Taller de Programación APREF 2020

https://tinyurl.com/Taller-APREF-2020

Programación orientada a objetos

1. ¿Qué es un objeto? ¿Cómo se compone? Ejemplifique.
2. ¿Qué es una clase? ¿Qué es una instancia?
3. ¿Cómo se crea un objeto? ¿Cuáles son los pasos que se siguen en Java para la creación de un Objeto?

Programación orientada a objetos

Describa (sin implementar) cómo resolvería el siguiente problema. Especifique: qué clases implementaría, cuáles serían sus atributos y métodos (mostrar sólo el encabezado), y qué concepto/s de POO aplicaría en el desarrollo.

Un comercio quiere representar la información de sus vendedores (por comisión y contratado).

Cualquier vendedor se caracteriza por su nombre y la cantidad de ventas realizadas. Los vendedores por comisión son vendedores que se caracterizan además por el monto total vendido. Los vendedores contratados son vendedores que se caracterizan además por su sueldo básico y su antigüedad.

Cualquier vendedor debe saber:

A) Devolver su representación con el siguiente formato “Soy X y realice Y ventas” donde X es el nombre e Y es la cantidad de ventas realizadas.

B) Calcular y devolver su sueldo a cobrar, teniendo en cuenta lo siguiente:

Los vendedores por comisión cobran el 20% del monto total vendido y un plus de 500$ si realizó más de 10 ventas.

Los vendedores contratados cobran el sueldo básico al cual se adicionan 100$ por año de antigüedad y un plus de 50$ por cada venta.

Además las clases deben proveer funcionalidad para acceder al valor de los atributos y modificar su valor.

Programación orientada a objetos

Queremos representar coordenadas. Una “Coordenada” se caracteriza por dos valores “x” e “y”, y debe saber cambiar su ubicación a una determinada por valores recibidos y devolver su representación en formato String.

Dada la siguiente clase:

```
public class Coordenada {
        int x=1, y=1;
        public void cambiarUbicacion (int unX, int unY){
                x = unX;  y = unY;              }
        public String toString (){
                String aux= “Valor x: ” + x + “ Valor y: ” + y;
                return aux;                     }                       }
```

A) Responda ¿Es correcta la implementación planteada? Justificar la respuesta.

B) Escriba las líneas de código necesarias para instanciar una coordenada, dando valores 10 y 15 para x e y respectivamente, e imprimir su representación en consola.

Constructores

1. Describa el concepto de constructor (objetivo, sintaxis de declaración, sintaxis de uso, concepto de sobrecarga).

2. Es posible generar una clase sin un constructor. Justifique.

Constructores

¿La sentencia MiClase c= new MiClase(); es correcta? Justifique.

```
public class MiClase {
        private int a=1, b=1;
        public MiClase(int valorA, int valorB){
                a = valorA; b = valorB;       }
        public void setA (int valor){
                a = valor;                    }
        public void setB (int valor){
                b = valor;                    }
        public int getA (){
                return a;                     }
        public int getB (){
                return b;                     }          }
```

Herencia

1. Defina el concepto de herencia. Describa dos ejemplos donde la herencia sea de utilidad para reutilizar código.

2. Describa los conceptos de clase abstracta y método abstracto, indicando además para qué se usan y sintaxis de declaración en Java.

3. Describa una jerarquía de clases donde aplique el uso de clase abstracta y método abstracto. Justifique la aplicación.

Herencia

Dadas las siguientes clases:

```
class A {
  private int valor = 3;
  public int getValor() {
    return valor;
  }
}
```

```
class B extends A {
  private int valor;
  public void setValor(int v) {
    valor = v;
  }
}
```

y el siguiente código:

```
B varb = new B();
varb.setValor(5);
System.out.println(varb.getValor());
```

¿Es correcto el código? En caso afirmativo ¿Qué imprime?

Herencia

```
public abstract class Alfa{
  private int uno=1;
  public int getUno(){
     return uno;
  }
  public void setUno(int valor){
     uno = valor;
  }
}
```

```
public class Beta extends Alfa{
   private int dos=2;
   private int tres=3;
   public int getDos(){
     return dos;
   }
   public void setDos(int valor){
     dos= valor;
   }
   public int getTres(){
     return tres;
   }
  public void setTres(int valor){
     tres= valor;
    }
}
```

Indique si la siguiente sentencia es correcta (justifique). **Alfa a = new Alfa();**

Dado un objeto instancia de la clase Beta: liste todas las variables de instancia almacenadas por dicho objeto y liste todos los mensajes que podría enviarle a dicho objeto.

Se quiere incorporar el método *mostrarDatos* en la clase Beta para mostrar en consola el estado del objeto. Indique si el siguiente código es correcto (Justifique).

```
public void mostrarDatos(){
        System.out.println(uno);
        System.out.println(dos);
        System.out.println(tres);        }
```

Herencia

```
public abstract class Epsilon{
  private int a=1;
  private int b=2;
  public int getA(){
     return a;
  }
  public void setA(int valor){
     a= valor;
  }
   public int getB(){
     return b;
   }
   public void setB(int valor){
     b= valor;
   }
}
```

```
public class Omega extends Epsilon{
   private int c=3;
   private int d=4;
   public int getC(){
     return c;
   }
  public void setC(int valor){
     c= valor;
    }
   public int getD(){
     return d;
   }
  public void setD(int valor){
     d= valor;
    }
}
```

a) Indique si la siguiente sentencia es correcta (justifique). Epsilon e = new Epsilon();

b) Dado un objeto instancia de la clase Omega: liste todas las variables de instancia almacenadas por dicho objeto y liste todos los mensajes que podría enviarle a dicho objeto.

c) Implemente un método toString que retorne la representación en formato String de un objeto “Omega”. La representación se forma a partir de los valores de todas las variables de instancia almacenadas por el objeto.