Green University of Bangladesh

Dept. of CSE

Course Title: Operating System Lab

Lab Report-03

Submitted By:	Submitted To:
Md. Tanvir Ahmed	Most. Rokeya Khatun
ID No : 191015102	Designation: Lecturer
Batch No: ED	Dept. of CSE
Semester : 8th	Green University Of Bangladesh
Performance Date:6/25/2021	Submission Date: 07/02/2021

Lab Performance-03

Name: Write a C program to multi-level queue scheduling algorithm Using FCFS

Introduction:

A common division is made between foreground processes and background processes. These two types of processes have different response-time requirements and so may have different scheduling needs. In addition, foreground processes may have priority over background processes. A multilevel queue scheduling algorithm partitions the ready queue into several separate queues. The processes are permanently assigned to one queue, generally based on some property of the process, such as memory size, process priority, or process type. Each queue has its own scheduling algorithm.

Algorithm:

Step 1: Input the processes along with their arrival time (AT) and burst time (BT).

Step 2: Find completion time for all processes.

Step 3: calculate turnaround time for all processes.

Step 4: calculate waiting time for all processes.

Step 6: Find average waiting time = total waiting time/ no of process.

Step 7: Find average turnaround time = total turnaround time/no of process

Code:

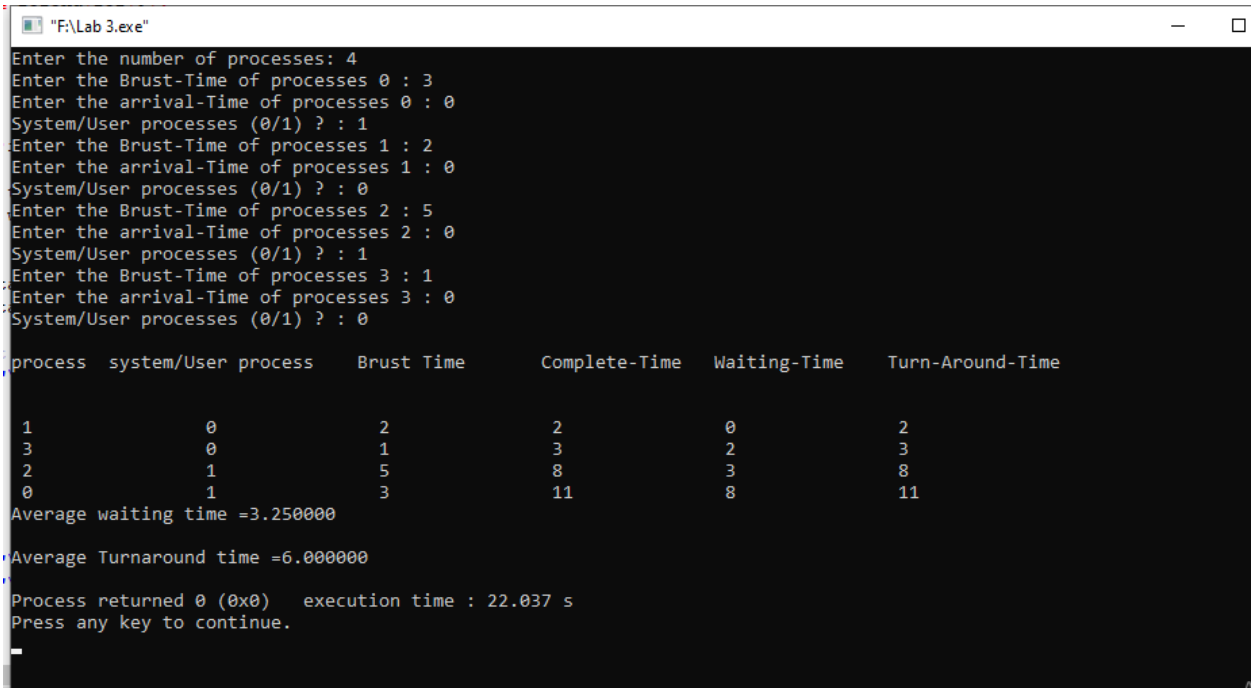
```
#include <stdio.h>
int main()
{
   int p[20], bt[20], at[20], su[20], wt[20], ct[20] = {0}, tat[20], i, j, k, n, temp, sum = 0;
   float wtavg, tatavg;
   printf("Enter the number of processes: ");
   scanf("%d", &n);
   for(i = 0; i < n; i++)
   {
      p[i]=i;
      printf("Enter the Brust-Time of processes %d : ",i);
      scanf("%d", &bt[i]);
      printf("Enter the arrival-Time of processes %d : ",i);
      scanf("%d", &at[i]);
      printf("System/User processes (0/1) ? : ");
      scanf("%d", &su[i]);
   }
   for(i = 0; i < n; i++)
   {
      for(k=i+1; k < n; k++)
      {
```

```
        if( su[i]>su[k] )
        {
            temp = p[i];
            p[i] = p[k];
            p[k] = temp;
            //Interchange brust time
            temp = bt[i];
            bt[i] = bt[k];
            bt[k] = temp;
            //Interchange system/user
            temp = su[i];
            su[i] = su[k];
            su[k] = temp;
        }
    }
}
wtavg = 0;
tatavg = 0;
for(j=0; j<n; j++)
{
    sum=sum+ bt[j];
    ct[j] = ct[j]+ sum;
}
```

```
    for(i=0; i<n; i++)
    {
        tat[i] = ct[i]-at[i];
        tatavg = tatavg+tat[i];
    }
    for(i=0; i<n; i++)
    {
        wt[i] = tat[i]-bt[i];
        wtavg = wtavg+wt[i];
    }
    wtavg=wtavg/n;
    tatavg=tatavg/n;
    printf("\nprocess\t system/User process\tBrust Time\t Complete-
Time\t Waiting-Time\t Turn-Around-Time \n\n");
    for(i=0; i<n; i++)
    {
        printf("\n %d \t\t  %d \t\t  %d \t\t  %d \t\t  %d \t\t
%d",p[i],su[i],bt[i],ct[i],wt[i],tat[i]);
    }
    printf("\nAverage waiting time =%f\n",wtavg);
    printf("\nAverage Turnaround time =%f\n",tatavg);

}
```

Output:



Discussion:

1) Multi-level queue scheduling algorithm is used in scenarios where the processes can be classified into groups.

2) Each queue will be assigned a priority and will have its own scheduling algorithm like round-robin.

3) A multi-level queue scheduling algorithm partitions the ready queue into several separate queues.

4) A multi-level queue scheduling algorithm, once assigned to a queue, the process will not move to any other queues.

Thank You