**The University of Azad Jammu and Kashmir, Muzaffarabad**



|  |  |
| --- | --- |
| **Submitted to:** | **Engr. Sidra Rafique** |
| **Course Title:** | **Operating System** |
| **Course Code:** | **SE-3205** |
| **Session:** | **2022-26** |
| **Semester:** | **6th** |
| **Roll No:** | **2022-SE-01** |
| **Submitted from:** | **Khurram Farman** |
| **Lab Task No** | **10** |
|  |  |

**Bachelors of Science in Software Engineering (2022-26)**

**Department of Software Engineering**

# ****CPU Scheduling Algorithms Report****

## ****Introduction****

This report presents the implementation and comparison of CPU scheduling algorithms. The focus is on **Shortest Job First (SJF – Preemptive)** and the **Priority Based Scheduling Algorithm**.  
These algorithms are widely used in process scheduling and play a significant role in determining CPU performance and efficiency.

## ****Objectives****

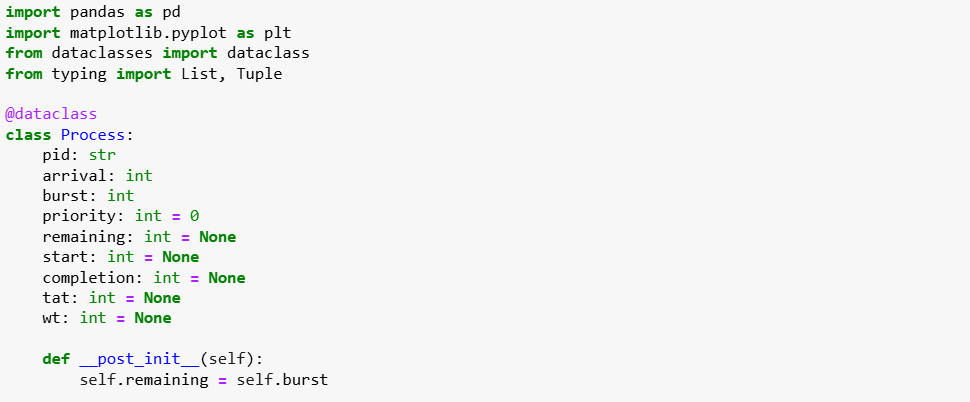
* To understand the working of SJF (Preemptive) and Priority scheduling algorithms.
* To analyze performance metrics such as Waiting Time and Turnaround Time.
* To compare results of both algorithms.

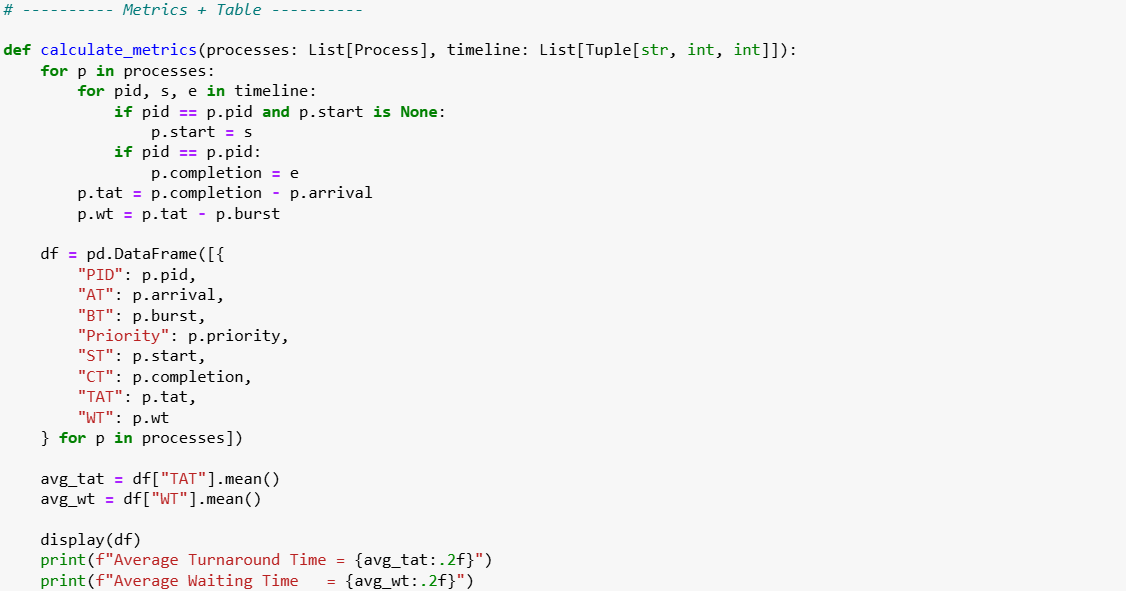
## ****Methodology****

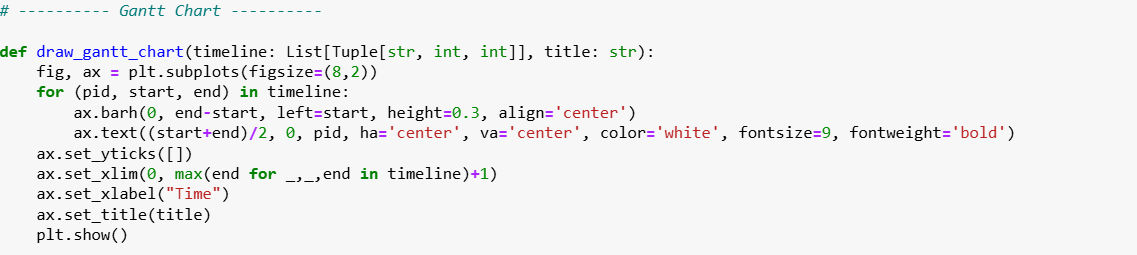
The algorithms were implemented in Python using Jupyter Notebook.  
The following steps were followed:

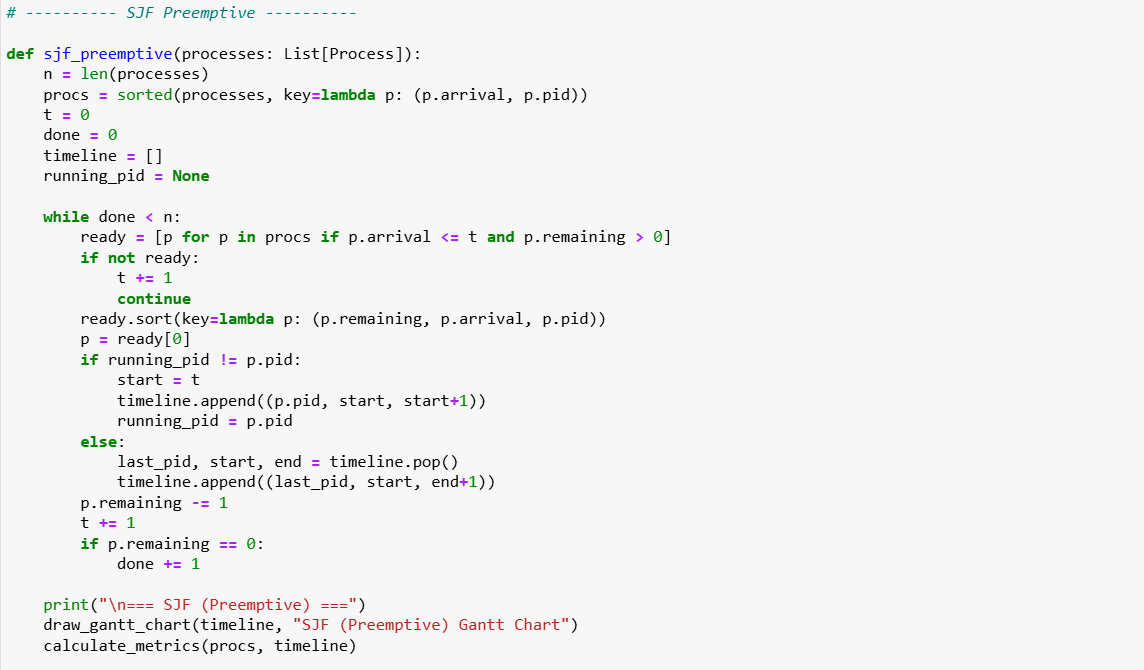
1. Define processes with their arrival time, burst time, and priority (for priority scheduling).
2. Apply **SJF Preemptive** algorithm.
3. Apply **Priority Scheduling** algorithm.
4. Compute completion time, turnaround time, waiting time, and their averages.
5. Compare results.

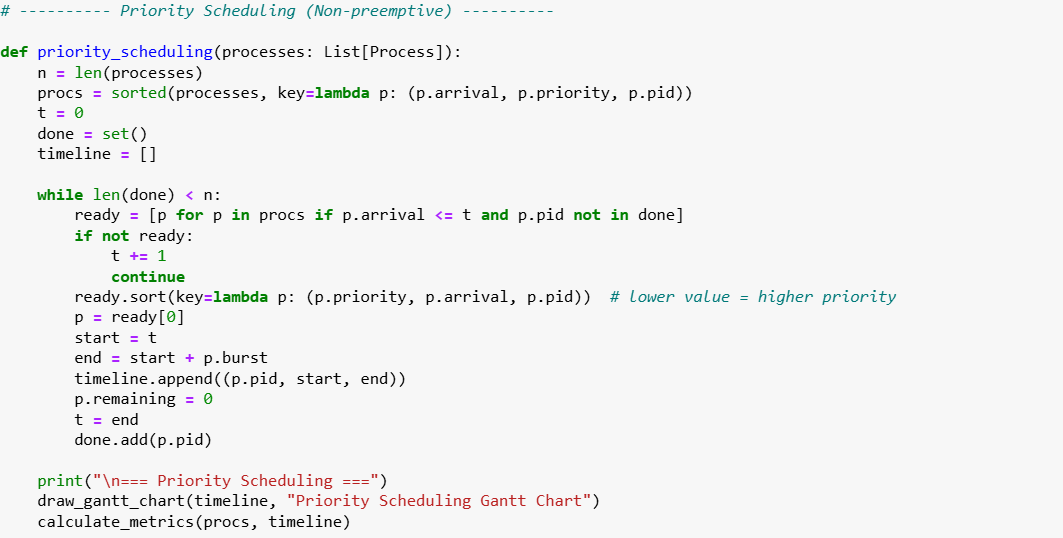
## ****Code Implementation****

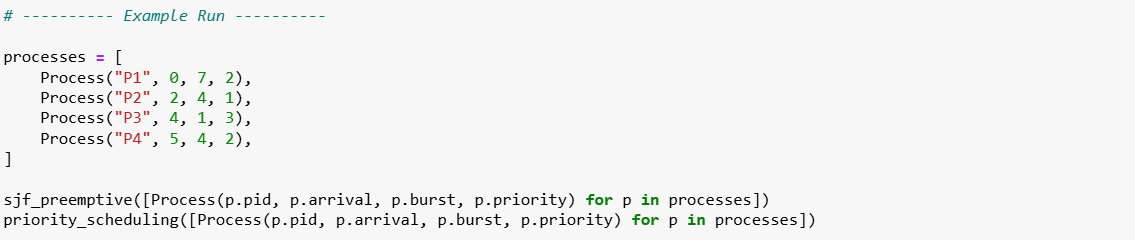


****

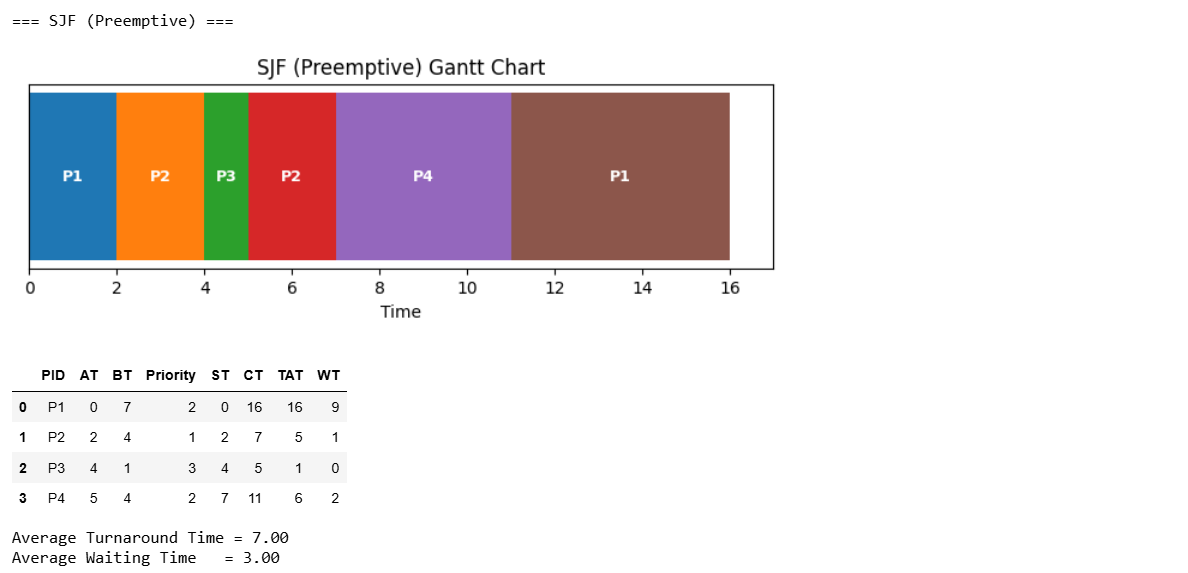
****

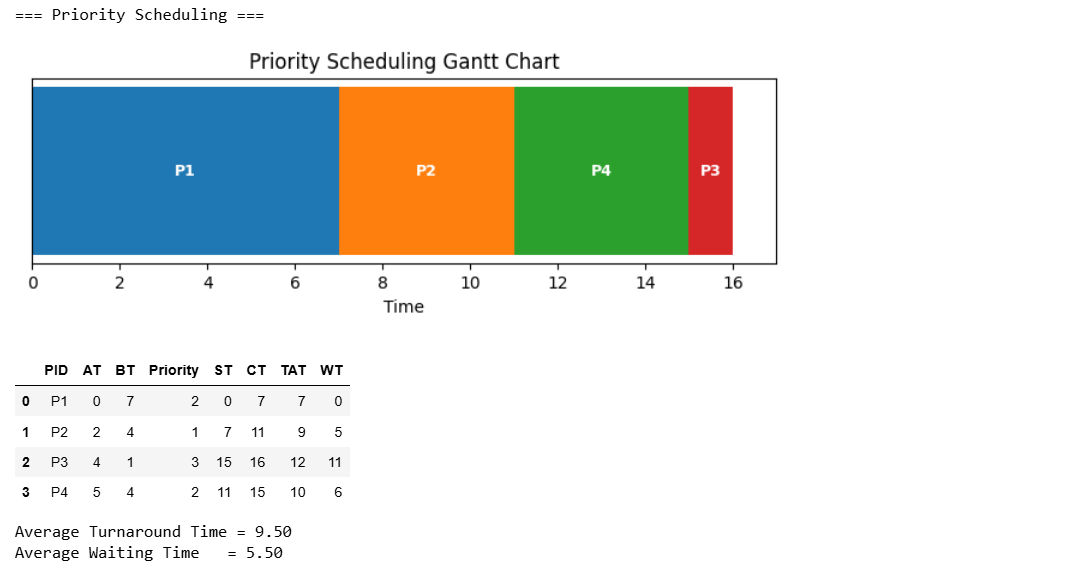
****

****

****

## ****Output****





## ****Results and Discussion****

* **SJF (Preemptive)** selects the process with the shortest remaining time whenever the CPU becomes free. This leads to lower average waiting times but may cause frequent context switches.
* **Priority Scheduling** selects the process with the highest priority. If two processes have the same priority, they are scheduled by their arrival order. This can lead to **starvation** for low-priority processes if not handled properly

### ****Comparison Table****

|  |  |  |
| --- | --- | --- |
| Algorithm | Average Waiting Time | Average Turnaround Time |
| SJF (Preemptive) | 7 ms | 3 ms |
| Priority Scheduling | 9.50 ms | 5.50 ms |

## ****Conclusion****

* **SJF (Preemptive)** provides better turnaround and waiting times but increases context switching overhead.
* **Priority Scheduling** ensures critical processes get CPU time earlier but may lead to starvation without aging.
* Both algorithms demonstrate how scheduling decisions affect CPU performance differently.

This study shows that selecting an appropriate scheduling algorithm depends on the nature of the workload and system requirements.