TP N° 3: Introducción a Java

Materia: Programación II

Nombre: Lopez Leandro

1. Registro de Estudiantes
a. Crear una clase Estudiante con los atributos: nombre, apellido, curso, calificación.
Métodos requeridos: mostrarInfo(), subirCalificacion(puntos), bajarCalificacion(puntos).

```
public class Estudiante {
    public String nombre;
    public String apellido;
    public int curso;
    public int calificacion;


    public void mostrarInfo(){
        System.out.println("Nombre: "+nombre+
                "\nApellido: "+apellido+
                "\nCurso: "+curso+
                "\nCalificacio: "+calificacion);
    }

    public void subirCalificacion(int nota){
        if (nota>0){
            calificacion = nota;
        }
     }

    public void bajarCalificacion(int nota){
        if (nota>0){
            calificacion = nota;
        }
     }

}
```

Tarea: Instanciar a un estudiante, mostrar su información, aumentar y disminuir calificaciones.

```
run:
Nombre: Leandro
Apellido: Lopez
Curso: 1
Calificacio: 0
Nombre: Leandro
Apellido: Lopez
Curso: 1
Calificacio: 9
Nombre: Leandro
Apellido: Lopez
Curso: 1
Calificacio: 5
BUILD SUCCESSFUL (total time: 0 seconds)
```

2. Registro de Mascotas

a. Crear una clase Mascota con los atributos: nombre, especie, edad. Métodos requeridos: mostrarInfo(), cumplirAnios().

```java
public class Mascota {
    String nombre;
    String especie;
    int edad;

    public void mostrarInfo(){
       System.out.println("Nombre: "+nombre+"\nEspecie: "+especie+"\nEdad: "+edad);
    }

    public void cumplirAnios(){
        edad += 1;
    }
}
```

Tarea: Crear una mascota, mostrar su información, simular el paso del tiempo y verificar los cambios.

```
run:
Nombre: Mora
Especie: American Bully
Edad: 3
Nombre: Mora
Especie: American Bully
Edad: 4
BUILD SUCCESSFUL (total time: 0 seconds)
```

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.

```java
public class Libro {
    private String libro;
    private String Autor;
    private int anioPublicacion;

    public String getLibro() {
        return libro;
    }

    public void setLibro(String libro) {
        this.libro = libro;
    }

    public String getAutor() {
        return Autor;
    }

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

```java
    public int getAnioPublicacion() {
        return anioPublicacion;
    }

    public void setAnioPublicacion(int anioPublicacion) {
        if (anioPublicacion>2025||anioPublicacion<0)
            System.out.println("Año incorrecto");
        else
        this.anioPublicacion = anioPublicacion;
    }


}
```

Tarea: Crear un libro, intentar modificar el año con un valor inválido y luego con uno válido, mostrar la información final.

```
run:
A�o incorrecto
Libro: Mil Manera de Morir
Autor: Leandro Lopez
A�o publicacion: 1998
BUILD SUCCESSFUL (total time: 0 seconds)
```

4. Gestión de Gallinas en Granja Digital

a. Crear una clase Gallina con los atributos: idGallina, edad, huevosPuestos. Métodos requeridos: ponerHuevo(), envejecer(), mostrarEstado().

```
public class Gallina {
  private int idGallina;
  private int edad;
  private int huevosPuestos;

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

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

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

  public void ponerHuevo(int cantidadHuevo){
      if (cantidadHuevo>0){
          huevosPuestos+=cantidadHuevo;
      }
    }
```

```
  public void envejecer(){
      edad+=1;
    }

  public void mostrarEstado(){
      System.out.println("Gallina: "+idGallina+"\nEdad: "+edad+""
              + "\nHuevos puestos: "+huevosPuestos);
  }
}
```

Tarea: Crear dos gallinas, simular sus acciones (envejecer y poner huevos), y mostrar su estado.

```
run:
Gallina: 1
Edad: 2
Huevos puestos: 2
Gallina: 1
Edad: 3
Huevos puestos: 7
BUILD SUCCESSFUL (total time: 0 seconds)
```

5. Simulación de Nave Espacial

Crear una clase NaveEspacial con los atributos: nombre, combustible. Métodos requeridos: despegar(), avanzar(distancia), recargarCombustible(cantidad), mostrarEstado().

```java
public class NaveEspacial {
    private String nombre;
    private int combustible;

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

    public void setCombustible(int combustible) {
        this.combustible = combustible;
    }

    public void despegar(){
        System.out.println("Despegando!!");
    }

    public void avanzar(int distancia){
        if (distancia>0&combustible>50){
            System.out.println("Avanzando!!");
        }else System.out.println("combustible insufuciente");
    }
```

```java
    public void recargarCombustible(int litros){
        if (litros>0&&litros<150){
            combustible+=litros;
        }
    }

    public void mostrarEstado(){
        System.out.println("Nave: "+nombre+"\nCombustible: "+combustible);
    }
}
```

Tarea: Crear una nave con 50 unidades de combustible, intentar avanzar sin recargar, luego recargar y avanzar correctamente. Mostrar el estado al final.

```java
public class Ejercicio5 {

    /**
     * @param args the command line argument
     */
    public static void main(String[] args) {
        NaveEspacial a = new NaveEspacial();

        a.setNombre("nave 1");
        a.setCombustible(50);

        a.despegar();
        a.avanzar(500);
        a.recargarCombustible(100);
        a.avanzar(500);
    }

}
```

```
- TrabajoPractico3 (run) ×
run:
Despegando!!
combustible insufuciente
Avanzando!!
BUILD SUCCESSFUL (total time: 0 seconds)
```