Task Manager.c

Introduction

This project is a simple task management system that allows users to add, view, complete, and delete tasks, with persistence enabled through file handling. The program uses a basic text-based menu for user interactions, making it accessible for users to manage tasks in a straightforward manner. It primarily focuses on functionality over complexity, suitable for beginners learning programming concepts such as structs, file handling, and modular functions.

Background Study

Task management applications help users organize and track their work efficiently. This project draws on fundamental programming techniques, utilizing data structures, file handling, error checking, and user input/output functions. By storing tasks in a file, it provides a persistent record of tasks across program runs. This type of system reflects basic principles of to-do list applications found in more complex systems, such as those used in personal productivity software.

The program is implemented in C and uses standard I/O libraries. Each task is defined as a struct, encapsulating the task's description, due date, completion status, activity status, and priority level. Using arrays to store tasks and separate functions for each task operation ensures that the program remains modular and straightforward.

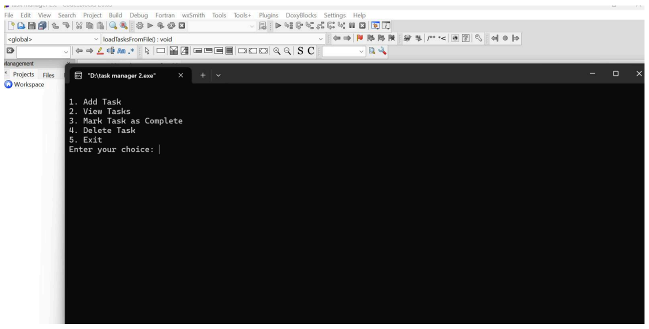
Visualization

Here is an overview of how the program interacts with the user:

1. Menu Display

The user is presented with a main menu consisting of five options:

1. Add a task
2. View tasks
3. Mark a task as complete
4. Delete a task
5. Exit the program



```
1. Add Task
2. View Tasks
3. Mark Task as Complete
4. Delete Task
5. Exit
Enter your choice:
```

2. Task Creation

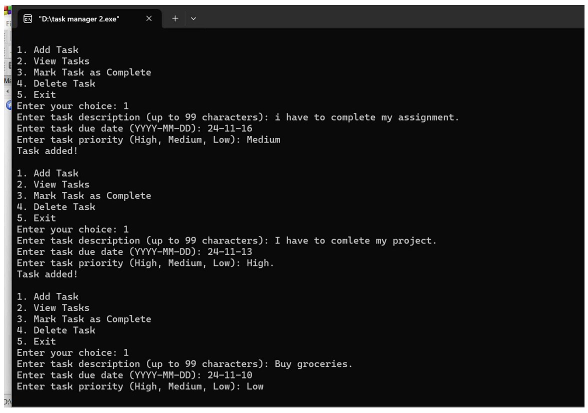
When the user selects “Add a task,” they are prompted to enter details for the task, including:

Task description

Due date in the format YYYY-MM-DD

Priority level (e.g., High, Medium, Low)

After entering these details, the task is saved, and the user is notified of successful creation.



```
1. Add Task
2. View Tasks
3. Mark Task as Complete
4. Delete Task
5. Exit
Enter your choice: 1
Enter task description (up to 99 characters): i have to complete my assignment.
Enter task due date (YYYY-MM-DD): 24-11-16
Enter task priority (High, Medium, Low): Medium
Task added!

1. Add Task
2. View Tasks
3. Mark Task as Complete
4. Delete Task
5. Exit
Enter your choice: 1
Enter task description (up to 99 characters): I have to comlete my project.
Enter task due date (YYYY-MM-DD): 24-11-13
Enter task priority (High, Medium, Low): High.
Task added!

1. Add Task
2. View Tasks
3. Mark Task as Complete
4. Delete Task
5. Exit
Enter your choice: 1
Enter task description (up to 99 characters): Buy groceries.
Enter task due date (YYYY-MM-DD): 24-11-10
Enter task priority (High, Medium, Low): Low
```

3. Viewing Tasks

Selecting “View tasks” displays all active tasks with details such as:

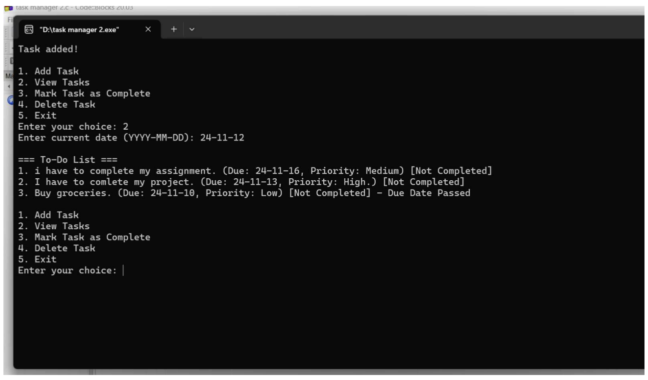
Task description

Due date

Priority level

Completion status (completed or not)

If the current date (entered by the user) is later than a task's due date, the task is marked as overdue.

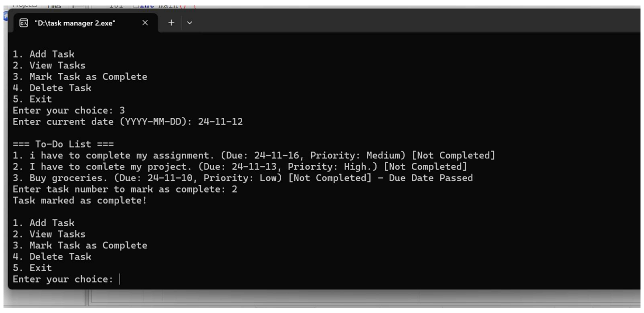


```
"D:\task manager 2.exe"
Task added!

1. Add Task
2. View Tasks
3. Mark Task as Complete
4. Delete Task
5. Exit
Enter your choice: 2
Enter current date (YYYY-MM-DD): 24-11-12

=== To-Do List ===
1. i have to complete my assignment. (Due: 24-11-16, Priority: Medium) [Not Completed]
2. I have to comlete my project. (Due: 24-11-13, Priority: High.) [Not Completed]
3. Buy groceries. (Due: 24-11-10, Priority: Low) [Not Completed] - Due Date Passed

1. Add Task
2. View Tasks
3. Mark Task as Complete
4. Delete Task
5. Exit
Enter your choice:
```

4. Marking Tasks as Complete

When choosing to mark a task as complete, the user is asked to enter the specific task number. The program then updates the task's completion status.

Afterward, the updated task list is saved to the file, confirming the completion.

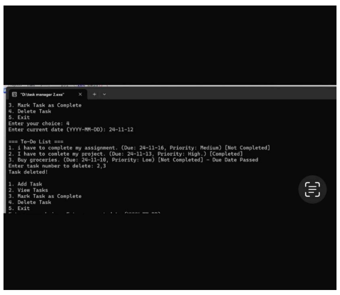


```
"D:\task manager 2.exe"
1. Add Task
2. View Tasks
3. Mark Task as Complete
4. Delete Task
5. Exit
Enter your choice: 3
Enter current date (YYYY-MM-DD): 24-11-12

=== To-Do List ===
1. i have to complete my assignment. (Due: 24-11-16, Priority: Medium) [Not Completed]
2. I have to comlete my project. (Due: 24-11-13, Priority: High.) [Not Completed]
3. Buy groceries. (Due: 24-11-10, Priority: Low) [Not Completed] - Due Date Passed
Enter task number to mark as complete: 2
Task marked as complete!

1. Add Task
2. View Tasks
3. Mark Task as Complete
4. Delete Task
5. Exit
Enter your choice:
```

5. Deleting Tasks

To delete a task, the user selects the task number, and the program marks it as inactive (effectively deleting it from the active list).

This change is also saved to the file, ensuring the task remains deleted across sessions.

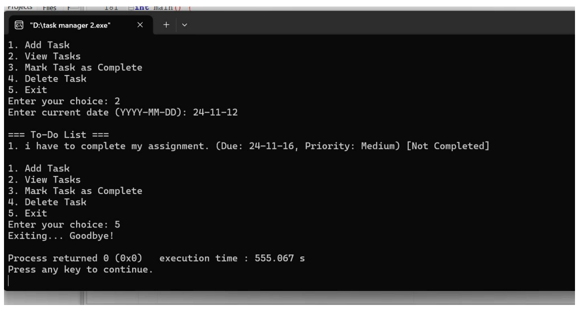


```
"D:\task manager 2.exe"
3. Mark Task as Complete
4. Delete Task
5. Exit
Enter your choice: 4
Enter current date (YYYY-MM-DD): 24-11-12

=== To-Do List ===
1. i have to complete my assignment. (Due: 24-11-16, Priority: Medium) [Not Completed]
2. I have to comlete my project. (Due: 24-11-13, Priority: High.) [Completed]
3. Buy groceries. (Due: 24-11-10, Priority: Low) [Not Completed] - Due Date Passed
Enter task number to delete: 2,3
Task deleted!

1. Add Task
2. View Tasks
3. Mark Task as Complete
4. Delete Task
5. Exit
```

6. File Persistence

Each time a task is added, marked complete, or deleted, the program updates a file to save these changes.

Upon restarting, the program loads all active tasks from this file, enabling data persistence between runs.



```
1. Add Task
2. View Tasks
3. Mark Task as Complete
4. Delete Task
5. Exit
Enter your choice: 2
Enter current date (YYYY-MM-DD): 24-11-12

=== To-Do List ===
1. i have to complete my assignment. (Due: 24-11-16, Priority: Medium) [Not Completed]

1. Add Task
2. View Tasks
3. Mark Task as Complete
4. Delete Task
5. Exit
Enter your choice: 5
Exiting... Goodbye!

Process returned 0 (0x0)   execution time : 555.067 s
Press any key to continue.
```

Limitations

Limited Data Storage: Tasks are stored in a single file, limiting scalability and making it unsuitable for larger task management needs.

No Task Categorization: Tasks are stored in a flat structure with no option for categorization, tags, or grouping.

Basic Error Handling: While there is some error checking, more robust error handling could improve usability, such as preventing invalid date inputs.

No Data Encryption: Since tasks are stored in plain text, there is no privacy or security feature to protect sensitive information.

Limited User Interface: The text-based interface, while simple, lacks graphical elements, which could make it more user-friendly and visually appealing.

Future Scope

Several enhancements could make the system more robust and versatile:

1. Category and Tag Support: Adding categories or tags would allow users to organize tasks by themes or projects.

2. Date and Time Validation: Implementing stricter date and time checks would improve input reliability and avoid errors in overdue task checks.

3. Enhanced File Handling: Moving from a single text file to a more robust storage format, like JSON or SQLite, could allow for better data management.

4. Graphical User Interface (GUI): Developing a GUI would make the program more intuitive and accessible, especially for users unfamiliar with text-based programs.

5. Security Features: Adding encryption for file storage could improve the privacy and security of task data.

6. Task Search and Filter: Adding search and filter capabilities would allow users to easily find specific tasks based on status, priority, or due date.

Conclusion

This project demonstrates the development of a basic task management system with persistent data storage using file handling in C. It provides an introduction to essential programming concepts, including data structures, modular functions, and user interaction through a text-based menu. Although limited in functionality and scalability, the program can serve as a foundation for more complex task management systems. By implementing additional features such as a GUI, enhanced error handling, and advanced file storage options, this program could evolve into a more sophisticated productivity tool.