Oracle

Exam 1z0-803

Java SE 7 Programmer I

Version: 11.5

[Total Questions: 216]

Question No : 1

Given:

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

What isthe result?

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

Answer: C

Explanation: Hello Java! 8

Question No : 2

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 No : 3

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 No : 4

Given:

```
public class ColorTest {

public static void main(String[] args) {

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

int count = 0;

for (String c : colors) {

if (count >= 4) {

break;
```

```
}

else {

continue;

}

if (c.length() >= 4) {

colors[count] = c.substring(0,3);

}

count++;

}

System.out.println(colors[count]);

}

}
```

What is the result?

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

Answer: C

Explanation: Theline, if (c.length() >= 4) {, is never reached. This causes a compilation error.

Note: The continue statement skips the current iteration of a for, while , or do-while loop. An unlabeled break statement terminates the innermost switch, for, while, or do-while statement, but a labeled break terminates an outer statement.

Question No : 5

Given the classes:

* AssertionError

* ArithmeticException

* ArrayIndexOutofBoundsException

* FileNotFoundException

* IllegalArgumentException

* IOError

* IOException

* NumberFormatException

* SQLException

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

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

Answer: A

Explanation: Not B: IOError and IOException are both checked errors.
Not C, not D, not E: FileNotFoundException is a checked error.
Note:
Checked exceptions:
* represent invalid conditions in areas outside the immediate control of the program (invalid user input, database problems, network outages, absent files)
* are subclasses of Exception
* a method is obliged to establish a policy for all checked exceptions thrown by its implementation (either pass the checked exception further up the stack, or handle it somehow)

Note:
Unchecked exceptions:
* represent defects in the program (bugs) - often invalid arguments passed to a non-private method. To quote from The Java Programming Language, by Gosling, Arnold, and Holmes: "Unchecked runtime exceptions represent conditions that, generally speaking,reflect errors in your program's logic and cannot be reasonably recovered from at run time."

* are subclasses of RuntimeException, and are usually implemented using IllegalArgumentException, NullPointerException, or IllegalStateException
* method is not obliged to establish a policy for the unchecked exceptions thrown by its implementation (and they almost always do not do so)

Question No : 6

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?

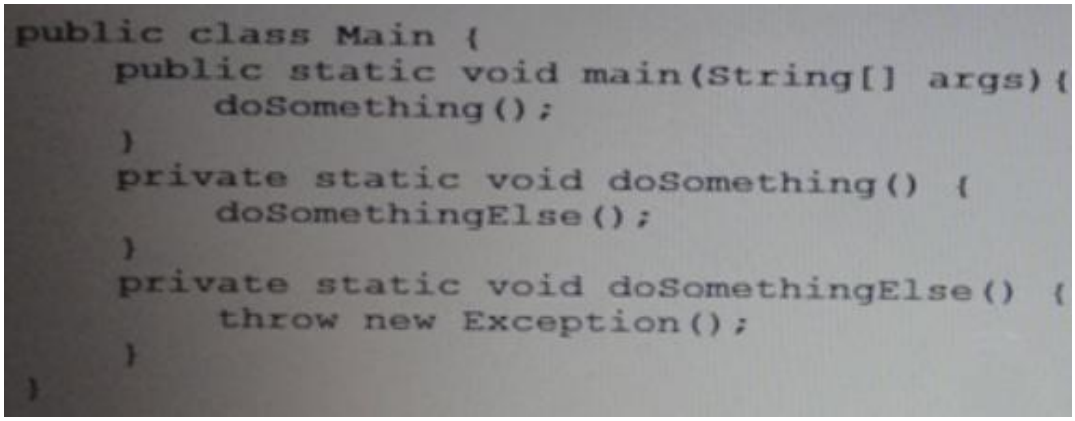
A. line 3
B. line4

C. line 5
D. line 6
E. line 10
F. line 11
G. line 12
H. line 13

Answer: C,D,H

Question No : 7

Given:



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

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

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

Answer: A

Explanation:

We need to catch the exception in the doSomethingElse() method.

Such as:

```
private static void doSomeThingElse() {
try {
throw new Exception();}
catch (Exception e)
{}
```

}

Note: One alternative, but not an option here, is the declare the exception in doSomeThingElse and catch it in the doSomeThing method.

Question No : 8

Given the code fragment:

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

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

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

Answer: A

Explanation:

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

Question No : 9

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 No : 10