OS Lab Assignment III

Varun Khadayate

NMIMS

# Definition

First Come First Serve (FCFS) is an operating system scheduling algorithm that automatically executes queued requests and processes in order of their arrival. It is the easiest and simplest CPU scheduling algorithm. In this type of algorithm, processes which requests the CPU first get the CPU allocation first. This is managed with a FIFO queue. The full form of FCFS is First Come First Serve.

As the process enters the ready queue, its PCB (Process Control Block) is linked with the tail of the queue and, when the CPU becomes free, it should be assigned to the process at the beginning of the queue.

# Characteristics of FCFS method

* It supports non-pre-emptive and pre-emptive scheduling algorithm.
* Jobs are always executed on a first-come, first-serve basis.
* It is easy to implement and use.
* This method is poor in performance, and the general wait time is quite high.

# Example of FCFS scheduling

A real-life example of the FCFS method is buying a movie ticket on the ticket counter. In this scheduling algorithm, a person is served according to the queue manner. The person who arrives first in the queue first buys the ticket and then the next one. This will continue until the last person in the queue purchases the ticket. Using this algorithm, the CPU process works in a similar manner.

# Implementation:

1. Input the processes along with their burst time (bt).
2. Find waiting time (wt) for all processes.
3. As first process that comes need not to wait so waiting time for process 1 will be 0 i.e. wt[0] = 0.
4. Find waiting time for all other processes i.e. for process i -> wt[i] = bt[i-1] + wt[i-1] .
5. Find turnaround time = waiting\_time + burst\_time for all processes.
6. Find average waiting time = total\_waiting\_time / no\_of\_processes.
7. Similarly, find average turnaround time = total\_turn\_around\_time / no\_of\_processes.

# CODE

#include<iostream>

#include<stdio.h>

using namespace std;

int main(){

int bt[10]={0},at[10]={0},tat[10]={0},wt[10]={0},ct[10]={0};

int n,sum=0;

float tottat=0,totwt=0;

printf("Enter number of processes: ");

scanf("%d",&n);

printf("Enter arrival time and burst time for each process\n\n");

for(int i=0;i<n;i++)

{

printf("Arrival time of process[%d]::",i);

scanf("%d",&at[i]);

printf("Burst time of process[%d]::",i);

scanf("%d",&bt[i]);

printf("\n");

}

for(int j=0;j<n;j++)

{

sum+=bt[j];

ct[j]+=sum;

}

for(int k=0;k<n;k++)

{

tat[k]=ct[k]-at[k];

tottat+=tat[k];

}

for(int k=0;k<n;k++)

{

wt[k]=tat[k]-bt[k];

totwt+=wt[k];

}

printf("\n\n");

printf("P#\t AT\t BT\t CT\t TAT\t WT\t\n\n");

for(int i=0;i<n;i++)

{

printf("P%d\t %d\t %d\t %d\t %d\t %d\n",i,at[i],bt[i],ct[i],tat[i],wt[i]);

}

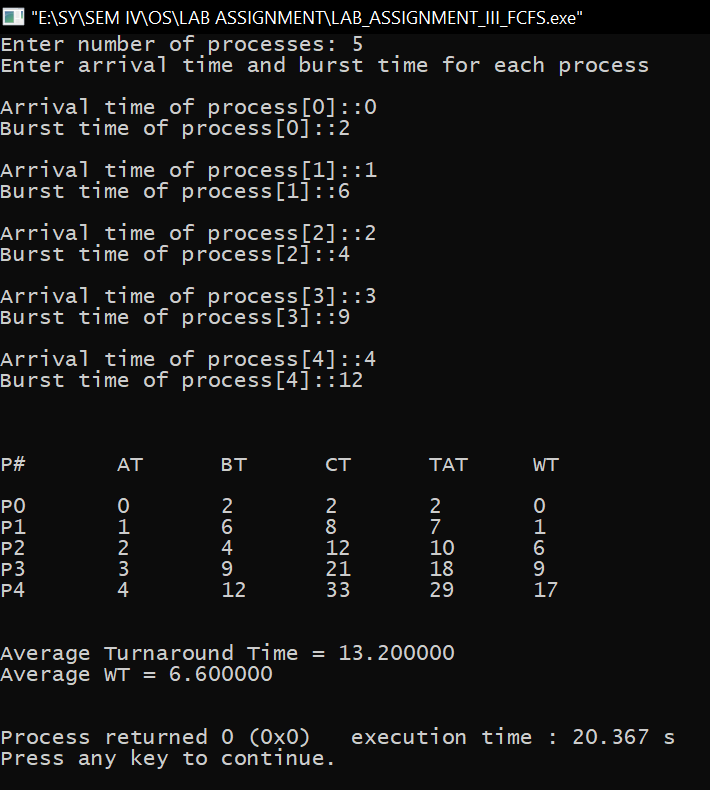
printf("\n\nAverage Turnaround Time = %f\n",tottat/n);

printf("Average WT = %f\n\n",totwt/n);

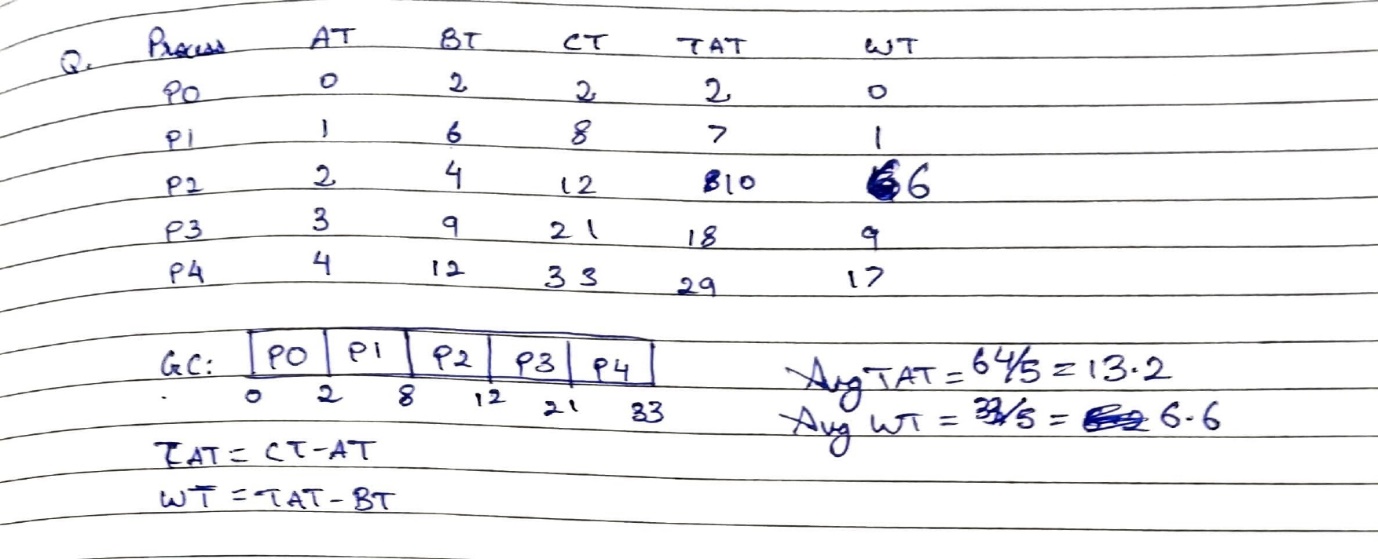
return 0;

}

# OUTPUT



# Via Formula and Steps check



# Advantages

* The simplest form of a CPU scheduling algorithm
* Easy to program
* First come first served

# Disadvantages of FCFS

* It is a Non-Pre-emptive CPU scheduling algorithm, so after the process has been allocated to the CPU, it will never release the CPU until it finishes executing.
* The Average Waiting Time is high.
* Short processes that are at the back of the queue have to wait for the long process at the front to finish.
* Not an ideal technique for time-sharing systems.
* Because of its simplicity, FCFS is not very efficient.