Preguntas explicadas

1.- Which declaration initializes a boolean variable?

- a) boolean m = null
- b) Boolean j = (1<5)
- c) boolean k = 0
- d) boolean h = 1

Opcion correcta b.

Esta es correcta ya que lo que realiza primero es una evaluacion y lo que regresa la evaluacion en un valor boleano.

2.- What is the DTO pattern used for?

- a) To Exchange data between processes
- b) To implement the data Access layer
- c) To implement the presentation layer

Opcion correcta a.

El patron DTO es un patron usado para transferir datos entre aplicaciones o subsistemas.

Los casos tipicos es donde necesitas pasar datos estre diferentes capas o diferentes sistemas.

3.- Int a = 10; int b = 37; int z = 0; int w = 0;

If (a==b) {z=3; } else if (a>b) {z=6;}

w = 10 * z;

What is the result?

- a) 30

- b) 0
- c) 60

Opcion correcta b.

Si bien no tiene problemas de compilación ya que compara valores del mismo tipo al realizar la comparacion (a==b) no son iguales por lo que regresa un valor “false”.
Al evaluar (a>b) tambien es falso, por lo que z mantiene su valor con el que se inicializó por lo que la respuesta es 0.

```
class Class1{String v1;}
class Class2{
      Class1 c1;
      String v2;
}
class Class3 {Class2 c1; String v3;}
```

4.- Which three options correctly describe the relationship between the classes?

- A. Class2 has-a v3.
- B. Class1 has-a v2.
- C. Class2 has-a v2.
- D. Class3 has a v1.
- E. Class2 has-a Class3.
- F. Class2 has-a Class1.

Respuestas correctas c,f.

El has-a se da cuando se hace referencia a una clase dentro de ota clase como en la opcion c , f.

5-

```
      try {
          // assume "conn" is a valid Connection object
          // assume a valid Statement object is created
          // assume rollback invocations will be valid
          // use SQL to add 10 to a checking account
          Savepoint s1 = conn.setSavePoint();
          // use SQL to add 100 to the same checking account
          Savepoint s2 = conn.setSavePoint();
          // use SQL to add 1000 to the same checking account
```

```
        // insert valid rollback method invocation here
    } catch (Exception e) {}
```

What is the result?

A) If conn.rollback(s1) is inserted, account will be incremented by 10.
B) If conn.rollback(s1) is inserted, account will be incremented by 1010.
C) If conn.rollback(s2) is inserted, account will be incremented by 100.
D) If conn.rollback(s2) is inserted, account will be incremented by 110.
E) If conn.rollback(s2) is inserted, account will be incremented by 1110.

Pendiente

6-

Which two statements are true an Abstract ?

A) An abstract class can implement an interface.
B) An abstract class can be extended by an interface.
C) An interface CANNOT be extended by another interface.
D) An interface can be extended by an abstract class.
E) An abstract class can be extended by a concr-ete class.
F) An abstract class CANNOT be extended by an abstract class.

Respuesta correcta a, b.

Una interfaz solo puede implementar o extender otra interface.
Las interfaces solo pueden ser implementadas por clases no extedidas.

Pregunta 7

```
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:21) at Main.main (Main.java:15).
B) Before if clause Exception in thread "main" java.lang.Exception at Main.doSomething (Main.java:21) at Main.main (Main.java:15) After if clause
C) Exception in thread "main" java.lang.Exception at Main.doSomething (Main.java:21) at Main.main (Main.java:15)
D) Before if clause After if clause

respuesta correcta a, b

Lo primero que pasa es que en el método main se manda a llamar el metodo doSomething() y lo primero que hace este metodo es imprimir "Before if clause"despues de esto dependiendo del si el metodo random da un valor meno o mayo a 5 da una excepcion y que caso de que retorne la excepción ya no se ejecuta mas del programa por lo que ya o se ejecutaria After if clause

Pregunta 8

```
public class MyFive {
    public static void main(String[] args) {
        //short kk = ?;
        short ii;
        short jj = 0;
        for (ii = kk; ii > 6; ii-=1) {
            jj++;
        }
        System.out.println("jj = " + jj);
    }
}
```

What value should replace kk in line 18 to cause jj = 5 to be output?

A) -1
B) 1
C) 5
D) 8

E) 11

Respuesta correcta es e.

Paque la condicion del bucle se cumpla kk tienen que ser mayo a 6 por lo que cual quier numero menor a 6 no es valido.
El loop va decrementando el valor hasta que se haga menor a 6 por que el unico que puede iterar 5 veces es 11.

```
public class Simple {

    public float price;
    public static void main(String[] args) {

        Simple price = new Simple();
        price = 4;
    }
}
```

Pregunta 9

What will make this code compile and run?

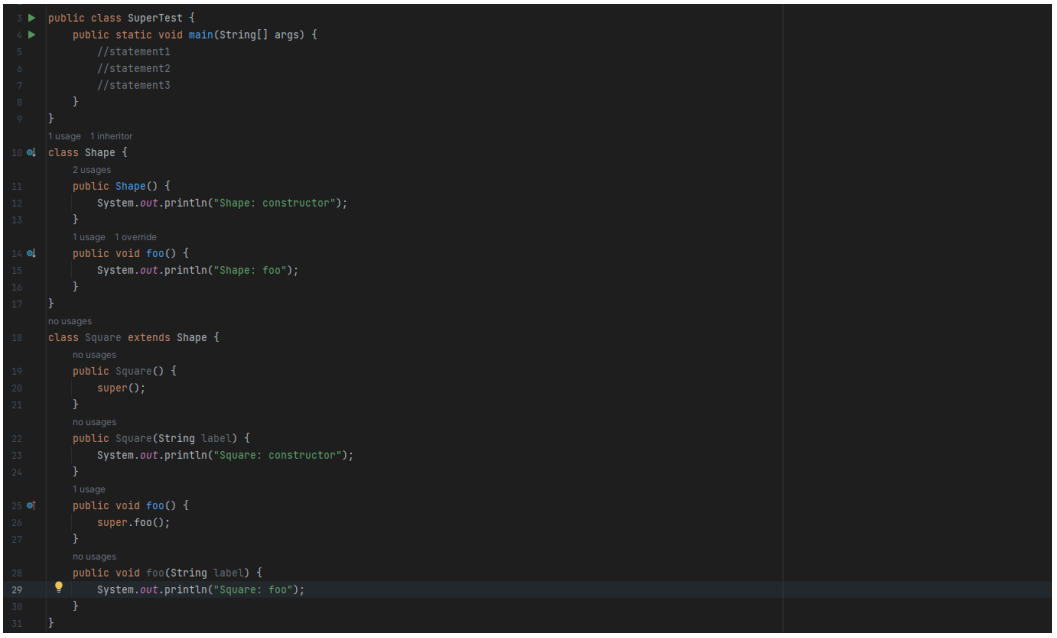
A. Change line 3 to the following: public int price;
B. Change line 7 to the following: int price = new Simple();
C. Change line 7 to the following: float price = new Simple();
D. Change line 7 to the following: price = 4f;
E. Change line 7 to the following: price.price = 4;

respuesta correcta

Al ser un metodo estatico no puede acceder a la variable no-static hasta crear una instancia de Simple.

Un Set pertenece a collection y no permite el tener elementos duplicados en su interior.

Los otros dos puntos hacen referencia a List y Map, respectivamente.



```java
public class SuperTest {
    public static void main(String[] args) {
        //statement1
        //statement2
        //statement3
    }
}
1 usage  1 inheritor
class Shape {
    2 usages
    public Shape() {
        System.out.println("Shape: constructor");
    }
    1 usage  1 override
    public void foo() {
        System.out.println("Shape: foo");
    }
}
no usages
class Square extends Shape {
    no usages
    public Square() {
        super();
    }
    no usages
    public Square(String label) {
        System.out.println("Square: constructor");
    }
    1 usage
    public void foo() {
        super.foo();
    }
    no usages
    public void foo(String label) {
        System.out.println("Square: foo");
    }
}
```

Pregunta 11

What should statement1, statement2, and statement3, be respectively, in order to produce the result?

Shape: constructor

Shape: foo

Square: foo

A. Square square = new Square("bar"); square.foo("bar"); square.foo();
B. Square square = new Square("bar"); square.foo("bar"); square.foo("bar");
C. Square square = new Square(); square.foo(); square.foo(bar);
D. Square square = new Square(); square.foo(); square.foo("bar");
E. Square square = new Square(); square.foo(); square.foo();

respuesta correcta D.

El constructor de la clase Square crea al constructor de la classe shape que imprime Sahape : constructor.

El metodo foo que se ejecuta siempre el del objeto creado que es Square y ese metodo ejecuta el metodo de la clase padre por lo que imprime Shape: foo.

El metodo que se ejecuta al llamar a square.foo("bar"); imprime square:foo ya que que metodo manda in string como argumento.

```java
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 { }
```

Pregunta 12

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

respuesta

Explicación:

getClass evalúa el la clase de el objeto, y no de la variable de referencia, por lo que al pasarle a sc el objeto AnotherSampleClass() , nos da este resultado.

```java
public abstract class Wow {
    private int wow;
    public Wow(int wow) { this.wow = wow; }
    public void wow() { }
    private void wowza() { }
}
```

14-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.

Compila ya que los metodos que son privados son concretos y no abstractos.
Las clases abstractas puede variables de instancia.
Las clases abstractas pueden tener constructores

14- The sigueton pattern allows :

A. Have a single instance of a class and this instance cannot be used by other classes.
B. Having a single instance of a class, while allowing all classes have access to that instance.
C. Having a single instance of a class that can only be accessed by the first methods that calls it.

Respuesta B.

Tener una única instancia de una clase, permitiendo al mismo tiempo que todas las clases tengan acceso a esa instancia.

```
public static void main(String[] args) {
    String[] table = {"aa", "bb", "cc"};
    int ii = 0;
    for (String ss : table) {
        while (ii < table.length) {
            System.out.println(ii); ii++;
            break;
        }
    }
}
```

15-How many times is 2 printend?

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

respuesta B

El bucles foreach realiza 3 iteraciones y depues entra al buche while mientas “ii” sea menor que la longitud del arreglo(3) pero al tener el “brake” basicamente solo itera una vez por cada iteracion del bucle foreach e incrementa uno por lo que solamente imprime una vez el dos.

```
public static void main(String[] args) {
    int [][] array2D = { {0, 1, 2}, {3, 4, 5, 6} };
    System.out.print(array2D[0].length + "");
    System.out.print(array2D[1].getClass().isArray() + "" );
    System.out.println(array2D[0][1]);
}
```

16-What is the result?

A. 3false1
B. 2true3
C. 2false3
D. 3true1
E. 3false3
F. 2true1
G. 2false1

respuesta D

La longitud del arreglo en su posicion 0 es de 3.

El arreglo en su posicion 1 al obtener su clase y verificar si es un arreglo, la respuesta es true.

17.- In Java the difference between throws and throw is:

A. Throws throws an exception and throw indicates the type of exception that the method.
B. Throws is used in methods and throw in constructors.
C. Throws indicates the type of exception that the method does not handle and throw an exception.

respuesta correcta c

throws: Se utiliza en la declaración de un método para indicar que el método puede lanzar una o más excepciones. Esto significa que el método no maneja esas excepciones por sí mismo, y cualquier código que llame a ese método debe manejar las excepciones que podrían ser lanzadas

18.- What is the result?

```
class Person {
String name = "No name";
public Person (String nm) {name=nm}
}
class Employee extends Person {
      String empID = "0000";
      public Employee(String id) { empID " //18
}
}
public class EmployeeTest {
      public static void main(String[] args) {
            Employee e = new Employee("4321");
            System.out.printiln(e.empID);
      }
}
```

A. 4321.
B. 0000.
C. An exception is thrown at runtime.
D. Compilation fails because of an error in line 18.

?????

19.- Which is true?

```
class Building {}
      public class Barn extends Building{
            public static void main(String[] args){
                  Building build1 = new Building();
                  Barn barn1 = new Barn();
                  Barn barn2 = (Barn) build1; //10
                  Object obj1 = (Object) build1; //11
                  String str1 = (String) build1; //12
                  Building build2 = (Building) barn1; //13
      }
}
```

A. If line 10 is removed, the compilation succeeds.
B. If line 11 is removed, the compilation succeeds.
C. If line 12 is removed, the compilation succeeds.
D. If line 13 is removed, the compilation succeeds.
E. More than one line must be removed for compilation to succeed.

Respuesta C

No puedes hacer un casting de build1 a String ya que no hereda la clase.(String no puede heredar).

20.- What is the result?

```
class Atom {
      Atom() {System.out.print("atom ");}
}
class Rock extends Atom {
      Rock(String type) {System.out.print(type);}
}
public class Mountain extends Rock {
      Mountain(){
            super("granite ");
            new Rock("granite ");
      }
      public static void main(String[] a) {new Mountain();}
      }
```

A. Compilation fails.
B. Atom granite.
C. Granite granite.
D. Atom granite granite.
E. An exception is thrown at runtime.
F. Atom granite atom granite

respuesta F.

Al llamar al constructor de mountain con super(“granite”) lo primero que hace es ejecutar el constructor de la clase abuela imprimiendo “atom” y lugo el de la clase padre imprimiendo granite (argumento enviado) y o mismo se repite al llamar al objeto Rock.

Pregunta 21

Which statement is true?

A. 420 is the output.

B. An exception is thrown at runtime.

C. All constructors must be declared public.

D. Constructors CANNOT use the private modifier.

E. Constructors CANNOT use the protected modifier.

```
class ClassA {
     public int numberOfInstances;
     protected ClassA(int numberOfInstances) {
          this.numberOfInstances = numberOfInstances;
     }
}
public class ExtendedA extends ClassA {
     private ExtendedA(int numberOfInstances) {
          super(numberOfInstances);
     }
     public static void main(String[] args) {
          ExtendedA ext = new ExtendedA(420);
          System.out.print(ext.numberOfInstances);
     }
}
```

respuesta correcta A.

Los constructores si pueden tener modificadores de acceso que no sean públicos. Un ejemplo de eso es el patron singleton.

Pregunta 22

What is the result?

A. 4 Null.

B. Null 4.

C. An IllegalArgumentException is thrown at run time.

D. 4 An ArrayIndexOutOfBoundsException is thrown at run time.

```
public class Test {
     public static void main(String[] args) {
          int[][] array = { {0}, {0,1}, {0,2,4}, {0,3,6,9},
{0,4,8,12,16} };
     System.out.println(array[4][1]);
     System.out.println(array[1][4]);
     }
}
```

respuesta D

Imprime 4 ya que los indices estan bien declarados y existe dentro del arreglo.

En la segunda impresión aarroja un excepción ya que el array en la posicion 1 solo tiene una longitud de 1

Pregunta 23

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 NullPointException is thrown at runtime.

F. A StringArrayIndexOutOfBoundsException is thrown at runtime.

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

respuesta c

La longitud del estring es de 11 pero los indices empiezan desde 0 porlo que solo se puede acceder hasta el indice 10.

Pregunta 24

What is the result?

A. The program prints 1 then 2 after 5 seconds.

B. The program prints: 1 thrown to main.

C. The program prints: 1 2 thrown to main.

D. The program prints: 1 then t1 waits for its notification.

```
public class Bees {
     public static void main(String[] args) {
          try {
               new Bees().go();
          }catch (Exception e) {
          System.out.println("thrown to main");
          }
     }

     synchronized void go() throws InterruptedException {
          Thread t1 = new Thread();
          t1.start();
          System.out.print("1 ");
          t1.wait(5000);
          System.out.print("2 ");
     }
}
```

?????

¿Cuál será el resultado?

```
public class SampleClass {
     public static void main(String[] args) {
          SampleClass sc, scA, scB;
          sc = new SampleClass();
          scA = new SampleClassA();
          scB = new SampleClassB();
          System.out.println("Hash is: " + sc.getHash() +
          ", " + scA.getHash() + ", " + scB.getHash());
     }
     public int getHash() {
          return 111111;
     }
}
class SampleClassA extends SampleClass {
     public int getHash() {
          return 44444444;
     }
}
class SampleClassB extends SampleClass {
     public int getHash() {
          return 999999999;
     }
}
```

a) Compilation fails
b) An exception is thrown at runtime
c) There is no result because this is not correct way to determine the hash code
d) Hash is: 111111, 44444444, 999999999.

sc es una instancia de SampleClass, scA es una instancia de SampleClassA y scB es una instancia de SampleClassB. Tanto scA y scB están haciendo un Override al método getHash();. El método main imprimirá los valores de las instancias, dando como resultado sc `111111`, scA `44444444` y scB `999999999`