➢ Vendor: Oracle

➢ Exam Code: 1Z0-808

➢ Exam Name: Java SE 8 Programmer I

➢ New Questions (Mar/2017)

NEW QUESTION 121
Given:

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

What is the result?

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

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

NEW QUESTION 122
Given:

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

What is the result?

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

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

NEW QUESTION 123
Given:

```
public class Case {
    public static void main(String[] args) {
        String product = "Pen";
        product.toLowerCase();
        product.contact(" 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

NEW QUESTION 124

Given:

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

Which is true?

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

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

NEW QUESTION 125
Given the code fragment:

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

What is the result?

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

Answer: C

NEW QUESTION 126
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.

NEW QUESTION 127
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

NEW QUESTION 128
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.

NEW QUESTION 129
Given:

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

What is the result?

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

Answer: D

NEW QUESTION 130

Given:

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

And the code fragment:

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

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

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

Answer: C

NEW QUESTION 131

Given:

```
public class Test {
public static void main(String[] args) {
try {
String[] arr =new String[4];
arr[1] = "Unix";
arr[2] = "Linux";
arr[3] = "Solarios";
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.

NEW QUESTION 132
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

NEW QUESTION 133
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.oraclexom/javase/tutorial/java/landl/defaultmethods.html

NEW QUESTION 134
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.

NEW QUESTION 135
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

NEW QUESTION 136
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?

Option A.

```
while(key == myarray[index++1]) {
    isFound == ture;
}
```

Option B.

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

Option C.

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

Option 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

NEW QUESTION 137

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("Overriden " + 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

NEW QUESTION 138
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 viod 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?

Option A.

```
SpecialException: Thrown at end of doSomething() method
```

Option B.

```
Error in thread "main" java.lang.ArrayIndexOutOfBoundsError
```

Option C.

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException:4
at ExceptionTest.doSomething(ExceptionTest.java:13)
at ExceptionTest.main(ExceptionTest.java:4)
```

Option 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

NEW QUESTION 139
Given the code fragments:

```
interface Contract { }
class Super implements Contract { }
class Sub extends Super { }
public class Ref {
    public static void main(String[] args) {
        List objs = new ArrayList();
        Contract c1 = new Super();
        Contract c2 = new Sub();    //line n1
        Super s1 = new Sub();
        objs.add(c1);
        objs.add(c2);
        objs.add(s1);    //line n2
        for(Object itm:objs) {
    System.out.println(itm.getClass().getName());
        }
    }
}
```

What is the result?

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

Answer: D

NEW QUESTION 140
Given:

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

What is the result?

A. true true
B. true false

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

Answer: E

NEW QUESTION 141
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

NEW QUESTION 142
.....