17/01/2017

Diferencia entre programación procedural y programación orientada a objetos:

Resuelve los problemas a través de funciones y los otros problemas entre objetos que interactúan entre sí.

Los objetos tienen: estado, comportamiento, identidad.

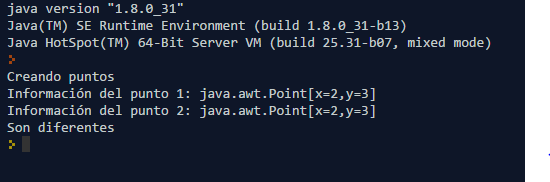
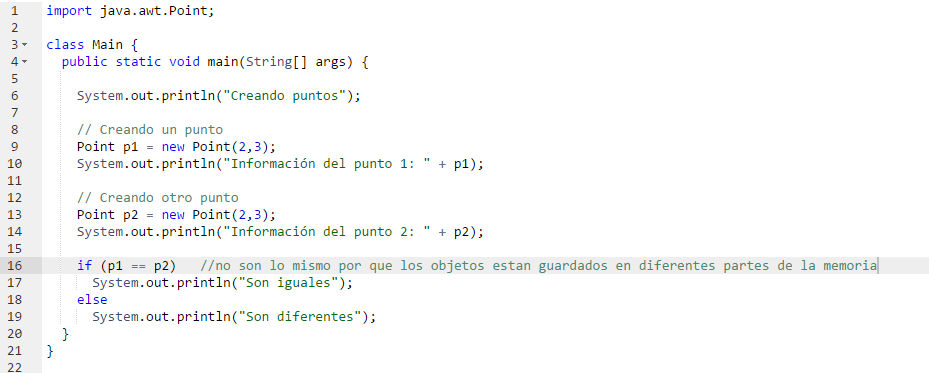
A el molde se le llama clase y a lo que va dentro del molde o lo que puedo hacer dentro de ellos se le llama instancia u objeto.

Por ejemplo, si vamos a construir una casa, antes de comenzar a construirla es necesario tener los planos con los cuales podremos construir dicha casa. Sin embargo, los planos es algo abstracto, a partir de esto podemos dar vida al proyecto de la casa.

De igual manera, en el mundo de la programación orientada a objetos, una clase nos permite definir la estructura de una parte de nuestro programa, es algo abstracto, y que no es realmente algo funcional hasta que el momento en que creemos un objeto.

Ahora, un objeto se conoce como la instancia de una clase. ¿Esto qué significa? Que hasta el momento que creemos un objeto es que podremos interactuar con lo que se definió en la clase.

<https://repl.it/FInn/1>

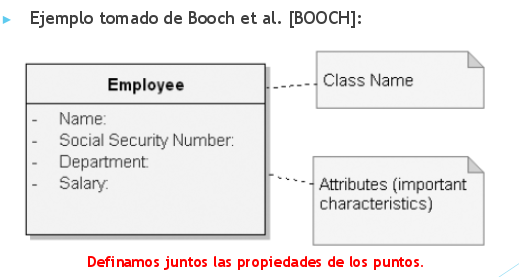


Estado de un objeto: son los valores que contiene el objeto en este caso son las coordenadas de p1 y p2. Y se pregunta poniendo en “if (p1.x ==p2.x && p1.x == p2.y)”

La “x” y la “y” son los atributos de la clase (import java.awt.Point) ya que creo dos coordenadadas (X,Y).

Comportamiento de un objeto: las acciones especificas del objeto

Identidad: es que no se puede usar la misma cosa en mismo lugar al mismo tiempo



Hay diferentes tipos de datos:

double----64 bits

float----32 bits

int-----32 bits

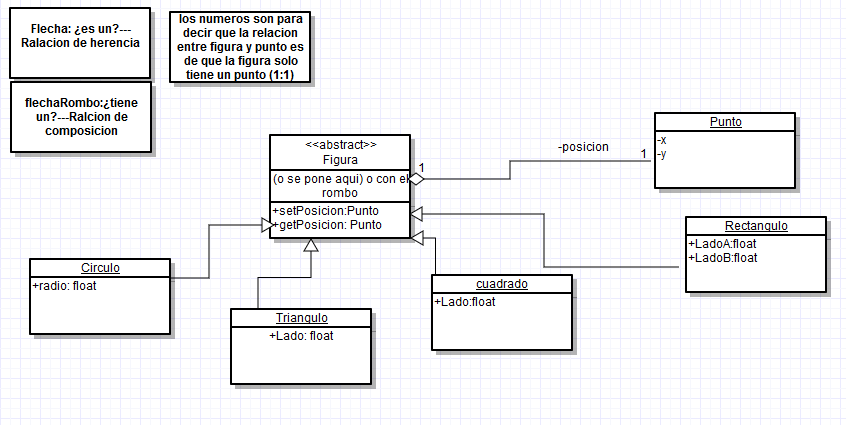
short----16 bits

**Herencia**

La herencia es un mecanismo que permite la definición de una clase a partir de la definición de otra ya existente. La herencia permite compartir automáticamente métodos y datos entre clases, subclases y objetos.

La herencia está fuertemente ligada a la reutilización del código en la OOP. Esto es, el código de cualquiera de las clases puede ser utilizado sin más que crear una clase derivada de ella, o bien una subclase. Hay dos tipos de herencia: Herencia Simple y Herencia Múltiple. La primera indica que se pueden definir nuevas clases solamente a partir de una clase inicial mientras que la segunda indica que se pueden definir nuevas clases a partir de dos o más clases iniciales. Java sólo permite herencia simple.

Las **clases más generales** se recomienda **hacerlas abstractas**, esto quiere decir que no puedes generar nuevos objetos a partir de ella.



El **polimorfismo** es un mecanismo que se da cuando una variable de tipo A se pueden meter instancias tipo B y C, siempre y cuando sean subtipos de A.

Perro animal= new Perro (); es lo mismo que Animal animal=new Perro ();

Esto quiere decir que si Perro es subtipo de Animal y es válido declarar de esa manera.

#----------------------------------------------------------------------------------------------------------------------#

ARCHIVOS:

Formatter txtWritter =new Formatter(“c:\\myfile.txt”)

txt.Writer.format(“Hola mi nombre es Eric”);

txt.Writter.format(“¡hola men!”);

txtWritter.close();