**Student Name: Anandita Pandey**

**Student ID: 11807433**

**Email Address:anandita.p23@gmail.com**

**GitHub Link:** [**https://github.com/Anandita23**](https://github.com/Anandita23)

**Code 1:**

// C program for implementation of FCFS

#include<stdio.h>

void findWaitingTime(int processes[], int n,

int bt[], int wt[])

{

// waiting time for first process is 0

wt[0] = 0;

// calculating waiting time

for (int  i = 1; i < n ; i++ )

wt[i] =  bt[i-1] + wt[i-1] ;

}

// Function to calculate turn around time

void findTurnAroundTime( int processes[], int n,

int bt[], int wt[], int tat[])

{

for (int  i = 0; i < n ; i++)

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

}

//Function to calculate average time

void findavgTime( int processes[], int n, int bt[])

{

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

//Function to find waiting time of all processes

findWaitingTime(processes, n, bt, wt);

//Function to find turn around time for all processes

findTurnAroundTime(processes, n, bt, wt, tat);

//Display processes along with all details

printf("Processes   Burst time   Waiting time   Turn around time\n");

// Calculate total waiting time and total turn

// around time

for (int  i=0; i<n; i++)

{

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

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

printf("   %d ",(i+1));

printf("       %d ", bt[i] );

printf("       %d",wt[i] );

printf("       %d\n",tat[i] );

}

int s=(float)total\_wt / (float)n;

int t=(float)total\_tat / (float)n;

printf("Average waiting time = %d",s);

printf("\n");

printf("Average turn around time = %d ",t);

}

int main()

{

//process id's

int processes[] = { 1, 2, 3};

int n = sizeof processes / sizeof processes[0];

//Burst time of all processes

int  burst\_time[] = {10, 5, 8};

findavgTime(processes, n,  burst\_time);

return 0;

}

**Code 2:**

#include <bits/stdc++.h>

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];

// Copy the burst time into rt[]

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

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

int complete = 0, t = 0, minm = INT\_MAX;

int shortest = 0, finish\_time;

bool check = false;

// Process until all processes gets

// completed

while (complete != n) {

// Find process with minimum

// remaining time among the

// processes that arrives till the

// current time`

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;

}

// Reduce remaining time by one

rt[shortest]--;

// Update minimum

minm = rt[shortest];

if (minm == 0)

minm = INT\_MAX;

// If a process gets completely

// executed

if (rt[shortest] == 0) {

// Increment complete

complete++;

check = false;

// Find finish time of current

// process

finish\_time = t + 1;

// Calculate waiting time

wt[shortest] = finish\_time -

proc[shortest].bt -

proc[shortest].art;

if (wt[shortest] < 0)

wt[shortest] = 0;

}

// Increment time

t++;

}

}

// Function to calculate turn around time

void findTurnAroundTime(Process proc[], int n,

int wt[], int tat[])

{

// calculating turnaround time by adding

// bt[i] + wt[i]

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;

// Function to find waiting time of all

// processes

findWaitingTime(proc, n, wt);

// Function to find turn around time for

// all processes

findTurnAroundTime(proc, n, wt, tat);

// Display processes along with all

// details

printf("Processes   Burst time   Waiting time   Turn around time\n");

// Calculate total waiting time and

// total turnaround time

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

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

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

printf("   %d ",(i+1));

printf("       %d ", bt[i] );

printf("       %d",wt[i] );

printf("       %d\n",tat[i]

}

printf("Average waiting time = %d",s);

printf("\n");

printf("Average turn around time = %d ",t);

}

// Driver code

int main()

{

Process proc[] = { { 1, 6, 1 }, { 2, 8, 1 },

{ 3, 7, 2 }, { 4, 3, 3 } };

int n = sizeof(proc) / sizeof(proc[0]);

findavgTime(proc, n);

return 0;

}

**Description:**

**Algorithm:**

**FCFS:**

1- Input the processes along with their burst time (bt).

2- Find waiting time (wt) for all processes.

3- As first process that comes need not to wait so

waiting time for process 1 will be 0 i.e. wt[0] = 0.

4- Find waiting time for all other processes i.e. for

process i ->

wt[i] = bt[i-1] + wt[i-1] .

5- Find turnaround time = waiting\_time + burst\_time

for all processes.

6- Find average waiting time =

total\_waiting\_time / no\_of\_processes.

7- Similarly, find average turnaround time =

total\_turn\_around\_time / no\_of\_processes.

**SJF:**

1- Traverse until all process gets completely

executed.

a) Find process with minimum remaining time at

every single time lap.

b) Reduce its time by 1.

c) Check if its remaining time becomes 0

d) Increment the counter of process completion.

e) Completion time of current process =

current\_time +1;

e) Calculate waiting time for each completed

process.

wt[i]= Completion time - arrival\_time-burst\_time

f)Increment time lap by one.

2- Find turnaround time (waiting\_time+burst\_time).

3.Calculate complexity of implemented algorithm. (Student must specify complexity of each line of code along with overall complexity)

**Complexity:**

**SJF: O(n log n)**

**FCFS: O(n)**

4.Explain all the constraints given in the problem. Attach the code snippet of the implemented constraint.

Code snippet:

// calculate turn around time

for (int  i = 0; i < n ; i++)

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

}

//Function to calculate average time

void findavgTime( int processes[], int n, int bt[])

{

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

//Function to find waiting time of all processes

findWaitingTime(processes, n, bt, wt);

//Function to find turn around time for all processes

findTurnAroundTime(processes, n, bt, wt, tat);

//Display processes along with all details

printf("Processes   Burst time   Waiting time   Turn around time\n");

// Calculate total waiting time and total turn

// around time

for (int  i=0; i<n; i++)

{

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

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

printf("   %d ",(i+1));

printf("       %d ", bt[i] );

printf("       %d",wt[i] );

printf("       %d\n",tat[i] );

}

int s=(float)total\_wt / (float)n;

int t=(float)total\_tat / (float)n;

printf("Average waiting time = %d",s);

printf("\n");

printf("Average turn around time = %d ",t);

}

5.If you have implemented any additional algorithm to support the solution, explain the need and usage of the same.

Description: I have only used FCFS and SJF Algorithm for implementation.

6. Explain the boundary conditions of the implemented code.

Description: The program should work till the number of processes available.

7.Explain all the test cases applied on the solution of assigned problem.

Description:

1. Q1 is for finding the burst time and waiting time.
2. Q2 is for finding the turn around time by adding BT and AT.
3. Q3 is for finding average turn around time.