Project Report

On

PROCESS SCHEDULING ALGORITHMS

In the partial fulfillment of the term work for

Course: CSE323

(Operating System Design)

Submitted by

Hridy Chowdhury(1931527042)

Ayesha Rahman(1931209)

Sabrina Akter Sabina(1931947042)

Aporbo Ghosh(1931458042)

Under the Guidance of

Rashed Mazumder

Title: PROCESS SCHEDULING

Objective:

The project entitled "CPU SCHEDULING", is basically a program which simulates the following scheduling algorithms:

1. FCFS (First Come First Served)

2. SJF (Shortest Job First)

3. SJF(Preemptive)

4. Priority Scheduling

5. Round-Robin

Introduction:

The main purpose of this application is to compare and select the best algorithm for a particular set of input pertaining to the processes and in conclusion display the best algorithm. CPU SCHEDULING is a key concept in computer multitasking, multiprocessing operating system and real-time operating system designs. Scheduling refers to the way processes are assigned to run on the available CPUs, since there are typically many more processes running than there are available CPUs. CPU scheduling deals with the problem of deciding which of the processes in the ready queue is to be allocated the CPU. By switching the CPU among processes, the operating system can make the computer more productive. A multiprogramming operating system allows more than one processes to be loaded into the executable memory at a time and for the loaded processes to share the CPU using time-multiplexing.

System Requirements Specification:

HARDWARE:

PROCESSOR: ANY INTEL PROCESSOR ABOVE 4TH GENERATION

RAM : 512MB DD RAM

HARD DISK : 25 GB

SOFTWARE:

FORNT END: JAVA, IDE NETBEANS 8.2

OS : WINDOWS 10

Analysis:

To clarify the performance of various scheduling algorithms an assumption of 50 processes with different values(randomly generated by system which has been simulated) for their three factors burst time (0–50), arrival time (0–10) and priority order (0–10) is submitted to the system simulated and various scheduling parameters like turnaround time, throughput, response time, waiting time analysis has been discussed below.
Here in all the horizontal axis represents the total time taken for the process to execute and the vertical axis represent the process number

- *First come first serve (FCFS) algorithm:* In case of Figure 1 and Figure 2, it shows us the waiting time and turnaround time for the random 50 process taken by the compiler and implemented on the basis of first come first serve algorithm

- *Shortest job first (non-preemptive) algorithm:* In case of Figure 3 and Figure 4, it shows us the waiting time and turnaround time for the random 50 process taken by the compiler and implemented on the basis of Shortest Job First (non-preemptive) algorithm.

- *Round robin algorithm:* In case of Figure 7 and Figure 8, it shows us the waiting time and turnaround time for the random 50 process taken by the compiler and implemented on the basis of Round Robin algorithm

- *Priority scheduling:* Here in case of Figure 9 , it shows us the waiting time and turnaround time for the random 50 process taken by the compiler and implemented on the basis of Priority Scheduling algorithm

- *Shortest job first (preemptive) algorithm:* Similarly, In case of Figure 5 and Figure 6, it shows us the waiting time and turnaround time for the random 50 process taken by the compiler and implemented on the basis of Shortest Job First (preemptive) algorithm

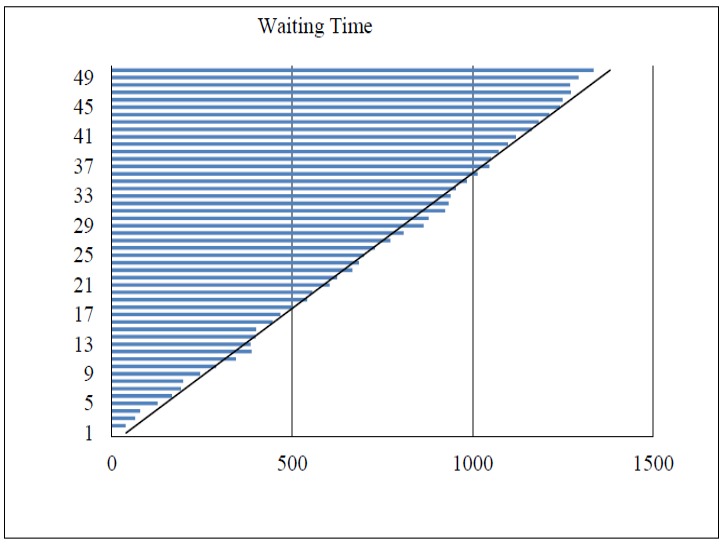


Fig. 1. *Average waiting time for FCFS*

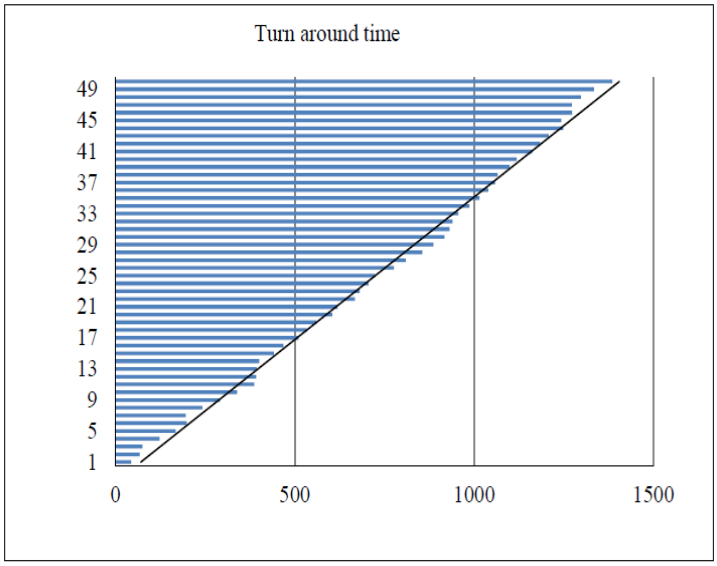


Fig. 2. *Average turnaround time for FCFS*

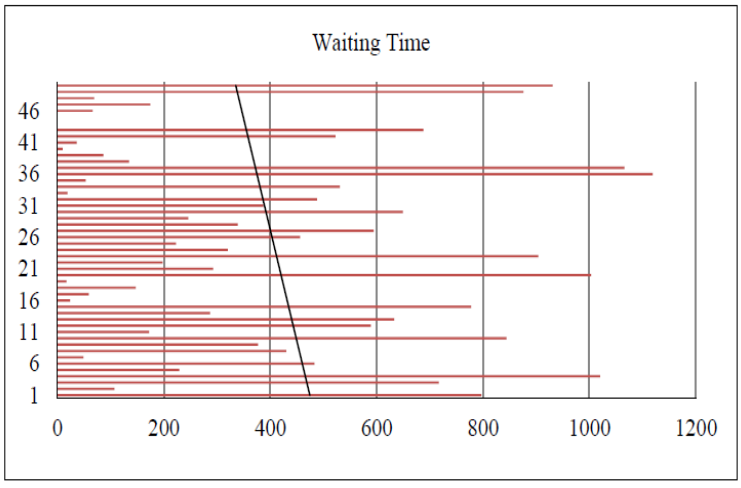


Fig.3. *Average waiting time for SJF*

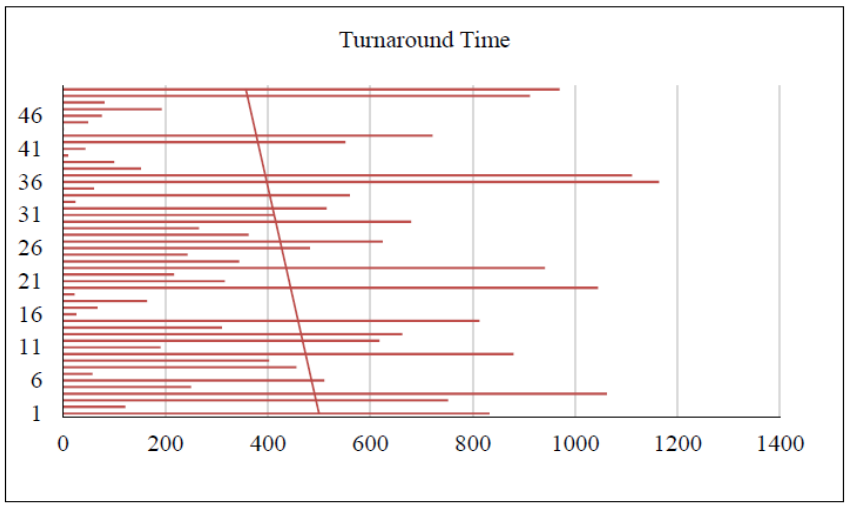


Fig.4. *Average turnaround time SJF*

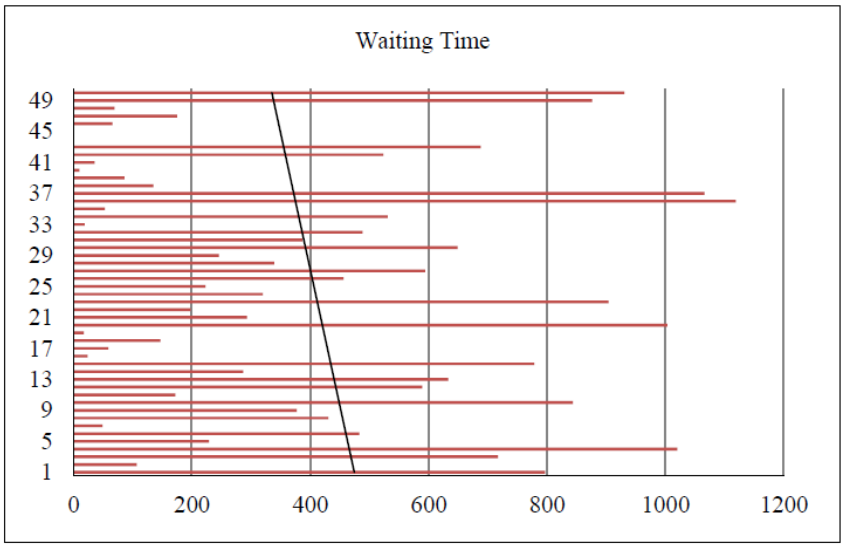


Fig.5. *Average waiting time SJF*

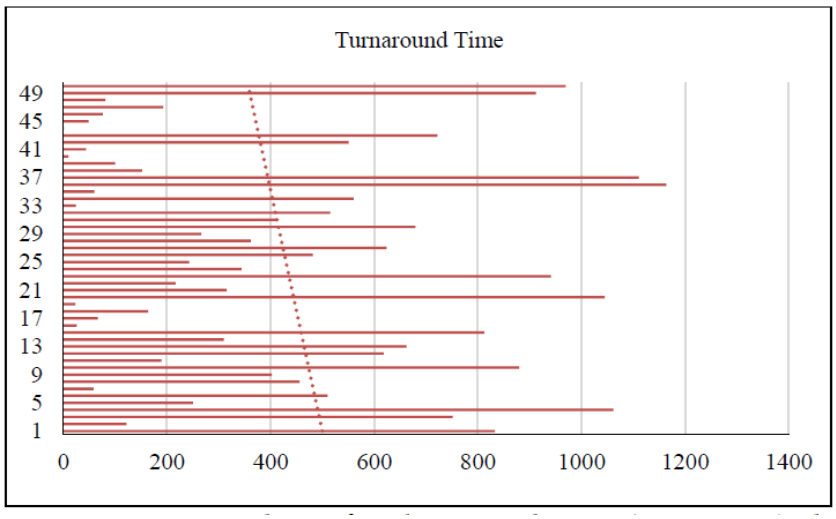


Fig.6. *Average turnaround time SJF*

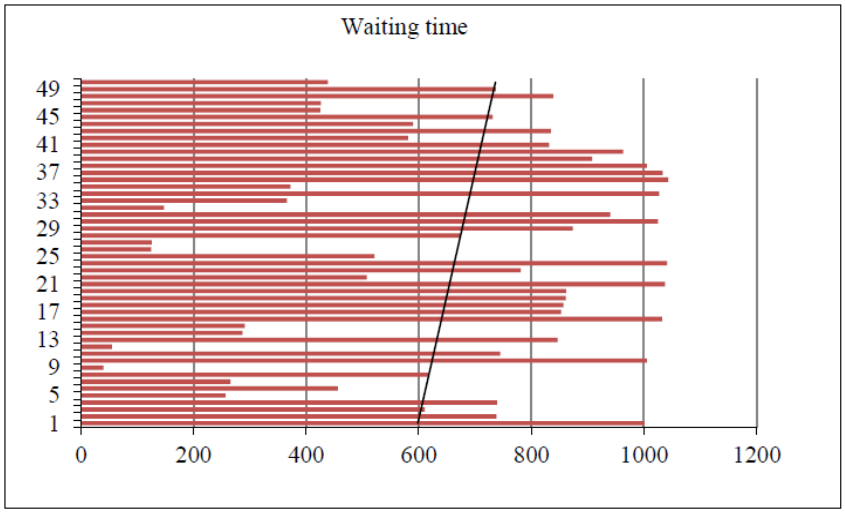


Fig.7. *Average waiting time RR*

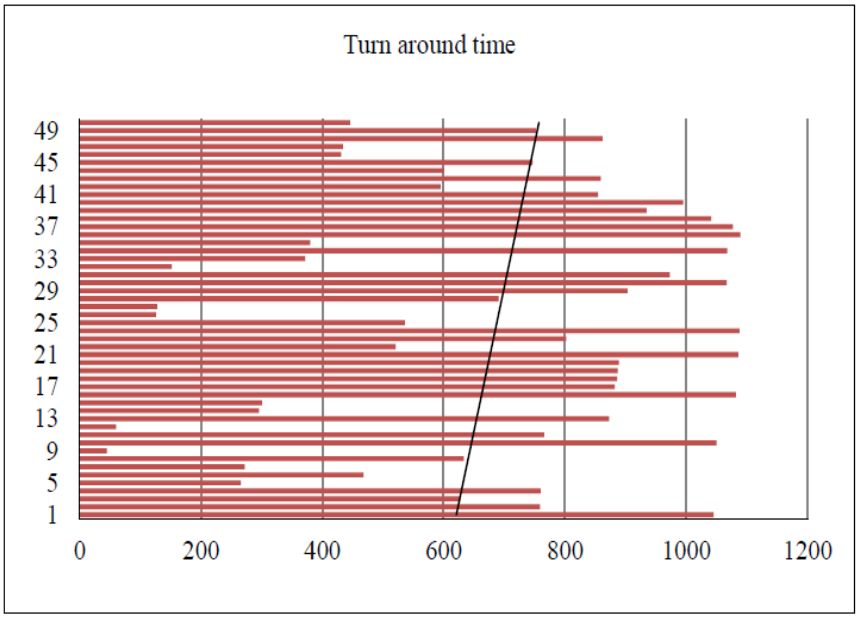


Fig.8. *Average turnaround time RR*

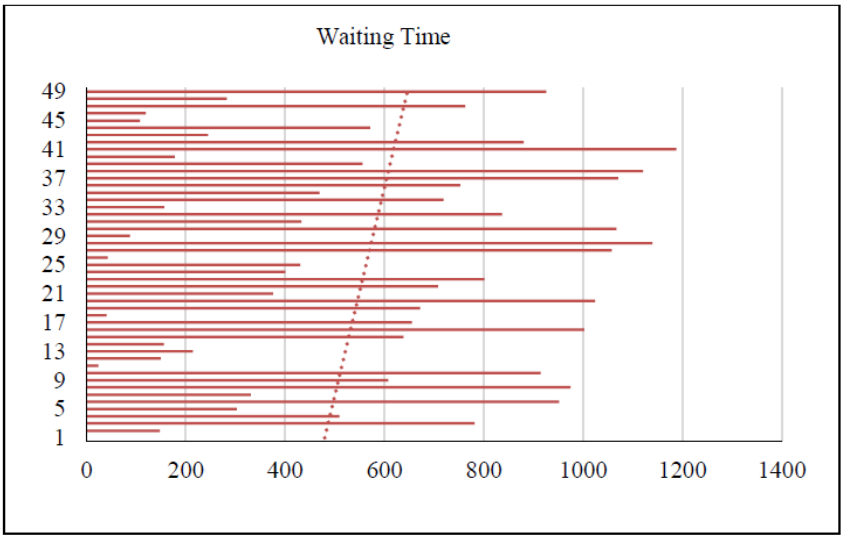


Fig.9. *Average waiting time priority*

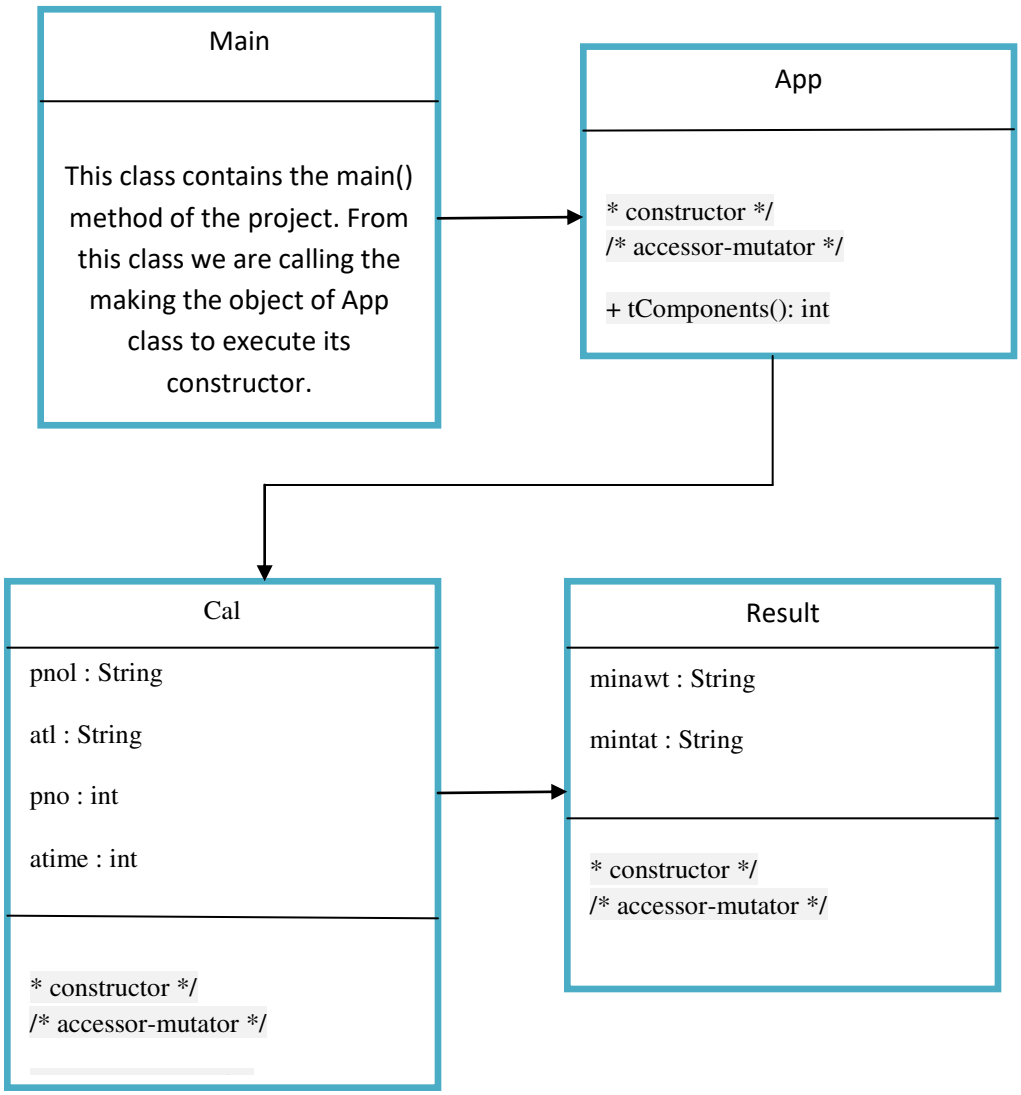
UML Diagram:

LIST OF CLASSES AND THEIR VARIOUS METHODS

PACKAGE: 1. com.algo

CLASSES:

1. Main
2. App
3. Cal
4. Result



Implementation:

The various list of methods and their implementation:

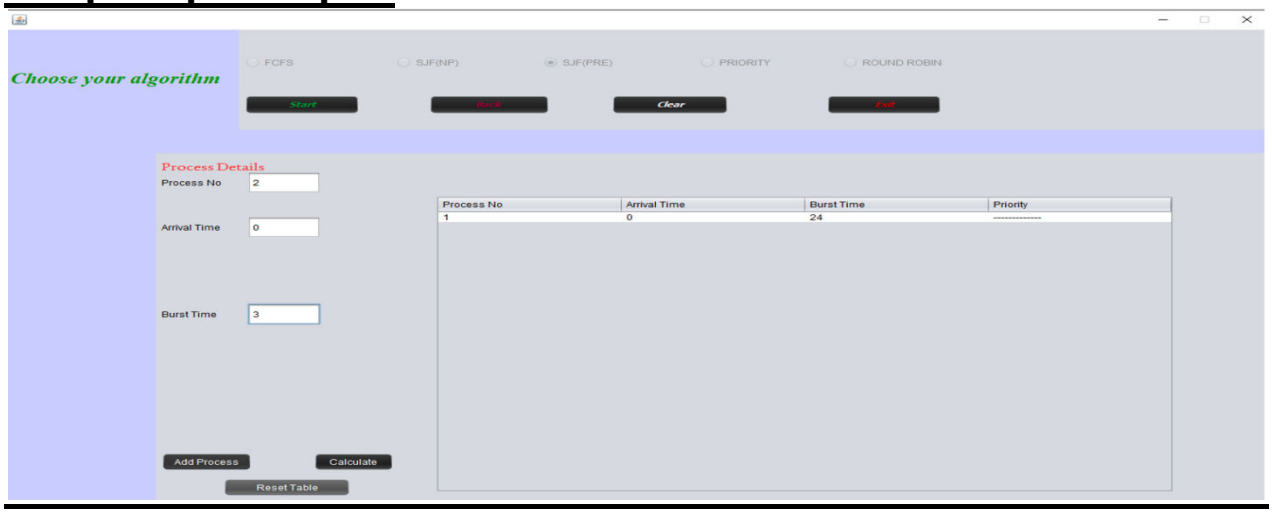
1) public static void main (String[] args) – It's the main method used to create the object of class App to execute its constructor.

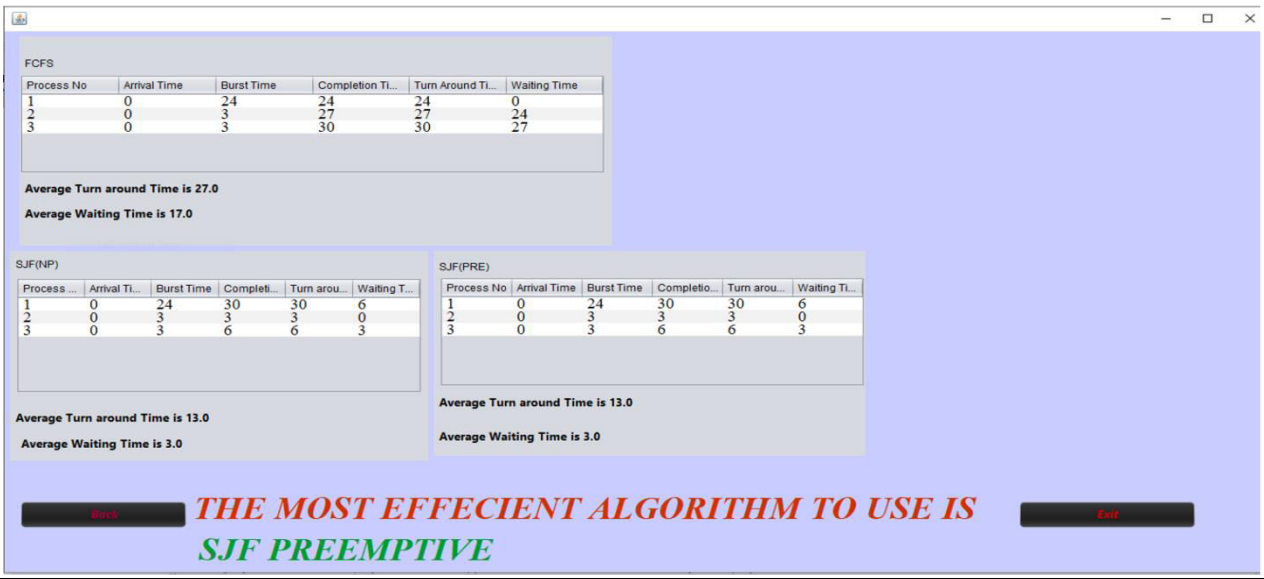
2) public App() – This method is used to display the Main Menu Screen and from here the user can choose whether they want proceed further or close the application. To proceed further an object of class Cal is created and its constructor is executed.

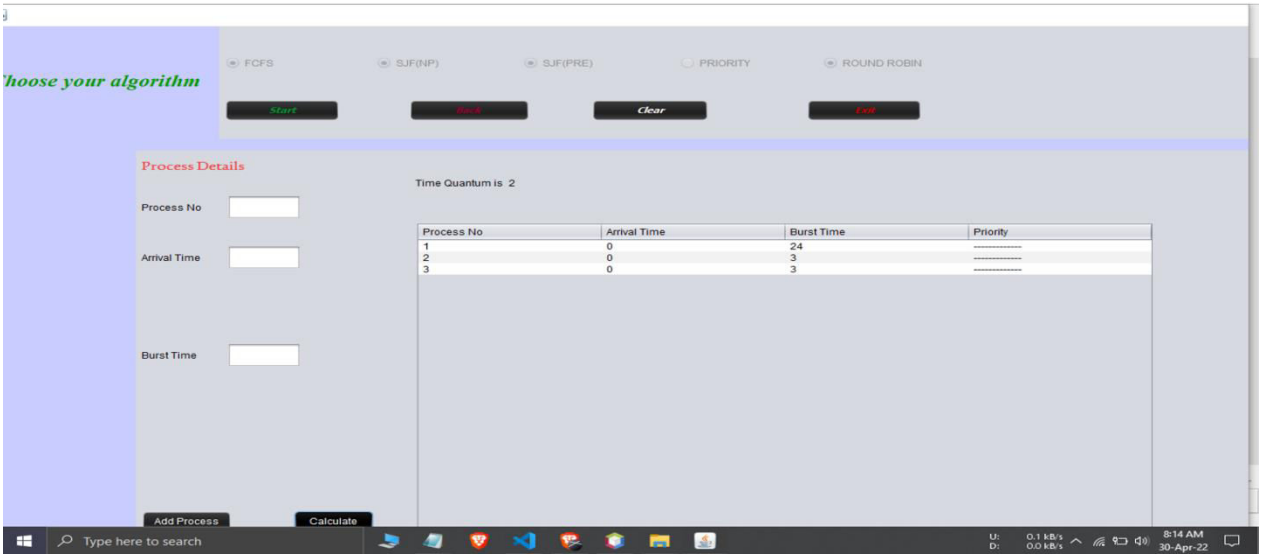
3) public Cal() – This method is used by the user for choosing the desired scheduling algorithm and inputting the various parameters. These parameters are stored in a table for the user to perceive and they can at any instance change the parameters or completely discard the previously entered parameters and start afresh. After the user is satisfied with the parameters they entered, they can click on Calculate Button to instantiate an object of class Result and proceed to the next form where the results are displayed.

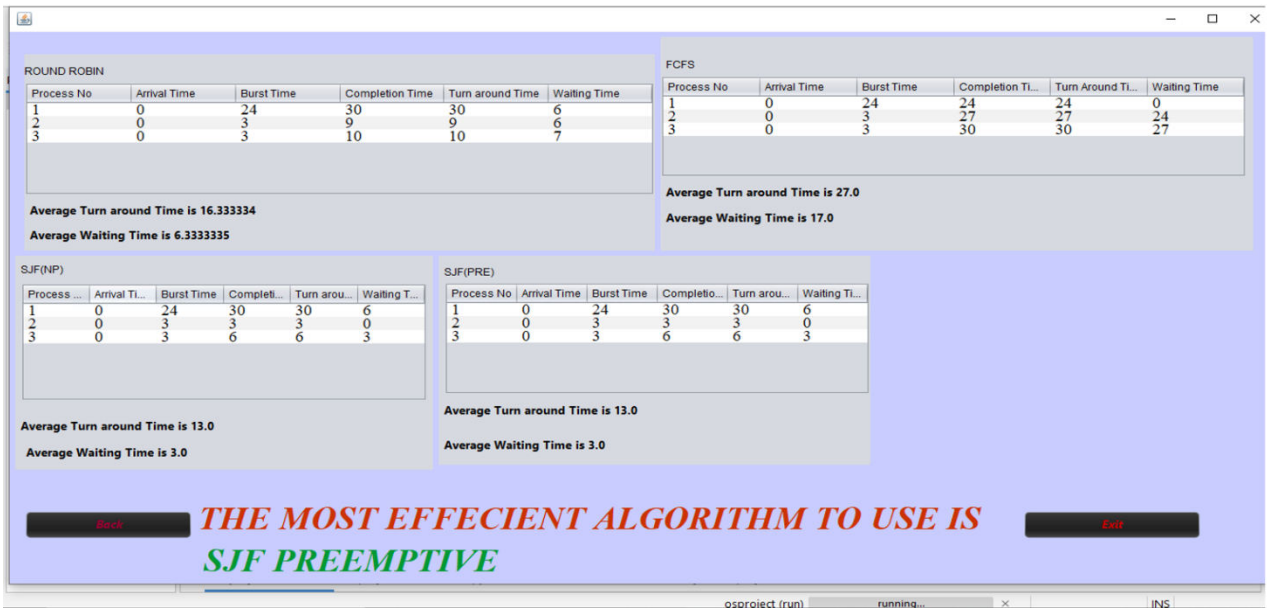
4) public Result() – This method is used by the application to display the Average Waiting Time and Average Turnaround Time for each of the algorithms the user previously selected. These parameters are displayed in a table, along with a message declaring which algorithm is the best to use. However due to combination of input parameters entered the Average Waiting Time and Average Turnaround Time for two or more algorithms can be same but only one algorithm will be displayed.

Sample input/output:




FCFS
Process No | Arrival Time | Burst Time | Completion Ti... | Turn Around Ti... | Waiting Time
1 | 0 | 24 | 24 | 24 | 0
2 | 0 | 3 | 27 | 27 | 24
3 | 0 | 3 | 30 | 30 | 27
Average Turn around Time is 27.0
Average Waiting Time is 17.0
SJF(NP)
Process ... | Arrival Ti... | Burst Time | Completi... | Turn arou... | Waiting T...
1 | 0 | 24 | 30 | 30 | 6
2 | 0 | 3 | 3 | 3 | 0
3 | 0 | 3 | 6 | 6 | 3
Average Turn around Time is 13.0
Average Waiting Time is 3.0
SJF(PRE)
Process No | Arrival Time | Burst Time | Completio... | Turn arou... | Waiting Ti...
1 | 0 | 24 | 30 | 30 | 6
2 | 0 | 3 | 3 | 3 | 0
3 | 0 | 3 | 6 | 6 | 3
Average Turn around Time is 13.0
Average Waiting Time is 3.0
Back
THE MOST EFFECIENT ALGORITHM TO USE IS
SJF PREEMPTIVE
Exit


hoose your algorithm
FCFS
SJF(NP)
SJF(PRE)
PRIORITY
ROUND ROBIN
Start
Back
Clear
Exit
Process Details
Time Quantum is 2
Process No
Arrival Time
Burst Time
Process No | Arrival Time | Burst Time | Priority
1 | 0 | 24 | ----------
2 | 0 | 3 | ----------
3 | 0 | 3 | ----------
Add Process
Calculate
Type here to search
8:14 AM
30-Apr-22



Conclusion:

As an output we get different average turn-around time and average waiting time from the calculations of several process scheduling algorithms. After plotting them on the graph with the help of Microsoft excel 2007 version one can easily interpret that waiting time escalate in more regular and even manner in case of first come first serve scheduling algorithm and so according to our graphs FCFS scheduling algorithm would be the best selection as compared to other scheduling algorithm for smaller burst time processes Out of all analyzed algorithms, SJF has the least average waiting time and turn- around time and so SJF is the best to process scheduling algorithm, and also if the process comes to processor simultaneously in case of SJF preemptive Scheduling. Round Robin is a comparatively good scheduling algorithm to alter the optimal average waiting time. Round Robin quantum time is set to more SJF algorithm or FCFS algorithm values. And priority Scheduling algorithm is based on the priority given to the process by the user or system and is suitable for applications with fluctuating time and resource requirements. Every algorithm is fine, but the processing speed depends on the processor load.

So all in all I can absolutely say that if I have to choose one algorithm between these algorithms in general, based on the experiments I did, I would definitely use the Shortest job first Algorithm as my first preference then I will go with priority scheduling then with round-robin scheduling and at last, I would opt for FCFS algorithm with keeping in mind all the pros and cons of the algorithms.