FACULTY OF SCIENCE AND TECHNOLOGY

ACADEMIC YE AR 2020/21

SEMESTER 1

COMP3101 – Operating Systems Final Assignment

Design Document

Date: November 30, 2020

Group Members

Lejandru Richards	620106944
Matthew Chevannes	620106393
Ranaldo Campbell	620106208
Romario Bennett	620110226
Shamar Nelson	620108459

Overview

The aim of this project was to show how an OS works in terms of Scheduling processes in a visual manner with proper pacing that allows the viewer to follow what is happening, with the aim of helping anyone who finds it difficult to see/visualize how the written and theoretical OS material really works.

For our Group we attempted to demonstrate the steps involved in 4 **Process Scheduling Algorithms**, which include:

First Come First Serve Algorithm (FCFS)
Shortest Job First Algorithm (SJF)
Priority Scheduling Algorithm
Round Robin Scheduling

Design

The overall goal of this assignment and program was to display in an animated or simulated manner, to anyone who might not understand fully the concept or execution of the mentioned OS Concepts.

Our Program provides this, a simple visual aid, as explained above. It was not intended to be a complex demonstration, but one that was simple enough to give some perspective into the theory. It was designed in an easy to follow manner where users would be guided, by way of a visual user interface, to enter data and view the results, having a basic understanding of Process Scheduling should be sufficient for them to benefit fully from our implementation.

Tool Stacks used in this assignment were
Python3.8

Tkinter GUI (a Python GUI Library)

Javasript

Html

CSS

Functional Specification

The user interface is presented with several buttons, labels and entry boxes each with separate functionalities, and descriptions of what should be displayed under each. We tried to make as intuitive as possible to the user

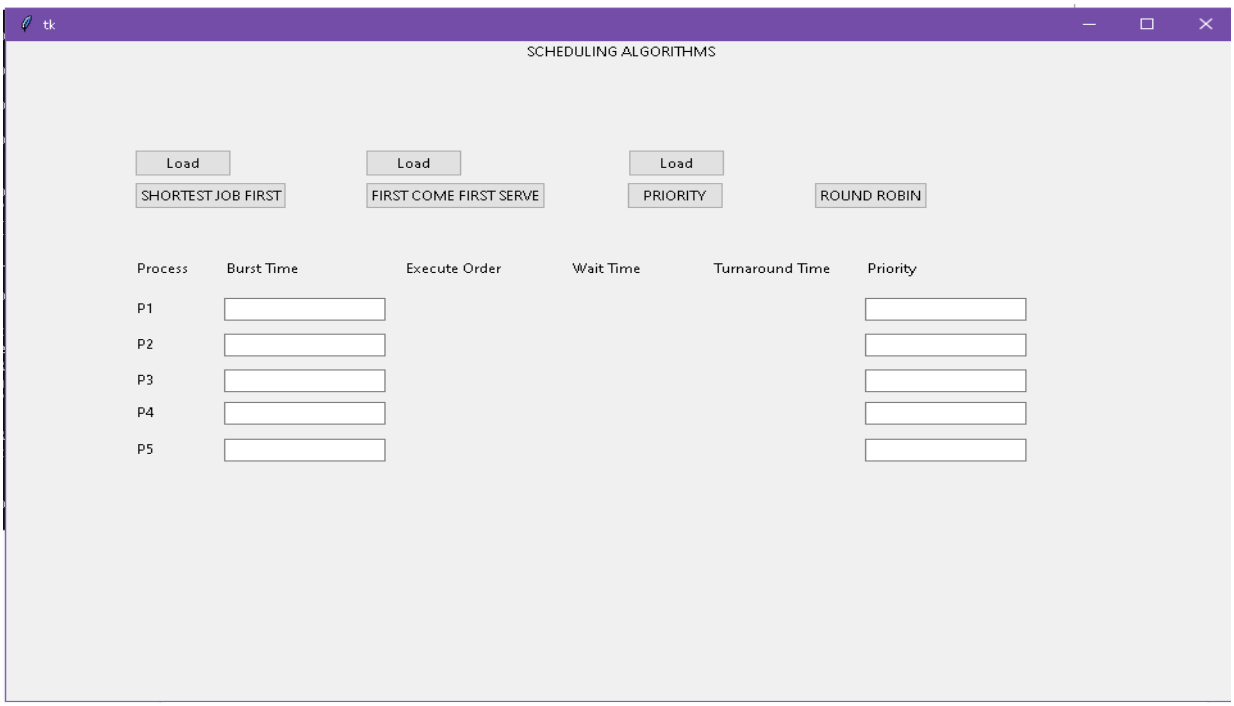


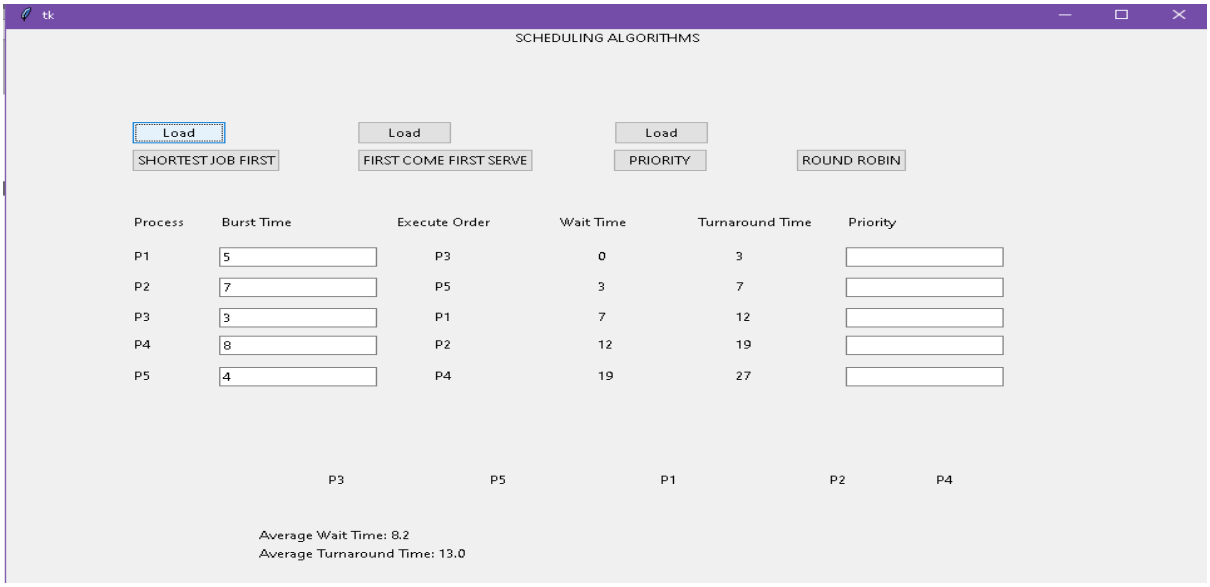
Fig1.1

Upon starting the application and the GUI is loaded, depending on which scheduling algorithm one might want to show the process for, different options selecting to reach the desires results. **(N.B). Application must be restarted after each successful demonstration of and execution (Fig1.1)**

Shortest Job First (Fig1.2 and 1.3)

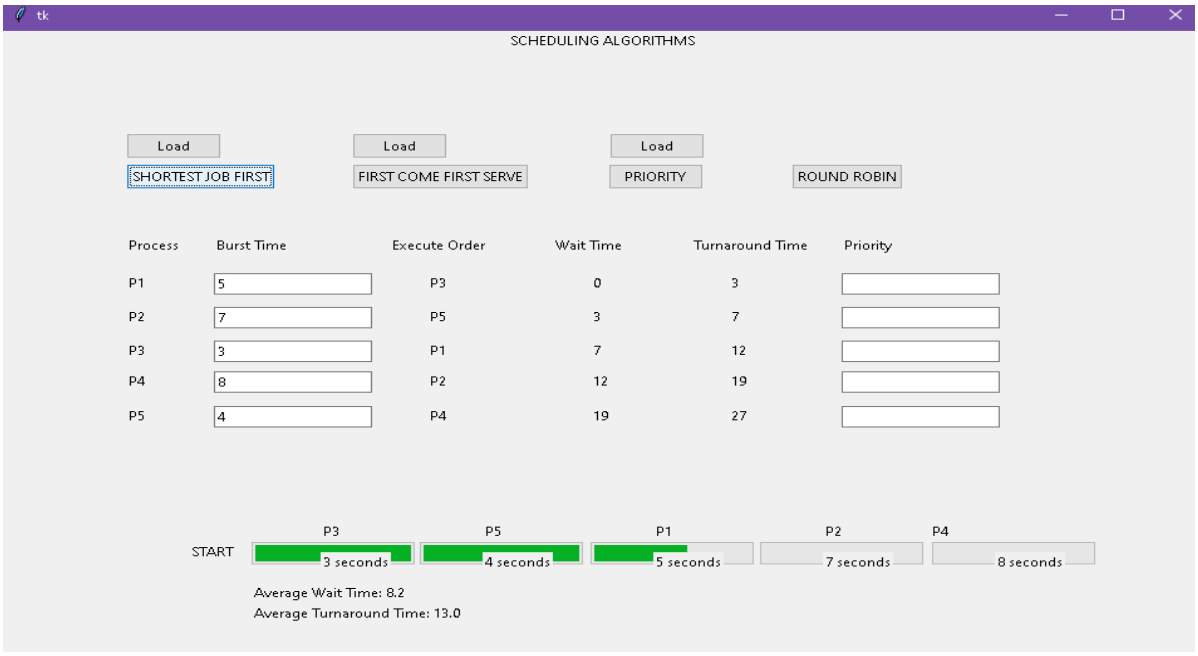
In Order to Display the Shortest Job First Algorithm

- Varying numbers are placed within each of the 5 entry boxes, under the ‘Burst Time’ heading to indicate the required burst times/ execution times of each waiting process.
- The “Load” button is clicked and values are then loaded into the application and the next execution is sorted by the shortest next burst time.
- Wait Time and turnaround time are also calculated, then displayed, as well as the average wait and turnaround time



(Fig1.2)

- The “Shortest Job First” button is clicked and then the execution of the algorithm and visual process is then displayed over a period of time after which execution is brought to an end.

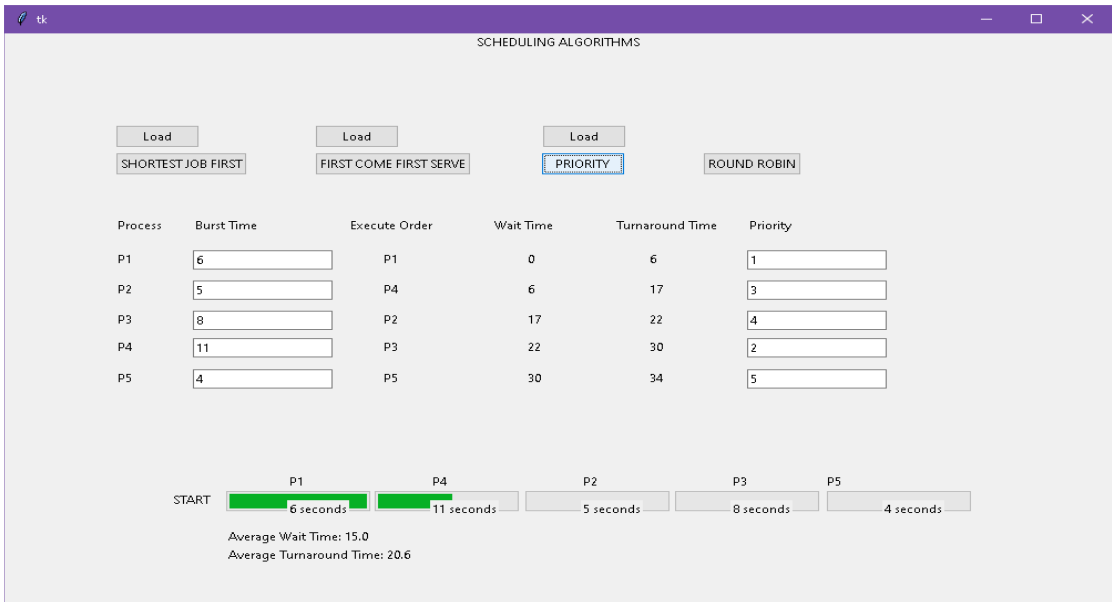


(Fig1.3)

Priority Scheduling Fig2.1

In Order to Visualize the Priority Scheduling Algorithm

- Varying numbers are placed within each of the 5 entry boxes, under the 'Burst Time' heading to indicate the required burst times/ execution times of each waiting process.
- The priority of Each Process required for execution is inputted by the user (The OS) ,indicating in what order the next to be executed process is to be ran, each executed one after the other. **(The lower the number the higher the priority)**
- The "Load" button is clicked and values are then loaded into the application and each process is and assigned to a process and is sorted by their individual priority
- Wait Time and turnaround time are also calculated, then displayed, as well as the average wait and turnaround time
- The "Priority" button is clicked and then the execution of the algorithm and visual process is then displayed over a period of time after which execution is brought to an end.

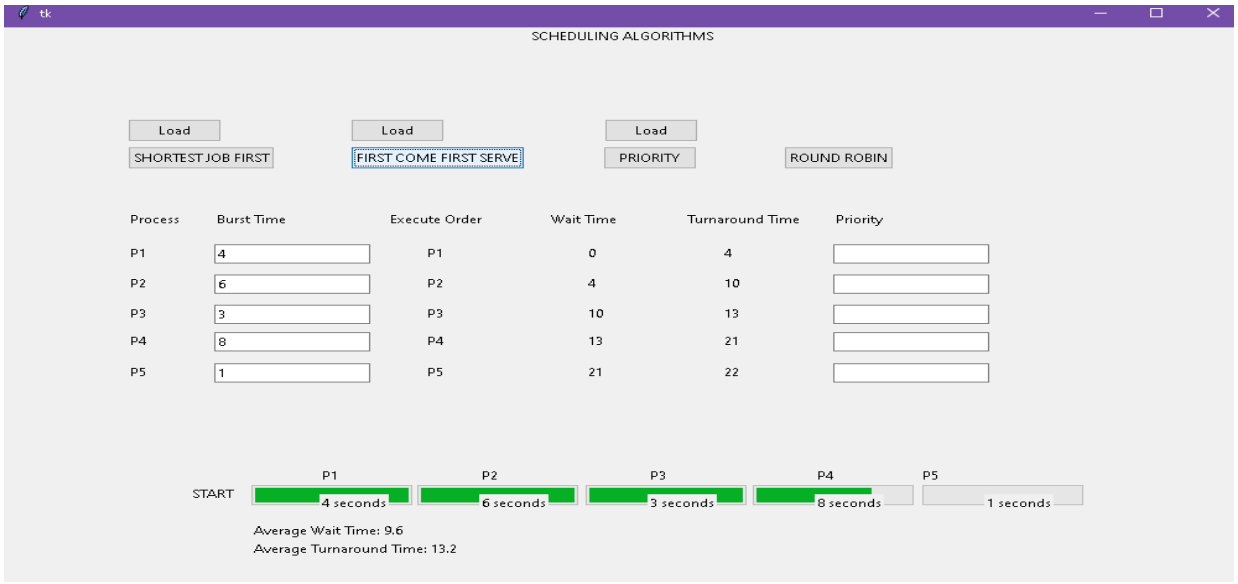


(Fig2.1)

First Come First Serve (Fig3.1)

In Order to Visualize the Priority Scheduling Algorithm

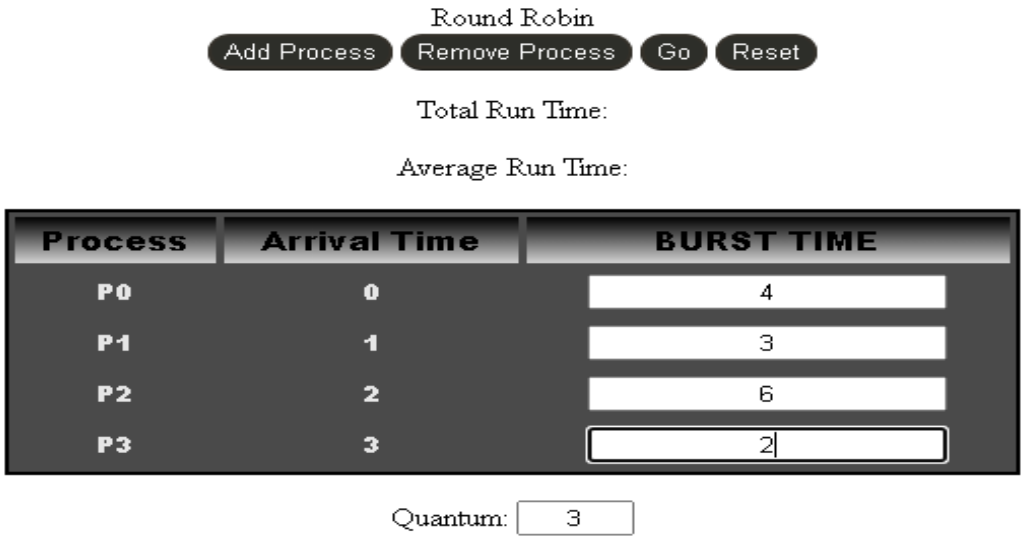
- Varying numbers are placed within each of the 5 entry boxes, under the 'Burst Time' heading to indicate the required burst times/ execution times of each waiting process.
- The "Load" button is clicked and values are then loaded into the application and stored and calculated based on the order in arrival in the processor **(from top to bottom order)**
- Wait Time and turnaround time are also calculated, then displayed, as well as the average wait and turnaround time
- The "First come First Serve" button is clicked and then the execution of the algorithm and visual process is then displayed over a period of time after which execution is brought to an end.



(Fig3.1)

Round Robin (Fig4.1)

The Round Robin scheduling Algorithm was done using a separate tech stack and tools, different from the previous 3, as a workaround to the issue of not being able to code it to the required specification using Python an Tkinter. It was visualized with HTML, CSS and jQuery and JavaScript commands and tools. Inspiration and resources were gathered from external resources and information in order to arrive at an ideal Solution

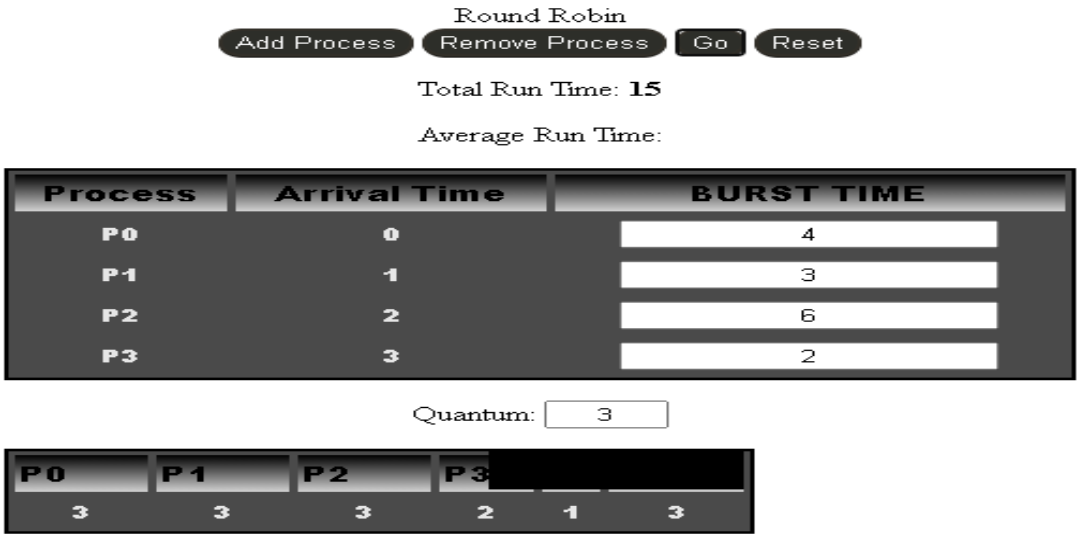


(Fig4.1)

Round Robin Scheduling (Fig4.2)

In Order to Visualize the Round Robin Scheduling Algorithm

- Varying numbers are placed within each of the entry boxes, under the 'Burst Time' heading to indicate the required burst times/ execution times of each waiting process.
- Unlike the previous implementation, processes can be added and removed just for extra demonstration purposes
- A quantum number is input which indicates how much processor time each process gets before switching to another process.
- It is Executed using the "Go' button which in linked to several functions which is responsible for the logic and calculations
- The processor is reset using the 'Reset' button and another demonstration can be performed after



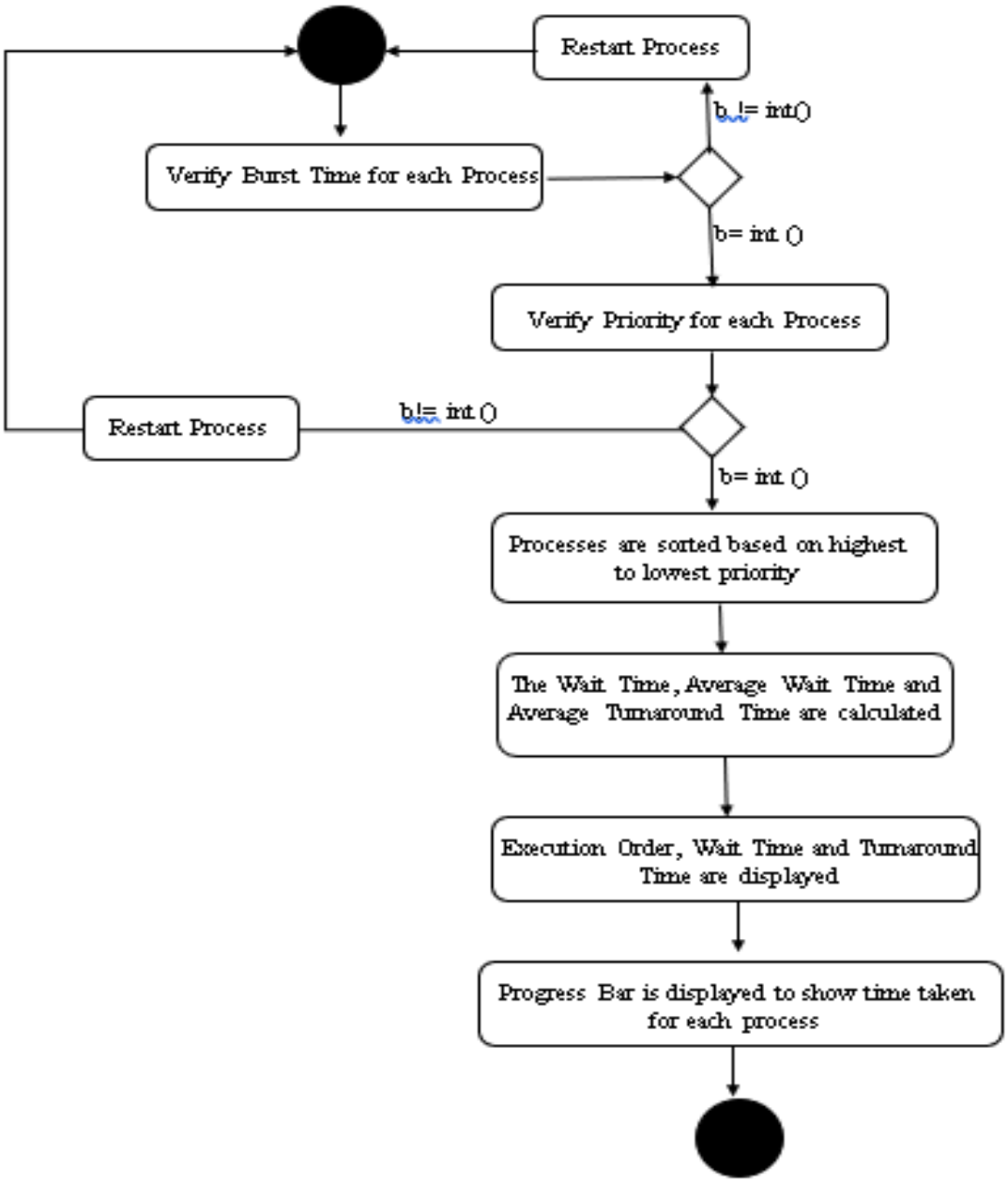
(Fig4.2)

Implementation

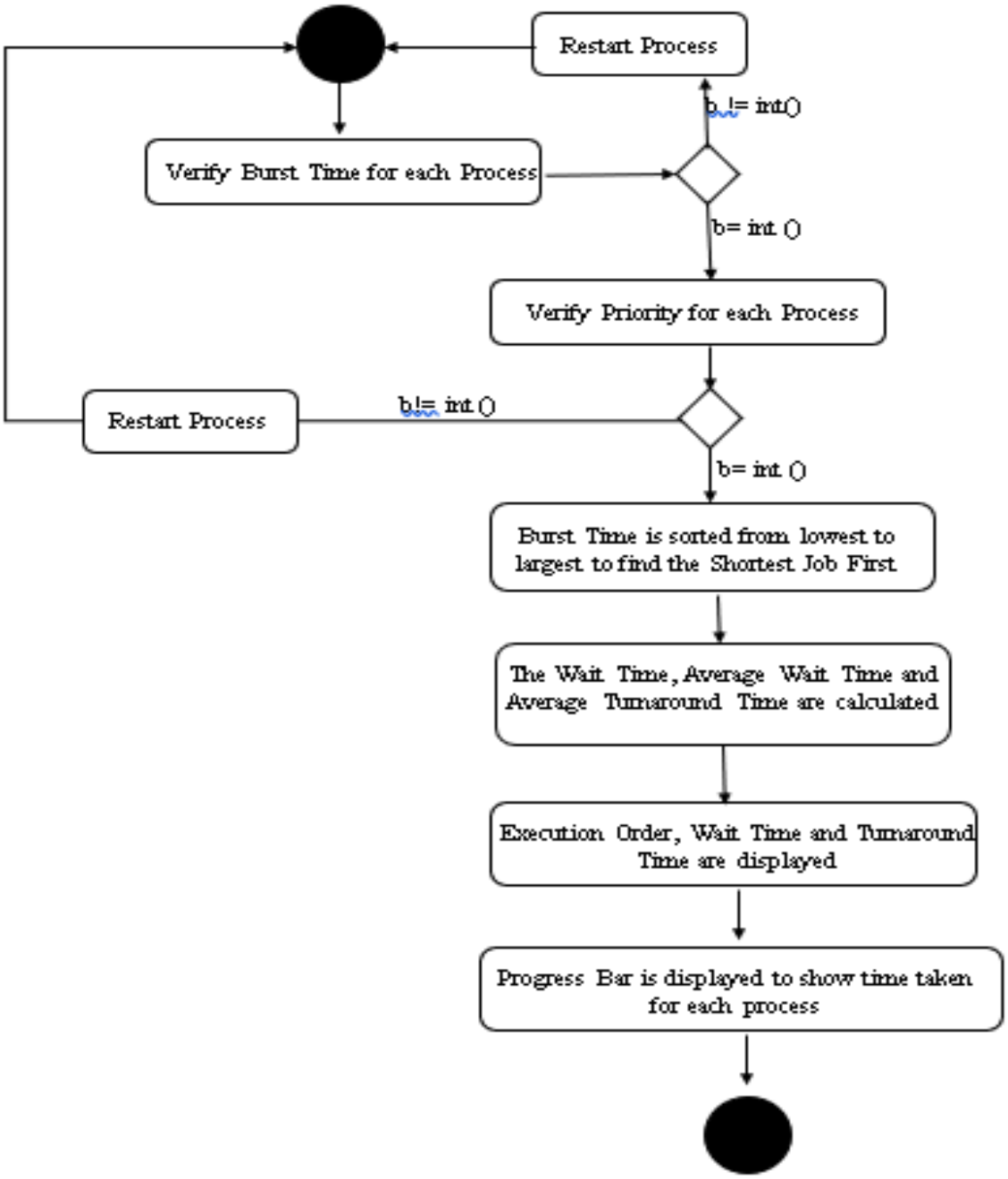
The implementations for the processing algorithm Visualizers for done in Python and Html/CSS

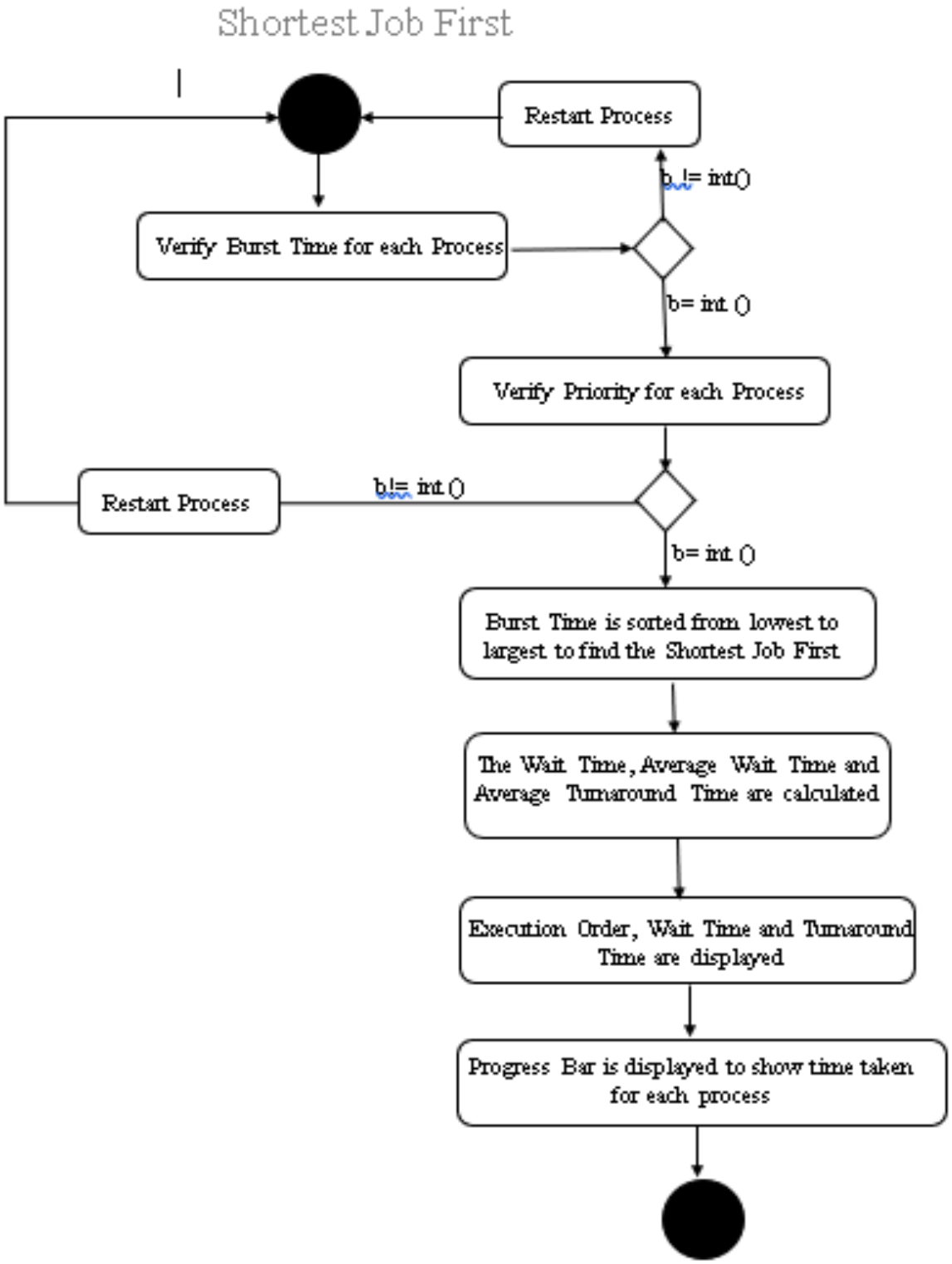
For Python, the Tkinter library was used for the representation of the GUI, in order to display and present values and the overall execution, it was also used as a front end for collecting the data to be processes by python functions and code. Similarly, HTML and CSS were also used, but for the Round Robin User interface instead, where JavaScript and jQuery tools and libraries were used to process the information on the back end

Round Robin Scheduling

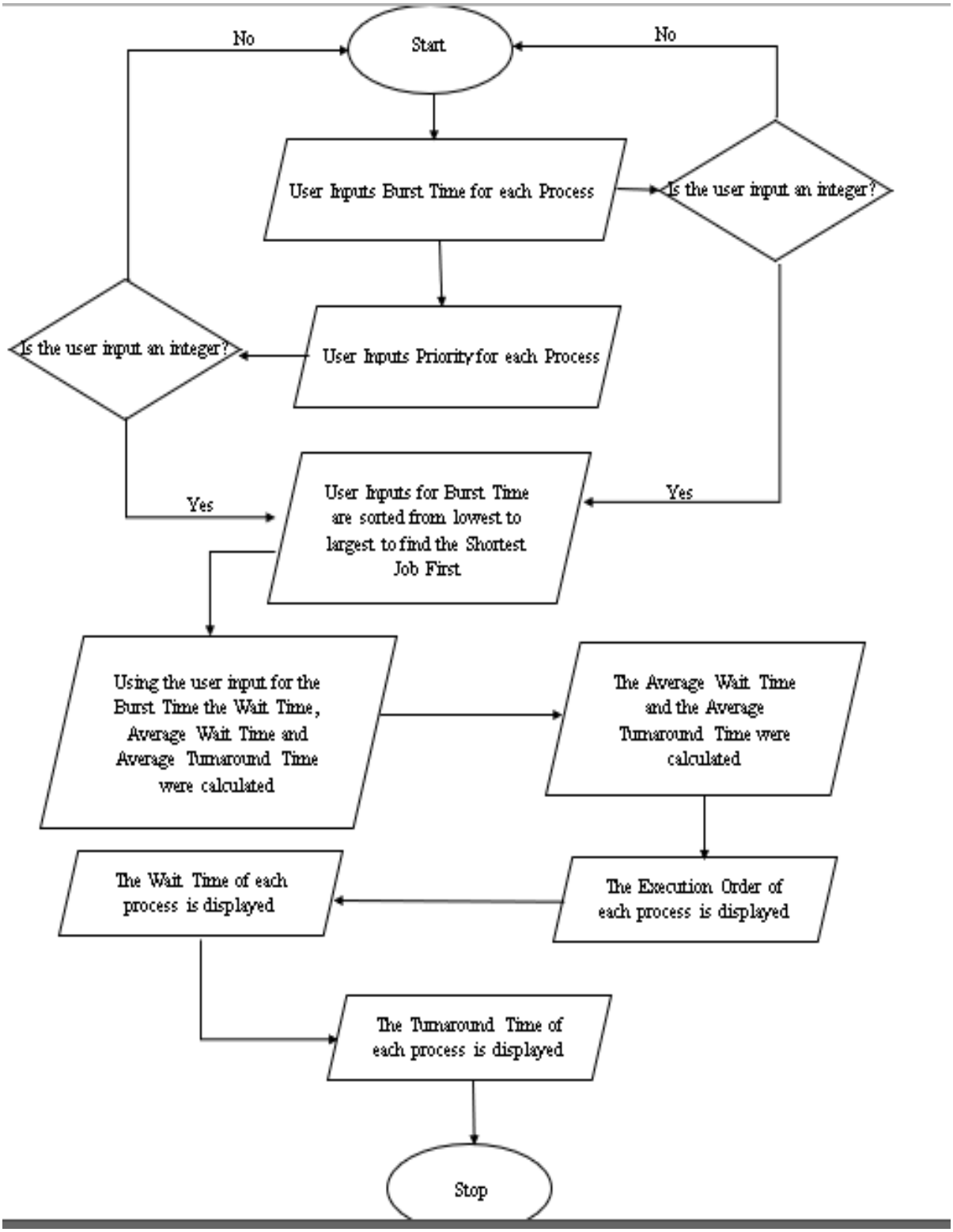


Priority Scheduling





Priority Schedule


Start
No
No
User Inputs Burst Time for each Process
Is the user input an integer?
Is the user input an integer?
User Inputs Priority for each Process
Yes
Yes
User Inputs for Burst Time are sorted from lowest to largest to find the Shortest Job First
Using the user input for the Burst Time the Wait Time, Average Wait Time and Average Turnaround Time were calculated
The Average Wait Time and the Average Turnaround Time were calculated
The Wait Time of each process is displayed
The Execution Order of each process is displayed
The Turnaround Time of each process is displayed
Stop