**Description of an application:**

A console-based program called Smart Task Manager assists users in organizing their everyday responsibilities. Users can add, remove, update, and view tasks. Every task has a title, description, due date, priority level, and status of completion.

The program uses **object-oriented programming (OOP)** concepts such as **classes, inheritance, encapsulation, and polymorphism** to structure the code efficiently and make it scalable.

**What does it do**

Allows users to make and maintain a to-do list.

Provides options to edit, delete, or mark tasks as completed.

Gives users the ability to change, remove, or mark jobs as finished.

Shows tasks according to their due date or priority.

Saves task information during the session using dynamic memory and file I/O (optional if implemented).

**How does it work**

The application operates in the terminal using a straightforward text-based menu system.

Individual task data is stored in the Task class.

A TaskManager class holds a vector of Task objects and offers methods to manipulate them (addTask, removeTask, sortTasks, etc.).

Uses class inheritance for different types of tasks (e.g., WorkTask, PersonalTask) that extend the base Task class.

The main function loops through user input and manages the user interface.

**Milestones**

Creating and storing tasks

Create new tasks based on user input.

Store them using dynamic memory and STL containers

Task Display & Sorting

Use STL containers and dynamic memory to store them.

Sort the jobs by priority or due date.

Filter to display only jobs that are finished or pending.

Editing and Finishing TasksUsing an index or title, edit or remove tasks.

Mark assignments as finished.

**Special features**

* **Data Persistence**  
   Save tasks to a file and reload them when the program starts.
* **Graphical User Interface (GUI)**  
   Convert the console app into a GUI app using a C++ library
* **Messages and Reminders**
* To remind users of impending tasks, include timed notifications.
* **Calendar Integration**
* Connect to third-party calendars, such as Google Calendar.
* **Mobile App Version**  
   Expand the codebase into a mobile app using cross-platform frameworks like Flutter or React Native (with a C++ backend).