like;  
public void method() throws Exception { throw new Exception();}

REFERENCE : <https://docs.oracle.com/javase/tutorial/essential/io/fileOps.html#exception>

The correct answer is: On line 1, fill in throws. On line 2, fill in throw new

**QUESTION:** 128

Given the code fragment:

Boolean b1 = true; Boolean b2 = false; int i = 0; while (foo) { }

Which one is valid as a replacement for foo?

- A. b1.compareTo(b2)
- B. i = 1
- C. i == 2? -1 : 0
- D. "foo".equals("bar")

**Answer:** D

**Explanation:**

Equals works fine on strings equals produces a Boolean value.

**QUESTION:** 129

Given:

Class A { } Class B { }

Interface X { } Interface Y { }

Which two definitions of class C are valid?

- A. Class C extends A implements X { }
- B. Class C implements Y extends B { }
- C. Class C extends A, B { }
- D. Class C implements X, Y extends B { }
- E. Class C extends B implements X, Y { }

**Answer:** A, E

**Explanation:**

extends is for extending a class.

implements is for implementing an interface.

Java allows for a class to implement many interfaces.

**QUESTION:** 130

Given the code fragment:

```
System.out.println ("Result: " +3+5); System.out.println ("Result: " + (3+5));
```

What is the result?

- A. Result: 8 Result: 8
- B. Result: 35 Result: 8
- C. Result: 8 Result: 35
- D. Result: 35 Result: 35



**Answer: B**

**Explanation:**

In the first statement 3 and 5 are treated as strings and are simply concatenated. In the first statement 3 and 5 are treated as integers and their sum is calculated.

**QUESTION: 131**

Given:

```
public class Natural { private int i;
void disp() {
while (i <= 5) {
for (int i=1; i <=5;) { System.out.print(i + " "); i++;
} i++;
}
}
public static void main(String[] args) { new Natural().disp();
}
}
```

What is the result?

- A. Prints 1 2 3 4 5 once
- B. Prints 1 3 5 once
- C. Prints 1 2 3 4 5 five times
- D. Prints 1 2 3 4 5 six times
- E. Compilation fails

**Answer: D**

**Explanation:**

1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5

**QUESTION: 132**

```
int i, j=0;
i = (3* 2 +4 +5 ) ;
j = (3 * ((2+4) + 5));
System.out.println("i:" + i + "\nj":+j);
What is the result?
```

- A. i: 16  
j: 33
- B. i: 15  
j: 33
- C. i: 33  
j: 23
- D. i: 15  
j: 23

- A. Option A  
B. Option B  
C. Option C  
D. Option D

**Answer:** B

**QUESTION:** 133

Given:

```
1.  import java.time.LocalDate;
2.  import java.time.Period;
3.
4.  public class Whizlabs {
5.      public static void main(String[] args) {
6.          LocalDate date = LocalDate.of(2015, 3, 26);
7.          Period p = Period.ofDays(1);
8.          System.out.println(date.plus(p));
9.      }
10. }
```

What is the output?

- A. 2015-03-27  
B. 2015-04-27

- C. 2015-02-27
- D. Compilation fails due to error at line 6.
- E. Compilation fails due to error at line 8.

**Answer:** A

**Explanation:**

To create we have used following method with LocalDate class; public static LocalDate of(int year, int month, int dayOfMonth)

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.

<https://docs.oracle.com/javase/tutorial/datetime/iso/datetime.html>

**QUESTION:** 134

Given:

```
class MarksOutOfBoundsException extends IndexOutOfBoundsException { } public class
GradingProcess {
void verify(int marks) throws IndexOutOfBoundsException { if (marks > 100) {
throw new MarksOutOfBoundsException();
}
if (marks > 50) { System.out.print("Pass");
} else
{ System.out.print("Fail"
);
}
}
public static void main(String[] args) { int marks = Integer.parseInt(args[2]); try {
new GradingProcess().verify(marks);
} catch(Exception e) { System.out.print(e.getClass());
}
}
}
```

And the command line invocation: Java grading process 89 50 104 What is the result?

- A. Pass
- B. Fail
- C. Class MarketOutOfBoundsException
- D. Class IndexOutOfBoundsException E.
- Class Exception

**Answer:** C

**Explanation:**

The value 104 will cause a MarketOutOfBoundsException

**QUESTION: 135**

Given:

```
public class Test3 {  
    public static void main(String[] args) {  
        String names[] = new String[3];  
        names[0] = "Mary Brown";  
        names[1] = "Nancy Red";  
        names[2] = "Jessy Orange";  
        try {  
            for(String n: names) {  
                try {  
                    String pwd = n.substring(0, 3)+n.substring(6,10);  
                    System.out.println(pwd);  
                }  
                catch (StringIndexOutOfBoundsException sie) {  
                    System.out.println("string out of limits");  
                }  
            }  
        }  
        catch (ArrayIndexOutOfBoundsException e) {  
            System.out.println("array out of limits");  
        }  
    }  
}
```

What is the result?

- A. Marrown String out of limits JesOran
- B. Marrown String out of limits Array out of limits
- C. Marrown String out of limits
- D. Marrown NanRed JesOran

**Answer: A**

**QUESTION: 136**

Given the code in a file Traveler.java:

```
class Tours {  
    public static void main(String[] args) {  
        System.out.print("Happy Journey! " + args[1]);  
    }  
}  
  
public class Traveler {  
    public static void main(String[] args) {  
        Tours.main(args);  
    }  
}
```

And the commands:

Javac Traveler.java

Java Traveler Java Duke What is the result?

- A. Happy Journey! Duke

- B. Happy Journey! Java
- C. An exception is thrown at runtime
- D. The program fails to execute due to a runtime error

**Answer:** D