TEST Java General

01m : 02s
to test end

## ☆ Multiple Choice

Which is the most restrictive level of access that allows a subclass in any package to access the members of a superclass?

### Pick one of the choices

○ **public**

○ **private**

● **protected**

○ **transient**

Clear selection

## ☆ Java interface implementation

Given the following sample code:

```
public interface Base {
    boolean m1 ();
    byte m2(short s);
}
```

Which 2 of the following fragments of code do compile?:

### Pick the correct choices

☑ interface Base2 implements Base {}

```
        }
    }
☑ abstract class Class2 implements Base {}
☑ abstract class Class2 implements Base {
        public boolean m1() {
            return (7 > 4);
        }
    }
□ abstract class Class2 implements Base {
        protected boolean m1() {
            return (5 > 7)
        }
    }
```

Clear selection

## ☆ Multiple Choice

Which is the most restrictive level of access that allows the members of a class to access the members of another class in the same package?

### Pick one of the choices

○ **Public**

○ **abstract**

○ **Protected**

○ **synchronized**

● **default access**

Clear selection

## ☆ Java basic operators

```
    public static void main(String [] args) {
        int x=20;
        String sup = (x < 15) ? "small" : (x < 22)? "tiny" : "huge";
        System.out.println(sup);
    }
}
```

### Pick one of the choices

○ **small**

● **Tiny**

○ **huge**

○ **Compilation fails**

Clear selection

## ☆ Multiple Choice

Which two of the following sentences cause a compilation error in Java?

### Pick the correct choices

☑ **float[ ] f = new float(3);**

☑ **float f2[ ] = new float[ ];**

□ **float[ ]f1 = new float[3];**

□ **float f3[ ] = new float[3];**

□ **float f5[ ] = {1.0f, 2.0f, 2.0f};**

Clear selection

Given the following Java class:

```
class A {
    protected int method1(int a, int b) {
        return 0;
    }
}
```

Which of the following is a valid method declaration in a subclass of A?

## Pick one of the choices

- ● **public int method1(int a, int b) {return 0; }**
- ○ **private int method1(int a, int b) { return 0; }**
- ○ **public short method1(int a, int b) { return 0; }**
- ○ **static protected int method1(int a, int b) { return 0; }**

Clear selection

# ☆ Java inner class implementation

Given the following code:

```
public class Outer {

    public void someOuterMethod(){
        //Line 5
    }

    public class Inner { }

    public static void main(String[] argv) {
        Outer ot = new Outer();
        //Line 10
    }
}
```

Which of the following fragments of code do compile?

○ new Inner(); //At line 10

● new ot.Inner(); //At line 10

○ new Outer.Inner(); //At line 10

Clear selection

## Multiple Choice

Which of the following sentences creates an array instance?

Pick one of the choices

● **int[ ] ia = new int[15];**

○ **float fa = new float[20];**

○ **char[ ] ca = "Some String";**

○ **int ia[ ] [ ] = { 4, 5, 6 }, { 1,2,3 };**

Clear selection

## Java objects references and equality

What are the outputs of the following code? (Choose all that apply)

```
public class Client {
     int codClient;
     String name;

    public Client(int codClient, String name) {
        this.codClient = codClient;
        this.name = name;
    }
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null) return false;
        if (getClass() != obj.getClass()) return false;
        Client other = (Client) obj;
        if (codClient != other.codClient)return false;
        return true;
    }

    public static void main(String[] args){
        Client c1 = new Client(1, "Pepe");
        Client c2 = c1;
        c2.codClient = 2;
        Client c3 = new Client(3, "Jose");
        Client c4 = new Client(1, "Pepe");
        Client c5 = new Client(3, "Pepe");

        if (c1 == c2) System.out.println("ONE");
        if (c1 == c4) System.out.println("TWO");
        if (c3 == c5) System.out.println("THREE");
        if (c1.equals(c4)) System.out.println("FOUR");
        if (c3.equals(c5)) System.out.println("FIVE");
    }
}
```

## Pick the correct choices

□ **ONE**

□ **TWO**

□ **THREE**

□ **FOUR**

□ **FIVE**

Clear selection

# ☆ Java equals() and hashCode()

## Pick the correct choices

□ If the hashCode() comparison == returns true, the equals() method must returns true.

□ If the hashCode() comparison == returns true, the equals() method might returns true.

□ If the equals() method returns true, the hashCode() comparison == might return false.

□ If the equals() method returns false, the hashCode() comparison == might return true.

□ If the hashCode() comparison != returns true, the equals() method might returns true.

Clear selection

10
11
12
13
14
15
16
17
18
19
20
21
22
23

# ☆ Java Collections implementations

Which implementation allows to store elements in the form of *key-value* pairs and grants synchronized access?

## Pick one of the choices

○ java.util.SortedMap

○ java.util.TreeMap

○ java.util.TreeSet

○ java.util.Hashtable

Clear selection

# ☆ Multiple Choice

Which one of the following sentences causes a compilation error in Java?

25

26

Pick one of the choices

○ int[ ] scores = {3, 5, 7};

○ int [ ][ ] scores = {2,7,6}, {9,3,45};

○ String cats[ ] = {"Fluffy", "Spot", "Zeus"};

○ boolean results[ ] = new boolean [] {true, false, true};

○ Integer results[ ] = {new Integer(3), new Integer(5), new Integer(8)};

Clear selection

## ☆ Java Collections interfaces

Which type allows to store elements in the form of *key-value* pairs?

Pick one of the choices

○ java.util.Map

○ java.util.Set

○ java.util.List

○ java.util.Collection

Clear selection

## ☆ Multiple Choice

Which interface is implemented by *java.util.Hashtable*?

○ **java.util.List**

○ **java.util.HashTable**

○ **java.util.Collection**

Clear selection

## ☆ Multiple Choice

You need to store elements in a *Collection* that guarantees there will be no duplicates and that all its elements are accessed in natural order. Which implementation must you use?

### Pick one of the choices

○ **java.util.Map**

○ **java.util.Set**

○ **java.util.List**

○ **java.util.Collection**

Clear selection

## ☆ Multiple Choice

Which type of *Collection* allows to grow or reduce its size and provides indexed access to its elements, but their operations are not *synchronized*?

### Pick one of the choices

○ **java.util.HashSet**

○ **java.util.ArrayList**

Clear selection

## Multiple Choice

Which implementation allows to store elements in the form of *key-value* pairs and retrieve them based on FIFO (first-in, first-out) strategy?

### Pick one of the choices

○ **java.util.ArrayList**

○ **java.util.LinkedHashMap**

○ **java.util.HashMap**

○ **java.util.TreeMap**

Clear selection

## Java Exceptions 1

What is the output of the following Java program?

```
class Exc0 extends Exception { }

class Exc1 extends Exc0 { } /* Line 2 */

public class Test {

    public static void main(String args[]){
        try {
            throw new Exc1(); /* Line 9 */
        } catch (Exc0 e0) { /* Line 11 */
            System.out.println("Ex0 caught");
        } catch (Exception e) {
```

### Pick one of the choices

○ **Ex0 caught**

○ **exception caught**

○ **Falla la compilación porque hay un error en la línea 2**

○ **Falla la compilación porque hay un error en la línea 9.**

Clear selection

## ☆ Java Exceptions 2

What is the output of the following program?

```
public class MyProgram {

    public static void main(String args[]){
        try {
            System.out.print("Hello world ");
        } finally {
            System.out.println("Finally executing ");
        }
    }
}
```

### Pick one of the choices

○ **Compilation fails because there is no Exception declared to be thrown**

○ **Compilation fails because there is no "catch()" block declared**

○ **Hello world.**

○ **Hello world Finally executing**

Clear selection

TEST Java General

01m : 02s
to test end

## ☆ Java Exceptions 3

What is the output of the following program?

```
public class X {

    public static void main(String [] args) {
        try {
            badMethod(); /* Line 7 */
            System.out.print("A");
        } catch (Exception ex) { /* Line 10 */
            System.out.print("B"); /* Line 12 */
        } finally { /* Line 14 */
            System.out.print("C"); /* Line 16 */
        }
        System.out.print("D"); /* Line 18 */
    }

    public static void badMethod(){
        throw new RuntimeException();
    }
}
```

### Pick one of the choices

○ **AB**

○ **BC**

○ **ABC**

○ **BCD**

Clear selection

## ☆ Java Exceptions 4

What is the output of the following program?

```
        System.out.print("throwit ");
        throw new RuntimeException();
    }

    public static void main(String [] args) {
        try {
            System.out.print("hello ");
            throwit();
        } catch (Exception re ) {
            System.out.print("caught ");
        } finally {
            System.out.print("finally ");
        }
        System.out.println("after ");
    }
}
```

## Pick one of the choices

○ **hello throwit caught**

○ **Compilation fails**

○ **hello throwit RuntimeException caught after**

○ **hello throwit caught finally after**

Clear selection

# ☆ Java Exceptions 5

What is the output of the following program?

```
public class X {
    public static void main(String [] args) {
        try {
            badMethod();
            System.out.print("A");
        } catch (Exception ex) {
            System.out.print("B");
        } finally {
            System.out.print("C");
    public static void badMethod(){
        throw new Error(); /* Line 22 */
    }
}
```

## Pick one of the choices

○ **ABCD**

○ **Compilation fails**

○ **Prints C and then shows an error message.**

○ **Prints BC and then shows an error message.**

Clear selection

# ☆ Java Exceptions 6

What is the output of the following Java program?

```
try {
    int x = 0;
    int y = 5 / x;
} catch (Exception e) {
    System.out.println("Exception");
} catch (ArithmeticException ae) {
    System.out.println(" Arithmetic Exception");
}
System.out.println("finished");
```

## Pick one of the choices

○ **Finished**

○ **Exception**

○ **Fails to compile**

Velocity PARTNERS

TEST Java General

01m : 02s
to test end

## ☆ Multiple Choice

Which of the following methods should be used to execute a Thread?

### Pick one of the choices

○ **init();**

○ **start();**

○ **run();**

○ **resume();**

Clear selection

## ☆ Multiple Choice

Which method must be implemented in a class implementing *java.lang.Runnable*?

### Pick one of the choices

○ **void run()**

○ **public void run()**

○ **public void start()**

○ **void run(int priority)**

Clear selection

Velocity PARTNERS

TEST Java General

01m : 02s
to test end

Which Java class or interface declares the methods wait(), notify(), and notifyAll()?

## Pick one of the choices

○ **Object**

○ **Thread**

○ **Runnable**

○ **Class**

Clear selection

## ☆ Multiple Choice

On Java, which of the following lines of code starts the executions of the thread?

```
class X implements Runnable {

    public static void main(String args[]){
        /* Missing code? */
    }

    public void run() {}
}
```

## Pick one of the choices

○ **Thread t = new Thread(X);**

○ **Thread t = new Thread(X); t.start();**

○ **X run = new X(); Thread t = new Thread(run); t.start();**

○ **Thread t = new Thread(); x.run();**

Clear selection

TEST Java General

01m : 02s
to test end

## ☆ Multiple Choice

Which are true? (Choose all that apply)

### Pick the correct choices

- □ **When a thread is waiting as a result of wait(), it releases it lock**
- □ **The notify() method is defined in the class java.lang.Thread**
- □ **To call wait(), an object must own the lock of the thread**
- □ **The difference between notify() and notifyAll() is that notifyAll() notifies all waiting threads, regardless of the object they are waiting on**
- □ **The notifyAll() method must be called from a synchronized context.**
- □ **The notify method causes a thread to inmediately release it lock**

Clear selection

Continue

About Privacy Policy Terms of Service