LOVELY PROFESSIONAL UNIVERSITY

CSE-316
PROJECT REPORT

Student Name: Abhishek Mishra
Student ID: 11803182 (B56)
Email Address: prabhishekgeetamishra@gmail.com
GitHub Link: https://github.com/imwazir/operating-system-project

Problem: Three students (a, b, c) are arriving in the mess at the same time. The id numbers of these students are 2132, 2102, 2453 and the food taken time from the mess table is 2, 4 and 8 minutes. If the two students have same remaining time so it is broken by giving priority to the students with the lowest id number. Consider the longest remaining time first (LRTF) scheduling algorithm and calculate the average turnaround time and waiting time.

Answer: This is a pre-emptive version of Longest Job First (LJF) scheduling algorithm. In this scheduling algorithm, we find the process with maximum remaining time and then process it. We check for the maximum remaining time after some interval of time (say 1 unit each) to check if another process having more Burst Time arrived up to that time.

Algorithm:

- **Step-1:** Create a structure of process containing all necessary fields like AT (Arrival Time), BT (Burst Time), CT (Completion Time), TAT (Turn Around Time), WT (Waiting Time).
- **Step-2:** Sort according to the AT;
- **Step-3:** Find the process having Largest Burst Time and execute for each single unit. Increase the total time by 1 and reduce the Burst Time of that process with 1.
- **Step-4:** When any process has 0 BT left, then update the CT (Completion Time ofthat process CT will be Total Time at that time).
- **Step-5:** After calculating the CT for each process, find TAT and WT.

Formulas Used:

Turn Around Time (TAT) = (Completion Time) - (Arrival Time) Also, Waiting Time (WT) = (Turn Around Time) - (Burst Time)

Arrival Time: Time at which the process arrives in the ready queue.
Completion Time: Time at which process completes its execution.
Burst Time: Time required by a process for CPU execution.
Turn Around Time: Time Difference between completion time and arrival time.

Turn Around Time = Completion Time - Arrival Time

Waiting Time (W.T): Time Difference between turnaround time and burst time.
Waiting Time = Turn Around Time - Burst Time

Description:

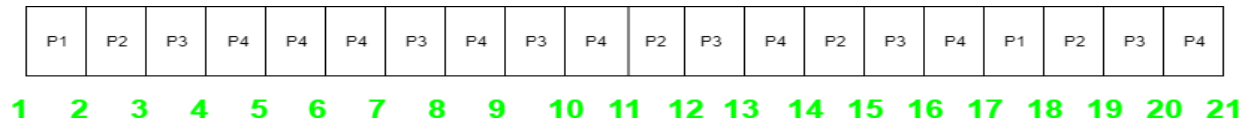
Procedure:

- **Step-1:** First, sort the processes in increasing order of their Arrival Time.
- **Step-2:** Choose the process having least arrival time but with most Burst Time. Then process it for 1 unit. Check if any other process arrives up to that time of execution or not.
- **Step-3:** Repeat the above both steps until execute all the processes.

Example: Consider the following table of arrival time and burst time for four processes P1, P2, P3 and P4.

Process	Arrival time	Burst Time
P1	1 ms	2 ms
P2	2 ms	4 ms
P3	3 ms	6 ms
p4	4 ms	8 ms

3.3 Gantt chart will be as following below,



Since, completion time (CT) can be directly determined by Gantt chart, and

Therefore,

Output:

Total Turn Around Time = 68 ms
So, Average Turn Around Time = 68/4 = 17.00 ms

And, Total Waiting Time = 48 ms
So, Average Waiting Time = 12.00 ms

Code:

```
#include <stdio.h>
struct student
{
   int Student_Id;
   int FoodTakenTime;
   int WaitingTime;
   int TurnAroundTime;
};

void get_data(struct student list[], int s);
void show(struct student list[], int s);
void scheduling(struct student list[], int s);
void waitingTime(struct student list[], int n);
void turnAroundTime(struct student list[], int n);

int main()
{
   struct student data[20];
   int n,i;
   char c='n';
   do
     {
   printf("Enter the No. of Students wants to eat in mess :- ");
   scanf("%d", &n);
   get_data(data, n);
   scheduling(data, n);
   waitingTime(data,n);
   turnAroundTime(data,n);
   show(data, n);
   printf("**********************if you want to run this program once more press 'y' : ");
   scanf("%s",&c);
   }while(c=='y');
   return 0;
}

void get_data(struct student list[80], int s)
{
   int i;
   for (i = 0; i < s; i++)
      {
```

```
      // printf("\n\nEnter data for Student %d", i + 1);

       printf("\nEnter Student id for Student %d\t", i + 1);
       scanf("%d", &list[i].Student_Id);

       printf("Enter time taken for food (minuts) for Student %d\t", i + 1);
       scanf("%d", &list[i].FoodTakenTime);
   }
}
void show(struct student list[80], int s)
{
   int i,AvgWaitingTime=0,AvgTurnAroundTime=0;
      int TotalWatingTime=0,TotalTurnAroundTime=0;
   printf("\n\nOutput according to LRTF\n");
   printf("\n|*************************************************************|");
   printf("\n|Student id\tFoodTakenTime\tWaitingTime\tTurnAroundTime |");
   printf("\n|*************************************************************|");
   for (i = 0; i < s; i++)
   {
      printf("\n|%d\t\t%d\t\t%d\t\t%d\t\t|", list[i].Student_Id,
list[i].FoodTakenTime,list[i].WaitingTime,list[i].TurnAroundTime);

printf("\a\n|=============================================================
====|");
                TotalWatingTime= TotalWatingTime+list[i].WaitingTime;
                TotalTurnAroundTime= TotalTurnAroundTime+list[i].TurnAroundTime;
      }
      printf("\n\n\t\t\t\t\tTotal Waiting Time is: = %d",TotalWatingTime);
      printf("\n\t\t\t\t\tTotal Turn around Time is: = %d\n\n",TotalTurnAroundTime);
      printf("\n\n\t\t\t\t\tAverage Waiting Time is: = %d",TotalWatingTime/s);
      printf("\n\t\t\t\t\tAverage Turn around Time is: = %d\n\n",TotalTurnAroundTime/s);
}

void scheduling(struct student list[80], int s)
{
   int i, j;
   struct student temp;

   for (i = 0; i < s - 1; i++)
   {
      for (j = 0; j < (s - 1-i); j++)
      {
```

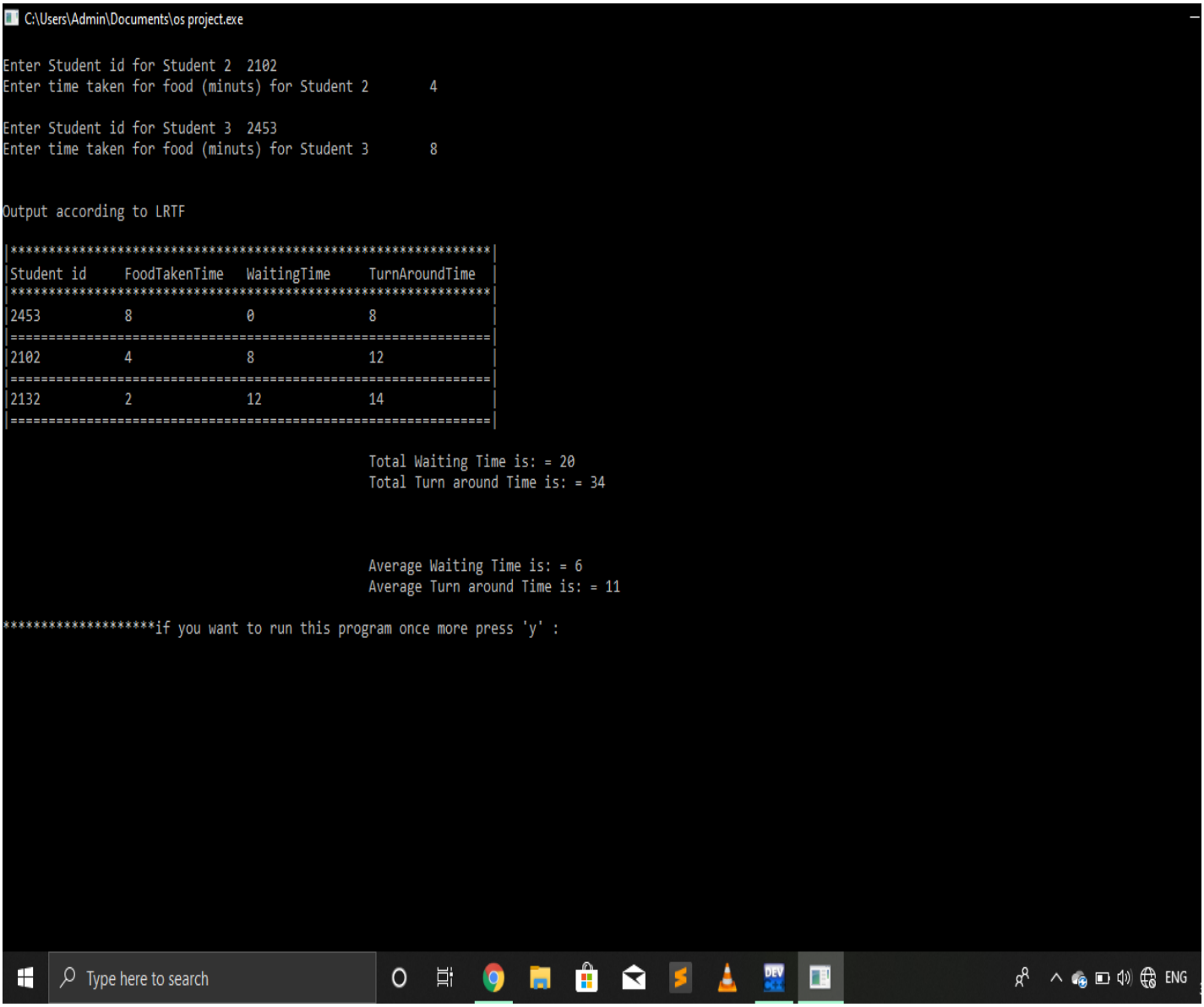
```
            if (list[j].FoodTakenTime < list[j + 1].FoodTakenTime)
            {
               temp = list[j];
               list[j] = list[j + 1];
               list[j + 1] = temp;
            }
            else if(list[j].FoodTakenTime == list[j + 1].FoodTakenTime)
            {
                  if(list[j].Student_Id > list[j + 1].Student_Id)
                  {
                  temp = list[j];
               list[j] = list[j + 1];
               list[j + 1] = temp;
               }
                           }
        }
    }
}


void waitingTime(struct student list[80], int n)
{
int j,total;
list[0].WaitingTime=0;
    for(j=1;j<n;j++)
    {
       list[j].WaitingTime=list[j-1].WaitingTime+list[j-1].FoodTakenTime;
    }
}


void turnAroundTime(struct student list[80], int n)
{
int j,total;

          for(j=0;j<n;j++)
         {
          list[j].TurnAroundTime=list[j].WaitingTime+list[j].FoodTakenTime;
         }
}
```

Output:


C:\Users\Admin\Documents\os project.exe
Enter Student id for Student 2 2102
Enter time taken for food (minuts) for Student 2 4
Enter Student id for Student 3 2453
Enter time taken for food (minuts) for Student 3 8
Output according to LRTF
Student id FoodTakenTime WaitingTime TurnAroundTime
2453 8 0 8
2102 4 8 12
2132 2 12 14
Total Waiting Time is: = 20
Total Turn around Time is: = 34
Average Waiting Time is: = 6
Average Turn around Time is: = 11
**********if you want to run this program once more press 'y' :
Type here to search
ENG