First Come First Serve (FCFS)

Experiment No : 3 Date :-

Aim :- To implement First Come First Serve(FCFS) CPU Scheduling Algorithm

Theory :

The **First Come First Served (FCFS)** algorithm, also known as First In First Out (FIFO), is a straightforward non-preemptive scheduling technique. It prioritizes processes based on their request order for CPU time. The earliest arriving process is allocated the CPU initially, following a sequential approach using a queue. Processes are added to the end of the queue upon arrival and removed from the front once their execution is complete.

For a given set of n processes with their respective burst times, the goal is to determine the average waiting time and average turnaround time using the FCFS algorithm. In this method, tasks are scheduled based on their arrival time, with the first arrival gaining CPU access first. It's important to note that this description assumes all processes have an arrival time of 0.

Nonetheless, FCFS scheduling may introduce the issue of starvation if the initial process has the longest burst time among all tasks. Efficient process scheduling is crucial to ensure work completion within deadlines. Processes involve both I/O and CPU time, and optimizing scheduling helps maximize CPU utilization, allocate resources fairly, and enhance throughput.

In the context of process scheduling, several terms hold significance:

Arrival Time: The moment a process enters the ready queue.

Completion Time: When a process finishes its execution.

Burst Time: The time needed for a process to execute on the CPU.

Turnaround Time: The interval between completion and arrival times: Turnaround Time = Completion Time – Arrival Time.

Waiting Time (W.T): The gap between turnaround and burst times: Waiting Time = Turnaround Time – Burst Time.

Example :

Processes	Burst Time
P1	**5**
P2	**3**
P3	**4**
P4	**2**

Calculating Waiting Time and Turnaround Time:

Waiting Time= Burst time of previous process + Waiting Time of previous process

TurnAround time = Waiting time+Burst time (current process)

- For P1,

Waiting time(wt[1])= 0

=0+5

=5

Turnaround Time = 5 + 0

=0

- For P2,

Waiting Time(wt[2]) =Burst time(P1)+Waiting Time(wt[1])

=5+0

=5

Turnaround Time = 3 + 5

=8

- For P3,

Waiting Time(wt[3]) =Burst time(P2)+Waiting Time(wt[2])

=3+5

=8

Turnaround Time = 4 + 8

=12

- For P4,

Waiting Time(wt[4]) =Burst time(P3)+Waiting Time(wt[3])

=4+8

=12

Turnaround Time = 2+ 12

=14

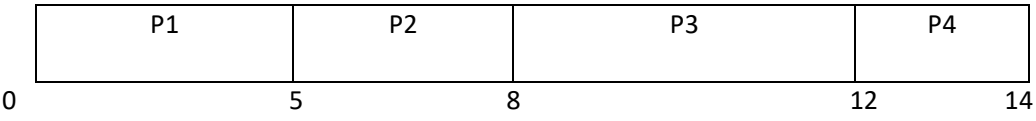
Processes	Burst Time	Waiting Time	Turnaround Time
1	5	0	5
2	3	5	8
3	4	8	12
4	2	12	14

Average Burst Time=3.5 seconds

Average Waiting Time=6.25

Average Turnaround Time=9.75

Gantt Chart



Code

```
#include<bits/stdc++.h>
using namespace std;
void input_burst_time(int n ,int burst_time[]){
   cout<<"Enter Burst Time"<<endl;
   for (int i = 0; i < n; i++){
      cin>>burst_time[i];
   }}
void calculate_waiting(int n ,int wt_time[],int
burst_time[]){
   for (int i = 1; i < n; i++){
      wt_time[i]= burst_time[i-1] + wt_time[i-
1];
   }
   cout<<endl;
}
void calulate_turn_around(int n , int
wt_time[],int burst_time[],int turn_around[]){
   for (int i = 0; i < n; i++){
      turn_around[i] = wt_time[i] +
burst_time[i];
   }
   cout<<endl;
}
void display(int n,int wt_time[],int
turn_around[]){
   cout<<"Waiting time of processes"<<endl;
   for (int i = 0; i < n; i++){
      cout<<wt_time[i]<<"  ";
   }
    cout<<"\nTurn around time of
processes"<<endl;
     for (int i = 0; i < n; i++){
        cout<<turn_around[i]<<"  ";
   }
   cout<<"\n";
}
void gchart(int n ,int burst_time[]){
   cout<<"\nGantt Chart:"<<endl;
   int current_time = 0;
   int size=3;
   for (int i = 0; i < n*7; i++){
      cout<<'-';
   }
   cout<<endl;
   cout<<"|  ";
   for (int i = 0; i < n; i++){
      cout<<'p'<<i+1<<"  |  ";
   }
   cout<<"\n";
for (int i = 0; i < n*7; i++){
      cout<<'-';
   }
   cout<<"\n";
    for (int i = 0; i < n; i++){
      cout<<current_time<<"      ";
      current_time+= burst_time[i];

      if(i == n-1){
         cout<<current_time;
      } }
   cout<<"\n";
}
```

```
void avg_tat(int n,int turn_around[],int wt_time[]){
   float avg_sum =0.0;
   float avg_wt = 0.0;
   for (int i = 0; i < n; i++)  {
        avg_sum += turn_around[i];
        avg_wt +=  wt_time[i];
   }
   cout<< "\nAverage waiting time is "<<(avg_wt/n)<<endl;
   cout<< "\nAverage Turn around time is "<<(avg_sum/n)<<endl;
}
int main(){
   int n;
   cout<<"Enter no of processes : "<<endl;
   cin>>n;
   int burst_time[n]={0};
   input_burst_time(n,burst_time);
   int wt_time[n]={0};
   calculate_waiting(n,wt_time,burst_time);
   int turn_around[n]={0};

calulate_turn_around(n,wt_time,burst_time,turn_around);
   display(n,wt_time,turn_around);
   cout<<"\n";
   avg_tat(n,turn_around,wt_time);
   cout<<"\n";
   gchart(n,burst_time);
}
```

Output :

```
diggaj@diggaj-ThinkPad-L420:~/Desktop/Shell Programs/Diggaj/Expt3$ g++ fcfs.cpp
diggaj@diggaj-ThinkPad-L420:~/Desktop/Shell Programs/Diggaj/Expt3$ ./a.out
Enter no of processes :
4
Enter Burst Time
5
3
4
2


Waiting time of processes
0  5  8  12
Turn around time of processes
5  8  12  14


Average waiting time is 6.25

Average Turn around time is 9.75


Gantt Chart:
----------------------------
|  p1  |  p2  |  p3  |  p4  |
----------------------------
0      5      8      12      14
```

Conclusion : First Come First Serve (FCFS) CPU Scheduling Algorithm was successfully studied and implemented.