Ejercicios Java 4

31. Creación de una clase persona con métodos y atributos

**· Persona.java**

**public** **class** Persona {

String nombre;

**int** edad;

**public** Persona(String nombre, **int** edad) { //Constructor

**this**.nombre = nombre;

**this**.edad = edad;

}

**public** **void** mostrarDatos() { //Muestra los datos

System.***out***.println("Nombre: " + nombre);

System.***out***.println("Edad: " + edad);

}

}

**· Principal.java**

**public** **class** Principal {

**public** **static** **void** main(String[] args) {

Persona p1 = **new** Persona("Juan", 20);

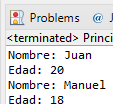
Persona p2 = **new** Persona("Manuel", 18);

p1.mostrarDatos(); //Muestra los datos de la primera persona

p2.mostrarDatos(); //Muestra los datos de la segunda persona

}

}

****

32. Uso de constructores sobrecargados

**· Persona.java**

**public** **class** Persona {

String nombre;

**int** edad;

**public** Persona() { //Constructor

**this**.nombre = "Desconocido";

**this**.edad = 0;

}

**public** Persona(String nombre) { //Constructor

**this**.nombre = nombre;

**this**.edad = 0;

}

**public** Persona(String nombre, **int** edad) { //Constructor

**this**.nombre = nombre;

**this**.edad = edad;

}

**public** **void** mostrarDatos() { //Muestra los datos

System.***out***.println("Nombre: " + nombre);

System.***out***.println("Edad: " + edad);

}

}

**· Principal.java**

**public** **class** Principal {

**public** **static** **void** main(String[] args) {

Persona p1 = **new** Persona();

Persona p2 = **new** Persona("Manuel");

Persona p3 = **new** Persona("Juan", 20);

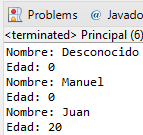
p1.mostrarDatos(); //Muestra los datos de la primera persona

p2.mostrarDatos(); //Muestra los datos de la segunda persona

p3.mostrarDatos(); //Muestra los datos de la tercera persona

}

}



33. Encapsulación y uso de getters y setters

**· Cuenta bancaria.java**

**public** **class** CuentaBancaria {

**private** **double** saldo;

**public** **void** depositar(**double** cantidad) { //Depositar dinero

saldo += cantidad;

}

**public** **void** retirar(**double** cantidad) { //Retirar dinero

saldo -= cantidad;

}

**public** **double** getSaldo() { //Mostrar saldo

**return** saldo;

}

}

**· Principal.java**

**public** **class** Principal {

**public** **static** **void** main(String[] args) {

CuentaBancaria cuenta = **new** CuentaBancaria();

cuenta.depositar(500);

cuenta.retirar(300);

System.***out***.println(cuenta.getSaldo());

}

}



34. Modificadores de acceso en una clase Empleado

**· Empleado.java**

**public** **class** Empleado {

String nombre;

**private** **double** salario;

**protected** String departamento;

**public** Empleado(String nombre, **double** salario, String departamento) {

**this**.nombre = nombre;

**this**.salario = salario;

**this**.departamento = departamento;

}

**public** **void** mostrarDatos() {

System.***out***.println(nombre);

System.***out***.println(salario);

System.***out***.println(departamento);

}

}

**· Subordinado.java**

**public** **class** Subordinado **extends** Empleado {

**public** Subordinado(String nombre, **double** salario, String departamento) {

**super**(nombre, salario, departamento);

}

}

**· Principal.java**

**public** **class** Principal {

**public** **static** **void** main(String[] args) {

Empleado e = **new** Empleado("Juan", 203.43, "Ventas");

Subordinado s = **new** Subordinado("Maria", 184.65, "Finanzas");

e.mostrarDatos();

s.mostrarDatos();

}

}



35. Uso de paquetes en Java

**· Mensaje.java**

**package** utilidades;

**public** **class** Mensaje {

**public** **static** **void** imprimirSaludo() { //Imprime el mensaje

System.***out***.println("¡Hola desde utilidades!");

}

}

**· Principal.java**

**package** utilidades;

**public** **class** Principal {

**public** **static** **void** main(String[] args) {

Mensaje.*imprimirSaludo*();

}

}

****

36. Uso de import static

**· Mensaje.java**

**package** utilidades;

**public** **class** Mensaje {

**public** **static** **void** imprimirSaludo() { //Imprime el mensaje

System.***out***.println("¡Hola desde utilidades!");

}

}

**· Principal.java**

**package** utilidades;

**import** **static** utilidades.Mensaje.\*; //Permite usar los métodos de mensaje sin nombrarlo

**public** **class** Principal {

**public** **static** **void** main(String[] args) {

*imprimirSaludo*();

}

}



37. Creación de una clase Vehículo y herencia

**· Vehiculo.java**

**public** **class** Vehiculo {

String marca;

String modelo;

}

**· Coche.java**

**public** **class** Coche **extends** Vehiculo {

**int** puertas;

**public** Coche(String marca, String modelo, **int** puertas) { //Constructor

**this**.marca = marca;

**this**.modelo = modelo;

**this**.puertas = puertas;

}

**public** **void** mostrarDatos() { //Muestra los datos

System.***out***.println("Marca: " + marca);

System.***out***.println("Modelo: " + modelo);

System.***out***.println("Número de puertas: " + puertas);

}

}

**· Principal.java**

**public** **class** Principal {

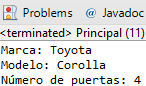
**public** **static** **void** main(String[] args) {

Coche coche = **new** Coche("Toyota", "Corolla", 4);

coche.mostrarDatos(); //Muestra los datos del objeto

}

}

****

38. Polimorfismo con métodos sobreescritos

**· Vehiculo.java**

**public** **class** Vehiculo {

String marca;

String modelo;

**public** **void** describir() { //Imprime el mensaje

System.***out***.println("Soy un vehículo");

}

}

**· Coche.java**

**public** **class** Coche **extends** Vehiculo {

**int** puertas;

**public** Coche(String marca, String modelo, **int** puertas) { //Constructor

**this**.marca = marca;

**this**.modelo = modelo;

**this**.puertas = puertas;

}

**public** **void** mostrarDatos() { //Muestra los datos

System.***out***.println("Marca: " + marca);

System.***out***.println("Modelo: " + modelo);

System.***out***.println("Número de puertas: " + puertas);

}

@Override

**public** **void** describir() { //Imprime el mensaje

System.***out***.println("Soy un coche de marca " + marca + " y modelo " + modelo);

}

}

**· Principal.java**

**public** **class** Principal {

**public** **static** **void** main(String[] args) {

Coche coche = **new** Coche("Toyota", "Corolla", 4);

coche.mostrarDatos(); //Muestra los datos del objeto

coche.describir(); //Imprime el mensaje

}

}

39. Creación de una clase Electrodoméstico y subclase

**· Electrodomestico.java**

**public** **class** Electrodomestico {

String marca;

**double** precio;

}

**· Lavadora.java**

**public** **class** Lavadora **extends** Electrodomestico {

**int** capacidadKg;

**public** Lavadora(String marca, **double** precio, **int** capacidadKg) { //Constructor

**this**.marca = marca;

**this**.precio = precio;

**this**.capacidadKg = capacidadKg;

}

**public** **void** mostrarDatos() { //Muestra los datos

System.***out***.println("Marca: " + marca);

System.***out***.println("Precio: " + precio);

System.***out***.println("Capacidad en KG: " + marca);

}

}

**· Televisor.java**

**public** **class** Televisor **extends** Electrodomestico{

**int** pulgadas;

**public** Televisor(String marca, **double** precio, **int** pulgadas) { //Constructor

**this**.marca = marca;

**this**.precio = precio;

**this**.pulgadas = pulgadas;

}

**public** **void** mostrarDatos() { //Muestra los datos

System.***out***.println("Marca: " + marca);

System.***out***.println("Precio: " + precio);

System.***out***.println("Pulgadas: " + pulgadas);

}

}

**· Figura.java**

**public** **abstract** **class** Figura {

**abstract** **double** calcularArea(); //Método abstracto

}

**· Circulo.java**

**public** **class** Circulo **extends** Figura {

**double** radio;

**public** Circulo(**double** radio) { //Constructor

**this**.radio = radio;

}

@Override

**double** calcularArea() { //Calcula el área del circulo

**return** Math.***PI*** \* radio \* radio;

}

}

**· Rectangulo.java**

**public** **class** Rectangulo **extends** Figura {

**double** base;

**double** altura;

**public** Rectangulo(**double** base, **double** altura) { //Constructor

**this**.base = base;

**this**.altura = altura;

}

@Override

**double** calcularArea() { //Calcula el área del rectángulo

**return** base \* altura;

}

}

**· Principal.java**

**public** **class** Principal {

**public** **static** **void** main(String[] args) {

Lavadora lavadora = **new** Lavadora("LG", 305.21, 34);

Televisor televisor = **new** Televisor("SONY", 486.54, 20);

Circulo circulo = **new** Circulo(2.43);

Rectangulo rectangulo = **new** Rectangulo(5.32, 2.13);

lavadora.mostrarDatos(); //Muestra los datos de la lavadora

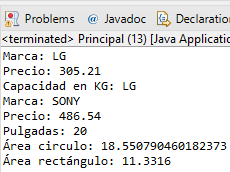
televisor.mostrarDatos(); //Muestra los datos del televisor

System.***out***.println("Área circulo: " + circulo.calcularArea()); //Muestra el área del circulo

System.***out***.println("Área rectángulo: " + rectangulo.calcularArea()); //Muestra el área del rectangulo

}

}



<https://github.com/Rodrigo-Garcia-Ortiz/Programacion.git>