Lab-3

(8) write a c program to simulate the following non-preemptive CPU scheduling algorithm to find turnaround time and waiting time.

i) FCFS
ii) SJF (Non-preemptive)

i) FCFS

```
#include <stdio.h>
// Fn to find waiting time and TAT.
void findWaitingTime(int processes[], int n,
                     int bt[], int wt[]){
    // waiting time for the first process is 0
    wt[0]=0;
    // calculating waiting time
    for(int i=1; i<n;i++){
        wt[i] = wt[i-1] + bt[i-1];
    }
}
void findTAT(int processes[], int n, int bt[]
             int wt[], int tat[]) {
    // TAT = Burst time + waiting time
    for(int i=0 ; i<n;i++) {
        tat[i] = bt[i] + wt[i];
    }
}
void findAvgTime(int processes[], int n,
                 int bt[])
    int wt[n], tat[n];
```

```
int total_wt = 0, total_tat = 0;
// finding waiting time for all processes
findWaitingTime(processes, n, bt, wt);
// finding TAT
findTAT(processes, n, bt, wt, tat);
printf("Processes  Burst time  Waiting time
        Turnaround time \n");
for(int i=0; i<n; i++) {
    total_wt += wt[i];
    total_tat += tat[i];
    printf(" %d\t  %d\t  %d\t  %d\n", processes[i], bt[i],
           wt[i], tat[i]);
}
float avg_wt = (float) total_wt / n;
float avg_tat = (float) total_tat / n;
printf("Average waiting time = %f\n",
       avg_wt);
printf("Average turnaround time = %f\n",
       avg_tat);
}
int main() {
    int n;
    printf("Enter the no. of processes: ");
    scanf("%d", &n);
```

```
int processes [n];
int burst_time[n];
printf("Enter burst time for each process:\n");
for(int i=0; i<n; i++){
    printf("Burst time for process %d:", i+1);
    scanf("%d", &burst_time[i]);
    processes[i]= i+1;
}
findAvgTime(processes, n, burst_time);
return 0;
}
```

output:

Enter the number of processes: 4
Enter burst time for each process:
Burst time for process 1: 5
" 2: 3
process 3: 8
" 4: 6

Process	Burst time	Waiting time	Turnaround time
1	5	0	5
2	3	5	8
3	8	8	16
4	6	16	22

Average waiting Time = 7.250000
Average TAT = 12.750000

amazon basics

② SJF

```
#include <stdio.h>
struct Process {
    int id;
    int arrival_time;
    int burst_time;
    int waiting_time;
    int turnaround_time;
};
void sjf_scheduling(struct Process processes[], int n) {
    int total_waiting_time = 0, total_turnaround_time = 0;
    for (int i = 0; i < n-1; i++) {
        for (int j = 0; j < n-i-1; j++) {
            if (processes[j].arrival_time > processes[j+1].arrival_time) {
                struct Process temp = processes[j];
                processes[j] = processes[j+1];
                processes[j+1] = temp;
            }
        }
    }
    for (int i = 0; i < n; i++) {
        if (i == 0) {
            processes[i].waiting_time = 0;
        } else {
            processes[i].waiting_time = processes[i-1].burst_time + processes[i-1].waiting_time;
```

```
    processes[i].turnaround_time =
    processes[i].waiting_time + processes[i].
                                    burst_time;
    total_waiting_time += processes[i].waiting_
                                    time;
    total_turnaround_time += processes[i]
                                .turnaround_time;
}
printf("Process\t Arrival Time\t Burst Time\t
    Waiting Time\t TurnAround Time\n");
for(int i=0; i<n; i++){
    printf("%d\t %d\t\t %d\t\t %d\t\t %d\n",
    processes[i].id, processes[i].arrival_time,
    processes[i].burst_time, processes[i].
    waiting_time, processes[i].turnaround_
                                time);
}
printf("\nAverage waiting time: %.2f\n",
                (float) total_waiting_time/n);
printf("Average TAT: %.2f\n",
                (float) total_turnaround_
                                time/n);
}
int main(){
    int n;
    printf("Enter the no. of processes\n");
    scanf("%d", &n);
    struct Process processes[n];
```

```
for(int i=0; i<n; i++){
    printf("Enter arrival time and burst time for process %d:", i+1);
    scanf("%d%d", &processes[i].arrival_time, &processes[i].burst_time);
    processes[i].id = i+1;
}
sjf_scheduling(processes, n);
return 0;
}
```

output:

Enter the no. of processes: 4

Enter arrival time and burst time for

process 1: 0 5

" process 2: 2 7

" process 3: 3 8

" process 4: 4 9

Process	Arrival Time	Burst Time	Waiting Time	TAT
1	0	5	0	5
2	2	7	5	12
3	3	8	12	20
4	4	9	20	29

Average Waiting Time = 9.25 ms

Average TAT : 16.50 ms.

16/5/2024

x