# **Experiment No: 3**

## AIM: Implementation of FCFS and SJF CPU scheduling algorithms.

**Objective:** To study and implement CPU scheduling algorithms.

#### Theory:

**FCFS Scheduling Algorithm:**

For FCFS scheduling algorithm, read the number of processes/jobs in the system, their CPU burst times. The scheduling is performed on the basis of arrival time of the processes irrespective of their other parameters. Each process will be executed according to its arrival time. Calculate the waiting time and turnaround time of each of the processes accordingly.

#### Program:

* 1. Start.
  2. Declare the array size.
  3. Read the number of processes to be inserted
  4. Read the arrival time of processes.
  5. Read the burst time of processes.
  6. Calculate the complition time of each process
  7. Calculate the waiting time of each process: wt[i]=tat[i]-bt[i]
  8. Calculate the turnaround time of each process: tat[i]=ct[i]-at[i]
  9. Calculate the average waiting and average turnaround time
  10. Display the values.
  11. Stop

**Source Code:**

#include<stdio.h>

int main(){

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

int n,sum=0,i,j, temp=0;

float totalTAT=0,totalWT=0;

printf("Enter number of processes ");

scanf("%d",&n);

printf("enter %d process:",n);

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

{

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

}

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+1);

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

printf("Burst time of process[%d] ",i+1);

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

printf("\n");

}

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

{

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

{

if(at[i]<at[j])

{

temp=at[i];

at[i]=at[j];

at[j]=temp;

temp=bt[i];

bt[i]=bt[j];

bt[j]=temp;

temp=p[j+1];

p[j+1]=p[j];

p[j]=temp;

}

}

}

//calculate completion time of processes

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

{

sum+=bt[j];

ct[j]+=sum;

}

//calculate turnaround time and waiting times

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

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

totalTAT+=tat[k];

}

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

{

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

totalWT+=wt[k];

}

printf("Solution: \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",p[i],at[i],bt[i],ct[i],tat[i],wt[i]);

}

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

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

return 0;}

**Output:**

#### SJF Scheduling algorithm:

For SJF scheduling algorithm, read the number of processes/jobs in the system, their CPU burst times. Arrange all the jobs in order with respect to their burst times. There may be two jobs in queue with the same execution tine, and then FCFS approach is to be performed. Each process will be executed according to the length of its burst time. Then calculate the waiting time and turnaround time of each of the processes accordingly.

Program:

1. Start
2. Declare the array size
3. Read the number of processes to be inserted
4. Read the Burst times of processes



1. Sort the Burst times in ascending order and process with shortest burst time
2. Calculate the waiting time of each process: wt[i] = wt[i-1] +bt[i-1]
3. Calculate the turnaround time of each process: tat[i] = tat[i-1] +bt[i]
4. Calculate the average waiting time and average turnaround time.
5. Display the values
6. Stop

**Source Code:**

#include<stdio.h>

void main()

{

int p[20], bt[20], wt[20], tat[20], i, k, n, temp; float wtavg, tatavg;

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

scanf("%d", &n);

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

{

p[i]=i;

printf("Enter Burst Time for Process %d -- ", i);

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

}

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

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

if(bt[i]>bt[k])

{

temp=bt[i];

bt[i]=bt[k];

bt[k]=temp;

temp=p[i];

p[i]=p[k];

p[k]=temp;

}

wt[0] = wtavg = 0;

tat[0] = tatavg = bt[0]; for(i=1;i<n;i++)

{

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

tat[i] = tat[i-1] +bt[i]; wtavg = wtavg + wt[i];

tatavg = tatavg + tat[i];

}

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

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

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

printf("\nAverage Waiting Time -- %f", wtavg/n);

printf("\nAverage Turnaround Time -- %f", tatavg/n);

}

**Output:**