**PROGRAMACIÓN II.**
**Modulo 3: Introducción a la POO.**

## Trabajo Práctico 3: Introducción a la POO.

**Alumno:** LEPKA AGUSTIN
**Comisión:** 13

**OBJETIVO GENERAL:**
Comprender los fundamentos de la Programación Orientada a Objetos, incluyendo clases, objetos, atributos y métodos, para estructurar programas de manera modular y reutilizable en Java.

| Concepto | Aplicación en el proyecto |
|---|---|
| Clases y Objetos | Modelado de entidades como Estudiante, Mascota, Libro, Gallina y NaveEspacial |
| Atributos y Métodos | Definición de propiedades y comportamientos para cada clase |
| Estado e Identidad | Cada objeto conserva su propio estado (edad, calificación, combustible, etc.) |
| Encapsulamiento | Uso de modificadores de acceso y getters/setters para proteger datos |
| Modificadores de acceso | Uso de private, public y protected para controlar visibilidad |
| Getters y Setters | Acceso controlado a atributos privados mediante métodos |
| Reutilización de código | Definición de clases reutilizables en múltiples contextos |

**Caso Práctico:**
Desarrollar en Java los siguientes ejercicios aplicando los conceptos de programación orientada a objetos:

**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.

**Código principal** donde se encuentra el Main del proyecto tp3programacionii:

```
package tp3programacionii;

public class Principal {


    public static void main(String[] args) {

        //Objeto ESTUDIANTE
        Estudiante estudiante1 = new Estudiante();

        //se establecen sus atributos
        estudiante1.setNombre("Juan");
        estudiante1.setApellido("Lopez");
        estudiante1.setCurso("Ultimo");
        estudiante1.setCalificacion(7.5);

        //se muestra el estado con los datos iniciales
        estudiante1.mostrarInformacion();

        //se aumenta la calificacion 2 puntos
        estudiante1.subirCalificacion(2);

        //se muestra el estado con la calificacion aumentada en 2
        estudiante1.mostrarInformacion();

        //se baja 5 puntos la calificacion
        estudiante1.bajarCalificacion(5);

        //se muestra el estado final del objeto
        estudiante1.mostrarInformacion();
}
}
```

**Código en la clase Estudiante.java** perteneciente al package tp3programacionii:

```
package tp3programacionii;

public class Estudiante {

    //atributos para el objeto
    private String nombre;
    private String apellido;
    private String curso;
    private double calificacion;

    //setters
    //asignar nombre
```

```
    public void setNombre(String nuevoNombre) {
        nombre = nuevoNombre;
    }
    //asignar apellido
    public void setApellido(String nuevoApellido) {
        apellido = nuevoApellido;
    }
    //asignar curso
    public void setCurso(String nuevoCurso) {
        curso = nuevoCurso;
    }
    //asignar calificacion
    public void setCalificacion(double aumentarCalificacion) {
        calificacion = aumentarCalificacion;
    }

    //metodos
    //mostrar informacion del objeto
    public void mostrarInformacion() {
        System.out.println("Estudiante: " + nombre + "\n" + apellido + "\n" +
"Curso: " + curso + "\n" + "Calificacion: " + calificacion);
    }

    //subir calificacion
    public void subirCalificacion(double puntos) { calificacion = calificacion +
puntos; }

    //bajar calificacion
    public void bajarCalificacion(double puntos) {
        calificacion = calificacion - puntos;
        if (calificacion < 0) calificacion = 0;
    }
}
```

**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.

**Código principal** donde se encuentra el Main del proyecto tp3programacionii:

```
package tp3programacionii;

public class Principal {


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

```
//Objeto MASCOTA

        Mascota mascota1 = new Mascota();

        //se establecen sus atributos
        mascota1.setNombre("Firulais");
        mascota1.setEspecie("Bulldog");
        mascota1.setEdad(3);

        //se muestra el estado inicial de la mascota
        mascota1.mostrarInfo();
        //se aumenta la edad de la mascota en 1
        mascota1.cumplirAnios();
        //se muestra el estado final de la mascota
        mascota1.mostrarInfo();

}
}
```

**Código en la clase Mascota.java** perteneciente al package tp3programacionii:

```
package tp3programacionii;

public class Mascota {

    //atributos para la mascota
    private String nombre;
    private String especie;
    private int edad;

    //setters
    //asignar nombre
    public void setNombre(String nuevoNombre) {
        nombre = nuevoNombre;
    }
    //asignar especie
    public void setEspecie(String nuevaEspecie) {
        especie = nuevaEspecie;
    }
    //asignar edad
    public void setEdad(int nuevaEdad) {
        edad = nuevaEdad;
    }

    //metodos

    //mostrar informacion del estado del objeto
    public void mostrarInfo() {
        System.out.println("Mascota: " + nombre + "\n" + "Especie: " + especie +
"\n" + "Edad: " + edad + " años");
```

```
    }

    //aumentar edad de la mascota
    public void cumplirAnios() { edad++; }
}
```

**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.

**Código principal** donde se encuentra el Main del proyecto tp3programacionii:

```
package tp3programacionii;


public class Principal {


    public static void main(String[] args) {
//objeto Libro
        Libro libro1 = new Libro();
        //se establecen los valores de los atributos para el libro
        libro1.setTitulo("Principito");
        libro1.setAutor("Antoine");
        libro1.setAñoPublicacion(1943);
        //se muestran los datos iniciales del libro
        libro1.mostrarInfo();

        //se intenta modificar el anio del libro con valor erroneo
        libro1.setAñoPublicacion(0);

        //se modifica a un anio correcto
        libro1.setAñoPublicacion(1943);

        //se muestran los datos finales del libro
        libro1.mostrarInfo();
        //se utilizan los getters para mostrar datos individualmente
        System.out.println(libro1.getAutor());
        System.out.println(libro1.getAñoPublicacion());
        System.out.println(libro1.getTitulo());

}
}
```

**Código en la clase Libro.java** perteneciente al package tp3programacionii:

```
package tp3programacionii;

public class Libro {

    //se establecen atributos para el libro
    private String titulo;
    private String autor;
    private int añoPublicacion;

    //setters
    //asignar titulo
    public void setTitulo(String nuevoTitulo) {
        titulo = nuevoTitulo;
    }
    //asignar autor
    public void setAutor(String nuevoAutor) {
        autor = nuevoAutor;
    }
    //asignar año de publicacion
    public void setAñoPublicacion(int año) {
        if (año > 0 && año <= 2025){
            añoPublicacion = año;
        }
        else {
            System.out.println("Año inválido: " + año);
        }
    }

    //getters
    //devolver titulo
    public String getTitulo() {
        return titulo;
    }
    //devolver autor
    public String getAutor() {
        return autor;
    }
    //devolver año de publicacion
    public int getAñoPublicacion() {
        return añoPublicacion;
    }

    //se muestra toda la informacion del objeto
    public void mostrarInfo() {
        System.out.println("Libro: " + titulo + "\n"+ "Autor: " + autor +  "\n"+ "Año: "
+ añoPublicacion);
    }
```

}

**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.

**Código principal** donde se encuentra el Main del proyecto tp3programacionii:

```
package tp3programacionii;


public class Principal {


    public static void main(String[] args) {

//Objeto GALLINA
        Gallina gallinita1 = new Gallina();
        //se establecen los valores para el objeto gallina
        gallinita1.setIdGallina(190);
        gallinita1.setEdad(2);
        //se suma cantidad de huevos puestos
        gallinita1.ponerHuevo();
        //se muestra el estado de la gallina
        gallinita1.mostrarEstado();
        //se aumenta la edad en 1
        gallinita1.envejecer();
        //se aumenta los huevos puestos en 1
        gallinita1.ponerHuevo();
        //se muestran los datos finales del objeto
        gallinita1.mostrarEstado();

}
}
```

**Código en la clase Gallina.java** perteneciente al package tp3programacionii:

```
package tp3programacionii;

public class Gallina {
    //se establecen atributos para objeto de clase gallina
    private int idGallina;
    private int edad;
    private int huevosPuestos;
```

```
    //se establecen los setters
    //asignar numero de identificacion de la gallina
    public void setIdGallina(int id) {
        idGallina = id;
    }
    //asignar la edad de la gallina
    public void setEdad(int nuevaEdad) {
        edad = nuevaEdad;
    }
    //metodos
    //simula poner 1 huevo mas de gallina
    public void ponerHuevo() { huevosPuestos++; }
    //simula envejecer 1 anio mas a la gallina
    public void envejecer() { edad++; }

    //se muestra la informacion del objeto
    public void mostrarEstado() {
        System.out.println("Gallina id: " + idGallina + "\n" +"Edad: " + edad + "\n" +
"Huevos puestos: " + huevosPuestos);
    }

}
```

**5. Simulación de Nave Espacial**

**a.** 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.

**Código principal** donde se encuentra el Main del proyecto tp3programacionii:

```
package tp3programacionii;


public class Principal {


    public static void main(String[] args) {
  //Objeto Nave Espacial
        NaveEspacial naveEspacial1 = new NaveEspacial();
        //se establecen los valores para el objeto nave espacial
        naveEspacial1.setNombre("Nave Principal");
        naveEspacial1.setCombustible(30);
        //se muestra el estado inicial de la nave
```

```
        naveEspacial1.mostrarEstado();
        //se intenta que la nave avance
        naveEspacial1.avanzar(20);
        //se carga combustible
        naveEspacial1.recargarCombustible(50);
        //mediante getter se muestra el dato de la cantidad de combustible
        naveEspacial1.getCombustible();
        //se intenta que la nave avance nuevamente
        naveEspacial1.avanzar(20);

        //se muestra el estado de la nave
        naveEspacial1.mostrarEstado();

}

}
```

**Código en la clase NaveEspacial.java** perteneciente al package tp3programacionii:

```
package tp3programacionii;

public class NaveEspacial {
//se establecen atributos para la nave
    private String nombre;
    private int combustible;
    private final int LIMITE_COMBUSTIBLE = 100;

    //setters
    //establecer nombre de la nave
    public void setNombre(String nuevoNombre) {
        nombre = nuevoNombre;
    }
    //establecer cantidad de combustible
    public void setCombustible(int nuevaCantidad) {
        combustible = nuevaCantidad;
    }

    //getter para la cantidad de combustible
    public void getCombustible(){
        System.out.println("la cantidad de combustible actual es: " + combustible);
    }

    //metodos

    //simular el despegue de la nave
    public void despegar() {
        if (combustible >= 10) {
            combustible = combustible - 10;
```

```
            System.out.println("La nave " + nombre + " ya despego" + "\n" +
"Combustible restante: " + combustible);

        }
        else {
            System.out.println("No alcanza el combustible para despegar.");
        }

    }

    //simular el avance de la nave
    public void avanzar(int distancia) {
        int consumo = distancia * 2;
        if (combustible >= consumo) {
            combustible = combustible - consumo;
            System.out.println(nombre + " avanzó " + distancia + " unidades." + "\n"
+ "Combustible restante: " + combustible);
        }
        else {
            System.out.println("No hay suficiente combustible para avanzar.");
        }

    }

    //simular carga de combustible de la nave
    public void recargarCombustible(int cantidad) {
        if (combustible + cantidad <= LIMITE_COMBUSTIBLE) {
            combustible = combustible + cantidad;
            System.out.println("Se recargo " + cantidad + " unidades." +"\n" +
"Combustible actual: " + combustible);
        }
        else {
            System.out.println("No se puede recargar, supera el límite de " +
LIMITE_COMBUSTIBLE);
        }
    }

    //muestra el estado de la nave
    public void mostrarEstado() {
        System.out.println("Nave: " + nombre + "\n" + "Combustible: " +
combustible + "\n" + "Limibte de combustible: " + LIMITE_COMBUSTIBLE);

    }
}
```