

# **Sistema De Gestión De Tareas En Java**

## **Introducción**

En este apartado se presenta una descripción un poco general sobre el proyecto desarrollado. El objetivo de este programa es permitir la gestión básica de tareas mediante una aplicación de consola desarrollada en el lenguaje de programación Java, aplicando los conceptos aprendidos en la universidad y reforzados en el curso, se aplicaron conceptos fundamentales de Programación Orientada a Objetos (POO), manejo de colecciones, control de excepciones simples y estructuras de control.

El sistema le permite al usuario agregar, visualizar, completar y eliminar tareas, garantizando un funcionamiento esperado mediante validaciones y manejo de errores.

## **Objetivos**

### **Objetivo general**

Desarrollar una aplicación en Java que permita el gestionamiento de tareas utilizando principios básicos de POO y estructuras de datos.

### **Objetivos específicos**

1. Crear una clase que represente una Tarea.
2. Implementar una clase que administre/gestione las listas de tareas.
3. Aplicar el manejo de excepciones simples para controlar errores.
4. Implementar un menú interactivo en consola con distintas opciones.
5. Validar la existencia de tareas antes de modificarlas u eliminarlas.

## **Descripción del sistema**

El sistema está compuesto por cuatro clases que trabajan en conjunto para ofrecer la funcionalidad requerida. La interacción con el usuario se realiza a través de un menú en consola, el cual permite ejecutar las distintas operaciones disponibles.

## **Estructura del proyecto**

## **Clase – Tarea**

Esta clase representa cada una de las tareas del sistema. Contiene atributos como el identificador, la descripción y el estado de la tarea. Además, incluye métodos para acceder sus atributos y un método ‘to-String()’ que se encarga de mostrar la información de la tarea de forma legible.

## **Clase – GestorTarea**

Esta clase es la encargada de administrar la lista de tareas.

Se encuentra encargada de:

- Agregar nuevas tareas.
- Mostrar datos.
- Buscar tareas.
- Marcar tareas como completadas.
- Eliminar tareas existentes.

En pocas palabras menos técnicas sería un tipo de CRUD, pero un poco más sencillo.

## **Clase – TareaNoEncontrada**

Se encarga de implementar una excepción personalizada para manejar situaciones en las que el usuario intente operar sobre una tarea que no existe. Esto permite mejorar el control de errores y evitar comportamientos incorrectos en el sistema.

## **Clase – Main**

Es la clase que contiene el método principal del programa. En esta clase se implementa un menú interactivo que sale por consola que le permite al usuario seleccionar las diferentes opciones del sistema y ejecutar las acciones en base a la opción que del usuario.

## **Manejo de excepciones**

El programa utiliza bloque ‘try-catch’ encargados de capturar errores durante la ejecución, como la de búsqueda de tareas inexistentes. El uso de excepciones personalizadas permite mensajes claros al usuario y evitar que el programa lance algún tipo de error que interfiera con la ejecución del mismo.

## **Funcionamiento del menú**

El menú presenta las siguientes opciones mostradas por consola:

- Agregar tarea

- Mostrar tareas
- Completar tarea
- Eliminar tarea
- Salir o cerrar el programa

Cada opción solicita los datos necesarios al usuario y ejecuta la acción en base a opción correspondiente.

## **Validaciones implementadas**

El sistema valida que, no se completen tareas insistentes, no se elimine una tarea que no existe y muestra un mensaje cuando nos hay tareas registradas en el programa. Estas validaciones evitan errores en la ejecución del programa.

## **Resultados obtenidos**

El programa funciona correctamente permitiendo gestionar tareas desde la consola. Las operaciones solicitadas se ejecutan sin errores y los mensajes mostrados al usuario son claros y comprensibles.

## **Conclusiones**

El programa se ejecuta de manera correcta y aplica los conceptos básicos de Java y POO. Además, se logró la implementación de un manejo adecuado de excepciones y una estructura clara del código, facilitando su comprensión y mantenimiento.

## **Referencias**

El material proporcionado de clases universitarias y reforzadas en el curso.