School of Engineering
TEMASEK POLYTECHNIC

Diploma in Computer Engineering

Computer Architecture & Operating Systems Project

Declaration of Originality

By submitting this work, I / we declare that

- I am / we are the originator(s) of this work.
- I / we have appropriately acknowledged all other original sources used in this work.
- I / We understand that Plagiarism is the act of taking and using the whole or any part of another person's work and presenting it as my/ our own without proper acknowledgement.
- I / We understand that Plagiarism is an academic offence and if I am/we are found to have committed or abetted the offence of plagiarism in relation to this submitted work, disciplinary action will be enforced.

Student Name:	Admin No:	Student Signature	Date
Camille Villarama	1500232B		12 Feb 18
Mohammed Shereif	1505491I		12 Feb 18
Muhammad Haikal	1506520D		12 Feb 18
Nazrul	1500983J		12 Feb 18

Project Report

1. Introduction

a. Project Description

The purpose of this project is to create an application that shows the design and functionalities of CPU Scheduling using Non-Preemptive Priority and First-Come, First-Serve (FCFS) Scheduling, as well as the Main Memory Management algorithm using Best-Fit Allocation.

The application was implemented using Python to achieve the following objectives:

i. Create, store and save the arrival time and the burst time of various processes for individual CPU Scheduling Algorithm
ii. Display the Gantt Chart for both CPU Scheduling Algorithms according to the values entered in the processes
iii. Calculate the average turnaround time and the waiting time using Non-Preemptive Algorithm and FCFS Scheduling Algorithm
iv. Demonstrate how Best-Fit Algorithm place processes in the main memory

b. Task Allocation

Project Segment	Action by
Non-Preemptive SJF Algorithm	Haikal
FCFS Scheduling Algorithm	Shereif
Main Memory Management	Nazrul
GUI Design	Camille

2. Theoretical Background

a. CPU Scheduling: Non-Preemptive SJF Algorithm

Shortest-Job-First can be preemptive or non-preemptive but in this application, non-preemptive is implemented.

Unlike Pre-emptive Algorithm, non-preemptive Algorithm

executes the process fully. It also takes into account the arrival time of each processes. So, in an instance where two process has the same burst time, the arrival time will be the priority. The algorithm is almost similar to FCFS Scheduling however, once the arrival time of the processes have passed, the process with the shortest burst time will be executed first.

b. **CPU Scheduling: First-Come, First-Serve Algorithm**

For this algorithm, the process that request the CPU first will be allocated the time to execute first. In our application, the user will then input the values for the arrival time and burst time (how long for process to be executed) before giving the Gantt chart and the calculated average of waiting time and turnaround time.

c. **Main Memory Management: Best-Fit Algorithm**

Best-fit Algorithm allocate processes into the main memory that can accomodate the size of the process memory . For an example, the given memory partitions for the main memory are 100 KB and 200 KB. The process that arrives has a memory of 150 KB. Using Best-fit Allocation, the process will be allocated into the 200KB memory partition, with 50 KB memory left. This 50 KB, is the value for the internal fragmentation.

In a case where the memory of a process is too big and couldn't be allocated into the main memory it will produce a value for the external fragmentation.

3. Software Design (Project Flow)

This application is split into 2 main parts: CPU Scheduling and Best-fit Allocation Memory Management.

a. **Project Flow**

i. **CPU Scheduling: First-Come, First-Serve Algorithm and Non-Preemptive SJF Scheduling Algorithm**

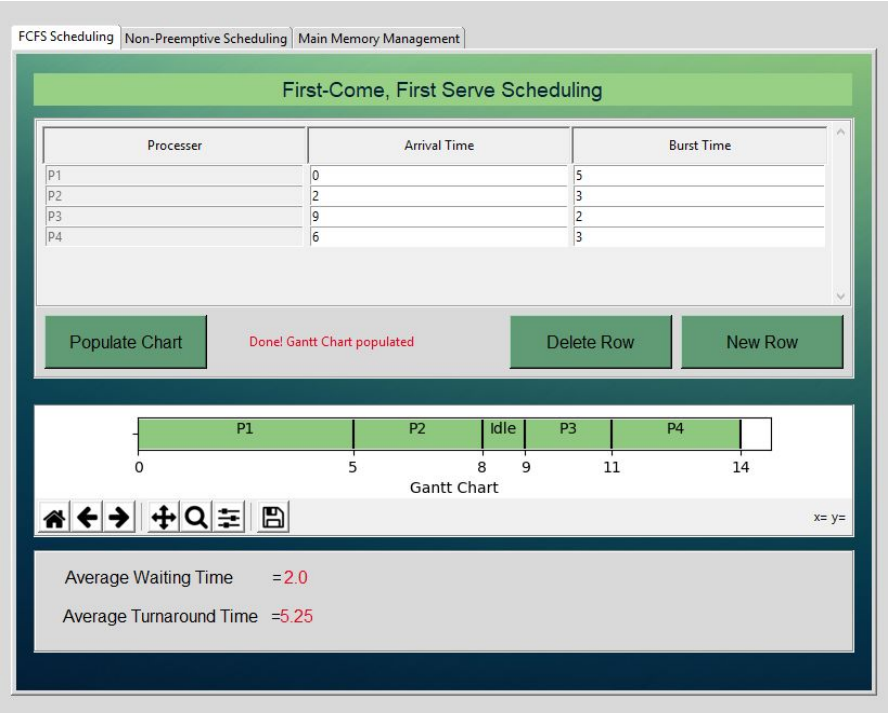


Figure A. FCFS GUI Design

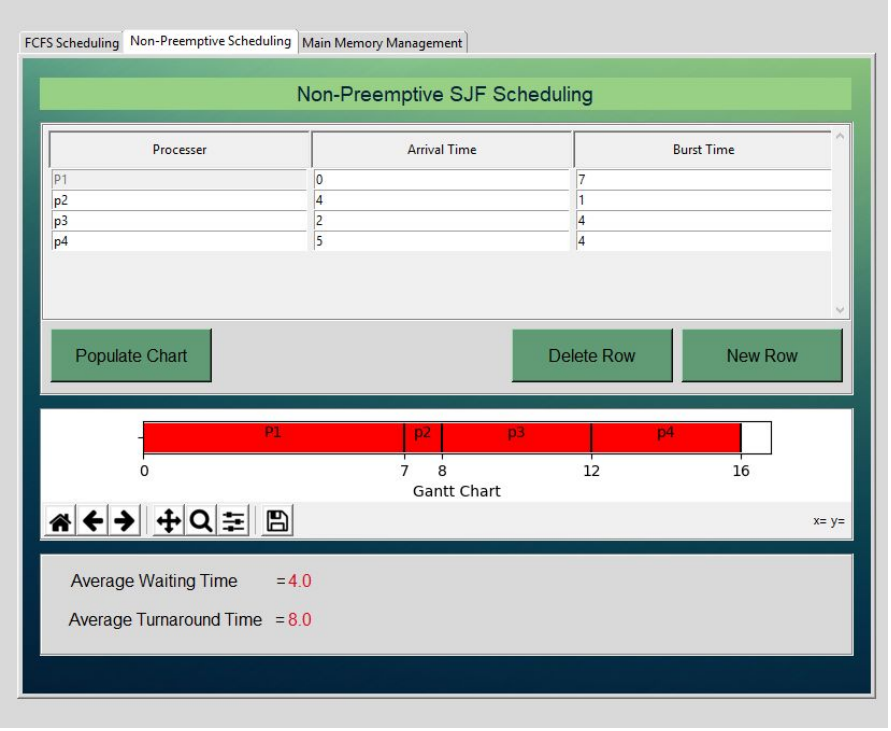


Figure B. Non-Preemptive GUI Design

Firstly, the user needs to fill in the values for the arrival time and burst time for the processes. In order to generate a row for a new process, the user just needs to click the button ‘New Row’.
The user is also able to delete a process if they wish to

discard the process in the calculation.

Once the ‘Populate Chart’ button is clicked, a Gantt chart is generated corresponding the arrival and burst time entered for the processes. At the same time, the average waiting time and turnaround time will also be calculated and shown.

Furthermore, there are features involving the Gantt chart where the user is able to save, zoom in and navigate left or right.

ii. **Main Memory Management: Best-Fit Algorithm**

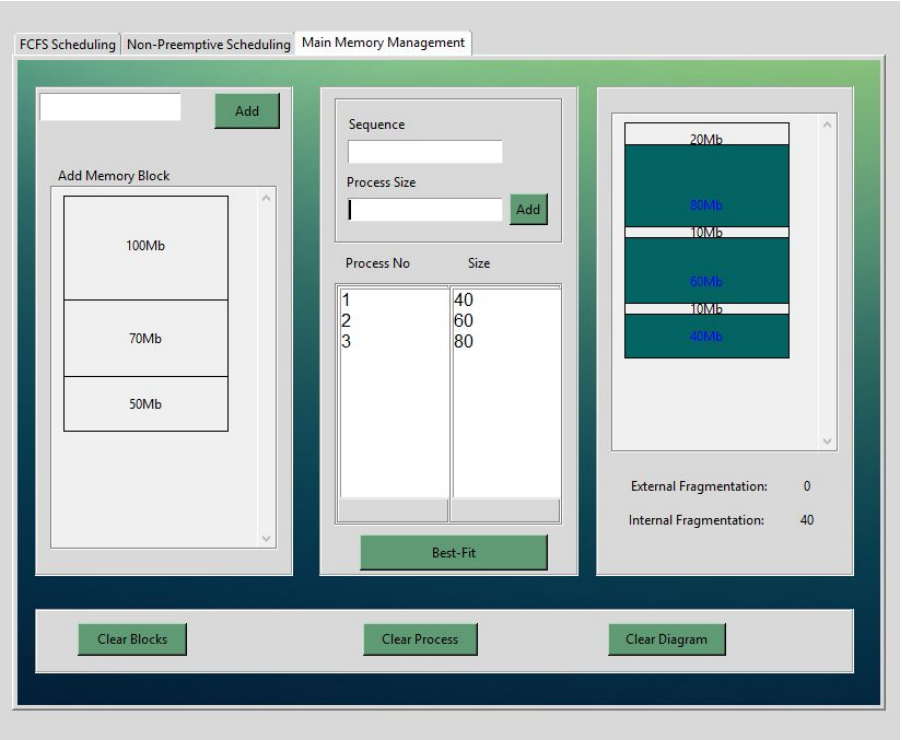


Figure C. Best-fit Allocation

To demonstrate how Best-fit Algorithm works, the user needs to select the third tab on the app notebook. After that, the user needs to add a few values for the memory block. Next, the user then need to input the sequence and process size of the processes that arrives.

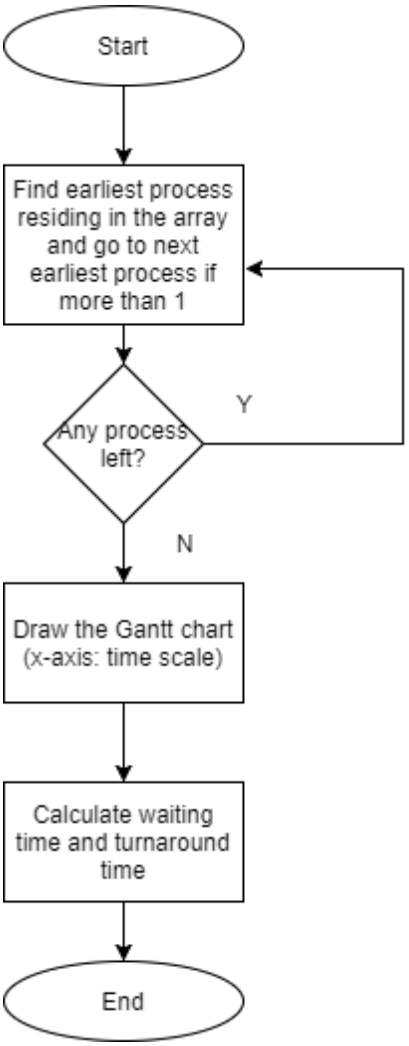
Once the necessary parameters are fulfilled, a diagram will be generated to demonstrate how Best-fit Allocation place processes in the main memory. At the same time, the values for the internal and external fragmentation will also be calculated and displayed.

There are also function button such as ‘Clear Block’ which resets the memory block diagram that was

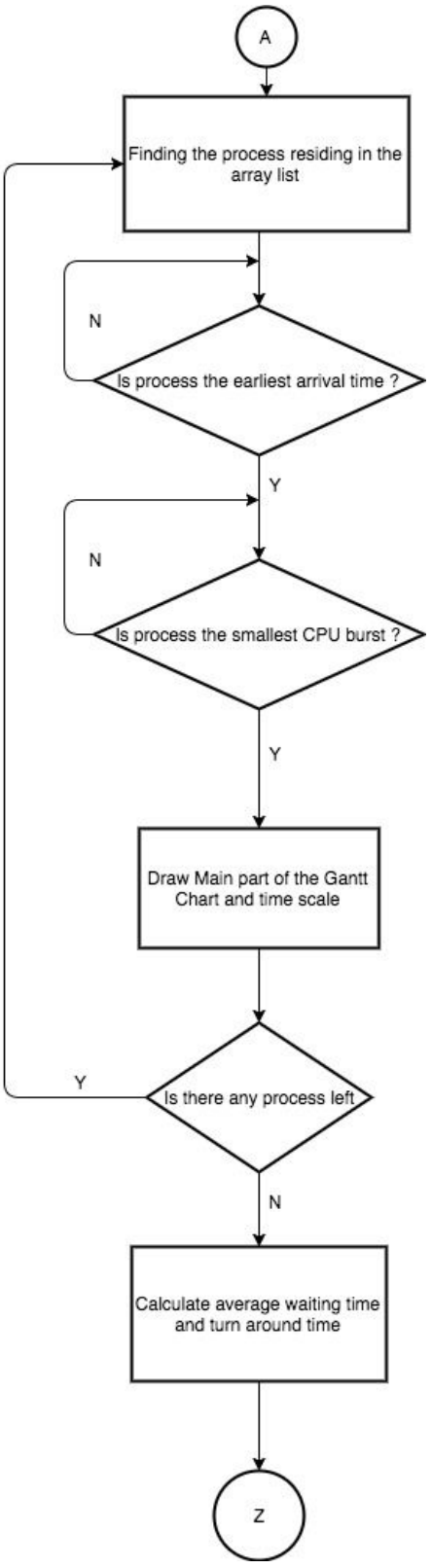
previously generated, ‘Clear Process’ which clears the table containing the processes and ‘Clear Diagram’ which deletes the memory map.

b. **Flowchart Diagrams**

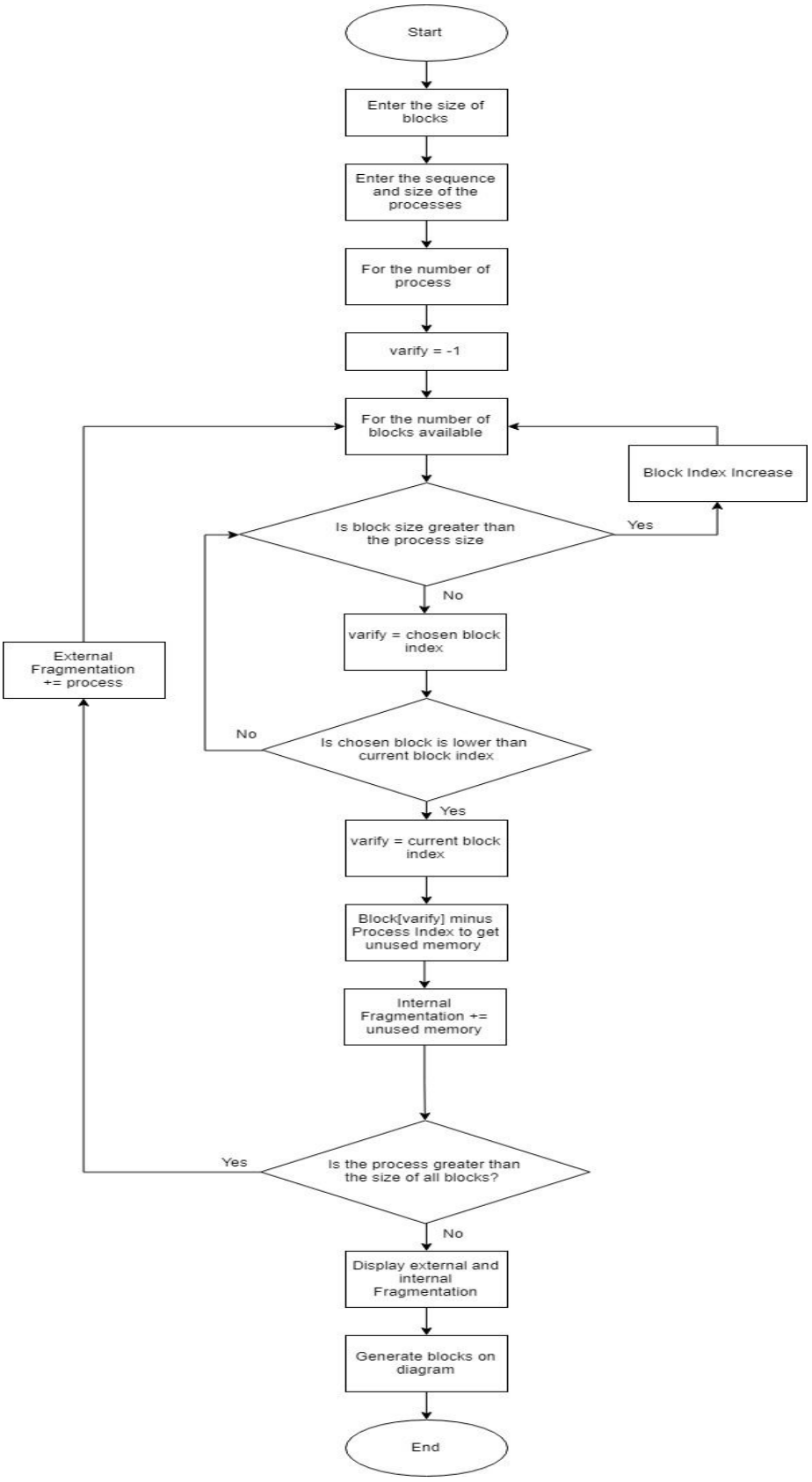
i. FCFS Scheduling



ii. Non-Preemptive SJF Scheduling



iii. Best-fit Allocation



4. Conclusion

In conclusion, all the objective stated above for this project has been fulfilled. The project has also been developed using a newly learned language, Python that enabled team to gain new knowledge and experience.