

Institute of Engineering Science and Technology, Shibpur.

Department of Information Technology

Data Structure Laboratory 2021

BATCH- HY

Assignment – 4

<https://github.com/Tathagata-Ghosh-Developer/Lab-Assignment-3rd-Semester>

```
/* Data Structure Lab Assignment 2021
Tathagata Ghosh --- 2020ITB065 ---- HY
01/10/2021 */

/*1. The priority queue will be a collection of strings. Lexicographically smaller
strings should be considered higher priority than lexicographically larger
ones. For example, "ping" is higher priority than "pong", regardless of
insertion order.
Design the following functions:
i) enqueue is used to insert a new element to the priority queue.
ii) extractMin returns the value of highest priority (i.e., lexicographically
smallest)
element in the queue and removes it.
iii) merge unifies the queues and returns their union as a new queue.*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct Queue
{
    int size, f, r;
    char **arr;
};

struct Queue *setQueue()
{
    struct Queue *q = (struct Queue *)malloc(sizeof(struct Queue));
    q->f = q->r = -1;
    q->size = 10;
    q->arr = (char **)malloc(q->size * sizeof(char *));
```

```

    for (int i = 0; i < q->size; i++)
    {
        q->arr[i] = (char *)malloc(100 * sizeof(char));
    }
    return q;
}

void display(struct Queue *q)
{
    for (int i = 0; i <= q->r; i++)
    {
        printf("%s\n", q->arr[i]);
    }
}

void enQueue(struct Queue *q, char *string)
{
    if (q->r == q->size - 1)
        printf("This Queue is full\n");
    else
    {
        q->r++;
        strcpy(q->arr[q->r], string);

        char temp[100];
        for (int j = q->r - 1; j > -1 && strcmp(q->arr[j], string) == 1; j--)
        {
            strcpy(temp, q->arr[j]);
            strcpy(q->arr[j], q->arr[j + 1]);
            strcpy(q->arr[j + 1], temp);
        }

        printf("%s is successfully enQueued.\n", string);
    }
}

char *extractMin(struct Queue *q)
{
    char *a = (char *)malloc(100 * sizeof(char));
    strcpy(a, " - X - ");

    if (q->f == q->r)
        printf("This Queue is empty\n");
    else
    {
        q->f++;
        strcpy(a, q->arr[q->f]);
    }
}

```

```

    return a;
}

struct Queue *merge(struct Queue *q)
{
    struct Queue *new = setQueue();
    while (q->f != q->r)
    {
        strcpy(new->arr[++new->r], extractMin(q));
    }
    return new;
}

int main()
{
    struct Queue *strings = setQueue();
    int numOfWorks;
    char word[100];

    printf("Enter the number of strings or words (0<=10) you want to enqueue:
");
    scanf("%d", &numOfWorks);
    printf("\nEnter the words:\n");
    for (int i = 0; i < numOfWorks; i++)
    {
        printf("Word %d: ", i + 1);
        scanf("%s", word);
        enqueue(strings, word);
    }

    struct Queue *newStrings = merge(strings);

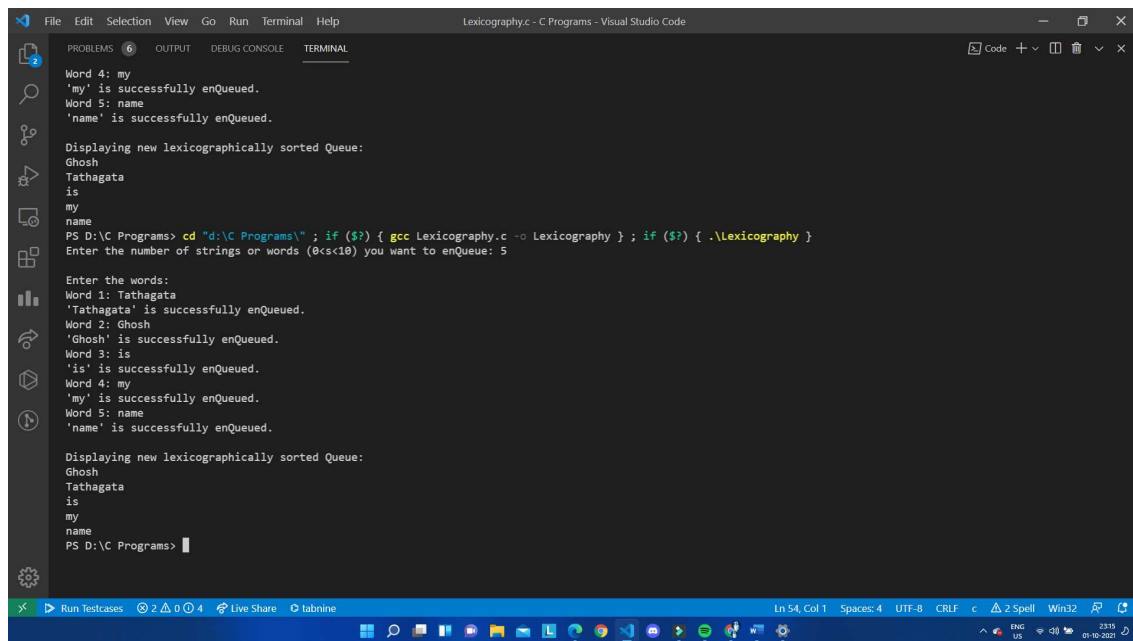
    free(strings);

    printf("\nDisplaying new lexicographically sorted Queue:\n");
    display(newStrings);

    return 0;
}

```

Output:



```
File Edit Selection View Go Run Terminal Help
Lexicography.c - C Programs - Visual Studio Code

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL

Word 4: my
'my' is successfully enQueued.
Word 5: name
'name' is successfully enQueued.

Displaying new lexicographically sorted Queue:
Ghosh
Tathagata
is
my
name

PS D:\C Programs> cd "d:\C Programs\" ; if ($?) { gcc Lexicography.c -o Lexicography } ; if ($?) { .\Lexicography }
Enter the number of strings or words (0<=10) you want to enQueue: 5

Enter the words:
Word 1: Tathagata
'Tathagata' is successfully enQueued.
Word 2: Ghosh
'Ghosh' is successfully enQueued.
Word 3: is
'is' is successfully enQueued.
Word 4: my
'my' is successfully enQueued.
Word 5: name
'name' is successfully enQueued.

Displaying new lexicographically sorted Queue:
Ghosh
Tathagata
is
my
name
PS D:\C Programs> |
```

```
/* Data Structure Lab Assignment 2021
Tathagata Ghosh --- 2020ITB065 ---- HY
01/10/2021 */

/*2. In road traffic maintenance we know the traffic signal is used. Depending
on the
real time traffic a particular signal is made on/off for a few seconds. If we
have
the traffic data available then write a program in C to manage the traffic sig
nal.
Consider there is a 5 road connector and you have to provide the signal for ea
ch
of the roads. Generate the time for each traffic between 10 to 20 sec randomly
and provide the signal for each of the roads one by one.
If for any reason one road is to be blocked then block that road and provide t
he
signal for remaining roads as earlier*/

#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include <time.h>
#define roads 5

struct Queue
{
    int size, f, r, count;
    char *arr;
};
```

```

struct Queue *setQueue()
{
    struct Queue *q = (struct Queue *)malloc(sizeof(struct Queue));
    q->size = roads, q->count = 0;
    q->f = 0, q->r = -1;
    q->arr = (char *)malloc(q->size * sizeof(char));
    return q;
}

void enqueue(struct Queue *q, char val)
{
    if (q->count != q->size)
    {
        do
        {
            q->r = (q->r + 1) % q->size;
        } while (q->arr[q->r] == 'B');
        q->arr[q->r] = val;
        q->count++;
    }
}

char dequeueF(struct Queue *q)
{
    if (q->count)
    {
        int index;
        do
        {
            index = q->f++;
            q->f %= q->size;
        } while (q->arr[index] == 'B');
        q->count--;
    }
}

void dequeueR(struct Queue *q)
{
    if (q->count)
    {
        int index;
        do
        {
            index = q->r--;
            q->r = (q->r == -1) ? q->size - 1 : q->r;
        } while (q->arr[index] == 'B');
        q->count--;
    }
}

```

```

}

void display(struct Queue *q)
{
    printf("\nRoad:\t");
    for (int i = 0; i < q->size; i++)
    {
        printf("%d\t", i + 1);
    }
    printf("\nSignal:\t");
    for (int i = 0; i < q->size; i++)
    {
        switch (q->arr[i])
        {
            case 'R':
                printf("Red\t");
                break;
            case 'Y':
                printf("Yellow\t");
                break;
            case 'G':
                printf("Green\t");
                break;
            case 'B':
                printf("Block\t");
                break;
        }
    }
    printf("\n");
}

int main()
{
    struct Queue *q = setQueue();
    while (q->count != q->size)
    {
        enqueue(q, 'R');
    }

    int blockedPath;
    printf("To Block any road, enter Road No. (1<=n<=5), else enter 0: ");
    scanf("%d", &blockedPath);
    (blockedPath) ? q->arr[blockedPath - 1] = 'B' : 1;

    for (int i = 1; i; i++)
    {
        dequeueR(q);
        dequeueR(q);
    }
}

```

```

        deQueueR(q);
        enqueue(q, 'R');
        enqueue(q, 'Y');
        enqueue(q, 'G');

        deQueueF(q);
        enqueue(q, 'Y');

        display(q);

        srand(time(NULL));
        int time = 10 + rand() % 11;
        sleep(time);
    }
    return 0;
}

```

The screenshot shows the Visual Studio Code interface with a C program named 'Traffic.c' open. The program is a simulation of a traffic light system with five roads. The terminal output shows the program's execution, where the user enters road numbers (1-5) and the program displays the corresponding traffic light colors (Yellow, Red, Green, Block, Yellow, Green, Red, Yellow, Red, Yellow, Green, Yellow, Red, Yellow, Green, Yellow, Red, Yellow, Green, Yellow, Red).

Visual Studio Code - Traffic.c - C Programs - Visual Studio Code

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL

PS D:\C Programs> cd "d:\C Programs\"; if (\$?) { gcc Traffic.c -o Traffic }; if (\$?) { .\Traffic }

To Block any road, enter Road No. (1<=n<=5), else enter 0: 3

Road: 1 2 3 4 5  
Signal: Yellow Red Block Yellow Green

Road: 1 2 3 4 5  
Signal: Green Yellow Block Red Yellow

Road: 1 2 3 4 5  
Signal: Yellow Green Block Yellow Red

Road: 1 2 3 4 5  
Signal: Red Yellow Block Green Yellow

Road: 1 2 3 4 5  
Signal: Yellow Red Block Yellow Green

Road: 1 2 3 4 5  
Signal: Green Yellow Block Red Yellow

Road: 1 2 3 4 5  
Signal: Yellow Green Block Yellow Red

Road: 1 2 3 4 5  
Signal: Red Yellow Block Green Yellow

Road: 1 2 3 4 5  
Signal: Yellow Red Block Yellow Green

Road: 1 2 3 4 5  
Signal: Green Yellow Block Red Yellow

Road: 1 2 3 4 5  
Signal: Yellow Green Block Yellow Red

Ln 14, Col 18 Spaces: 4 UTF-8 CRLF 2 Spell Win32 7345 09-10-2021

