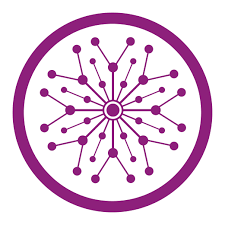
**

**The Superior University**

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

**📌 Project Title**

“Printing task scheduler using FCFS”

**👥 Group Members**

* Sara Ch (SU92-BSSEM-F23-080)

**📂 GitHub Repository**

✅ **Both the Python code file and this documentation must be uploaded to a public GitHub repository.**

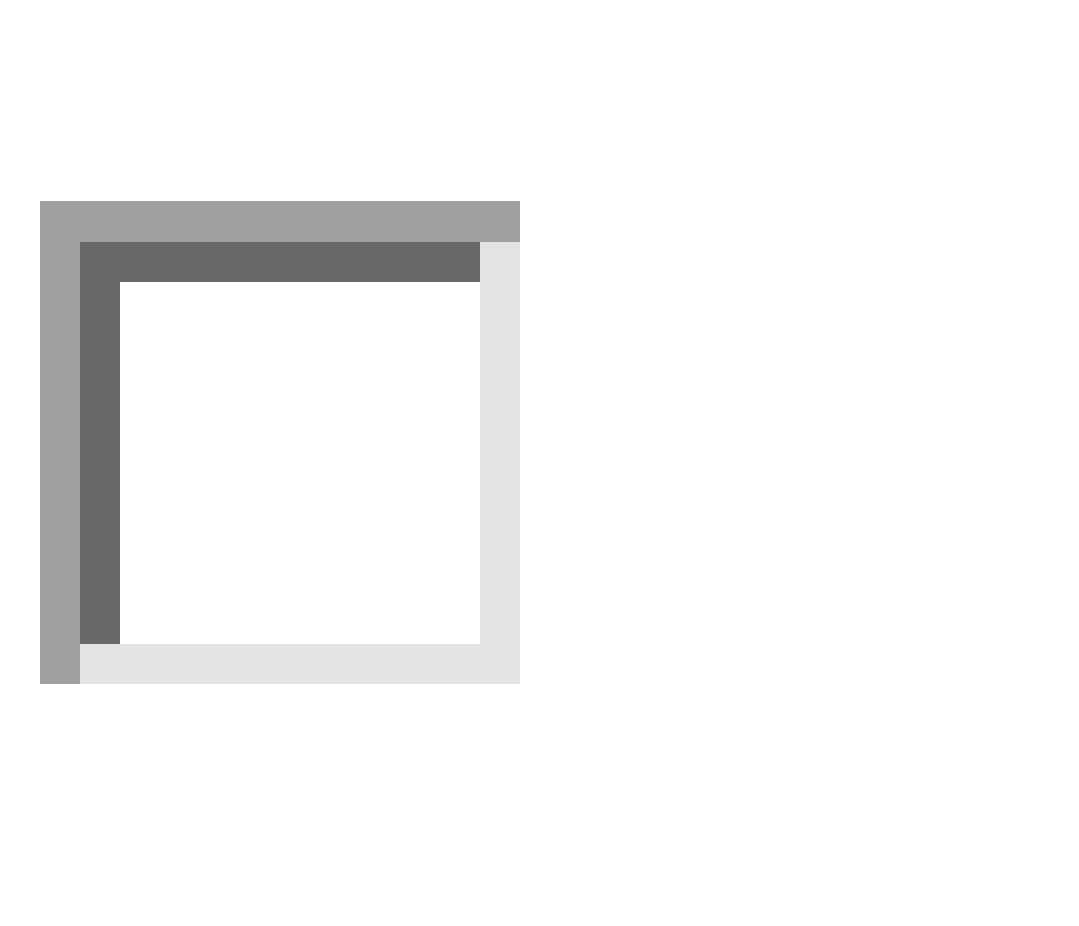
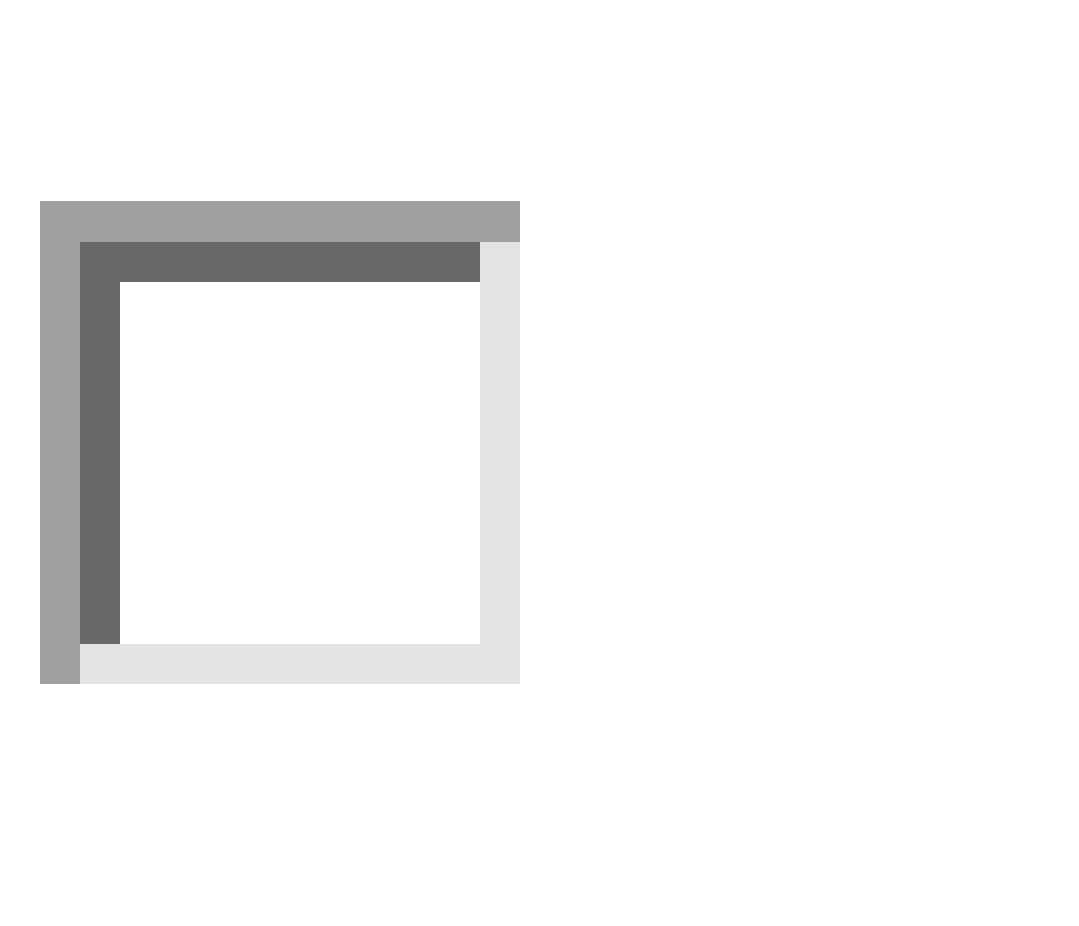
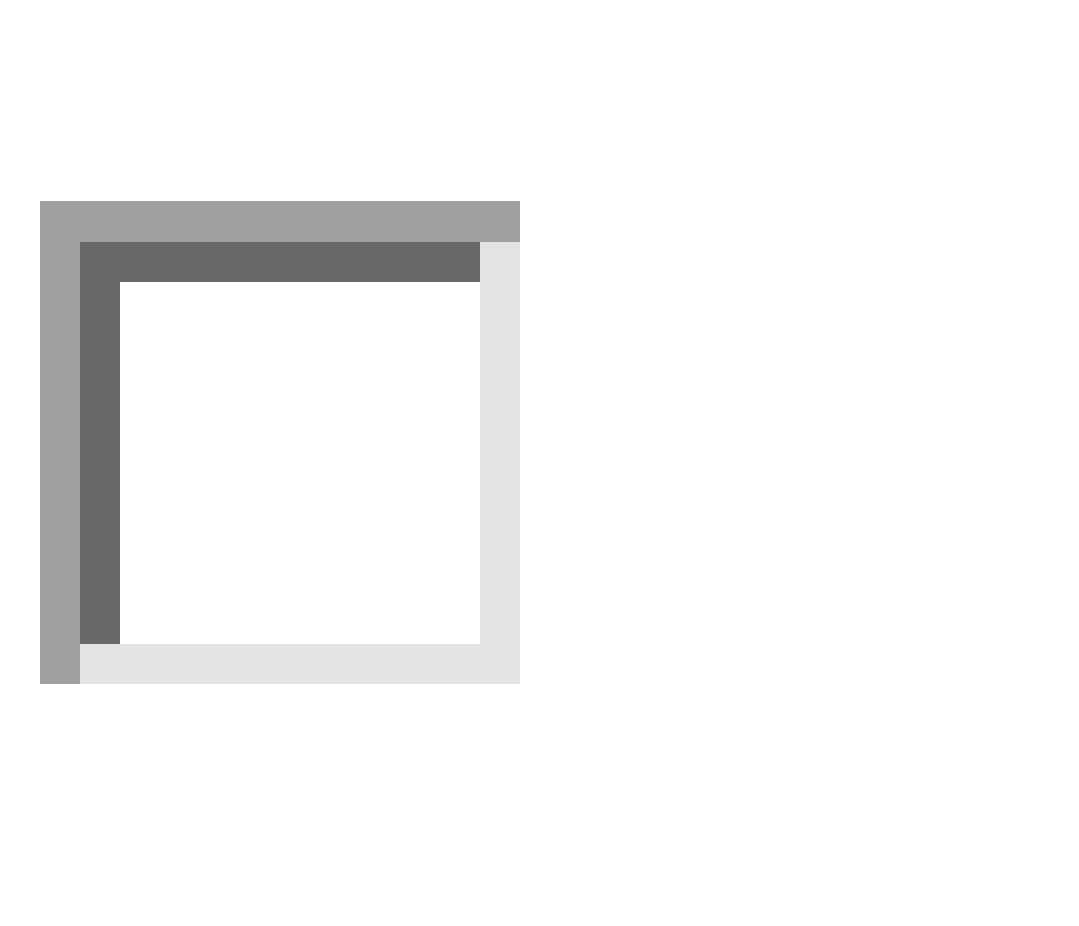
**GitHub Repository Link:**

<https://github.com/SaraCh28/Printing-task-scheduler->

Your repository should include:

* ✅ printing-task-scheduler.py (Python file with FCFS implementation)
* ✅ README.md (Project overview and instructions)
* ✅ This completed documentation (.docx)
* ✅ 2–3 output screenshots (input, output table, average metrics.docx)

**🔧 Scheduling Algorithm Implemented**

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

**📄 Project Description**

This project simulates a basic **printing task scheduler** using the **First Come First Serve (FCFS)** algorithm. It mimics how print jobs are handled by a printer based on the order in which they arrive.

**Inputs Required:**

* Arrival Time of each print job
* Burst Time (time needed to print the job)

**Outputs Generated:**

* Completion Time of each job
* Turnaround Time (Completion - Arrival)
* Waiting Time (Turnaround - Burst)
* Average Waiting and Turnaround Times

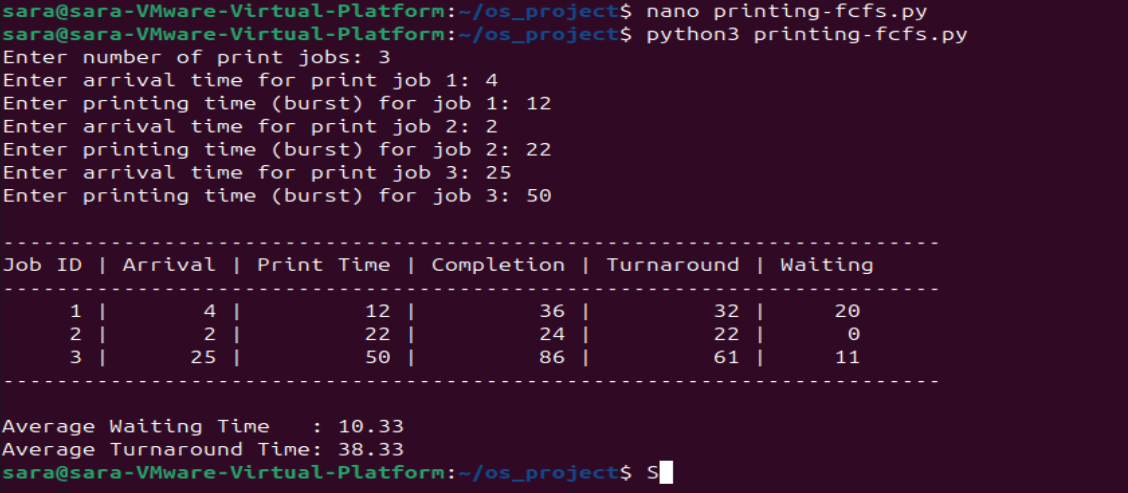
**Implementation:**

* The scheduler accepts multiple print jobs.
* It sorts them by arrival time.
* Each job is executed in order of arrival.
* Idle time is managed by fast-forwarding the clock if no jobs are available.
* Average metrics are computed and displayed in a structured format.

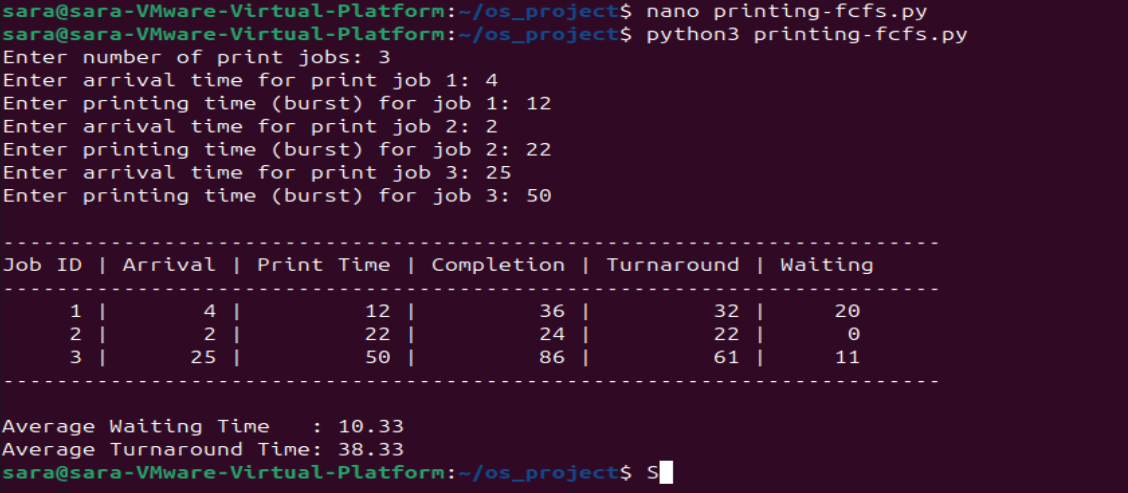
**📸 Output Screenshots**

Attach 2–3 screenshots of:

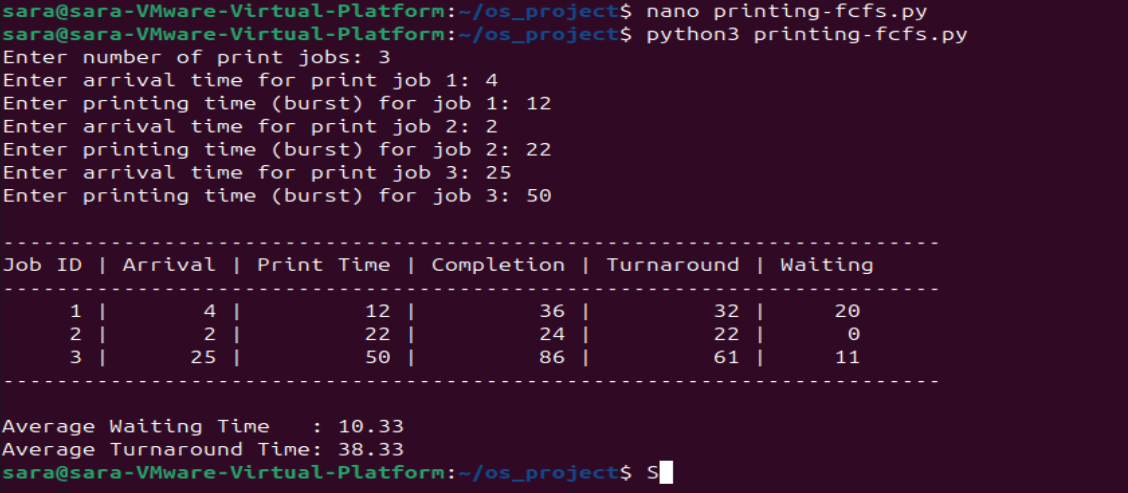
* Input values entered



* Result table with Completion, Turnaround, Waiting Times



* Final calculated waiting and turnaround times



**🧠 Code Structure & Explanation**

* **Main Script:** The Python code is written in a linear, user-friendly format using dictionaries and lists.
* **Core Logic:**
  + Jobs are stored in a list with arrival and burst times.
  + Sorted by arrival time.
  + Each job’s completion, turnaround, and waiting times are calculated.
  + The average times are computed and displayed.
* **No external libraries** are used, making the code fully portable and beginner-friendly.
* **Commenting & Readability:** Comments and print formatting are included for clarity.

**📊 Performance Metrics**

| **Metric** | **Value** |
| --- | --- |
| Average Waiting Time | 34.33 |
| Average Turnaround Time | 116.33 |
| Time Quantum (if RR) | N/A |

**🛠️ Challenges Faced**

1. **Incorrect Waiting Time Calculation:** Initially, I didn't account for printer idle time between jobs. This was fixed by adding a check to fast-forward current\_time if the next job arrives later than the previous job finishes.
2. **Output Alignment Issues:** Aligning the output table was tricky. I used formatted string printing to make it clean and readable.
3. **Average Time Accuracy:** Ensuring the average waiting and turnaround times were computed only **after all jobs were processed**, not during the loop, was important to get accurate results.