Shrihari K Mangle
CSB2-74   12010517

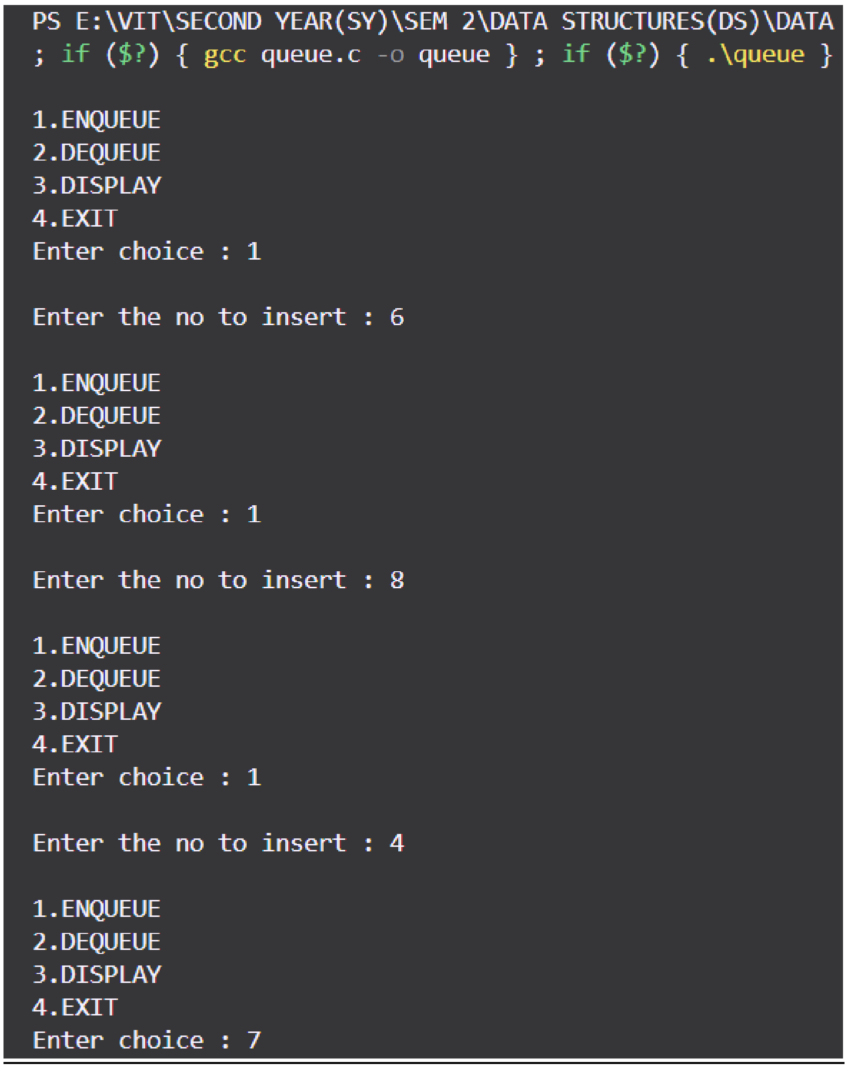
LAB-5

5. Implement Linear, Double Ended and Circular Queue. (Note: Linear Queue has to be implemented using Linked Lists.)

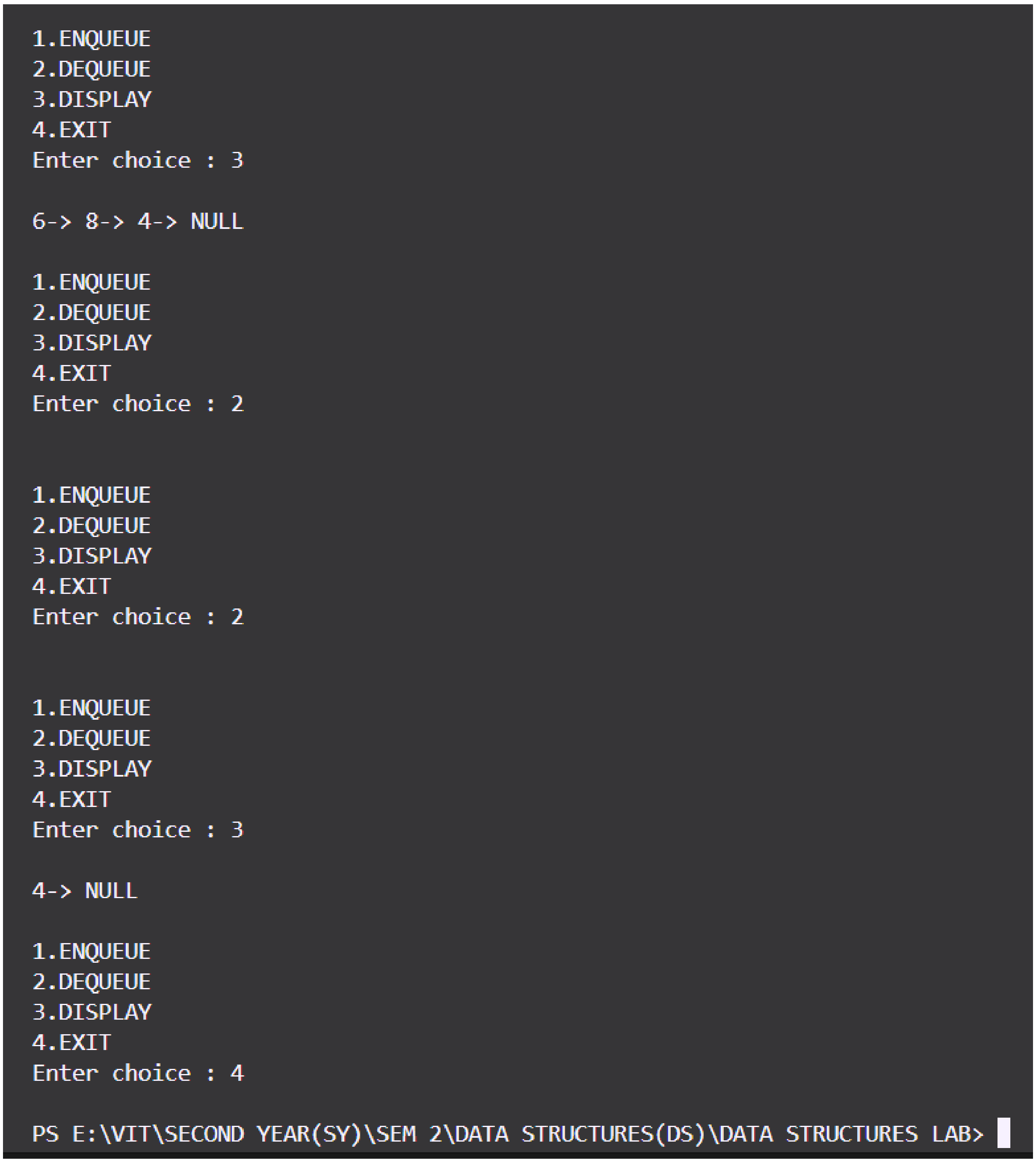
LINEAR QUEUE USING LL :

```c
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

typedef struct node
{
    int data;
    struct node *next;
} node;

node *front = NULL, *rear = NULL;

void enqueue(int x)
{
    if (rear == NULL)
    {
        rear = (node *)malloc(sizeof(node));
        rear->next = NULL;
        rear->data = x;
        front = rear;
    }
    else
    {
        node *temp = (node *)malloc(sizeof(node));
        temp->next = NULL;
        temp->data = x;
        rear->next = temp;
        rear = temp;
    }
}

void dequeue()
{
    if (front == NULL)
    {
        printf("Queue is empty\n");
        return;
    }
    else if (front->next != NULL)
    {
        node *temp = front;
        front = front->next;
        free(temp);
    }
    else  // ONLY ONE NODE REMAINING
    {
        printf("%d\n", front->data);
        free(front);
        front = rear = NULL;
    }
}

void display()
{
    node *temp = front;
    if (front == NULL)
    {
        printf("Queue is empty\n");
        return;
    }
    else
    {
        while (temp != NULL)
        {
            printf("%d-> ", temp->data);
            temp = temp->next;
        }
        printf("NULL\n");
    }
}

int main()
{

    int no, choice;
    while (1)
    {
        printf("\n1.ENQUEUE\n2.DEQUEUE\n3.DISPLAY\n4.EXIT\n");
        printf("Enter choice : ");
        scanf("%d", &choice);
        printf("\n");

        switch (choice)
        {
        case 1:
            printf("Enter the no to insert : ");
            scanf("%d", &no);
            enqueue(no);
            break;

        case 2:
            dequeue();
            break;

        case 3:
            display();
            break;

        case 4:
            exit(0);

        default:
            break;
        }
    }

    return 0;
}
```

OUTPUT :



```
PS E:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA
; if ($?) { gcc queue.c -o queue } ; if ($?) { .\queue }

1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 1

Enter the no to insert : 6

1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 1

Enter the no to insert : 8

1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 1

Enter the no to insert : 4

1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 7
```



```
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 3

6-> 8-> 4-> NULL

1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 2


1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 2


1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 3

4-> NULL

1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 4

PS E:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA STRUCTURES LAB>
```

Shrihari K Mangle
CSB2-74 12010517

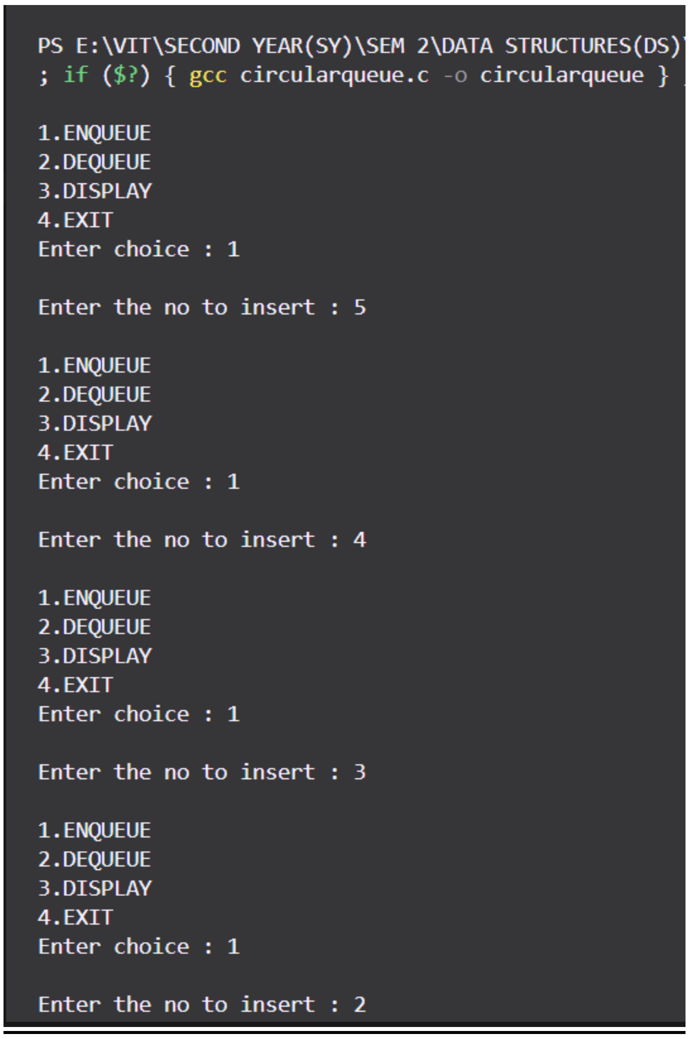
CIRCULAR QUEUE :

```c
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#define MAX 5

typedef struct queue
{
    int data[MAX];
    int rear, front;
} queue;

void init(queue *p)
{
    p->rear = -1;
    p->front = -1;
}

int empty(queue *p)
{
    if (p->rear == -1)
    {
        return 1;
    }
    return 0;
}

int full(queue *p)
{
    if ((p->rear + 1) % MAX == p->front)
    {
        return 1;
    }
    return 0;
}

void enqueue(queue *p, int x)
{
    if (full(p))
    {
        printf("Queue is full!!\n");
    }
    else if (empty(p))
    {
        p->front = p->rear = 0;
        p->data[p->rear] = x;
    }
    else
    {
        p->rear = (p->rear + 1) % MAX;
        p->data[p->rear] = x;
    }
}

int dequeue(queue *p)
{
    if (empty(p))
    {
        printf("Queue is empty\n");
    }

    int x;
    x = p->data[p->front];

    if (p->front == p->rear)
    {
        init(p);
    }
    else
    {
        p->front = (p->front + 1) % MAX;
    }

    return x;
}

void display(queue *p)
{
    if (empty(p))
    {
        printf("Empty!!!\n");
        return;
    }
    else
    {
        int i;
        i = p->front;
        while (i != p->rear)
        {
            printf("%d ", p->data[i]);
            i = (i + 1) % MAX;
        }
        printf("%d ", p->data[i]);
        printf("\n");
    }
}
```

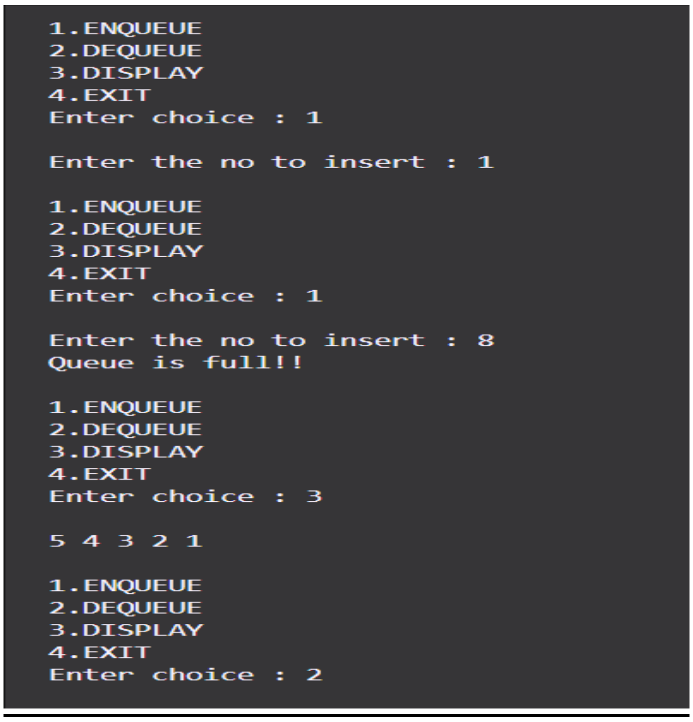
```c
int main()
{
    queue *p = (queue *)malloc(sizeof(queue));
    init(p);
    int no, choice;
    while (1)
    {
        printf("\n1.ENQUEUE\n2.DEQUEUE\n3.DISPLAY\n4.EXIT\n");
        printf("Enter choice : ");
        scanf("%d", &choice);
        printf("\n");

        switch (choice)
        {
        case 1:
            printf("Enter the no to insert : ");
            scanf("%d", &no);
            enqueue(p,no);
            break;

        case 2:

            dequeue(p);
            break;

        case 3:
            display(p);
            break;

        case 4:
            exit(0);



        default:
            break;
        }
    }

    return 0;
}
```

Shrihari K Mangle
CSB2-74 12010517

OUTPUT :

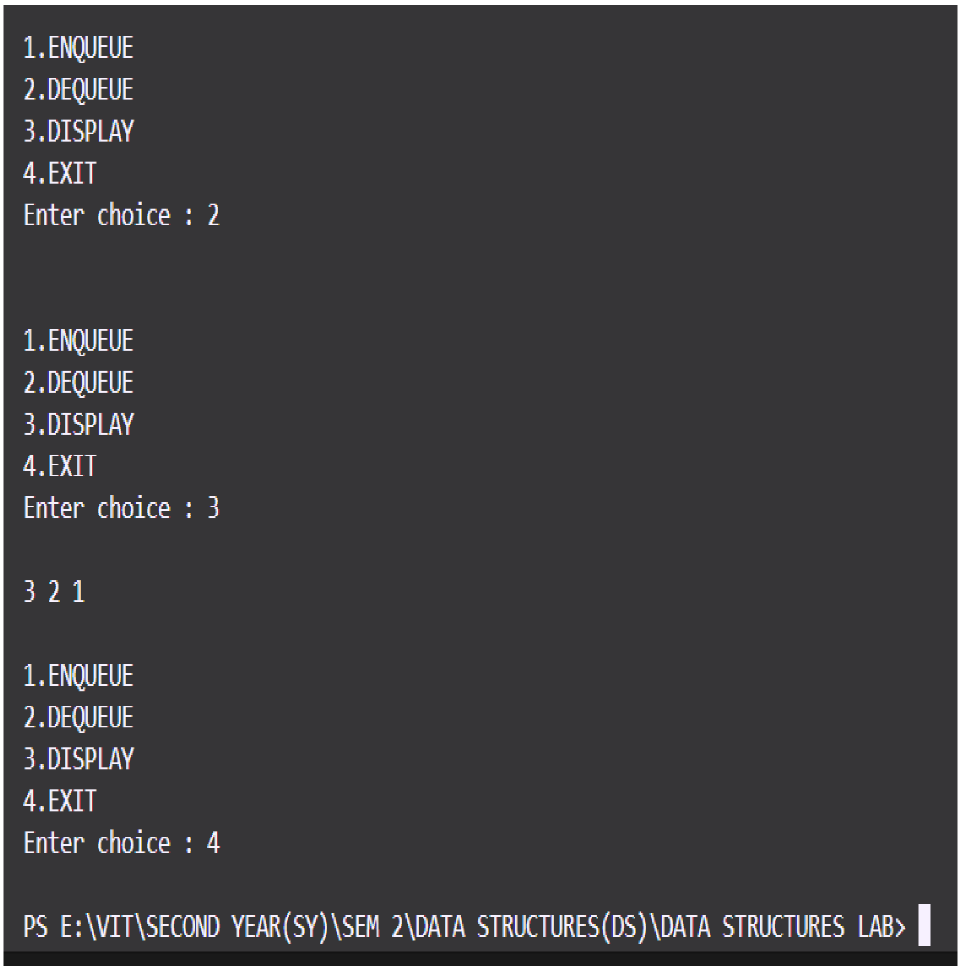


```
PS E:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)
; if ($?) { gcc circularqueue.c -o circularqueue }

1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 1

Enter the no to insert : 5

1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 1

Enter the no to insert : 4

1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 1

Enter the no to insert : 3

1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 1

Enter the no to insert : 2
```



```
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 1

Enter the no to insert : 1

1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 1

Enter the no to insert : 8
Queue is full!!

1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 3

5 4 3 2 1

1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 2
```

Shrihari K Mangle
CSB2-74   12010517



```
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 2


1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 3

3 2 1

1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT
Enter choice : 4

PS E:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA STRUCTURES LAB>
```

Shrihari K Mangle
CSB2-74 12010517

DOUBLE ENDED CIRCULAR QUEUE :

```c
#include <stdio.h>
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#define MAX 5
typedef struct queue
{
    int data[MAX];
    int front,rear;
}queue;
queue *q;
void init(queue *q);
void inqueuerear(queue *q,int x);
void inqueuefront(queue *q,int x);
int dequeuefront(queue *q);
int dequeuerear(queue *q);
void display(queue *q);
int isempty(queue *q);
int isfull(queue *q);
int main()
{
    queue *p=(queue *)malloc(sizeof(queue));
    init(p);
    while(1)
    {

    int ch,x,del;
    printf("Enter the choice: \n1.Insert using rear\n2.Insert using 
front\n3.Delete using rear\n4.Delete using front\n5.Display\n6.Exit");
    scanf("%d",&ch);
    switch(ch)
    {
        case 1:
            printf("Enter data to insert using rear ");
            scanf("%d",&x);
            inqueuerear(p,x);
            break;
        case 2:
            printf("Enter data to insert using front ");
            scanf("%d",&x);
            inqueuefront(p,x);
            break;
        case 3:
            del=dequeuerear(p);
            printf("%d is deleted from queue ",del);
            break;
        case 4:
```

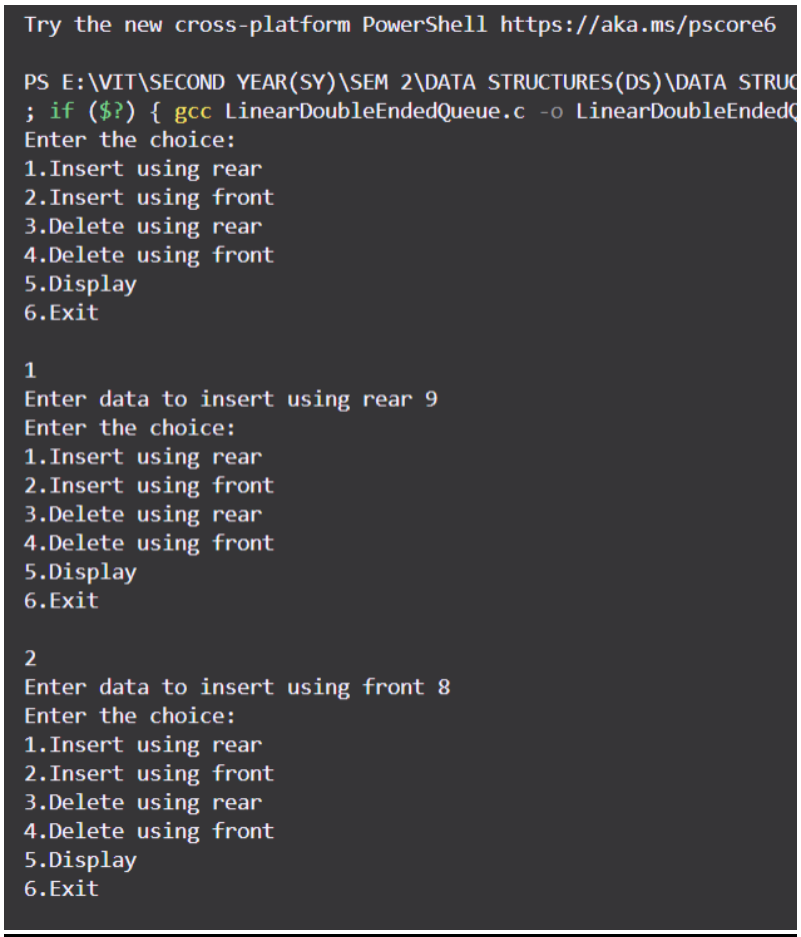
Shrihari K Mangle
CSB2-74   12010517

```
            del=dequeuefront(p);
            printf("%d is deleted from queue ",del);
            break;
        case 5:
            display(p);
            break;
        case 6:
            exit (0);
        default:
            printf("Please enter valid choice");
    }
}
}
void init(queue *q)
{
    q->front=-1;
    q->rear=-1;
}
int isempty(queue *q)
{
    if(q->rear==-1)
    {
        return 1;
    }
    else
    {
        return 0;
    }
}
int isfull(queue *q)
{
    if((q->rear+1)%MAX==q->front)
    {
        return 1;
    }
    else
    {
        return 0;
    }
}
void inqueuerear(queue *q,int x)
{
    if(isempty(q))
    {

        q->front=0;
        q->rear=0;
        q->data[q->rear]=x;
    }
    else if(isfull(q))
    {
```

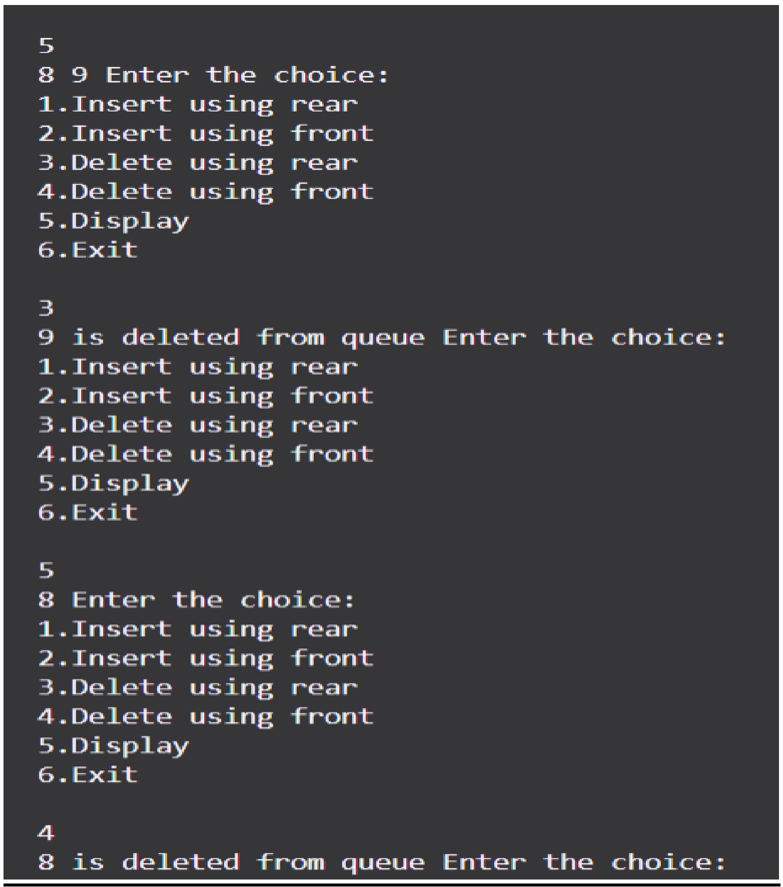
```c
        printf("Queue is full");
    }
    else
    {
        q->rear=(q->rear+1)%MAX;
        q->data[q->rear]=x;
    }
}
void inqueuefront(queue *q,int x)
{
    if(isempty(q))
    {
        q->front=0;
        q->rear=0;
        q->data[q->front]=x;
    }
    else if(isfull(q))
    {
        printf("Queue is full");
    }
    else
    {
        q->front=(q->front-1+MAX)%MAX;
        q->data[q->front]=x;
    }
}
int dequeuefront(queue *q)
{
    int x;
    x=q->data[q->front];
    if(isempty(q))
    {
        printf("Queue is empty");
    }
    else if(q->front==q->rear)
    {
        init(q);
    }
    else
    {
        q->front=(q->front+1)%MAX;
    }
    return x;

}
int dequeuerear(queue *q)
{
    int x;
    x=q->data[q->rear];
    if(isempty(q))
    {
        printf("Queue is empty");
    }
    else if(q->front==q->rear)
    {
        init(q);
    }
    else
    {
        q->rear=(q->rear-1+MAX)%MAX;
    }
    return x;
}
void display(queue *q)
{
    if(isempty(q) )
    {
        printf("Queue is empty");
    }
    else
    {
        int x;
        x=q->front;
        while(x!=q->rear)
        {
            printf("%d ",q->data[x]);
            x=(x+1)%MAX;
        }
        printf("%d ",q->data[q->rear]);
    }
}
```

<u>OUTPUT :</u>

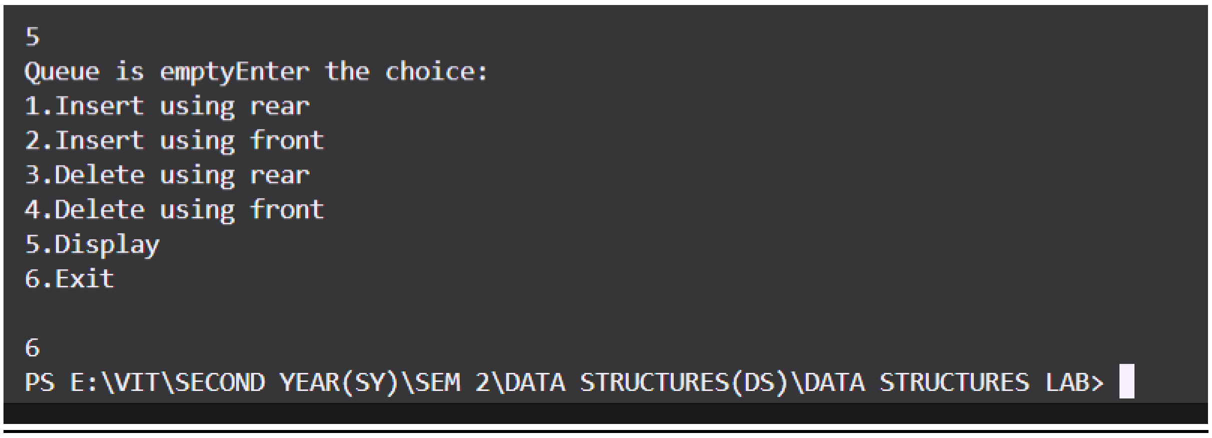
Shrihari K Mangle
CSB2-74   12010517



```
Try the new cross-platform PowerShell https://aka.ms/pscore6

PS E:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA STRUC
; if ($?) { gcc LinearDoubleEndedQueue.c -o LinearDoubleEndedQ
Enter the choice:
1.Insert using rear
2.Insert using front
3.Delete using rear
4.Delete using front
5.Display
6.Exit

1
Enter data to insert using rear 9
Enter the choice:
1.Insert using rear
2.Insert using front
3.Delete using rear
4.Delete using front
5.Display
6.Exit

2
Enter data to insert using front 8
Enter the choice:
1.Insert using rear
2.Insert using front
3.Delete using rear
4.Delete using front
5.Display
6.Exit
```



```
5
8 9 Enter the choice:
1.Insert using rear
2.Insert using front
3.Delete using rear
4.Delete using front
5.Display
6.Exit

3
9 is deleted from queue Enter the choice:
1.Insert using rear
2.Insert using front
3.Delete using rear
4.Delete using front
5.Display
6.Exit

5
8 Enter the choice:
1.Insert using rear
2.Insert using front
3.Delete using rear
4.Delete using front
5.Display
6.Exit

4
8 is deleted from queue Enter the choice:
```



```
5
Queue is emptyEnter the choice:
1.Insert using rear
2.Insert using front
3.Delete using rear
4.Delete using front
5.Display
6.Exit

6
PS E:\VIT\SECOND YEAR(SY)\SEM 2\DATA STRUCTURES(DS)\DATA STRUCTURES LAB>
```