Code:

#include<iostream>

#include<algorithm>

using namespace std;

struct Process {

int processID;

int burstTime;

int arrivalTime;

int ct;

};

void findWaitingTime(Process processes[], int n, int wt[]) {

sort(processes, processes + n, [](const Process& a, const Process& b) {

return a.arrivalTime < b.arrivalTime;

});

int ct = 0;

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

ct += processes[i].burstTime;

processes[i].ct = ct; /

wt[i] = ct-processes[i].burstTime - processes[i].arrivalTime;

}

}

void findTurnAroundTime(Process processes[], int n, int wt[], int tat[]) {

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

tat[i] = processes[i].burstTime + wt[i];

}

void findavgTime(Process processes[], int n) {

int wt[n], tat[n], total\_wt = 0, total\_tat = 0;

findWaitingTime(processes, n, wt);

findTurnAroundTime(processes, n, wt, tat);

cout << "Processes " << " Arrival time " << " Burst time "

<< " Waiting time " << " Turn around time\n";

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

total\_wt = total\_wt + wt[i];

total\_tat = total\_tat + tat[i];

cout << " " << processes[i].processID << "\t\t" << processes[i].arrivalTime << "\t\t"

<< processes[i].burstTime << "\t " << wt[i] << "\t\t " << tat[i] << endl;

}

cout << "Average waiting time = " << (float)total\_wt / (float)n;

cout << "\nAverage turn around time = " << (float)total\_tat / (float)n;

}

int main() {

int n;

cout << "Enter the number of processes: ";

cin >> n;

Process processes[n];

cout << "Enter burst time and arrival time for each process:\n";

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

cout << "Enter burst time for process " << i + 1 << ": ";

cin >> processes[i].burstTime;

cout << "Enter arrival time for process " << i + 1 << ": ";

cin >> processes[i].arrivalTime;

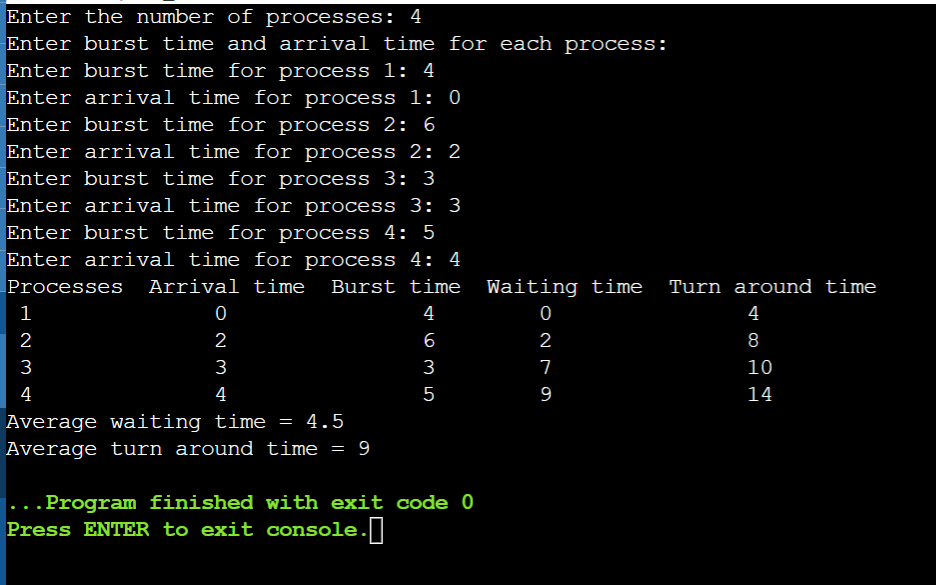
processes[i].processID = i + 1;

}

findavgTime(processes, n);

return 0;

}

Output:

Code:

//SJF

#include <iostream>

#include <limits>

using namespace std;

struct Process {

int pid; // Process ID

int bt; // Burst Time

int art; // Arrival Time

};

// Function to find the waiting time for all processes

void findWaitingTime(Process proc[], int n, int wt[]) {

int rt[n];

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

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

int complete = 0, t = 0, minm = 99;

int shortest = 0, finish\_time;

bool check = false;

while (complete != n) {

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

if ((proc[j].art <= t) && (rt[j] < minm) && rt[j] > 0) {

minm = rt[j];

shortest = j;

check = true;

}

}

if (check == false) {

t++;

continue;

}

rt[shortest]--;

minm = rt[shortest];

if (minm == 0)

minm = 99;

if (rt[shortest] == 0) {

complete++;

check = false;

finish\_time = t + 1;

wt[shortest] = finish\_time - proc[shortest].bt - proc[shortest].art;

if (wt[shortest] < 0)

wt[shortest] = 0;

}

t++;

}

}

// Function to calculate turn around time

void findTurnAroundTime(Process proc[], int n, int wt[], int tat[]) {

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

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

}

// Function to calculate average time

void findavgTime(Process proc[], int n) {

int wt[n], tat[n], total\_wt = 0, total\_tat = 0;

findWaitingTime(proc, n, wt);

findTurnAroundTime(proc, n, wt, tat);

cout << " P\tBT\tAT\tWT\tTAT\n";

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

total\_wt = total\_wt + wt[i];

total\_tat = total\_tat + tat[i];

cout << " " << proc[i].pid << "\t" << proc[i].bt << "\t" << proc[i].art

<< "\t" << wt[i] << "\t" << tat[i] << endl;

}

cout << "\nAverage waiting time = " << (float)total\_wt / (float)n;

cout << "\nAverage turn around time = " << (float)total\_tat / (float)n;

}

int main() {

int n;

cout << "Enter the number of processes: ";

cin >> n;

Process proc[n];

cout << "Enter burst time and arrival time for each process:\n";

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

cout << "Enter burst time for process " << i + 1 << ": ";

cin >> proc[i].bt;

cout << "Enter arrival time for process " << i + 1 << ": ";

cin >> proc[i].art;

proc[i].pid = i + 1;

}

findavgTime(proc, n);

return 0;

}

Output:

