

Práctica 4

Herencia

Objetivo. Trabajar con el concepto de herencia y polimorfismo.

Nota: Trabajar sobre la carpeta “tema4” del proyecto

1-A- Incluya la clase Triángulo a la jerarquía de figuras vista (carpeta tema4). Triángulo debe *heredar* de Figura todo lo que es común y *definir* su constructor y sus atributos y métodos propios. Además debe *redefinir* el método toString.

B- De igual manera, incluya la clase Círculo a la jerarquía de figuras.

C- Añada a la representación String el valor del perímetro. Piense ¿qué método toString debe modificar: el de cada subclase o el de Figura?

D- Añada el método despintar que establece los colores de la figura a línea “negra” y relleno “blanco”. Piense ¿dónde debe definir el método: en cada subclase o en Figura?

E- Realizar un programa que instancie un triángulo y un círculo. Muestre en consola la representación String de cada uno. Pruebe el funcionamiento del método despintar.

2- Queremos representar a los empleados de un club: jugadores y entrenadores.

- Cualquier *empleado* se caracteriza por su nombre, sueldo básico y antigüedad.
- Los *jugadores* son empleados que se caracterizan por el número de partidos jugados y el número de goles anotados.
- Los *entrenadores* son empleados que se caracterizan por la cantidad de campeonatos ganados.

A- Implemente la jerarquía de clases declarando atributos, métodos para obtener/modificar su valor y *constructores* que reciban los datos necesarios.

B- Cualquier empleado debe responder al mensaje calcularEfectividad. La efectividad del entrenador es el promedio de campeonatos ganados por año de antigüedad, mientras que la del jugador es el promedio de goles por partido.

C- Cualquier empleado debe responder al mensaje calcularSueldoACobrar. El sueldo a cobrar es el sueldo básico más un 10% del básico por cada año de antigüedad y además:

- Para los *jugadores*: si el promedio de goles por partido es superior a 0,5 se adiciona un plus de otro sueldo básico.
- Para los *entrenadores*: se adiciona un plus por campeonatos ganados (5000\$ si ha ganado entre 1 y 4 campeonatos; \$30.000 si ha ganado entre 5 y 10 campeonatos; 50.000\$ si ha ganado más de 10 campeonatos).

D- Cualquier empleado debe responder al mensaje toString, que devuelve un String que lo representa, compuesto por nombre, sueldo a cobrar y efectividad.

F- Realizar un programa que instancie un jugador y un entrenador. Informe la representación String de cada uno.

NOTA: para cada método a implementar piense en que clase/s debe definir el método.

Taller de Programación 2022 – Módulo POO

3-A- Implemente las clases para el siguiente problema. Una garita de seguridad quiere identificar los distintos tipos de personas que entran a un barrio cerrado. Al barrio pueden entrar: *personas*, que se caracterizan por nombre, DNI y edad; y *trabajadores*, estos son personas que se caracterizan además por la tarea realizada en el predio.

Implemente constructores, getters y setters para las clases. Además tanto las personas como los trabajadores deben responder al mensaje `toString` siguiendo el formato:

- Personas "Mi nombre es **Mauro**, mi DNI es **11203737** y tengo **70** años"
- Trabajadores "Mi nombre es **Mauro**, mi DNI es **11203737** y tengo **70** años. Soy **jardinero**."

B- Realice un programa que instancie una persona y un trabajador y muestre la representación de cada uno en consola.

NOTA: Reutilice la clase *Persona* (carpeta tema2).

4- Un objeto *visor de figuras* se encarga de mostrar en consola cualquier figura que reciba y también mantiene cuántas figuras mostró. Analice y ejecute el siguiente programa y responda: ¿Qué imprime? ¿Por qué?

<pre>public class VisorFiguras { private int mostradas; public VisorFiguras(){ mostradas=0; } public void mostrar(Figura f){ System.out.println(f.toString()); mostradas++; } public int getMostradas() { return mostradas; } }</pre>	<pre>public class MainVisorFiguras { public static void main(String[] args) { VisorFiguras visor = new VisorFiguras(); Cuadrado c1 = new Cuadrado(10,"Violeta","Rosa"); Rectangulo r= new Rectangulo(20,10,"Azul","Celeste"); Cuadrado c2= new Cuadrado(30,"Rojo","Naranja"); visor.mostrar(c1); visor.mostrar(r); visor.mostrar(c2); System.out.println(visor.getMostradas()); } }</pre>
--	--

5-A- Modifique la clase *VisorFiguras*: ahora debe permitir guardar las figuras a mostrar (a lo sumo 5) y también mostrar todas las figuras guardadas. Use la siguiente estructura.

<pre>public class VisorFigurasModificado { private int guardadas; private int capacidadMaxima=5; private Figura [] vector; public VisorFigurasModificado(){ //completar } public void guardar(Figura f){ //completar } //sigue a la derecha -> }</pre>	<pre>public boolean quedaEspacio(){ //completar } public void mostrar(){ //completar } public int getGuardadas() { return guardadas; } }</pre>
--	---

B- Realice un programa que instancie el visor, guarde dos cuadrados y un rectángulo en el visor y por último haga que el visor muestre sus figuras almacenadas.