**Lab Manual for Operating System**

**Lab No. 7**

# **First Come First Serve & Shortest Job First Scheduling Algorithm**

Objectives

To understand basic concept, working and usage of scheduling algorithm.

**LAB # 07**

**FCFS & SJF Scheduling Algorithm**

## **Short-Term Scheduling**

Short-term scheduling, or dispatching, involves scheduling processes to execute on a processor. Whenever the CPU becomes idle, the operating system must select one of the processes in the ready queue to be executed. The selection process is carried out by the short-term scheduler. It selects a process from the processes in memory that are ready to execute and allocates the CPU to that process.

There are various algorithms available for scheduling processes. Some of these are discussed in this manual while the rest are left for the students as tasks.

## **First Come First Serve (FCFS)**

FCFS is a relatively simple algorithm, which treats the processes strictly in order of arrival. Processor is allocated to the process which arrives first. Once the processor is allocated to a process, it is released when that process finishes. When the processor is available, the scheduler selects the next process to be executed.

The codes provided in Example#01 and Example#02 are implementations of the FCFS algorithms. In the code provided in Example#01, it is assumed that all processes are present in the system at the same time; that is, all processes have arrived in the system. However, the processes usually arrive at different times. In order to accommodate processes arriving at different times, the code in Example#01 can be modified as given in Example#02.

* **Arrival Time**: The time at which process enter into ready queue.
* **Burst time**: Time duration taken by the CPU to get executed.
* **Completion Time**: Time at which process completes its execution.
* **Turn Around Time**: *Turn Around Time* ***= Completion Time – Arrival Time***
* **Waiting Time** = Turn Around Time – Burst Time

Table

Description automatically generated with medium confidence

**Example#01:**

#include<stdio.h>

void main (void)

{

int st[20], wt[20], tat[20], i, n;

float wt\_avg, tat\_avg;

printf("\nEnter the number of processes: ");

scanf("%d", &n);

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

{

printf("\nEnter Burst/Service time for process%d: ", i);

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

}

wt[0] = wt\_avg = 0;

tat[0] = tat\_avg = st[0];

for (i = 1; i < n; i++)

{

wt[i] = wt[i-1] + st[i-1];

tat[i] = tat[i-1] + st[i];

wt\_avg = wt\_avg + wt[i];

tat\_avg = tat\_avg + tat[i];

}

printf("\n PROCESS \t SERVICE TIME \t WAITING TIME \t TURNAROUND TIME\n");

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

{

printf("\nP%d \t\t%d \t\t%d \t\t%d\n", i, st[i], wt[i], tat[i]);

}

printf("\nAverage waiting time:%f ", wt\_avg/n);

printf("\nAverage turnaround time:%f \n", tat\_avg/n);

}

![Text

Description automatically generated]()

**Example#02:**

#include<stdio.h>

void main (void)

{

int at[20], st[20], wt[20], ft[20], tat[20], i, n;

float wt\_avg, tat\_avg;

printf("\nEnter the number of processes: ");

scanf("%d", &n);

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

{

printf("\nEnter the Arrival time for process%d: ", i);

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

printf("\nEnter Burst/Service time for process%d: ", i);

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

}

wt[0] = wt\_avg = 0;

tat[0] = tat\_avg = st[0];

ft[0] = st[0];

for (i = 1; i < n; i++)

{

if (at[i] <= at[i+1])

{

wt[i] = wt[i-1] + st[i-1];

ft[i] = wt[i] + st[i];

tat[i] = ft[i] - at[i];

wt\_avg = wt\_avg + wt[i];

tat\_avg = tat\_avg + tat[i];

}

}

printf("\n PROCESS \t SERVICE TIME \t WAITING TIME \t FINISH TIME \t TURNAROUND TIME\n");

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

{

printf("\nP%d \t\t%d \t\t%d \t\t%d \t\t%d\n", i, st[i], wt[i], ft[i], tat[i]);

}

printf("\nAverage waiting time:%f ", wt\_avg/n);

printf("\nAverage turnaround time:%f \n", tat\_avg/n);

}

![Text

Description automatically generated]()

## **Shortest job first**

Shortest job first (SJF) or shortest job next, is a scheduling policy that selects the waiting process with the smallest execution time to execute next. SJN, also known as Shortest Job Next (SJN), is a non-preemptive algorithm.

* Sort all the process according to the arrival time.
* Then select that process which has minimum arrival time and minimum Burst time.
* After completion of process make a pool of process which after till the completion of previous process and select that process among the pool which is having minimum Burst time.

Diagram

Description automatically generated with medium confidence

## **Lab Tasks**

1. The FCFS source code given above assumes that the arrival times for processes are provided in ascending order of time. Modify the code in C language for random process arrival times.
2. Modify the FCFS code of example#02 to be able to calculate the mean of Normalized Turnaround Time. Also the modified code should display the Normalized Turnaround Time of each process.
3. Implement the Shortest Job First algorithm with the help of 4 non-preemptive processes.