Student Name: Dorababu Pragada

Student Id: 11802429

Email Address: pragadadorababu419@gamil.com

GitHub link:

Code:

#include<stdio.h>

#include<stdlib.h>

typedef struct

{

int arr\_time,bur\_time,que;

int process\_id;

}process;

int main()

{

int size,quantum,done =0;//Number of processes

int count = 0;

int time,smallest,waiting\_time[50],turnaround\_time[50],completion[50],temp\_burst[50];

double average\_waiting,average\_turnaround;

printf("\nEnter the number of processes:");

scanf("%d",&size);

process no[size];

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

{

printf("\nProcess ID:\t");

scanf("%d", &no[i].process\_id);

printf("Arrival Time:\t");

scanf("%d", &no[i].arr\_time);

printf("Burst Time:\t");

scanf("%d", &no[i].bur\_time);

printf("Queue1/Queue2(1/2):\t");

scanf("%d", &no[i].que);

}

printf("\nEnter the time quantum for Round Robin:");

scanf("%d",&quantum);

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

{

temp\_burst[i] = no[i].bur\_time;//storing the burst time in a array for further use.

}

no[size+1].bur\_time = 999;// assign a max burst for comparison

for(time=0;count!=size;time++) // shortest remaining time first scheduling

{

smallest = size+1;

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

{

if(no[i].arr\_time<=time && no[i].bur\_time<no[smallest].bur\_time && no[i].bur\_time>0 && no[i].que == 1)

{

smallest=i;

}

}

if(count >= size/2) //round robin scheduling

{

while(1)

{

for(int j=0;j<size;j++)

{

if(no[j].arr\_time<=time && no[j].bur\_time > quantum && no[j].que == 2 )

{

no[j].bur\_time -= quantum;

smallest = j;

}

else if(no[j].arr\_time<=time && no[j].bur\_time < quantum && no[j].que == 2)

{

no[j].bur\_time = 0;

smallest = j;

goto a;

}

}

}

}

no[smallest].bur\_time--; //decrementing the burst time

a: if(no[smallest].bur\_time == 0)

{

count++;

completion[smallest] = time+1;

turnaround\_time[smallest] = completion[smallest] - no[smallest].arr\_time;

waiting\_time[smallest] = turnaround\_time[smallest] - temp\_burst[smallest];

}

}

printf("\n\nProcess Id\tArrival Time\t Burst Time\t Waiting Time\tTurnaround Time");

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

{

printf("\n P%d \t\t%d\t\t%d \t\t%d\t\t%d",no[i].process\_id,no[i].arr\_time,temp\_burst[i],waiting\_time[i],turnaround\_time[i]);

average\_waiting += waiting\_time[i];

average\_turnaround += turnaround\_time[i];

}

printf("\n\nAverage waiting time = %lf\n",average\_waiting/size);

printf("Average Turnaround time = %lf",average\_turnaround/size);

}

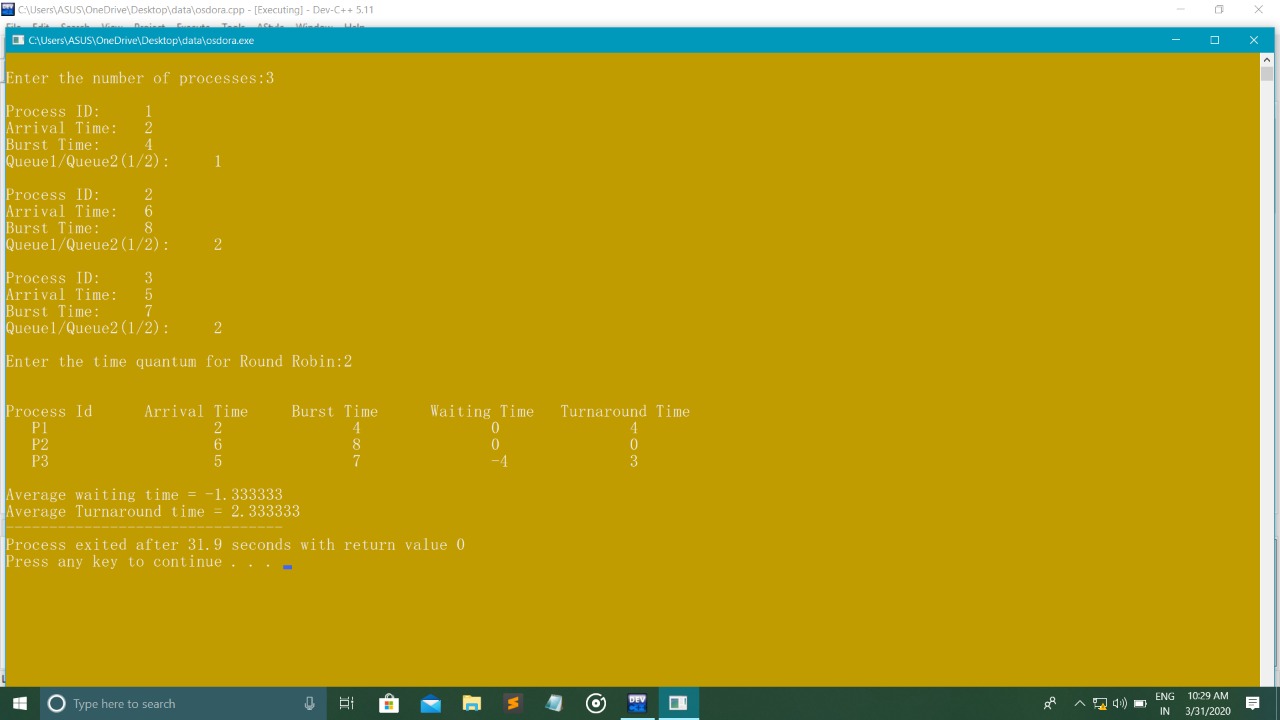
**DESCRIPTION:**

In the program to inputs are we have enter the processers and all are in positive integer. And also give the arrival time and burst time and given values to them and get the output turnaround time and waiting time and also we get average turnaround time and average waiting time. That execution and all types of process will be done by CPU.

In the given question the preemptive shortest, scheduler with multilevel queue which will schedule the processes on the basis of preemptive shortest remaining in the processing time first algorithm (SROT) In given program to execution you are giving value is greater than “0” and there are positive integers. We also use round robin time quantum value all positive integers.

In operating system in there are we have to how find the average turnaround time and average waiting time and in the queue 1 has higher priority than queue 2 and enter 1or 2. The program solved by using with shortest remaining time(SROT) Algorithm and Round robin Algorithm.

**Snap shots:**



**COMPLEXITY:**

* In the function there are 76 lines and complexity is 20 this the overall complexity.

|  |
| --- |
|  |
| **Function Name** | **NLOC** | **Complexity** |  |  |
| Main | 76 | 20 |  |  |

**BOUNDARY CONDITIONS:**

* The process given the all positive value that is greater than ‘0’
* The queue 1 is has higher priority than queue 2.
* Time queue should be greater than ‘0’ are positive integers.
* Round robin time should be greater than ‘0’ are positive integers.

**TEST-CASES:**

OUTPUT: Enter the no of processes.

INPUT: Enter any no greater than 0

OUTPUT: Enter 1 ' Queue is higher priority than 2’ Queue (1/2):

INPUT: Enter 1 or 2

OUTPUT: Enter the time quantum round robin

INPUT: Enter all positive integers.

OUTPUT: Enter the Arrival time

INPUT: To Enter, time 12:00 am we should enter 1200.

OUTPUT: Enter the Burst time

INPUT: Enter a positive integer.

The same test cases are repeated for each process and the desired output is given.

Output: The average turnaround time and the average waiting time is displayed for each

Process along with their process no, arrival time and burst times.

**GITHUB LINK:**