**Descripción del Proyecto:**

**Administrador de Tareas Personales**

Este proyecto en C++ tiene como objetivo desarrollar una aplicación que permita al usuario gestionar sus tareas personales de manera eficiente. El sistema proporciona funcionalidades básicas como la creación, edición, eliminación y visualización de tareas. Cada tarea incluye detalles como un título, descripción y fecha límite. El programa está diseñado para ser interactivo mediante un menú simple, permitiendo al usuario realizar las operaciones mencionadas con facilidad. Además, el sistema se basa en el uso de clases y estructuras avanzadas de datos para manejar la lista de tareas.

**Funcionalidad del Código**

1. **Clase Tarea:**
   * Esta clase representa cada tarea individual con atributos como:
     + id: Un identificador único para cada tarea.
     + titulo: El nombre de la tarea.
     + descripcion: Una descripción detallada de la tarea.
     + fechaLimite: Fecha en la que la tarea debe completarse.
   * Métodos principales:
     + getId(), getTitulo(), getDescripcion(), getFechaLimite(): Para obtener los valores de los atributos de la tarea.
     + setTitulo(), setDescripcion(), setFechaLimite(): Permiten modificar los valores de la tarea.
     + mostrarTarea(): Imprime en consola los detalles de la tarea.
2. **Clase AdministradorDeTareas:**
   * Esta clase gestiona una colección de objetos Tarea mediante el uso de un vector (std::vector).
   * Atributos:
     + listaDeTareas: Vector que almacena todas las tareas creadas.
     + idActual: Un contador que se incrementa con cada tarea añadida para asignar un id único.
   * Métodos principales:
     + agregarTarea(): Crea una nueva tarea con los detalles proporcionados (título, descripción, fecha límite) y la añade a la lista.
     + editarTarea(): Busca una tarea por su id y permite modificar sus atributos.
     + eliminarTarea(): Busca una tarea por su id y la elimina de la lista si existe.
     + mostrarTareas(): Imprime en consola todas las tareas actualmente almacenadas en la lista.
3. **Interfaz del Usuario (funcionalidad principal en main()):**
   * El programa proporciona un menú interactivo en el que el usuario puede:
     + **Añadir Tarea:** Solicita al usuario que ingrese el título, descripción y fecha límite de la tarea y la añade al sistema.
     + **Editar Tarea:** Permite al usuario editar una tarea existente seleccionándola por su ID.
     + **Eliminar Tarea:** Elimina una tarea seleccionada por su ID.
     + **Mostrar Tareas:** Muestra todas las tareas actuales con sus detalles.
     + **Salir del programa.**
   * El programa continúa ejecutándose en un bucle do-while hasta que el usuario elige la opción de salir.

**Explicación Funcional del Código**

1. **Al iniciar el programa,** el usuario verá un menú donde puede elegir entre las diferentes opciones (añadir, editar, eliminar, mostrar tareas).
2. **Añadir Tarea:**
   * Cuando el usuario selecciona esta opción, se le pide que ingrese el título, descripción y fecha límite.
   * El programa llama al método agregarTarea() para crear una nueva instancia de Tarea y añadirla al vector listaDeTareas.
3. **Editar Tarea:**
   * El usuario ingresa el ID de la tarea que desea modificar. Si la tarea existe, puede actualizar el título, descripción y fecha límite.
4. **Eliminar Tarea:**
   * Similar al caso de editar, el usuario proporciona el ID de la tarea que desea eliminar. El programa busca la tarea en el vector y, si la encuentra, la elimina.
5. **Mostrar Tareas:**
   * Muestra en consola todas las tareas almacenadas en el sistema, imprimiendo el ID, título, descripción y fecha límite de cada tarea.
6. **Salir:**
   * El bucle se repite hasta que el usuario selecciona la opción de salir.

**Posibles Extensiones**

* **Priorización de tareas:** Permitir asignar prioridades a las tareas (alta, media, baja).
* **Búsqueda de tareas:** Implementar una función para buscar tareas por título o fecha límite.
* **Recordatorios:** Añadir una funcionalidad que alerte al usuario cuando una tarea está cerca de su fecha límite.