UTN - TUPaD

PROGRAMACIÓN II Trabajo Práctico 3: Introducción a POO

Aldo Manfredi - 2c2025

..

https://onecompiler.com/java

Todo en el repo

****

```
import java.util.*;

public class Main {
    public static void main(String[] args) {
        Persona p1 = new Persona();
        p1.setNombre("KJuan");
        System.out.println(p1.getNombre());
    }
}

// Esta clase no es pública, por lo que puede coexistir en el mismo archivo
// con la clase Main (que sí es pública).
class Persona {
    private String nombre;

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
}
```

****

****

1
Registro de Estudiantes
a. Crear una clase Estudiante con los atributos: nombre, apellido, curso, calificación.
Métodos requeridos: mostrarInfo(), subirCalificacion(puntos), bajarCalificacion(puntos).
Tarea: Instanciar a un estudiante, mostrar su información, aumentar y disminuir calificaciones.

```
// Estudiante.java
```

```
package kata1;
class Estudiante  {
    private String nombre;
    private String apellido;
    private String curso;
    private double calificacion;

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getApellido() {
        return apellido;
    }

    public void setApellido(String apellido) {
        this.apellido = apellido;
    }

    public String getCurso() {
        return curso;
    }

    public void setCurso(String curso) {
        this.curso = curso;
    }

    public double getCalificacion() {
        return calificacion;
    }

    public void setCalificacion(double calificacion) {
        this.calificacion = calificacion;
    }

    // funciones: Métodos requeridos: mostrarInfo(), subirCalificacion(puntos),
    // bajarCalificacion(puntos).
    public void mostrarInfo(){
        System.out.println("Estudiante: " + apellido + " " + nombre);
        System.out.println("Curso: " + curso);
        System.out.println("Calificacion: " + calificacion);
    }

    public double subirCalificacion(double puntos){
```

```
        double nuevaCalificacion = calificacion + puntos;
        calificacion = nuevaCalificacion;
        return calificacion;
    }

    public double bajarCalificacion(double puntos){
        double nuevaCalificacion = calificacion - puntos;
        calificacion = nuevaCalificacion;
        return calificacion;
    }

}
```

```
// Main.java
package kata1;
public class Main {
    public static void main(String[] args) {

        Estudiante p1 = new Estudiante();
        p1.setNombre("juan");
        p1.setApellido("zamora");
        p1.setCurso("bbdd");
        p1.setCalificacion(7.5);
        // funciones: Métodos requeridos: mostrarInfo(), subirCalificacion(puntos),
bajarCalificacion(puntos).
        p1.mostrarInfo();

        p1.subirCalificacion(2.2);
        p1.mostrarInfo();

        p1.bajarCalificacion(5.2);
        p1.mostrarInfo();

    }
}
```

2. Registro de Mascotas
a. Crear una clase Mascota con los atributos: nombre, especie, edad.
Métodos requeridos: mostrarInfo(), cumplirAnios().
Tarea: Crear una mascota, mostrar su información, simular el paso del tiempo y verificar los cambios.

```
// Mascota.java
package kata2;
```

```
public class Mascota {
    // nombre, especie, edad.
    private String nombre;
    private String especie;
    private int edad;

    public String getNombre() {
        return nombre;
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public String getEspecie() {
        return especie;
    }

    public void setEspecie(String especie) {
        this.especie = especie;
    }

    public int getEdad() {
        return edad;
    }

    public void setEdad(int edad) {
        this.edad = edad;
    }

    // Métodos requeridos: mostrarInfo(), cumplirAnios().
    public void mostrarInfo(){
        System.out.println("nombre: " + nombre);
        System.out.println("Especie: " + especie);
        System.out.println("edad: " + edad);
    }

    public int cumplirAnios(){
        int nuevaEdad = edad + 1;
        edad = nuevaEdad;
        return edad;
    }


}


// Main.java
```

```
package kata2;
public class Main {

    public static void main(String[] args) {
        Mascota mascota1 = new Mascota();
        mascota1.setNombre("leopoldo");
        mascota1.setEspecie("gato");
        mascota1.setEdad(3);
        mascota1.mostrarInfo();

        mascota1.cumplirAnios();
        mascota1.mostrarInfo();

    }
}
```

3. Encapsulamiento con la Clase Libro
a. Crear una clase Libro con atributos privados: titulo, autor, añoPublicacion. Métodos requeridos: Getters para todos los atributos. Setter con validación para añoPublicacion.
Tarea: Crear un libro, intentar modificar el año con un valor inválido y luego con uno válido, mostrar la información final.

```
// prueba haciendo todo en un solo archivo Libro.java
import java.time.Year;
class Libro {
    private String titulo;
    private String autor;
    private int añoPublicacion;

    public String getTitulo() {
        return titulo;
    }

    public void setTitulo(String titulo) {
        this.titulo = titulo;
    }

    public String getAutor() {
        return autor;
    }
```

```
    public void setAutor(String autor) {
        this.autor = autor;
    }

    public int getAñoPublicacion() {
        return añoPublicacion;
    }

    public void setAñoPublicacion(int añoPublicacion) {
        int añoActual = Year.now().getValue();
        if (añoPublicacion > 0 && añoPublicacion <= añoActual) {
            this.añoPublicacion = añoPublicacion;
        } else {
            System.err.println("Error: El año de publicación no es válido.");
        }
    }

    public void mostrarInfo(){
        System.out.println("Info solicitada:");
        System.out.println("Título: " + titulo);
        System.out.println("Autor: " + autor);
        System.out.println("Año de publicación: " + añoPublicacion);
    }
}

public class Main {
    public static void main(String[] args) {
        Libro miLibro = new Libro();
        miLibro.setTitulo("rayuela");
        miLibro.setAutor("cortazar");
        miLibro.setAñoPublicacion(1934);
        miLibro.mostrarInfo();

        System.out.println("\n--- Probando validación del año ---");
        miLibro.setAñoPublicacion(2050); // Intento de valor inválido
        miLibro.mostrarInfo();
    }
}
```

4. Gestión de Gallinas en Granja Digital
a. Crear una clase Gallina con los atributos: idGallina, edad, huevosPuestos.
Métodos requeridos: ponerHuevo(), envejecer(), mostrarEstado().
Tarea: Crear dos gallinas, simular sus acciones (envejecer y poner huevos), y mostrar su estado.

```
package kata4;
class Gallina {
    // Atributos: idGallina, edad, huevosPuestos
    private String idGallina;
    private int edad;
    private int huevosPuestos;

    public String getIdGallina() {
        return idGallina;
    }

    public void setIdGallina(String idGallina) {
        this.idGallina = idGallina;
    }

    public int getEdad() {
        return edad;
    }

    public void setEdad(int edad) {
        this.edad = edad;
    }

    public int getHuevosPuestos() {
        return huevosPuestos;
    }

    public void setHuevosPuestos(int huevosPuestos) {
        this.huevosPuestos = huevosPuestos;
    }

    // Métodos: ponerHuevo(), envejecer(), mostrarEstado()
    public void ponerHuevo(){
        this.huevosPuestos++;
    }

    public void envejecer(){
        this.edad++; // sumo 1 a la edad
    }

    public void mostrarEstado(){
        System.out.println("\n--- Estado actual ---");
```

```
        System.out.println("ID gallina: " + idGallina);
        System.out.println("Edad: " + edad);
        System.out.println("Huevos puestos: " + huevosPuestos);
    }
}


public class Main {
    public static void main(String[] args) {
        Gallina galli1 = new Gallina();
        galli1.setIdGallina("ramona");
        galli1.setEdad(1);
        galli1.setHuevosPuestos(19);
        galli1.mostrarEstado();

        galli1.ponerHuevo();
        galli1.envejecer();
        galli1.mostrarEstado();

        Gallina galli2 = new Gallina();
        galli2.setIdGallina("lucre");
        galli2.setEdad(3);
        galli2.setHuevosPuestos(29);
        galli2.mostrarEstado();

        galli2.ponerHuevo();
        galli2.envejecer();
        galli2.mostrarEstado();
    }
}
```

5. Simulación de Nave Espacial
Crear una clase NaveEspacial con los atributos: nombre, combustible.
Métodos requeridos: despegar(), avanzar(distancia), recargarCombustible(cantidad), mostrarEstado().
Reglas: Validar que haya suficiente combustible antes de avanzar y evitar que se supere el límite al recargar.
Tarea: Crear una nave con 50 unidades de combustible, intentar avanzar sin recargar, luego recargar y avanzar correctamente. Mostrar el estado al final.

```
package kata5;
// Archivo: Main.java

public class Main {
    public static void main(String[] args) {
```

```
        // Tarea: Crear una nave. Java usa el constructor por defecto.
        NaveEspacial miNave = new NaveEspacial();
        miNave.setNombre("lucrecia");
        miNave.mostrarEstado();
        miNave.avanzar(60);
        miNave.recargarCombustible(50);
        miNave.avanzar(60);
        miNave.recargarCombustible(20);
        miNave.mostrarEstado();
    }
}


class NaveEspacial {
    private String nombre;
    private int combustible = 50; // Se inicializa directamente
    private final int CAPACIDAD_MAXIMA = 100;
    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public void despegar() {
        System.out.println(" ¡La nave " + nombre + " está despegando!");
    }

    public void avanzar(int distancia) {
        if (combustible >= distancia) {
            combustible -= distancia;
            System.out.println("La nave " + nombre + " avanza " + distancia + "
unidades.");
        } else {
            System.out.println("¡Error! No hay suficiente combustible para avanzar " +
distancia + " unidades.");
        }
    }

    public void recargarCombustible(int cantidad) {
        if (combustible + cantidad <= CAPACIDAD_MAXIMA) {
            combustible += cantidad;
            System.out.println("La nave " + nombre + " ha recargado " + cantidad + "
unidades de combustible.");
        } else {
            int exceso = (combustible + cantidad) - CAPACIDAD_MAXIMA;
            System.out.println("¡Error! La recarga de " + cantidad + " superaría el límite
de combustible.");
            System.out.println("Se supera el límite por: " + exceso + " unidades.");
        }
    }
```

```
    public void mostrarEstado() {
        System.out.println("\n--- Estado de la nave " + nombre + " ---");
        System.out.println("Combustible actual: " + combustible + " / " +
CAPACIDAD_MAXIMA);
        System.out.println("-----------------------------------");
    }
}
```