DS Assignment 6

Implement double ended priority queue The library is managing request for Book in first come first serve manner. Teacher's requests are given higher priority than that of students. Design the task of library using the most suitable Data Structure. Sample Input:(Assuming execution of every request is 1 sec)

Student request at time in seconds: 1,4,6,7,8

Teacher's request at time in seconds: 1,3,5,7,10

Source Code:

```
#include<stdio.h>

#include<stdlib.h>

#include<stdbool.h>

#define SIZE 100

int queue[SIZE];

int front=0;

int rear=0;

bool isEmpty();

bool isFull();
```

```
char dequeueFront();
void enqueueFront(char x);
void enqueueRear(char x);
int space();
int main(){
    int stu=5, tea=5, i,j,k, s[stu], t[tea];
    printf("Enter the Student request in seconds ");
    for(i=0;i<stu;i++){
        scanf("%d",&s[i]);
    }
    printf("Enter the Teacher request in seconds ");
    for(j=0;j<tea;j++){
        scanf("%d",&t[j]);
    }
    printf("The Dequeue order is:\n");
    j=0,i=0,k=1;
    while(i<SIZE){
        if(i<stu && j<tea){
            if(s[i]==t[j] && s[i]<=k){
```

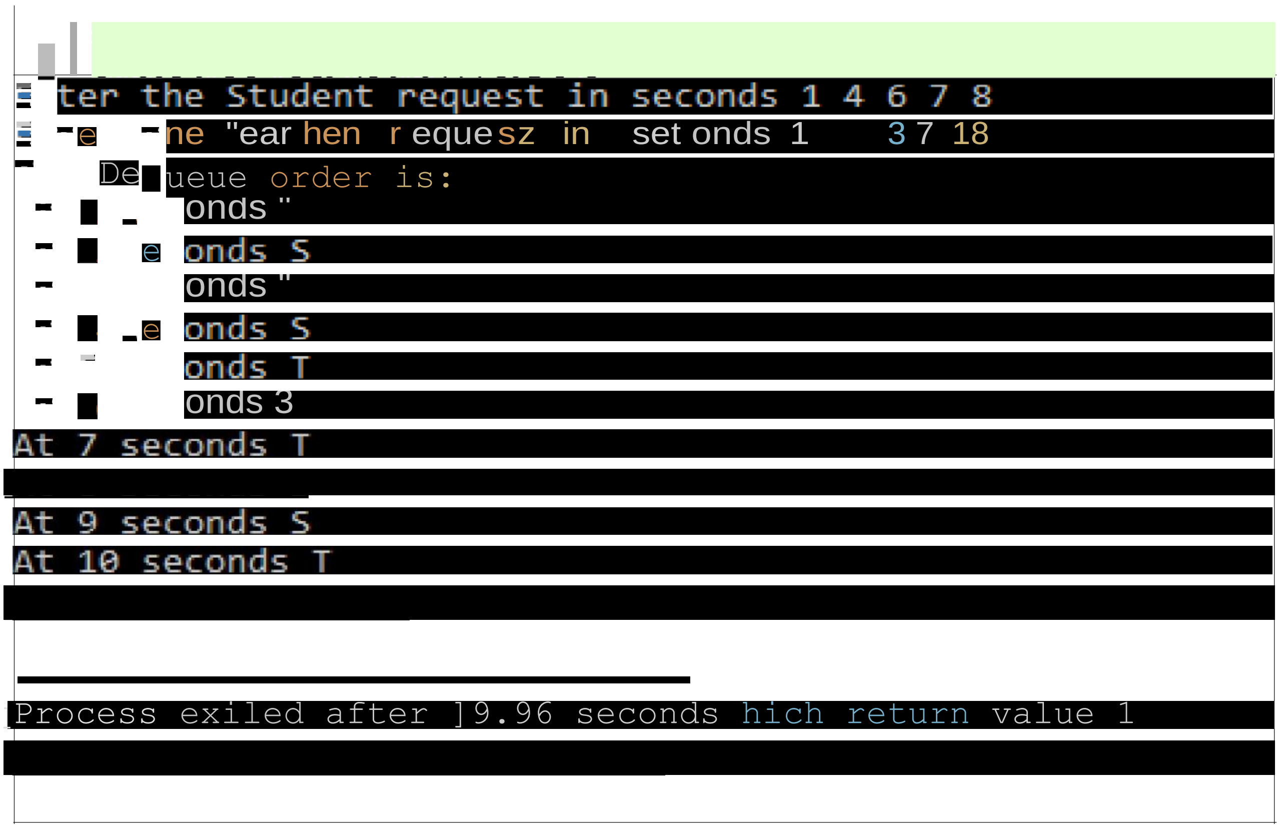
```
            enqueueFront('T');
            enqueueRear('S');
            i++;
            j++;
        }
        else if(s[i]<t[j] && s[i]==k){
            enqueueRear('S');
            i++;
        }
        else if(t[j]<s[i] && t[j]==k){
            enqueueFront('T');
            j++;
        }
    }
        else if(i==stu && t[j]==k){
            enqueueFront('T');
            j++;
        }
        else if(j==tea && s[i]==k){
```

```
                        enqueueRear('S');
                        i++;
                    }
            printf("At %d seconds %c \n",k,dequeueFront());
                k++;
            }
    }

    bool isEmpty(){
            return space()== 0;
    }

    bool isFull(){
            return space()==SIZE-1;
    }

    char dequeueFront(){
            if(isEmpty()){
                printf("Now Queue is empty.\n");
```

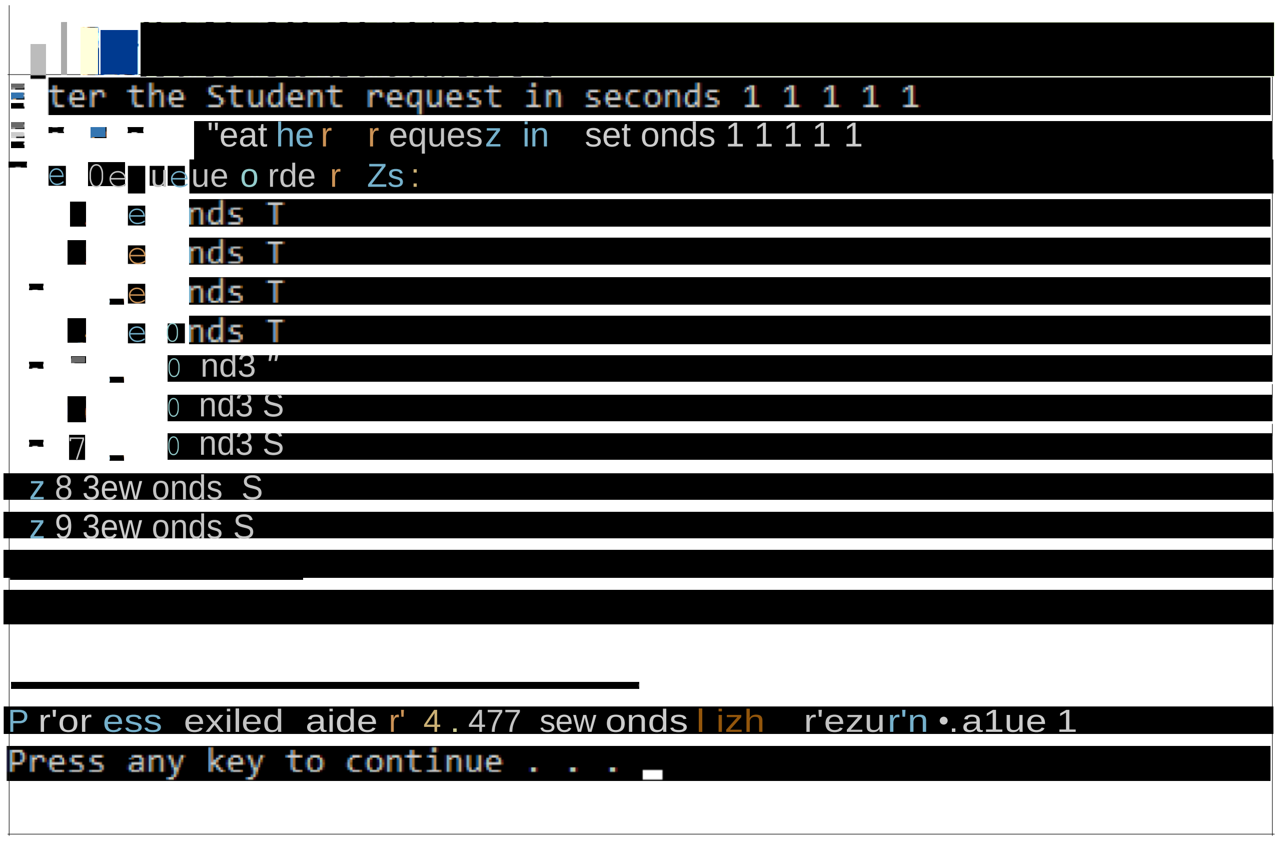
```
        exit(1);
    }
    else{
        char x = queue[front];
        queue[front]='\0';
        if(front==SIZE-1){
            front=0;
        }
        else
            front++;
        return x;
    }
}
int space(){
    if(front <= rear){
        return rear-front;
    }
    else{
        return rear- front +SIZE;
        }
}
void enqueueFront(char x){
        if(isFull()){
                printf("Queue is full!\n");
                return ;
        }
        else{
                if(front==0){
                front=SIZE-1;
                queue[front]=x;
        }
        else{
                front--;
                queue[front]=x;
        }
  }
}
```

```
void enqueueRear(char x){
        if(isFull()){
                printf("Queue is full!\n");
                return ;
        }
        else{
                if(rear==SIZE-1){
                        queue[rear]=x;
                        rear=0;
        }
        else{
                        queue[rear]=x;
                        rear++;
        }
  }
}
```

Output:



```
ter the Student request in seconds 1 4 6 7 8
e ne "ear hen r equesz in set onds 1 3 7 18
De ueue order is:
onds "
e onds S
onds "
e onds S
onds T
onds 3
At 7 seconds T

At 9 seconds S
At 10 seconds T


Process exiled after ]9.96 seconds hich return value 1
```



```
ter the Student request in seconds 1 1 1 1 1
"eat he r r equesz in set onds 1 1 1 1 1
e 0e ue ue o rde r Zs:
e nds T
e nds T
e nds T
e 0nds T
0 nd3 "
0 nd3 S
7 0 nd3 S
z 8 3ew onds S
z 9 3ew onds S


P r'or ess exiled aide r' 4 . 477 sew onds l izh r'ezur'n •.a1ue 1
Press any key to continue . . .
```