Estructuras utilizadas – Sistema de Gestión de Tareas

Universidad Nacional de Colombia

Autores:

Sara Sepúlveda

Samuel Mejía

Antonio Moreno

Octubre 2025

El proyecto tiene como objetivo ofrecer una herramienta que permita al usuario gestionar sus tareas de manera ordenada y ordenando las tareas con base a diferentes factores, como la propia prioridad de la tarea y su fecha de entrega.

Se escogieron tres estructuras adecuadas para garantizar un manejo óptimo de la información, teniendo en cuenta lo que se quiere lograr con cada una. Estas son: **Arreglo dinámico, pila y cola con prioridad.** Cada una cumple un rol diferente dentro del diseño, a continuación, se especifican los objetivos de implementarlas.

1. Arreglo Dinámico

Con esta estructura se realiza el almacenamiento de todas las tareas, con sus respectivos atributos, (descripción, prioridad, etc.) además, las ordena por fecha de creación. Esta estructura es ideal para guardar y modificar las tareas, pues las mantiene en un orden determinado y no hace uso de herramientas avanzadas para dicho almacenamiento. Además, su mutabilidad le permite cambiar su tamaño dependiendo de la cantidad de tareas a diferencia de los arreglos estáticos.

**Implementación:**

En esta línea se declara la variable “listaTareas” utilizando la clase ArrayList (arreglo dinámico), especificando que solo puede recibir instancias de la clase Tarea:



Asignación a la variable:



A continuación, cada tarea se irá agregando al final del arreglo de esta forma:



A la hora de remover una tarea, se hace de la siguiente forma sin afectar la posición del resto:

Texto

El contenido generado por IA puede ser incorrecto.

El arreglo actúa como una base de datos en memoria del sistema. Tomando los datos del archivo de texto Tareas.txt

1. Pila

Se hizo uso de la pila LIFO (Last In First Out) para guardar todas las acciones de manera que la última quedara en el “tope” de esta y se pueda interactuar con las acciones en ese orden.

El usuario puede cometer errores a la hora de añadir, eliminar o editar tareas, la pila funciona como un historial para las acciones y permite deshacerlas, asegurándose de que la última acción sea la primera que pueda deshacerse.

**Implementación:**

Se creó la clase “Accion” que tiene como atributos el tipo de acción, la tarea asociada a esa acción y si es necesario para el tipo de acción el valor del atributo antes de que fuera modificado:

Texto

El contenido generado por IA puede ser incorrecto.

Para poder deshacer las acciones, se crea el método “revertir” que, dependiendo del nombre de la acción, la deshace de la siguiente forma:

Pantalla de computadora

El contenido generado por IA puede ser incorrecto.

Por ejemplo, si la última acción fue agregar una tarea, esta se elimina o si la última acción fue eliminar una tarea, esta se agrega nuevamente.

En las siguientes líneas se implementa la pila bajo el nombre historialAc (historial de acciones), esta solo va a recibir instancias de la clase “Accion”, luego se hace la asignación a la variable:





Cada vez que se agregue una tarea al arreglo, se guarda dicha acción en la pila:



Al eliminar una tarea, se guarda la acción:



Cuando se cambia el título, la descripción, la fecha de entrega, la prioridad o el estado de una tarea, se guarda de las siguientes formas:











En las siguientes líneas se deshace la última acción haciendo uso del método pop propio de las pilas y del método revertir de la clase “Accion” vista anteriormente:



Si la pila está vacía, se imprime el mensaje “No hay acciones para deshacer”:



1. Cola con prioridad

En una cola común, se trabaja con los datos en orden FIFO (First In First Out), sin embargo, en este tipo de cola, se atienden los datos con base en un criterio definido por el usuario.

Esta estructura se implementó con el objetivo de organizar las tareas a partir del atributo “prioridad” que indica la urgencia de realizar cada tarea, siendo 1 poca urgencia y 5 muy urgente. En caso de que dos o mas tareas tengan la misma prioridad, las tareas se organizarán a partir de su fecha límite de entrega.

**Implementación:**

En las siguientes líneas se declara la variable cola\_p que solo recibirá instancias de la clase Tarea y se asigna la variable:





La cola solamente almacenará tareas con estado pendiente, al agregar una tarea al arreglo automáticamente tendrá este estado, de esta forma se agrega también a la cola:



Cuando se borra una tarea del arreglo, también se borrará de la cola:



Para poder mantener el orden deseado, el siguiente método actualiza constantemente la cola cuando se hacen actualizaciones en la prioridad o el estado de una tarea:

Texto

El contenido generado por IA puede ser incorrecto.