SISTER NIVEDITA UNIVERSITY

School of Engineering

Subject Name:	DATA STRUCTURES AND ALGORITHMS
Subject Code:	
Department:	B.TECH IN COMPUTER SCIENCE AND ENGINEERING

Name:	CHANDREYEE SHOME
Enrolment Number:	2212200001166
Registration Number:	220010415539
Semester:	2nd
Academic Year:	1st (2022 - 2023)

ASSIGNMENT 4

Q 1. Implement a Circular Queue by using array then do the enqueue and dequeue operation.

CODE:

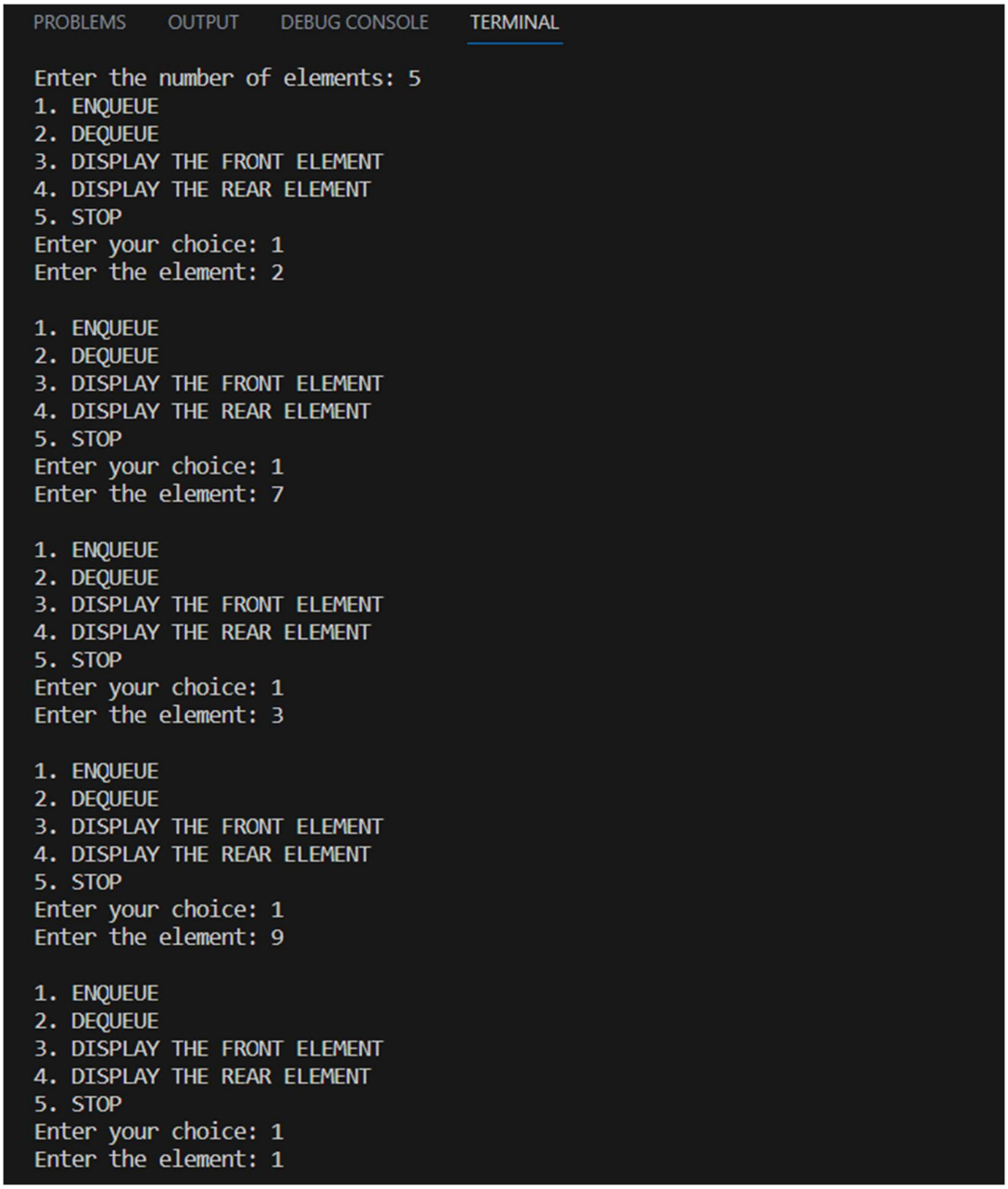
```
//CIRCULAR QUEUE PROGRAM

//header files
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

//user defined datatype Queue
typedef struct Queue
{
    int front;
    int rear;
    int capacity;
    int size;
    int *arr;
}Q;

//function decleration
void init(Q *q,int cap);
int isfull(Q *q);
int EnQueue(Q *q, int item);
int isempty(Q *q);
int DeQueue(Q *q);
int Efront(Q *q);
int Erear(Q *q);

//initializing the parts of queue
void init(Q *q, int cap)
{
    q->front = q->rear = -1;
    q->size = 0;
    q->capacity = cap;
    q->arr = (int*)malloc(cap * sizeof(int));
}

int main()
{
    Q queue;
    int size;
    system ("cls");
    printf("Enter the number of elements: ");
    scanf("%d",&size);
    init(&queue,size);
```

```
    int choice=0, item;

    do
    {
        printf("1. ENQUEUE\n2. DEQUEUE\n3. DISPLAY THE FRONT ELEMENT\n4.
DISPLAY THE REAR ELEMENT\n5. STOP\nEnter your choice: ");
        scanf("%d",&choice);
        switch(choice)
        {
            case 1:
                {
                    printf("Enter the element: ");
                    scanf("%d",&item);
                    int EQ = EnQueue(&queue, item);
                    if(EQ == 0)
                    {
                        printf("\nQueue Overflow\n");
                    }
                }
                break;
            case 2:
                {
                    int DQ = DeQueue(&queue);
                    if(DQ==404)
                    {
                        printf("\nQueue Underflow\n");
                    }
                    else
                    {
                        printf("\nThe Dequeued element: %d\n",DQ);
                    }
                }
                break;
            case 3:
                    printf("\nThe Front element: ");
                    printf("%d\n",Efront(&queue));
                    break;
            case 4:
                    printf("\nThe Rear element: ");
                    printf("%d\n",Erear(&queue));
                    break;
            case 5:
                    break;
            default:
                    printf("\nWrong Choice. Try Again.\n");
        }
        printf("\n");
    }while(choice!=5);
```

```
}

//function to check if queue is full
int isfull(Q *q)
{
    return(q->rear+1 == q->front || q->front==0 && q->rear==q->capacity-1);
}

//function to insert element in queue
int EnQueue(Q *q, int item)
{
    if(!isfull(q))
    {
        if(q->front == -1 && q->rear == -1)
        {
            q->front=0;
            q->rear=0;
        }
        else if(q->rear == q->capacity-1)
        {
            q->rear=0;
        }
        else
        {
            q->rear = q->rear+1;
        }
        q->arr[q->rear] = item;
        q->size = q->size+1;
        return 1;
    }
    else
    return 0;
}

//function to check if queue is empty
int isempty(Q *q)
{
    return(q->front==-1);
}

//function to delete elements from queue
int DeQueue(Q *q)
{
    int item;
    if(!isempty(q))
    {
        item = q->arr[q->front];
        q->size = q->size-1;
```

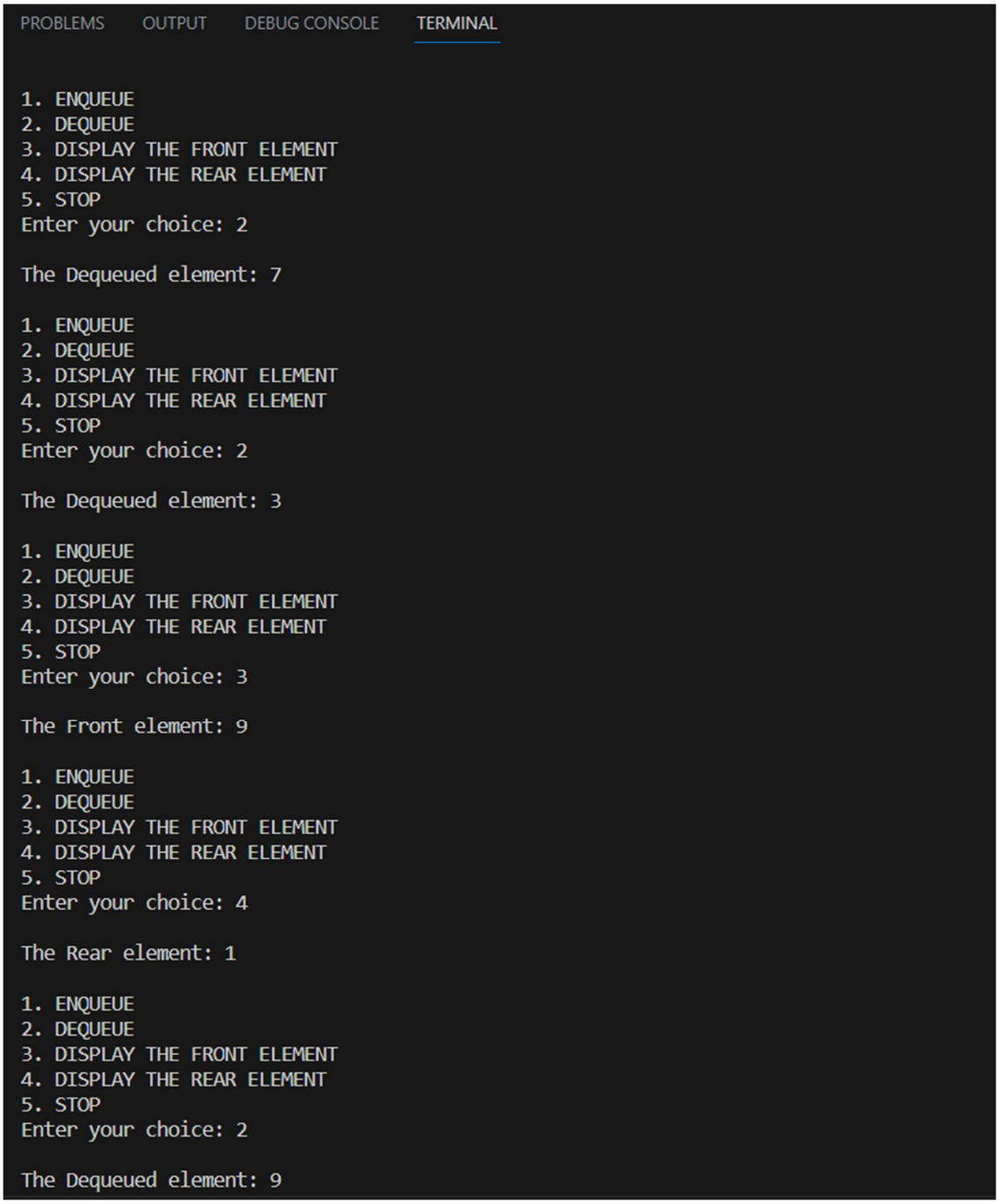
```
        if(q->front == q->rear)
        {
            q->front=q->rear=-1;
        }
        else if(q->front == q->capacity-1)
        {
            q->front = 0;
        }
        else
        {
            q->front = q->front+1;
        }
        return item;
    }
    else
    {
        return 404;
    }
}

//function to display front element of queue
int Efront(Q *q)
{
    return(q->arr[q->front]);
}

//function to display rear element of queue
int Erear(Q *q)
{
    return(q->arr[q->rear]);
}
```

OUTPUT:


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
Enter the number of elements: 5
1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 1
Enter the element: 2
1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 1
Enter the element: 7
1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 1
Enter the element: 3
1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 1
Enter the element: 9
1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 1
Enter the element: 1

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 1
Enter the element: 0

Queue Overflow

1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 3

The Front element: 2

1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 4

The Rear element: 1

1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 2

The Dequeued element: 2
```


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 2
The Dequeued element: 7
1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 2
The Dequeued element: 3
1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 3
The Front element: 9
1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 4
The Rear element: 1
1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 2
The Dequeued element: 9

```
1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 2

The Dequeued element: 1

1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 2

Queue Underflow

1. ENQUEUE
2. DEQUEUE
3. DISPLAY THE FRONT ELEMENT
4. DISPLAY THE REAR ELEMENT
5. STOP
Enter your choice: 5

PS C:\Users\CHANDREYEE SHOME\Desktop\C C++>
```