

**The Superior University**

**📝 Operating Systems Lab – Project Documentation Template**

**📌 Project Title**

**Project Management Scheduler using FCFS Algorithm in Python**

**(❁´◡`❁) Group Members**

*List of all team members with roll numbers:*

* Muhammad Hammas BSSE - 051
* Alisha Mustafa BSSE - 031
* Mahnoor Fatima BSSE - 059

**📂 GitHub Repository**

<https://github.com/mahnoor-004/Project-Management.-Scheduler/edit/main/README.md> (Mahnoor Fatima)

<https://github.com/Hammas004/Project-Management-Scheduler/blob/main/README.md> (Muhammad Hammas)

<https://github.com/alisha-008/Project-Management-Scheduler/edit/main/README.md> (Alisha Mustafa)

**🔧 Scheduling Algorithm Implemented**

✅ Tick the scheduling algorithm your group implemented:

* FCFS (First Come First Serve)
* SJF (Shortest Job First – Non-Preemptive)
* SJF (Preemptive)
* Round Robin

**📄 Project Description**

**What problem does your project solve?**

This project addresses the challenge of scheduling and managing tasks within a team-based environment, such as software development or project planning. Using the **First-Come, First-Served (FCFS)** algorithm, it provides a simple and fair method to assign and process tasks in the exact order they are received. It helps team leads or managers ensure that no task is skipped or delayed unfairly due to prioritization bias, offering a transparent scheduling process.

**📥 What inputs are required?**

The project requires the following inputs from the user:

1. **Member Name** – The name of the person who can be assigned tasks.
2. **Task Name** – A label for identifying each task (e.g., "Design", "Testing").
3. **Burst Time** – The time required to complete the task.
4. **Assigned Member (optional)** – A team member to whom the task is assigned.

**📤 What outputs are generated?**

After scheduling the tasks using the FCFS algorithm, the program produces:

* **Start Time** and **End Time** for each task
* **Waiting Time** for each task (how long it waited before execution)
* **Turnaround Time** (total time from task addition to completion)
* **Averages**:
  + Average Waiting Time
  + Average Turnaround Time
* A formatted table displaying all task details

**⚙️ How is the algorithm implemented?**

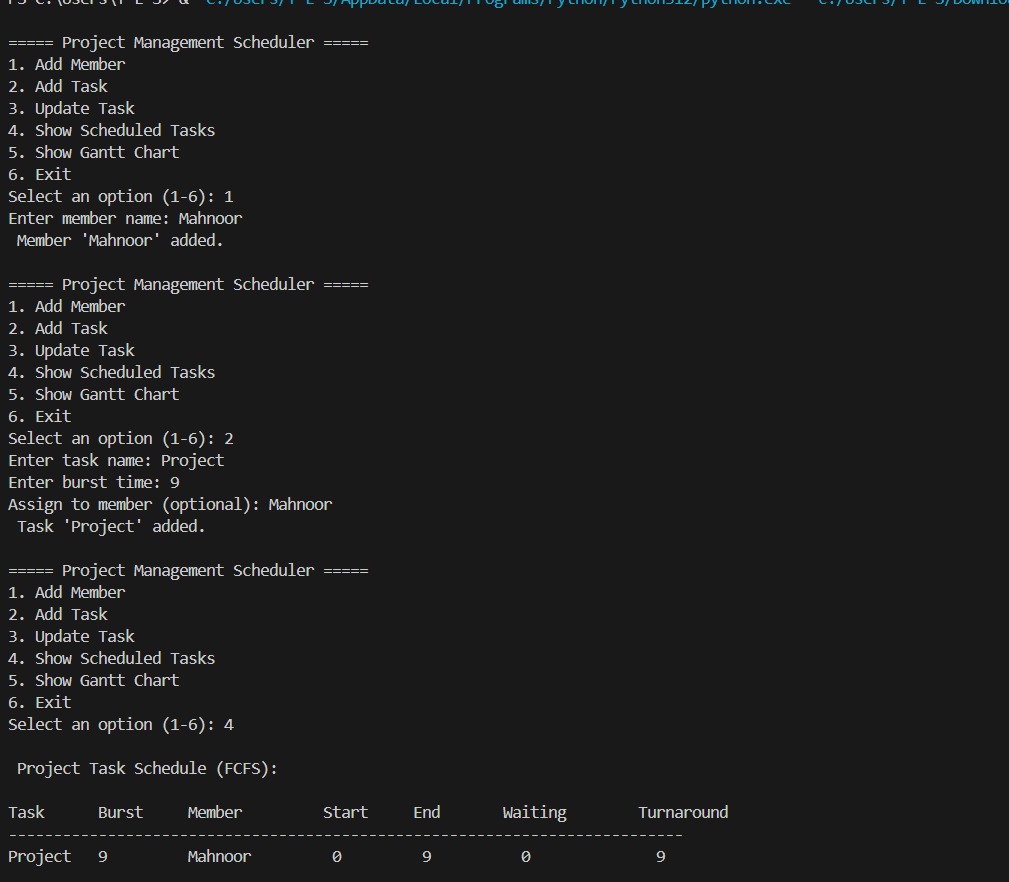
The FCFS (First-Come, First-Served) algorithm is implemented using the following logic:

* Tasks are stored in a list in the order they are added.
* **The scheduler calculates:**
  + **start\_time = current time**
  + **end\_time = start\_time + burst\_time**
  + **waiting\_time = start\_time**
  + **turnaround\_time = end\_time**
* After each task, the current time is updated to the end time of the last task.
* The results are then displayed in a clear tabular format.

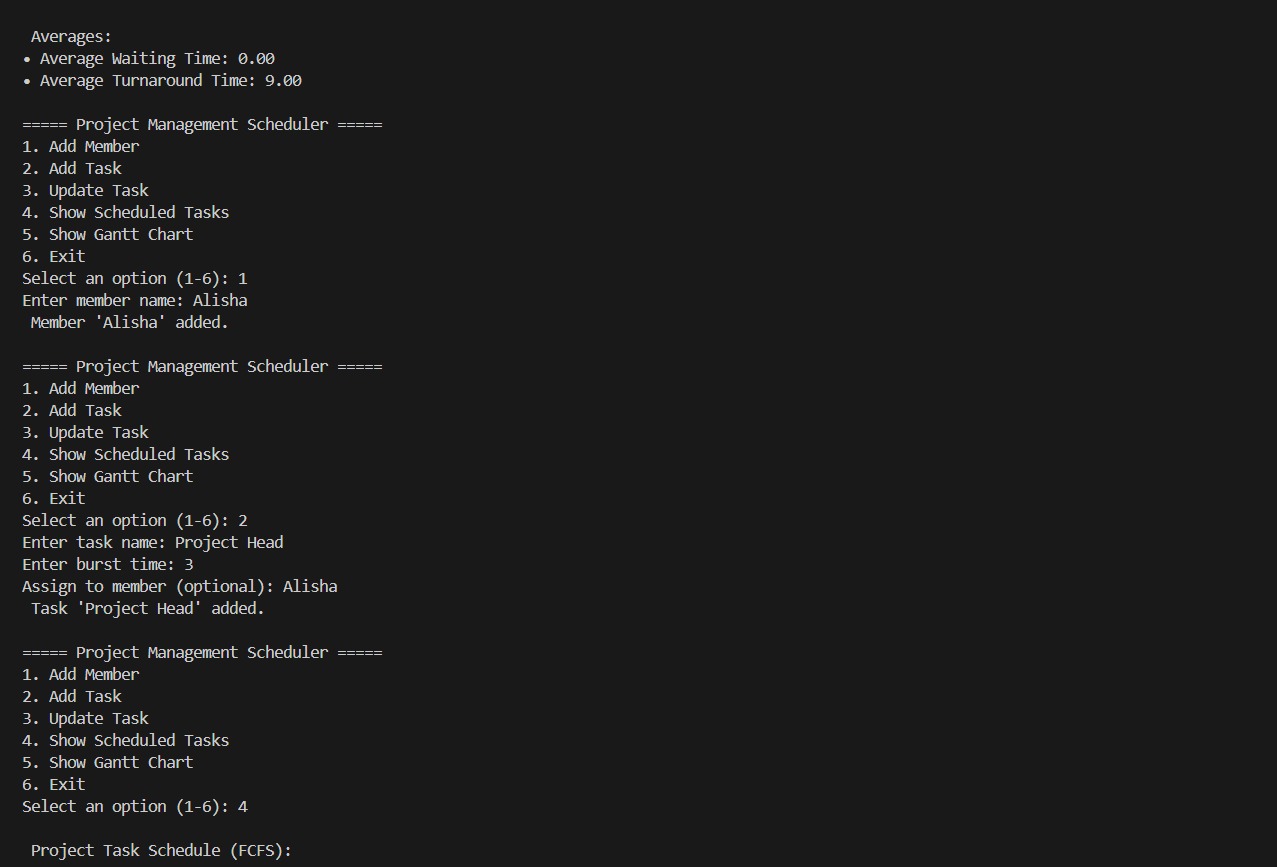
This non-preemptive algorithm ensures that tasks are executed sequentially, respecting their arrival order.

**📸 Output Screenshots**

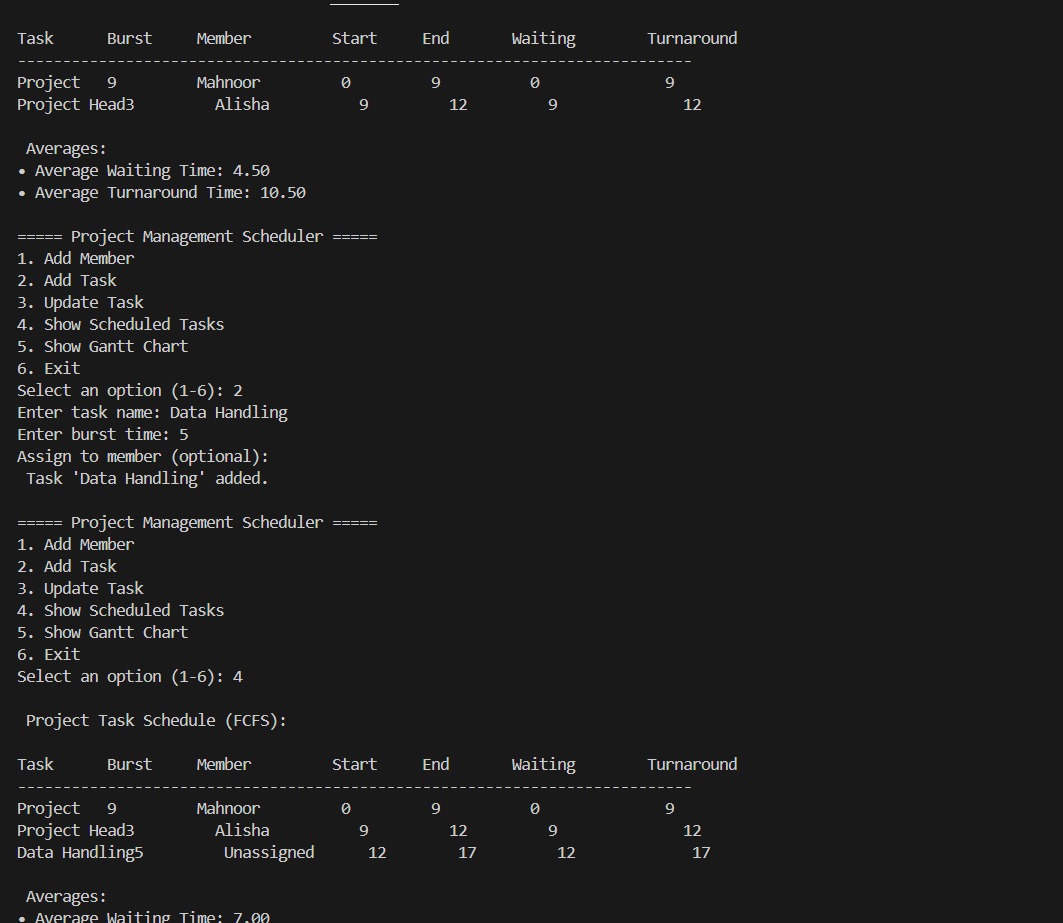
**Adding members:**

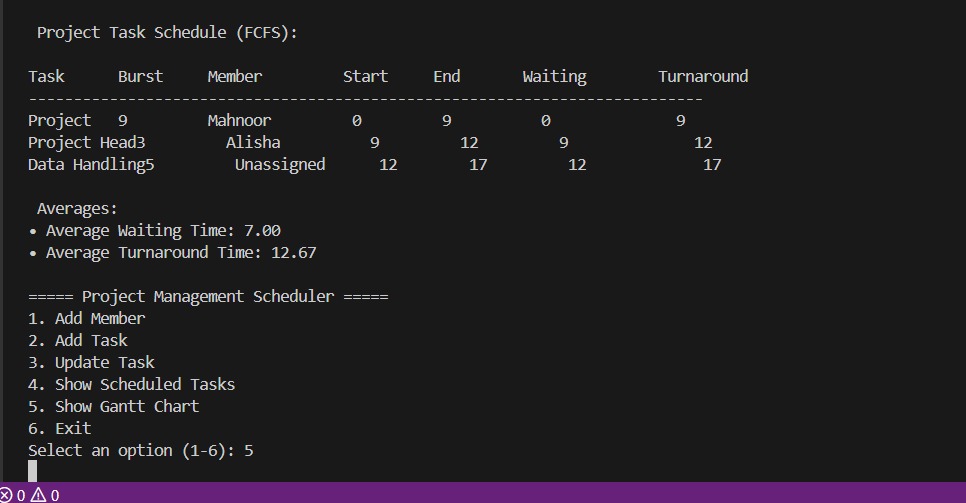
****

**Adding Tasks:**

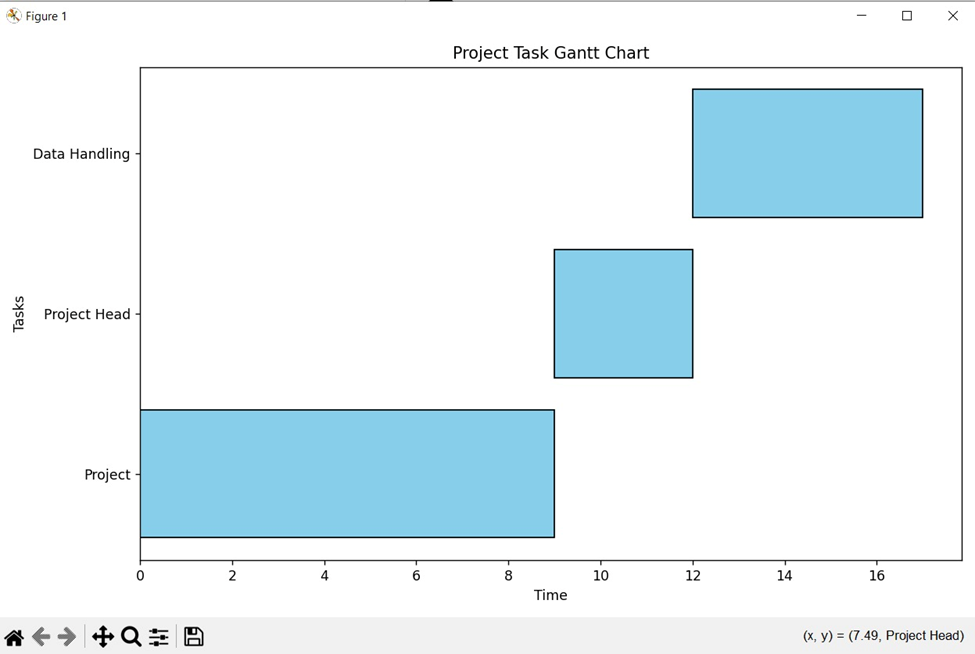
****

**Task Scheduling:**

****

****

**Gantt Chart with output:**

****

**🧠 Code Structure & Explanation**

**Class Structure**

**🔹 Task Class**

This class represents an individual task in the project.

**Attributes:**

* name: Task name
* burst\_time: Time required to complete the task
* assigned\_member: The member assigned to the task
* waiting\_time: Time task has to wait before execution
* turnaround\_time: Total time from task arrival to completion
* start\_time: When the task starts execution
* end\_time: When the task finishes execution

**Method:**

* update(): Updates the burst time or assigned member if new valid values are provided.

**🔹 Project Management Scheduler Class**

This is the main class that manages all project tasks and team members.

**Attributes:**

* tasks: A list storing all the tasks
* members: A list storing team members' names

**Methods:**

* add\_member(): Adds a new team member if not already added.
* add\_task(): Adds a new task with name, burst time, and optional member.
* update\_task(): Updates an existing task's burst time or assigned member based on user input.
* schedule(): Implements the **FCFS scheduling algorithm** by setting start time, end time, waiting time, and turnaround time for each task.
* display\_schedule(): Displays all scheduled tasks with their timing details and computes average waiting and turnaround times.

**⚙️ Main Function Workflow (main())**

1. Presents a menu to the user with options to:
   * Add a team member
   * Add a new task
   * Update a task
   * View scheduled tasks
   * Exit the program
2. Takes user input to select an option.
3. Calls the appropriate method from Project Management Scheduler based on the user's choice.

**📊 Performance Metrics**

|  |  |
| --- | --- |
| **Metric** | **Value** |
| **Average Waiting Time** | Calculated dynamically based on task inputs (e.g., 5.25) |
| **Average Turnaround Time** | Calculated dynamically based on task inputs (e.g., 10.75) |
| **Time Quantum (if RR)** | *Not Applicable (N/A)* – This project uses **FCFS**, not Round Robin |

**🛠️ Challenges Faced**

1. **Incorrect Time Calculations:**  
   Waiting and turnaround times were miscalculated initially. We fixed this by properly updating current- time and applying the correct formulas.
2. I**nvalid Member Assignment:**  
   Tasks could be assigned to non-existent members. We added validation to ensure only existing members can be assigned.
3. **Update Input Errors:**  
   Blank or invalid inputs during task updates caused issues. We used error handling and input checks to prevent crashes.