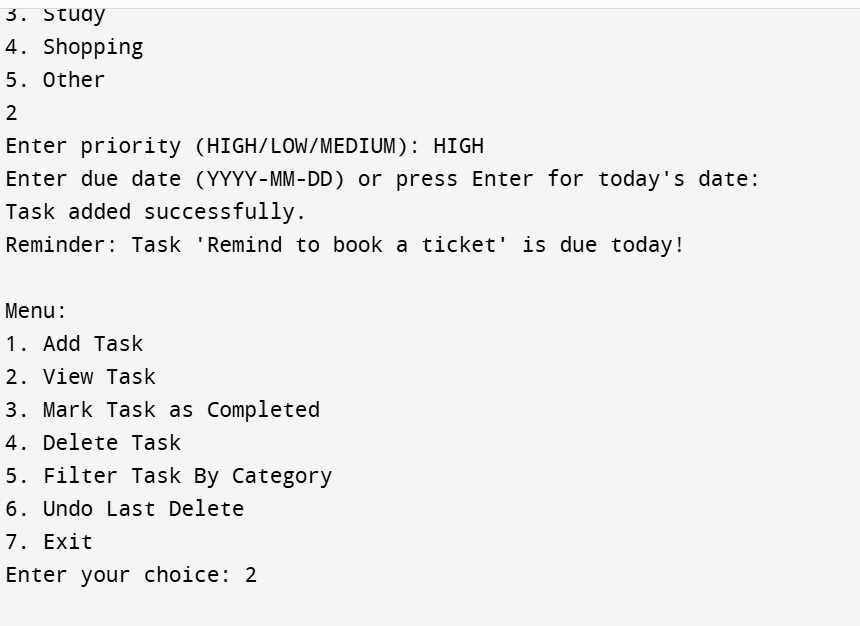
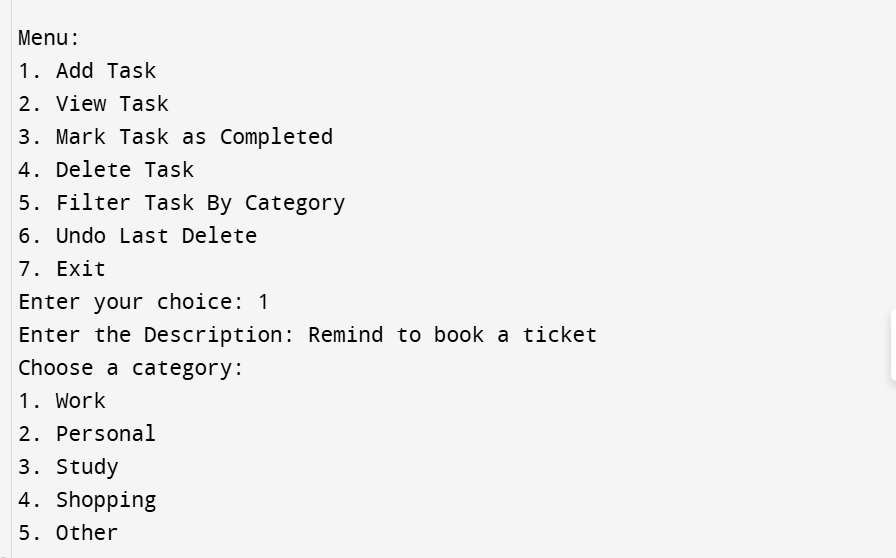
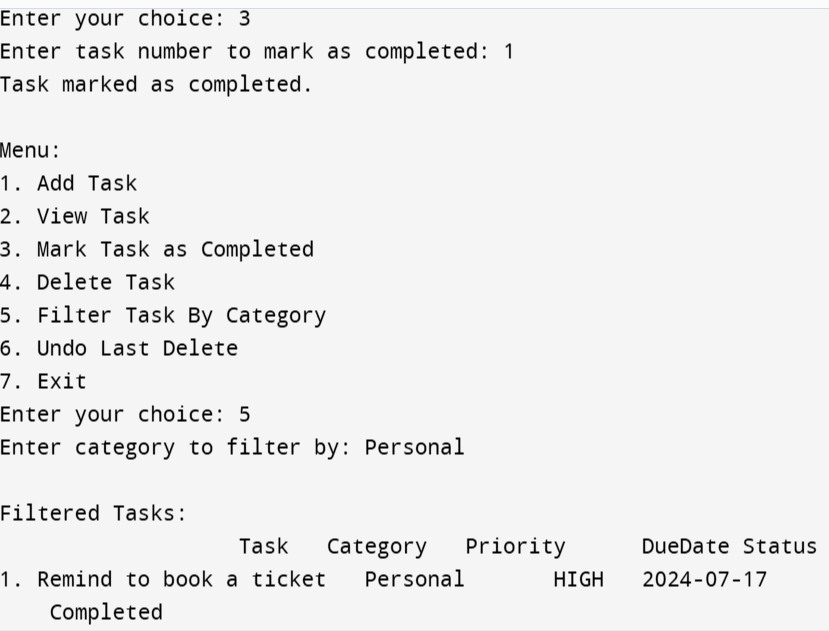
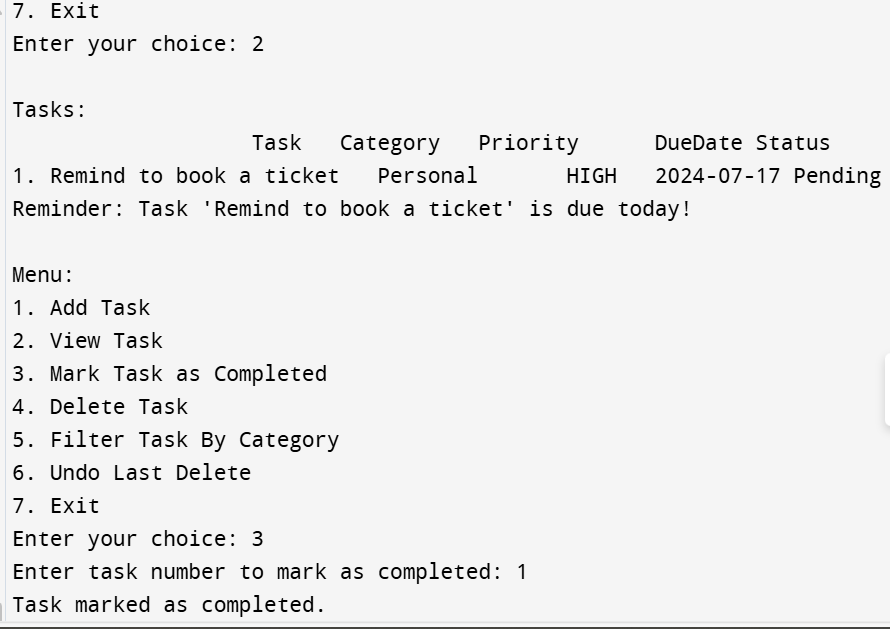
1. To do list Application:

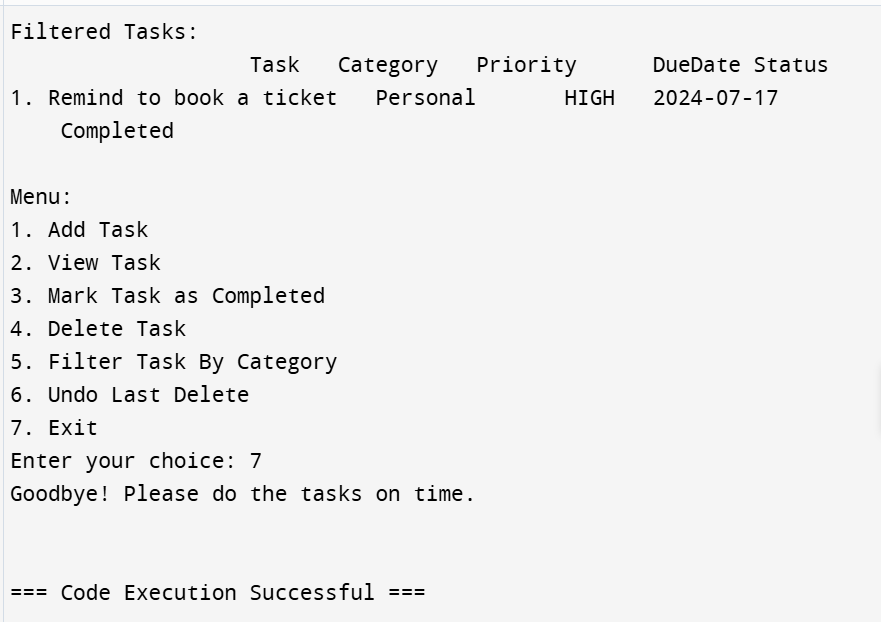
Introduction:

This project is a console-based task management system in C++ that allows users to add, view, complete, delete, filter, and manage tasks. The system categorizes tasks by priority (HIGH, MEDIUM, LOW) and allows for various operations to keep track of pending tasks efficiently. Users can also undo deletions and receive reminders for tasks due on the current date.

Output:







Introduction about functions:

 **Task Class Constructor:**

Task(string description, string category, string priority, string dueDate)

* **Purpose:** Initializes a task object with a description, category, priority, and due date. Sets the task status to not completed.
* **Parameters:**
  + description: Description of the task.
  + category: Category of the task (e.g., Work, Personal).
  + priority: Priority level of the task (HIGH, MEDIUM, LOW).
  + dueDate: Due date for the task.
* **getCurrentDate Function:**

string getCurrentDate()

* **Purpose:** Retrieves the current date in the format "YYYY-MM-DD".
* **Returns:** A string representing the current date.
* **compareTasksByPriority Function:**

bool compareTasksByPriority(const Task& a, const Task& b)

* **Purpose:** Compares two tasks based on their priority.
* **Parameters:**
  + a: The first task to compare.
  + b: The second task to compare.
* **Returns:** A boolean indicating whether the priority of task a is higher than task b.
* **displayTasks Function:**

void displayTasks()

* **Purpose:** Displays all tasks sorted by priority.
* **Details:** If no tasks are available, prompts the user to add tasks first.
* **markTaskAsCompleted Function:**

void markTaskAsCompleted()

* **Purpose:** Marks a task as completed based on user input.
* **Details:** Prompts the user for the task number and updates the task status to completed if the task number is valid.
* **deleteTask Function:**

void deleteTask()

* **Purpose:** Deletes a task based on user input.
* **Details:** Prompts the user for the task number, removes the task from the list if the task number is valid, and saves the deleted task for possible undo.
* **undoDelete Function:**

void undoDelete()

* **Purpose:** Restores the last deleted task.
* **Details:** If there is a task in the deleted tasks list, it adds it back to the main tasks list.
* **filterTaskByCategory Function:**

void filterTaskByCategory()

* **Purpose:** Filters and displays tasks based on the specified category.
* **Details:** Prompts the user for a category and displays tasks that match the category.
* **addTask Function:**

void addTask()

* **Purpose:** Adds a new task to the task list.
* **Details:** Prompts the user for task details (description, category, priority, and optionally due date), and adds the task to the list. Defaults to the current date if no due date is provided.
* **remindTasks Function:**

void remindTasks()

* **Purpose:** Reminds the user of tasks that are due today.
* **Details:** Checks for tasks with a due date matching the current date and notifies the user if any tasks are due.
* **displayMenu Function:**

void displayMenu()

* **Purpose:** Displays the main menu and handles user input for various task management operations.
* **Details:** Presents options to the user, including adding, viewing, completing, deleting, filtering tasks by category, undoing the last deletion, and exiting the program.
* **main Function:**

int main()

* **Purpose:** The entry point of the program.
* **Details:** Calls the displayMenu function to start the task management system