LOVELY PROFESSIONAL UNIVERSITY

Transforming Education Transforming India

CSE 316

Operating System Assignment

Problem no: 14

Name : Priyansh Rawat

Student ID : 11803937

Section : K18SJ

Email : rawatpriyansh@gmail.com

GitHub link : https://github.com/priyansh-rawat/Teacher-Student-priority-queue-in-c

Problem Statement Q14:

Write a C program to solve the following problem:

If a teacher is being served at the food mess and during the period when he is being served, another teacher comes, then that teacher would get the service (food) next. This process might continue leading to increase in waiting time of students to get food. Ensure in your program that the waiting time of students is minimized.

Problem Analysis:

In this problem the priority algorithm is implemented. In priority algorithm the processes with higher priority are served before the processes with lower priority. In case two processes have the same priority, they will be served in the order they arrive. This may also cause the processes of lower priorities from never getting a chance of being implemented thus causing starvation. In such a case after a particular interval of time the priority of such processes increases.

This program can be solved in C language with the help of multiple arrays. The program inputs the teachers and students in 2 separate queues. The teachers shall be served before the students irrespective of the time they first enter. If 2 teachers or 2 students enter at the same time, they will be served on first come first serve basis. A special priority queue has also been implemented to serve teachers and students who have waited for a definite period of time. In such a case first the teachers and students will be served and only then will more people be allowed to enter the queue.

Algorithm:

```
1.  Set check=0,t_point=0,s_point=0
2.  while (check == 0)
        1.  c=read("would you like to take input");
        2.  while(c=='y')
                1. choice = read("Student/Teacher (0/1)");
                2. switch(choice)
                       Case 0 : input burst time
                       Case 1: input burst time
                3.  c=read("would you like to take more input");
        3.  if(t_point<t_total)
                1. if((int)tw[t_point]>100)
                       1.  y=priority_t(t_point,t_total,tw,tb,t,s_point,s_total,sw);
                       2.  t_point=t_point+y;
        4.  if(s_point<s_total)
                1. if((int)sw[s_point]>100)
                       1.  y=priority_s(t_point,t_total,tw,s_point,s_total,sw,s,sb);
                       2.  t_point=t_point+y;
        5.  if(t_point<t_total)
                1.  write("Teacher %d will be served",t[t_point]+1)
                2. write("waiting time", tw[t_point])
                3.  x=tb[t_point];
                4.  t_point++;
                5.  op1(t_point,t_total,tw,x);
                6.  op1(s_point,s_total,sw,x);
        6.  else if(t_point==t_total && s_point<s_total)
                1.  write("Seacher %d will be served",s[s_point]+1)
                2. write("waiting time", sw[s_point])
                3.  x=sb[s_point];
                4.  s_point++;
                5.  op1(s_point,s_total,sw,x);
        7.  else
                1. check=1;

3.  end.
```

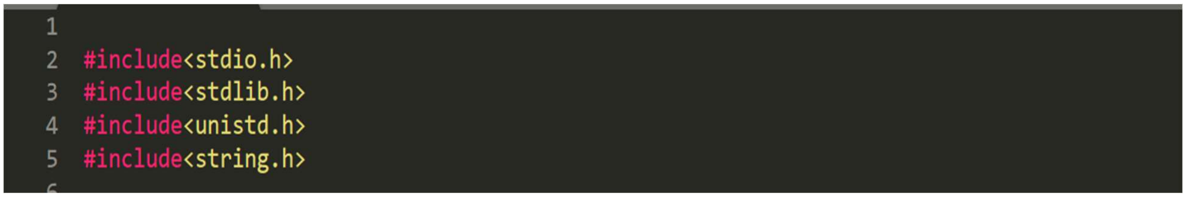
Complexity:

In worst case, the program runs with time complexity of **O(n^2)** and in best case, the program runs with time complexity **O(1)**.

The program does not require any storage of data and thus the space complexity of the program is **O(1).**

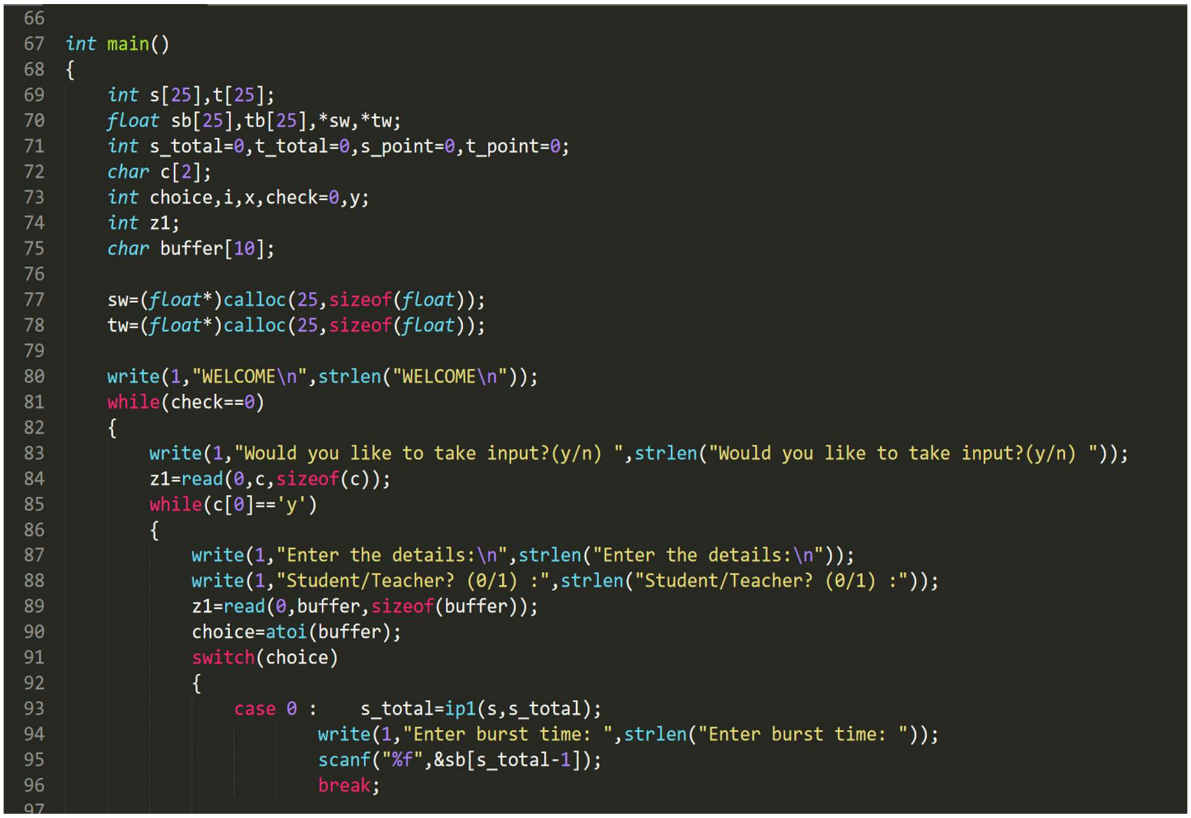
Code:

Header files –

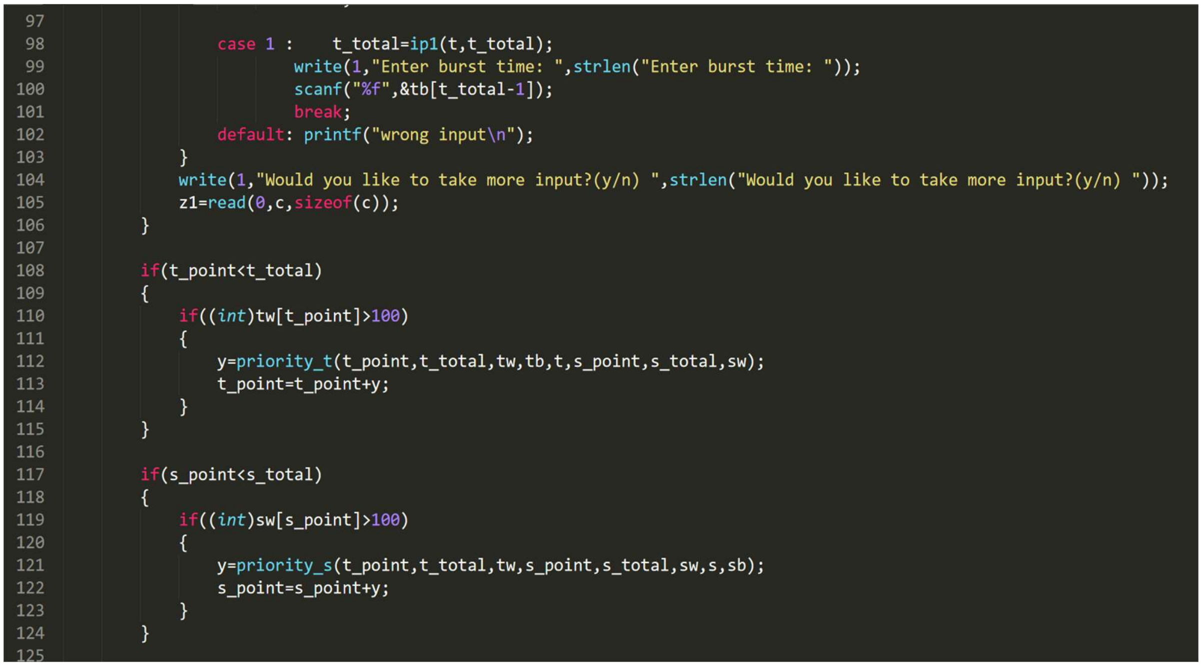


```c
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<string.h>
```

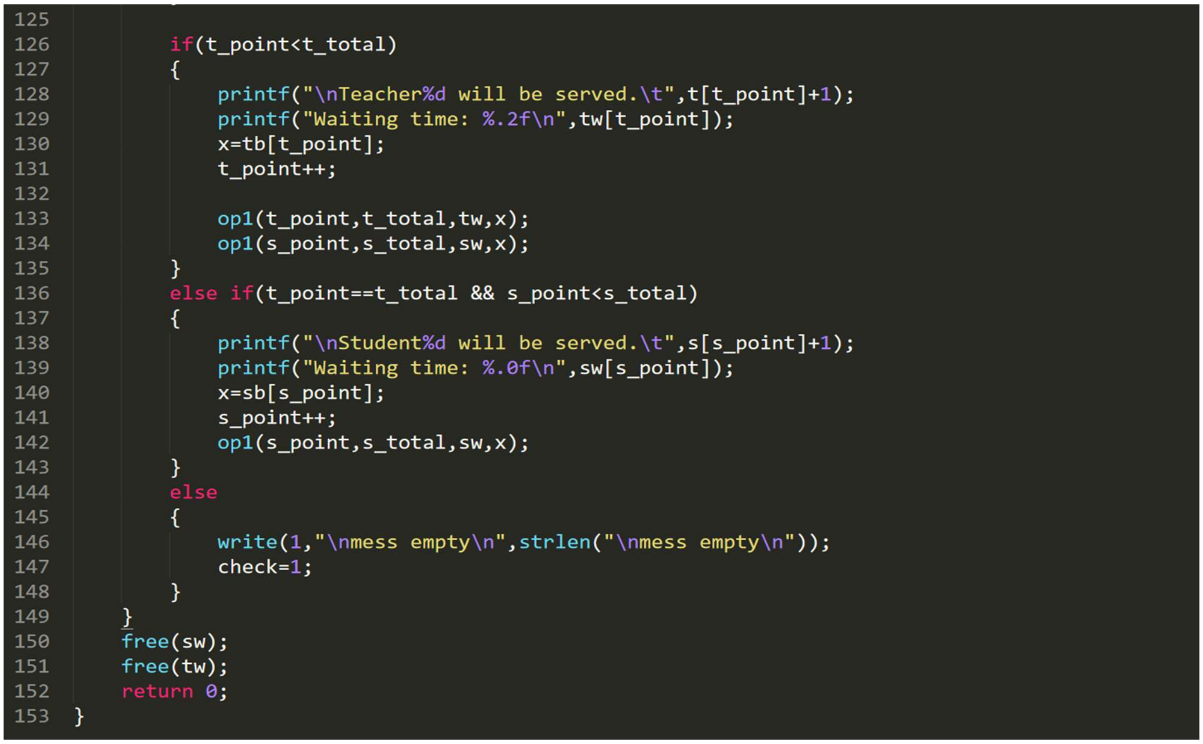
Main function –



```c
int main()
{
    int s[25],t[25];
    float sb[25],tb[25],*sw,*tw;
    int s_total=0,t_total=0,s_point=0,t_point=0;
    char c[2];
    int choice,i,x,check=0,y;
    int z1;
    char buffer[10];

    sw=(float*)calloc(25,sizeof(float));
    tw=(float*)calloc(25,sizeof(float));

    write(1,"WELCOME\n",strlen("WELCOME\n"));
    while(check==0)
    {
        write(1,"Would you like to take input?(y/n) ",strlen("Would you like to take input?(y/n) "));
        z1=read(0,c,sizeof(c));
        while(c[0]=='y')
        {
            write(1,"Enter the details:\n",strlen("Enter the details:\n"));
            write(1,"Student/Teacher? (0/1) :",strlen("Student/Teacher? (0/1) :"));
            z1=read(0,buffer,sizeof(buffer));
            choice=atoi(buffer);
            switch(choice)
            {
                case 0 :    s_total=ip1(s,s_total);
                        write(1,"Enter burst time: ",strlen("Enter burst time: "));
                        scanf("%f",&sb[s_total-1]);
                        break;
```

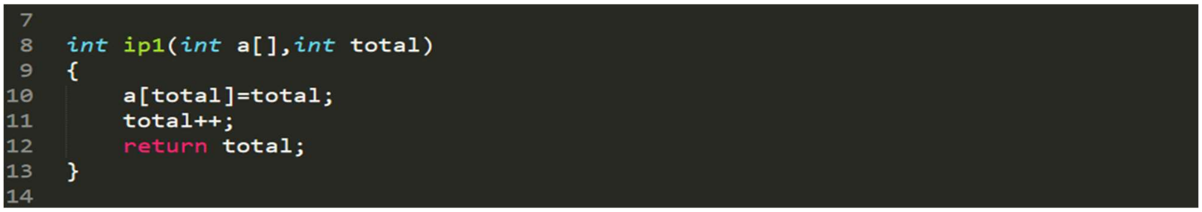


```c

                case 1 :    t_total=ip1(t,t_total);
                        write(1,"Enter burst time: ",strlen("Enter burst time: "));
                        scanf("%f",&tb[t_total-1]);
                        break;
                default: printf("wrong input\n");
            }
            write(1,"Would you like to take more input?(y/n) ",strlen("Would you like to take more input?(y/n) "));
            z1=read(0,c,sizeof(c));
        }

        if(t_point<t_total)
        {
            if((int)tw[t_point]>100)
            {
                y=priority_t(t_point,t_total,tw,tb,t,s_point,s_total,sw);
                t_point=t_point+y;
            }
        }

        if(s_point<s_total)
        {
            if((int)sw[s_point]>100)
            {
                y=priority_s(t_point,t_total,tw,s_point,s_total,sw,s,sb);
                s_point=s_point+y;
            }
        }
```



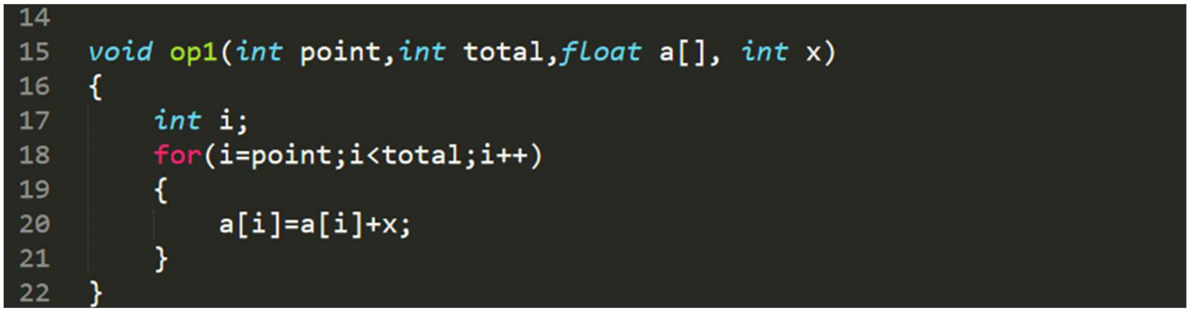
```c

        if(t_point<t_total)
        {
            printf("\nTeacher%d will be served.\t",t[t_point]+1);
            printf("Waiting time: %.2f\n",tw[t_point]);
            x=tb[t_point];
            t_point++;

            op1(t_point,t_total,tw,x);
            op1(s_point,s_total,sw,x);
        }
        else if(t_point==t_total && s_point<s_total)
        {
            printf("\nStudent%d will be served.\t",s[s_point]+1);
            printf("Waiting time: %.0f\n",sw[s_point]);
            x=sb[s_point];
            s_point++;
            op1(s_point,s_total,sw,x);
        }
        else
        {
            write(1,"\nmess empty\n",strlen("\nmess empty\n"));
            check=1;
        }
    }
    free(sw);
    free(tw);
    return 0;
}
```

Teacher/Student input function



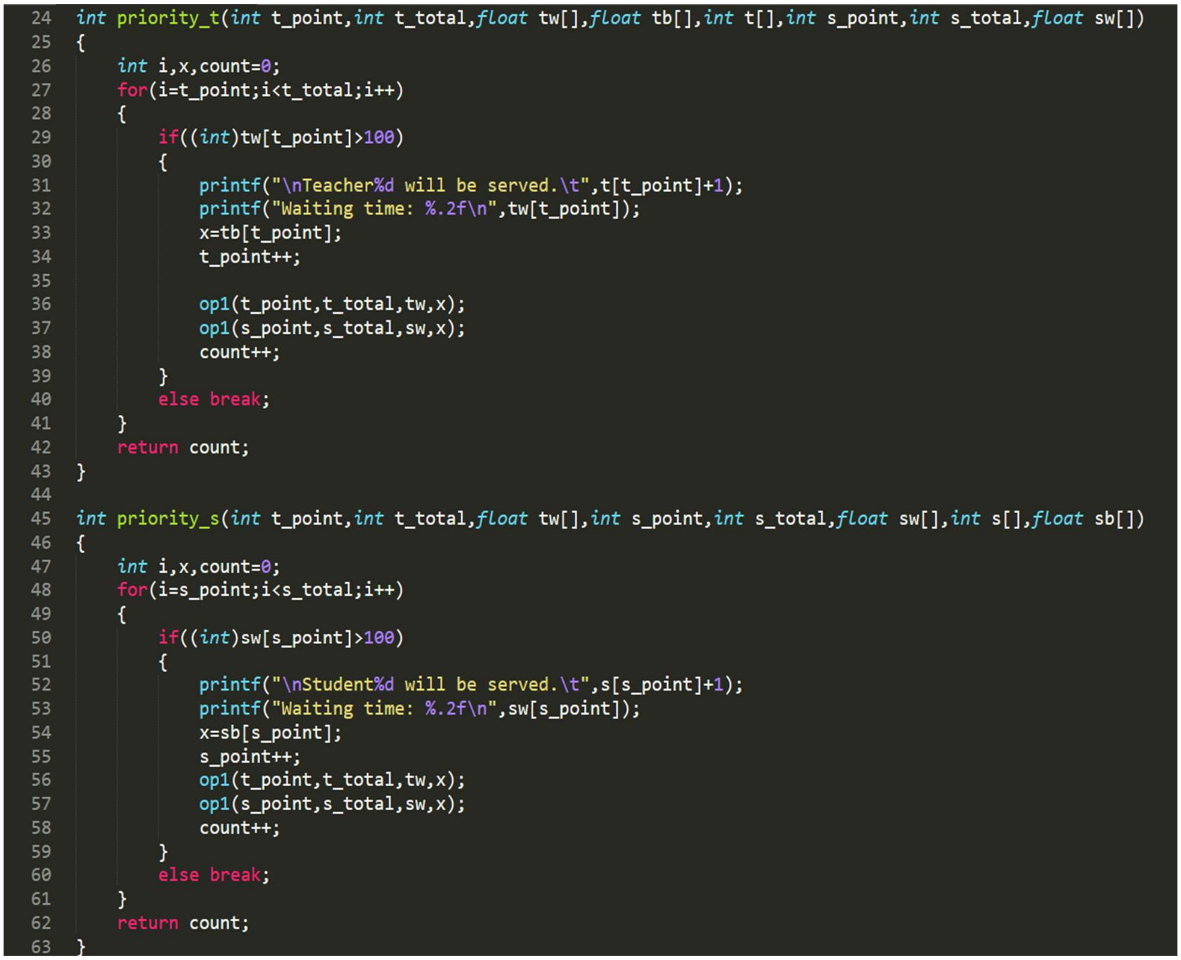
```c

int ip1(int a[],int total)
{
    a[total]=total;
    total++;
    return total;
}

```

Function to add waiting times



```c

void op1(int point,int total,float a[], int x)
{
    int i;
    for(i=point;i<total;i++)
    {
        a[i]=a[i]+x;
    }
}
```

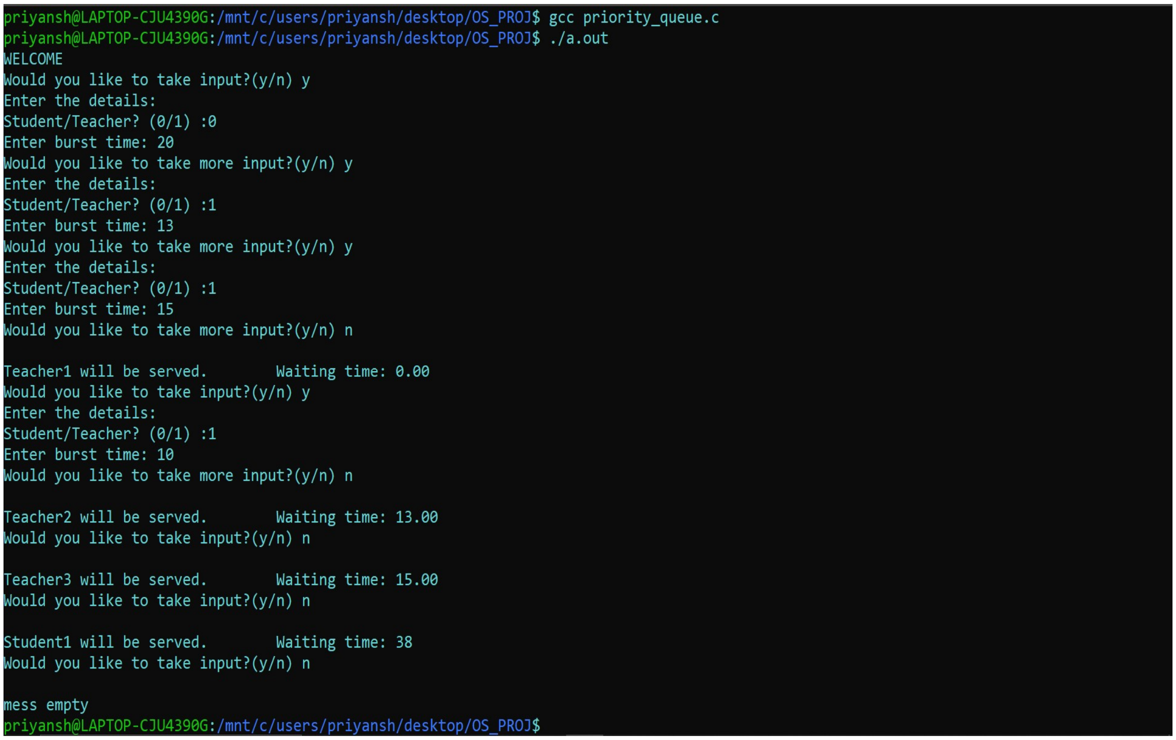
Function to increase priority



```c
int priority_t(int t_point,int t_total,float tw[],float tb[],int t[],int s_point,int s_total,float sw[])
{
    int i,x,count=0;
    for(i=t_point;i<t_total;i++)
    {
        if((int)tw[t_point]>100)
        {
            printf("\nTeacher%d will be served.\t",t[t_point]+1);
            printf("Waiting time: %.2f\n",tw[t_point]);
            x=tb[t_point];
            t_point++;

            op1(t_point,t_total,tw,x);
            op1(s_point,s_total,sw,x);
            count++;
        }
        else break;
    }
    return count;
}

int priority_s(int t_point,int t_total,float tw[],int s_point,int s_total,float sw[],int s[],float sb[])
{
    int i,x,count=0;
    for(i=s_point;i<s_total;i++)
    {
        if((int)sw[s_point]>100)
        {
            printf("\nStudent%d will be served.\t",s[s_point]+1);
            printf("Waiting time: %.2f\n",sw[s_point]);
            x=sb[s_point];
            s_point++;
            op1(t_point,t_total,tw,x);
            op1(s_point,s_total,sw,x);
            count++;
        }
        else break;
    }
    return count;
}
```

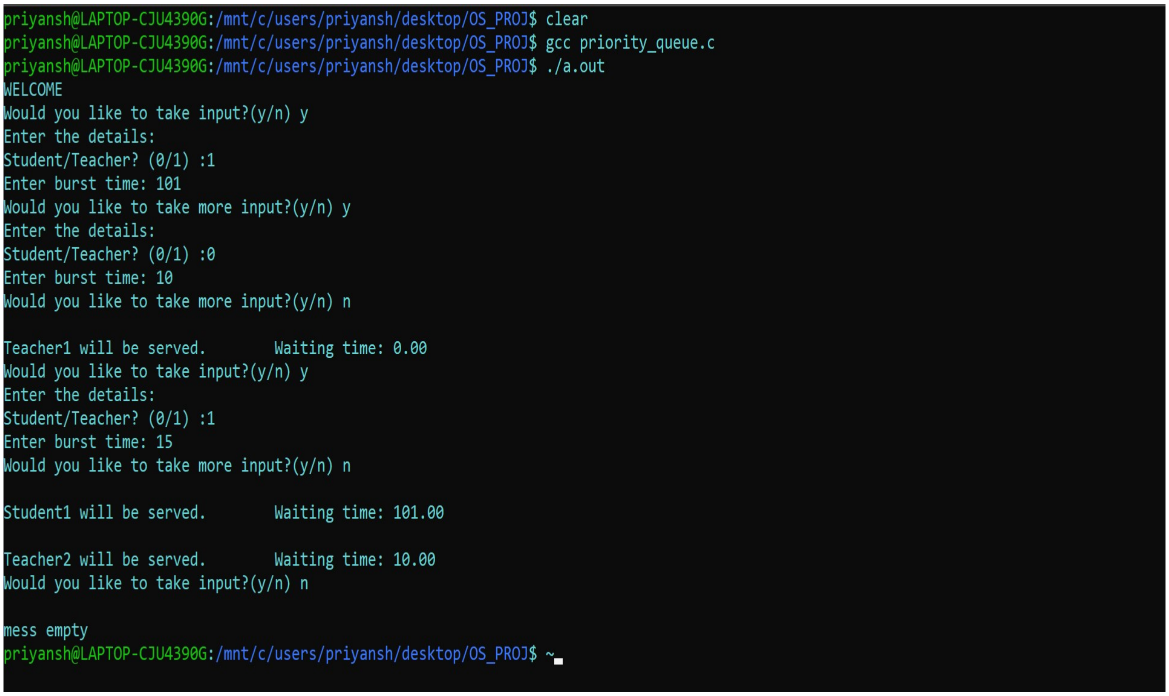
Output

Output showing teacher prioritized ahead of student



```
priyansh@LAPTOP-CJU4390G:/mnt/c/users/priyansh/desktop/OS_PROJ$ gcc priority_queue.c
priyansh@LAPTOP-CJU4390G:/mnt/c/users/priyansh/desktop/OS_PROJ$ ./a.out
WELCOME
Would you like to take input?(y/n) y
Enter the details:
Student/Teacher? (0/1) :0
Enter burst time: 20
Would you like to take more input?(y/n) y
Enter the details:
Student/Teacher? (0/1) :1
Enter burst time: 13
Would you like to take more input?(y/n) y
Enter the details:
Student/Teacher? (0/1) :1
Enter burst time: 15
Would you like to take more input?(y/n) n

Teacher1 will be served.        Waiting time: 0.00
Would you like to take input?(y/n) y
Enter the details:
Student/Teacher? (0/1) :1
Enter burst time: 10
Would you like to take more input?(y/n) n

Teacher2 will be served.        Waiting time: 13.00
Would you like to take input?(y/n) n

Teacher3 will be served.        Waiting time: 15.00
Would you like to take input?(y/n) n

Student1 will be served.        Waiting time: 38
Would you like to take input?(y/n) n

mess empty
priyansh@LAPTOP-CJU4390G:/mnt/c/users/priyansh/desktop/OS_PROJ$
```

Output showing student prioritized ahead of teacher (waiting time>100)



```
priyansh@LAPTOP-CJU4390G:/mnt/c/users/priyansh/desktop/OS_PROJ$ clear
priyansh@LAPTOP-CJU4390G:/mnt/c/users/priyansh/desktop/OS_PROJ$ gcc priority_queue.c
priyansh@LAPTOP-CJU4390G:/mnt/c/users/priyansh/desktop/OS_PROJ$ ./a.out
WELCOME
Would you like to take input?(y/n) y
Enter the details:
Student/Teacher? (0/1) :1
Enter burst time: 101
Would you like to take more input?(y/n) y
Enter the details:
Student/Teacher? (0/1) :0
Enter burst time: 10
Would you like to take more input?(y/n) n

Teacher1 will be served.        Waiting time: 0.00
Would you like to take input?(y/n) y
Enter the details:
Student/Teacher? (0/1) :1
Enter burst time: 15
Would you like to take more input?(y/n) n

Student1 will be served.        Waiting time: 101.00

Teacher2 will be served.        Waiting time: 10.00
Would you like to take input?(y/n) n

mess empty
priyansh@LAPTOP-CJU4390G:/mnt/c/users/priyansh/desktop/OS_PROJ$ ~
```