**LAB ASSIGNMENT-5**

**1. First Come First Serve (FCFS)**

c

#include<stdio.h>

struct Process {

int pid, at, bt, ct, tat, wt;

};

int main() {

int n;

printf("Enter number of processes: ");

scanf("%d", &n);

struct Process p[n];

printf("Enter arrival and burst times:\n");

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

p[i].pid = i+1;

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

}

*// Sort by arrival time*

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

for(int j=0; j<n-i-1; j++) {

if(p[j].at > p[j+1].at) {

struct Process temp = p[j];

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

p[j+1] = temp;

}

}

}

*// Calculate times*

int current = 0;

float avg\_wt = 0, avg\_tat = 0;

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

if(current < p[i].at) current = p[i].at;

p[i].ct = current + p[i].bt;

p[i].tat = p[i].ct - p[i].at;

p[i].wt = p[i].tat - p[i].bt;

current = p[i].ct;

avg\_wt += p[i].wt;

avg\_tat += p[i].tat;

}

printf("\nPID\tAT\tBT\tCT\tTAT\tWT\n");

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

printf("%d\t%d\t%d\t%d\t%d\t%d\n",

p[i].pid, p[i].at, p[i].bt, p[i].ct, p[i].tat, p[i].wt);

}

printf("\nAverage Waiting Time: %.2f", avg\_wt/n);

printf("\nAverage Turnaround Time: %.2f", avg\_tat/n);

return 0;

}

**Sample Output:**

text

Enter number of processes: 3

Enter arrival and burst times:

0 5

0 11

0 11

PID AT BT CT TAT WT

1 0 5 5 5 0

2 0 11 16 16 5

3 0 11 27 27 16

Average Waiting Time: 7.00

Average Turnaround Time: 16.00

**2. Shortest Job First (SJF) - Non-Preemptive**

c

#include<stdio.h>

struct Process {

int pid, bt, tat, wt;

};

int main() {

int n;

printf("Enter number of processes: ");

scanf("%d", &n);

struct Process p[n];

printf("Enter burst times:\n");

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

p[i].pid = i+1;

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

}

*// Sort by burst time*

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

for(int j=0; j<n-i-1; j++) {

if(p[j].bt > p[j+1].bt) {

struct Process temp = p[j];

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

p[j+1] = temp;

}

}

}

*// Calculate times*

int current = 0;

float avg\_wt = 0, avg\_tat = 0;

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

p[i].wt = current;

p[i].tat = p[i].wt + p[i].bt;

current += p[i].bt;

avg\_wt += p[i].wt;

avg\_tat += p[i].tat;

}

printf("\nPID\tBT\tTAT\tWT\n");

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

printf("%d\t%d\t%d\t%d\n",

p[i].pid, p[i].bt, p[i].tat, p[i].wt);

}

printf("\nAverage Waiting Time: %.2f", avg\_wt/n);

printf("\nAverage Turnaround Time: %.2f", avg\_tat/n);

return 0;

}

**Sample Output:**

text

Enter number of processes: 4

Enter burst times:

6 8 7 3

PID BT TAT WT

4 3 3 0

1 6 6 0

3 7 13 6

2 8 21 13

Average Waiting Time: 4.75

Average Turnaround Time: 10.75

**3. Round Robin Scheduling**

c

#include<stdio.h>

struct Process {

int pid, at, bt, rt, tat, wt;

};

int main() {

int n, quantum;

printf("Enter number of processes: ");

scanf("%d", &n);

printf("Enter time quantum: ");

scanf("%d", &quantum);

struct Process p[n];

printf("Enter arrival and burst times:\n");

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

p[i].pid = i+1;

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

p[i].rt = p[i].bt;

}

int time = 0, complete = 0;

while(complete < n) {

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

if(p[i].rt > 0 && p[i].at <= time) {

int execute = (p[i].rt > quantum) ? quantum : p[i].rt;

time += execute;

p[i].rt -= execute;

if(p[i].rt == 0) {

p[i].tat = time - p[i].at;

p[i].wt = p[i].tat - p[i].bt;

complete++;

}

}

}

}

float avg\_wt = 0, avg\_tat = 0;

printf("\nPID\tBT\tTAT\tWT\n");

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

printf("%d\t%d\t%d\t%d\n",

p[i].pid, p[i].bt, p[i].tat, p[i].wt);

avg\_wt += p[i].wt;

avg\_tat += p[i].tat;

}

printf("\nAverage Waiting Time: %.2f", avg\_wt/n);

printf("\nAverage Turnaround Time: %.2f", avg\_tat/n);

return 0;

}

**Sample Output:**

text

Enter number of processes: 3

Enter time quantum: 4

Enter arrival and burst times:

0 10

0 5

0 8

PID BT TAT WT

1 10 22 12

2 5 7 2

3 8 20 12

Average Waiting Time: 8.67

Average Turnaround Time: 16.33