FCFS

#include <stdio.h>

int main() {

int n;

printf("Enter number of processes: ");

scanf("%d", &n);

int pid[n], at[n], bt[n], ct[n], tat[n], wt[n];

// Input process ID, arrival time, burst time

printf("Enter Process ID, Arrival Time and Burst Time:\n");

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

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

}

// Sort processes by Arrival Time (using bubble sort)

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

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

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

// Swap arrival time

int temp = at[j];

at[j] = at[j+1];

at[j+1] = temp;

// Swap burst time

temp = bt[j];

bt[j] = bt[j+1];

bt[j+1] = temp;

// Swap process ID

temp = pid[j];

pid[j] = pid[j+1];

pid[j+1] = temp;

}

}

}

int time = 0;

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

if (time < at[i])

time = at[i]; // Wait until the process arrives

ct[i] = time + bt[i];

time = ct[i];

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

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

}

// Display output

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

}

return 0;

}

// pseudo code

START

1. Input number of processes (n)

2. For each process i from 0 to n-1:

a. Input Process ID (pid[i])

b. Input Arrival Time (at[i])

c. Input Burst Time (bt[i])

3. Sort all processes by Arrival Time using bubble sort:

- Swap pid[], at[], and bt[] accordingly

4. Initialize current time = 0

5. For each process i from 0 to n-1:

a. If current time < arrival time, set current time = arrival time

b. Completion Time (ct[i]) = current time + burst time

c. Update current time = ct[i]

d. Turnaround Time (tat[i]) = ct[i] - at[i]

e. Waiting Time (wt[i]) = tat[i] - bt[i]

6. Print all process details:

PID, AT, BT, CT, TAT, WT

END