



Vendor: Oracle

Exam Code: 1Z0-803

Exam Name: Java SE 7 Programmer I

Version: DEMO

QUESTION 1

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: CD

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 2

Given:

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

What is the result?

- A. hello
- B. default
- C. Compilation fails

- D. The program prints nothing
- E. An exception is thrown at run time

Answer: A

Explanation:

The program compiles fine.

The program runs fine.

The output is: hello

QUESTION 3

Given:

```
public class SampleClass {
    public static void main(String[] args) {
        AnotherSampleClass asc = new AnotherSampleClass();
        SampleClass sc = new SampleClass();
        // TODO code application logic here
    }
}
class AnotherSampleClass extends SampleClass {
}
```

Which statement, when inserted into line "// TODO code application logic here ", is valid change?

- A. asc = sc;
- B. sc = asc;
- C. asc = (object) sc;
- D. asc= sc.clone ()

Answer: B

Explanation:

Works fine.

QUESTION 4

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 5

Which code fragment is illegal?

- A.

```
class Base1{  
    abstract class Abs1{ }}
```
- B.

```
abstract class Abs1{  
    void doit () { }  
}
```
- C.

```
class Base1 {  
    abstract class Abs1extends Base1 {
```
- D.

```
abstract int var1= 89;
```

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 6

Given the code fragment:

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

What is the result?

- A. 1

- 2
B. 0
1
C. 1
1
D. 2
2

Answer: A

Explanation:

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

QUESTION 7

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. AStringIndexOutOfBoundsException is thrown at runtime
- D. AnArrayIndexOutOfBoundsException 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 8

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?

- 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

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 9

Given:

```
#1
package handy.dandy;
public class KeyStroke {
    public void typeExclamation() {
        System.out.println("!")
    }
}

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

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

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

Answer: CEF

Explanation:

Three separate solutions:

C: the full class path to the method must be stated (when we have not imported the package)

E: We can import the hold dandy class

F: we can import the specific method

Thank You for Trying Our Product

PassLeader Certification Exam Features:

- ★ More than **99,900** Satisfied Customers Worldwide.
- ★ Average **99.9%** Success Rate.
- ★ **Free Update** to match latest and real exam scenarios.
- ★ **Instant Download** Access! No Setup required.
- ★ Questions & Answers are downloadable in **PDF** format and **VCE** test engine format.
- ★ Multi-Platform capabilities - **Windows, Laptop, Mac, Android, iPhone, iPod, iPad**.
- ★ **100%** Guaranteed Success or **100%** Money Back Guarantee.
- ★ **Fast**, helpful support **24x7**.



View list of all certification exams: <http://www.passleader.com/all-products.html>



Microsoft



ORACLE



CITRIX



JUNIPER
NETWORKS



EMC²
where information lives[®]

10% Discount Coupon Code: STNAR2014