**Name:** Kushal Kishor Shankhapal **Date:** 30/07/2023

**Roll No:** 56 **Subject:** OS Lab

**SJF: Shortest Job First**

**Code:**

#include <stdio.h>

#include <limits.h> // For INT\_MAX to represent a large number

#define MAX\_PROCESSES 10 // Define the maximum number of processes supported

int main() {

// Arrays to store arrival times, burst times, and temporary burst times

int arrival\_time[MAX\_PROCESSES], burst\_time[MAX\_PROCESSES], temp[MAX\_PROCESSES];

int remaining\_time[MAX\_PROCESSES]; // Array to track the remaining burst times for each process

int completion\_time[MAX\_PROCESSES], waiting\_time[MAX\_PROCESSES], turnaround\_time[MAX\_PROCESSES];

int i, smallest, count = 0, time, limit;

int total\_wait\_time = 0, total\_turnaround\_time = 0;

float average\_waiting\_time, average\_turnaround\_time;

// Prompt the user to enter the number of processes

printf("\nEnter the Total Number of Processes (max %d):\t", MAX\_PROCESSES);

scanf("%d", &limit);

// Check if the number of processes exceeds the maximum allowed

if (limit > MAX\_PROCESSES) {

printf("Number of processes cannot exceed %d\n", MAX\_PROCESSES);

return 1;

}

// Read the arrival and burst times for each process

printf("\nEnter Details of %d Processes\n", limit);

for(i = 0; i < limit; i++) {

printf("\nProcess %d:\n", i + 1);

printf("Enter Arrival Time:\t");

scanf("%d", &arrival\_time[i]);

printf("Enter Burst Time:\t");

scanf("%d", &burst\_time[i]);

remaining\_time[i] = burst\_time[i]; // Initialize remaining time with the burst time

temp[i] = burst\_time[i]; // Store the original burst time for later calculations

}

int completed[MAX\_PROCESSES] = {0}; // Array to keep track of which processes have been completed

// Main loop to simulate time and schedule processes

for(time = 0; count != limit; time++) {

smallest = -1; // Initialize smallest as -1 to indicate no process selected yet

// Find the process with the smallest remaining burst time that has arrived and is not yet completed

for(i = 0; i < limit; i++) {

if (arrival\_time[i] <= time && !completed[i]) {

if (smallest == -1 || remaining\_time[i] < remaining\_time[smallest]) {

smallest = i; // Update smallest to the current process with the shortest remaining time

}

}

}

if (smallest != -1) {

remaining\_time[smallest]--; // Execute the selected process for one unit of time

if (remaining\_time[smallest] == 0) {

// Process is completed

count++;

completed[smallest] = 1; // Mark the process as completed

completion\_time[smallest] = time + 1; // Calculate completion time of the process

waiting\_time[smallest] = completion\_time[smallest] - arrival\_time[smallest] - temp[smallest];

turnaround\_time[smallest] = completion\_time[smallest] - arrival\_time[smallest];

total\_wait\_time += waiting\_time[smallest]; // Accumulate total waiting time

total\_turnaround\_time += turnaround\_time[smallest]; // Accumulate total turnaround time

}

}

}

// Calculate average waiting time and turnaround time

average\_waiting\_time = (float)total\_wait\_time / limit;

average\_turnaround\_time = (float)total\_turnaround\_time / limit;

// Print the process details including arrival time, burst time, completion time, waiting time, and turnaround time

printf("\nProcess No\tAT\tBT\tCT\tTAT\tWT\n");

for(i = 0; i < limit; i++) {

printf("%d\t\t%d\t%d\t%d\t%d\t%d\n", i + 1, arrival\_time[i], temp[i], completion\_time[i], turnaround\_time[i], waiting\_time[i]);

}

// Print the average waiting time and average turnaround time

printf("\nAverage Waiting Time:\t%f", average\_waiting\_time);

printf("\nAverage Turnaround Time:\t%f\n", average\_turnaround\_time);

return 0;

}

**Output:**

pl-17@pl17-OptiPlex-3020:~$ cd Kushal\_Files

pl-17@pl17-OptiPlex-3020:~/Kushal\_Files$ g++ sjf\_p1.cpp -o sjf.out

pl-17@pl17-OptiPlex-3020:~/Kushal\_Files$ ./sjf.out

Enter the Total Number of Processes (max 10): 4

Enter Details of 4 Processes

Process 1:

Enter Arrival Time: 1

Enter Burst Time: 3

Process 2:

Enter Arrival Time: 2

Enter Burst Time: 4

Process 3:

Enter Arrival Time: 1

Enter Burst Time: 2

Process 4:

Enter Arrival Time: 4

Enter Burst Time: 4

Process No AT BT CT TAT WT

1 1 3 6 5 2

2 2 4 10 8 4

3 1 2 3 2 0

4 4 4 14 10 6

Average Waiting Time: 3.000000

Average Turnaround Time: 6.250000