

**Universidad Tecnológica Nacional - Facultad Regional San Nicolás**

**Programación II**

**Profesor coordinador:** Carlos Martínez

**Alumno:** Alex Austin Nahuel.

**Comisión N°:** 17

**Profesor tutor:** Juan Cruz Robledo

**Actividad N°:** 3

**REPOSITORIOS DE GITHUB:** <https://github.com/AlexNahuelAustin/Programacion-II-UTN-TUPaD-2025.git>

1. a

package actividad1;

import java.util.Scanner;

public class RegistroDeAlumno {

/\*

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.

\*/

public static void main(String[] args) {

Scanner teclado = new Scanner(System.in);

//declamos el objeto y pedimos los datos por teclado

Alumno alum1 = new Alumno();

System.out.println("Ingrese el nombre del estudiante: ");

alum1.nombre = teclado.nextLine();

System.out.println("Ingrese el apellido del estudiante: ");

alum1.apellido = teclado.nextLine();

System.out.println("Ingrese el el curso del estudiante: ");

alum1.curso = teclado.nextLine();

System.out.println("Ingrese la calificacion del 1 al 10 del estudiante: ");

alum1.calificacion = Double.parseDouble(teclado.nextLine());

// mostramos por consola los datos del estudiante

alum1.motrarInfo(alum1.nombre, alum1.apellido, alum1.curso, alum1.calificacion);

//pedimos los puntos extra del estudiante

System.out.println("-------puntos extras----------");

System.out.println("Ingrese los puntos extra del estudiante: ");

alum1.subirCalificacion(alum1.calificacion);

System.out.println("La calificacion del alumno por su puntos extras es: " + alum1.calificacion);

//pedimos la baja de punto del estudiante

System.out.println("-------baja de puntos----------");

System.out.println("Ingrese la baja de punto del estudiante: ");

alum1.bajarCalificacion(alum1.calificacion);

System.out.println("La calificacion final del alumno: " + alum1.calificacion);

}

}

**package actividad1;**

**import java.util.Scanner;**

**public class Alumno {**

//importamos el metodo scanner y declaramos los atributos

Scanner teclado = new Scanner(System.in);

String nombre;

String apellido;

String curso;

double calificacion;

public Alumno() {

}

// validamos que la note este entre 1 a 10

public void setcalificaciones(Scanner teclado) {

boolean calificacionValida = false;

while (!calificacionValida) {

System.out.println("Ingrese la calificacion del 1 a al 10: ");

calificacion = Double.parseDouble(teclado.nextLine());

}

}

// metodo para mostrar la informacion del estudiante

public void motrarInfo(String nombre, String apellido, String curso, double calificacion) {

System.out.println("----------------- Informacion del estudiante -----------------");

System.out.println("Nombre: " + nombre + "\nApellido: " + apellido + "\nCurso: " + curso + "\nCalificacion: " + calificacion);

}

//metodo para subir la nota

public double subirCalificacion(double puntoExtra) {

puntoExtra = Double.parseDouble(teclado.nextLine());

calificacion += puntoExtra;

if (calificacion > 10) {

calificacion = 10;

}

return calificacion;

}

//metodo para bajar la nota

public double bajarCalificacion(double bajarNota) {

bajarNota = Double.parseDouble(teclado.nextLine());

calificacion -= bajarNota;

if (calificacion <= 0) {

calificacion = 1;

}

return calificacion;

}

}

package actividad2;

/\*\*

\* 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.

\*/

public class Actividad2 {

public static void main(String[] args) {

// declaramos el objeto

Mascota mascota1 = new Mascota();

mascota1.Nombre = "Milo";

mascota1.Especie = "Gato";

mascota1.Edad = 6;

//mostramos por consola los datos de nuestra mascota

mascota1.mostrarInfo();

//simulamos el paso del tiempo y mostramos su nueva edad.

mascota1.cumplirAnios(4);

mascota1.mostrarInfo();

}

}

**package actividad2;**

**public class Mascota {**

// nombramos los atributos de la clase

String Nombre;

String Especie;

int Edad;

public void mascota() {

}

// para mostrar la informacion por consola

public void mostrarInfo() {

System.out.println("----------Informacion de su mascota----------");

System.out.println("El nombre de la mascotas es: " + Nombre);

System.out.println("La especie de la mascotas es: " + Especie);

System.out.println("La edad de la mascotas es: " + Edad);

}

// retornamos la edad actualizada de nuestra mascota

public int cumplirAnios(int anio) {

Edad += anio;

return Edad;

}

}

package actividad3;

public class Actividad3 {

/\*

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.

\*/

public static void main(String[] args) {

//declaramos el objeto

Libro libro1 = new Libro("El mentalista", "Jhon Mayer", 2007);

// Msotramos por cosola los datos

libro1.mostrarInfo();

System.out.println("-----------------------------------------------");

//moficamos un año invalido

libro1.setAnioPublicacion(3500);

libro1.mostrarInfo();

// pasamos un años valido

System.out.println("-----------------------------------------------");

libro1.setAnioPublicacion(2009);

libro1.mostrarInfo();

}

}

**package actividad3;**

**public class Libro {**

// Nombramos los atributos de la clase

private String Titulo;

private String Autor;

private int anioPublicacion;

public Libro(String Titulo, String Autor, int anioPublicacion) {

this.Titulo = Titulo;

this.Autor = Autor;

this.anioPublicacion = anioPublicacion;

}

public String getTitulo() {

return Titulo;

}

public String getAutor() {

return Autor;

}

public int getAnioPublicacion() {

return anioPublicacion;

}

public void setAnioPublicacion(int anioPublicacion) {

int anioactual = 2025;

if (anioPublicacion > 0 && anioPublicacion <= anioactual) {

this.anioPublicacion = anioPublicacion;

} else {

System.out.println("años de publicacion es invalido: " + anioPublicacion);

}

}

//metodo para mostrar la informacion

public void mostrarInfo() {

System.out.println("-------------Informacion de libro-------------");

System.out.println("Titulo: " + Titulo);

System.out.println("Autor: " + Autor);

System.out.println("Año de publicacion: " + anioPublicacion);

}

}

1. a

package actividad4;

public class Actividad4 {

/\*

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.

\*/

public static void main(String[] args) {

// creamos las dos gallinas

Gallina gallina1 = new Gallina(01, 7, 25);

Gallina gallina2 = new Gallina(02, 10, 15);

//mostramos la informacion inicial.

gallina1.mostraEstado();

gallina2.mostraEstado();

// envejecemos a las gallinas

gallina1.envejecer(5);

gallina2.envejecer(5);

//puesta de huevos

gallina1.ponerHuevos(19);

gallina2.ponerHuevos(6);

// mostramos la informacion actual de las gallinas

gallina1.mostraEstado();

gallina2.mostraEstado();

}

}

**package actividad4;**

**public class Gallina** {

// declararamos los atributos de la gallina

private final int idGallina;

private int edad;

private int huevoPuestos;

// creamos el constructor

public Gallina(int idGallina, int edad, int huevoPuestos) {

this.idGallina = idGallina;

this.edad = edad;

this.huevoPuestos = huevoPuestos;

}

// hacemos los metodo para las las acciones de la gallina

public void ponerHuevos(int ponerHuevos) {

if (ponerHuevos >= 1) {

huevoPuestos += ponerHuevos;

}

}

public void envejecer(int anios) {

if (anios >= 1) {

edad += anios;

}

}

// declaramos la para mostrar la informacion de la gallinas.

public void mostraEstado() {

System.out.println("------------Informacion de la gallina------------");

System.out.println("Id de gallina: " + idGallina);

System.out.println("Edad de la gallina: " + edad + " años");

System.out.println("Total de huevos puestos: " + huevoPuestos);

}

}

package actividad5;

public class Actividad5 {

/\*

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.

\*/

public static void main(String[] args) {

// nombramos al objeto nave

NaveEspacial nave1 = new NaveEspacial("Estrella de la muerte ", 45);

// llamamos al metodo despegar

nave1.despegar();

//llamamos al metodo avanzar

nave1.avanzar(11);

//llamamos el metodo para recargar combustiblr

nave1.recargarCombustible(31);

//llamamos al metdo avanzar nuevamente.

nave1.avanzar(20);

//mostramos el estado

nave1.mostrarEstado();

}

}

**package actividad5;**

**public class NaveEspacial** {

// ponemos los atributos a la nave

private String nombre;

private int combustible;

private final int TANQUE\_CAPACIDAD\_MAX = 50;

private int distanciaRecorrida;

public NaveEspacial(String nombre, int combustible) {

this.nombre = nombre;

this.combustible = combustible;

}

//metodo para el despege

public void despegar() {

if (combustible >= 5) {

combustible -= 5;

System.out.println(nombre + " ha despegado. cantidad de combustible restante: " + combustible);

} else {

System.out.println(" Cantidad de conbustible insuficiente: ");

}

}

//metodo para avanzar

public void avanzar(int distancia) {

int consumos = distancia \* 2;

distanciaRecorrida += distancia;

if (consumos <= combustible) {

combustible -= consumos;

System.out.println(nombre + " Recorrio: " + distanciaRecorrida + "Km " + " combustible restante " + combustible);

} else {

System.out.println(nombre + " combustible insuficiente " + " distacia recorrida " + distancia + " Km");

}

}

// metofdo para recargar combustible

public void recargarCombustible(int cantidad) {

if ((combustible + cantidad) < TANQUE\_CAPACIDAD\_MAX) {

combustible += cantidad;

System.out.println("Se recargaron: " + cantidad + " unidades de combustible. Total: " + combustible);

} else {

combustible = TANQUE\_CAPACIDAD\_MAX;

System.out.println("Tanque lleno");

}

}

// metodo para mostrar el estado de la nave espacial

public void mostrarEstado() {

System.out.println("---------------Informacion de la nave espacial---------------");

System.out.println("Nombre de la nave: " + nombre);

System.out.println("distancia total recorrida: " + distanciaRecorrida + "Km");

System.out.println("combustible restante: " + combustible);

}

}