QUEUES

Write a menu driven C program to implement Queues and Circular Queue using arrays and perform the following operation (i) Insert (ii)Delete (iii)is empty() (iv) Is full() (v) Display

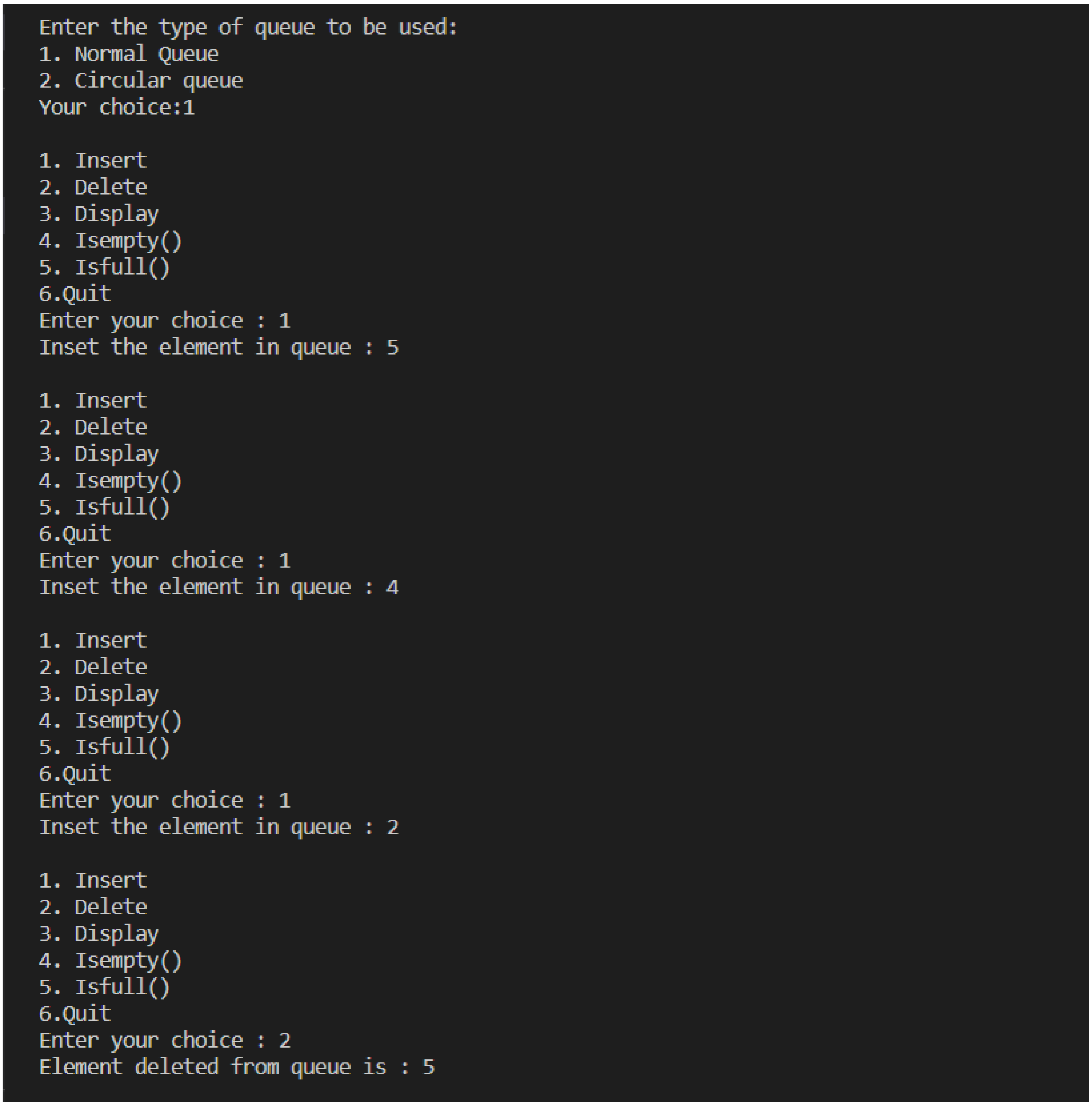
PROGRAM

```c
#include<stdio.h>

#define MAX 5
int queue_array[MAX];
int rear = - 1;
int front = - 1;

void insert()
{
    int add_item;
    if (rear == MAX - 1)
    printf("Queue Overflow \n");
    else
    {
        if (front == - 1)
        front = 0;
        printf("Inset the element in queue : ");
        scanf("%d", &add_item);
        rear = rear + 1;
        queue_array[rear] = add_item;
    }
}

void delete()
{
    if (front == - 1 || front > rear)
    {
        printf("Queue Underflow \n");
        return ;
    }
    else
    {
        printf("Element deleted from queue is : %d\n", queue_array[front]);
        front = front + 1;
    }
}

void qdisplay()
{
    int i;
    if (front == - 1)
        printf("Queue is empty \n");
    else
    {
        printf("Queue is : \n");
        for (i = front; i <= rear; i++)
            printf("%d ", queue_array[i]);
        printf("\n");
    }
}

void qisempty()
{
    if(front>rear || (front==-1&&rear==-1))
    printf("\nQueue is empty");
    else
    printf("\nQueue is not empty");
}

void qisfull()
{
    if(rear==MAX-1)
    printf("\nQueue is full");
    else
    printf("\nQueue is not full");
}

// Check if the circular queue is full
int isFull() {
  if ((front == rear + 1) || (front == 0 && rear == MAX - 1)) return 1;
  return 0;
}

// Check if the circular queue is empty
int isEmpty() {
  if (front == -1) return 1;
  return 0;
}

// Adding an element
void enQueue() {
    int element;
    printf("\nEnter the element to be inserted:");
    scanf("%d",&element);
  if (isFull())
    printf("\n Queue is full!! \n");
  else
    {
    if (front == -1) front = 0;
    rear = (rear + 1) % MAX;
    queue_array[rear] = element;
    printf("\n Inserted -> %d", element);
    }
}

// Removing an element
void deQueue() {
  int element;
  if (isEmpty()) {
    printf("\n Queue is empty !! \n");
    return (-1);
  } else {
    element = queue_array[front];
    if (front == rear) {
      front = -1;
      rear = -1;
    }
    else {
      front = (front + 1) % MAX;
    }
    printf("\n Deleted element -> %d \n", element);
  }
}

// Display the queue
void display() {
  int i;
  if (isEmpty())
    printf(" \n Empty Queue\n");
  else {
    printf("\n Items -> ");
    for (i = front; i != rear; i = (i + 1) % MAX) {
      printf("%d ", queue_array[i]);
    }
    printf("%d ", queue_array[i]);
  }
}

void main()
{
    int choice,x,qchoice;
    printf("\n\nEnter the type of queue to be used:\n1. Normal Queue\n2. Circular queue\nYour choice:");
    scanf("%d",&qchoice);
    if(qchoice==1)
    {
        while (1)
        {
            printf("\n1. Insert \n");
            printf("2. Delete\n");
            printf("3. Display\n");
            printf("4. Isempty()\n");
            printf("5. Isfull()\n");
            printf("6.Quit \n");
            printf("Enter your choice : ");
            scanf("%d", &choice);
            switch (choice)
            {
                case 1:
                insert();
                break;
                case 2:
                delete();
                break;
                case 3:
                qdisplay();
                break;
                case 4:
                qisempty();
                break;
                case 5:
                qisfull();
                break;
                case 6:
                exit(0);
                break;
                default:
                printf("Wrong choice \n");
            }
        }
    }
    else if(qchoice==2)
    {
    while (1)
        {
            printf("\n1. Insert \n");
            printf("2. Delete\n");
            printf("3. Display\n");
            printf("4. Isempty()\n");
            printf("5. Isfull()\n");
            printf("6. Quit \n");
            printf("Enter your choice : ");
            scanf("%d", &choice);
            switch (choice)
            {
                case 1:
                enQueue();
                break;
                case 2:
                deQueue();
                break;
                case 3:
                display();
                break;
                case 4:
                x=isEmpty();
                if(x==1) printf("\nQueue is empty\n");
                else printf("\nQueue is not empty\n");
                break;
                case 5:
                x=isFull();
                if(x==1) printf("\nQueue is full\n");
                else printf("\nQueue is not full\n");
                break;
                case 6:
                exit(0);
                break;
                default:
```

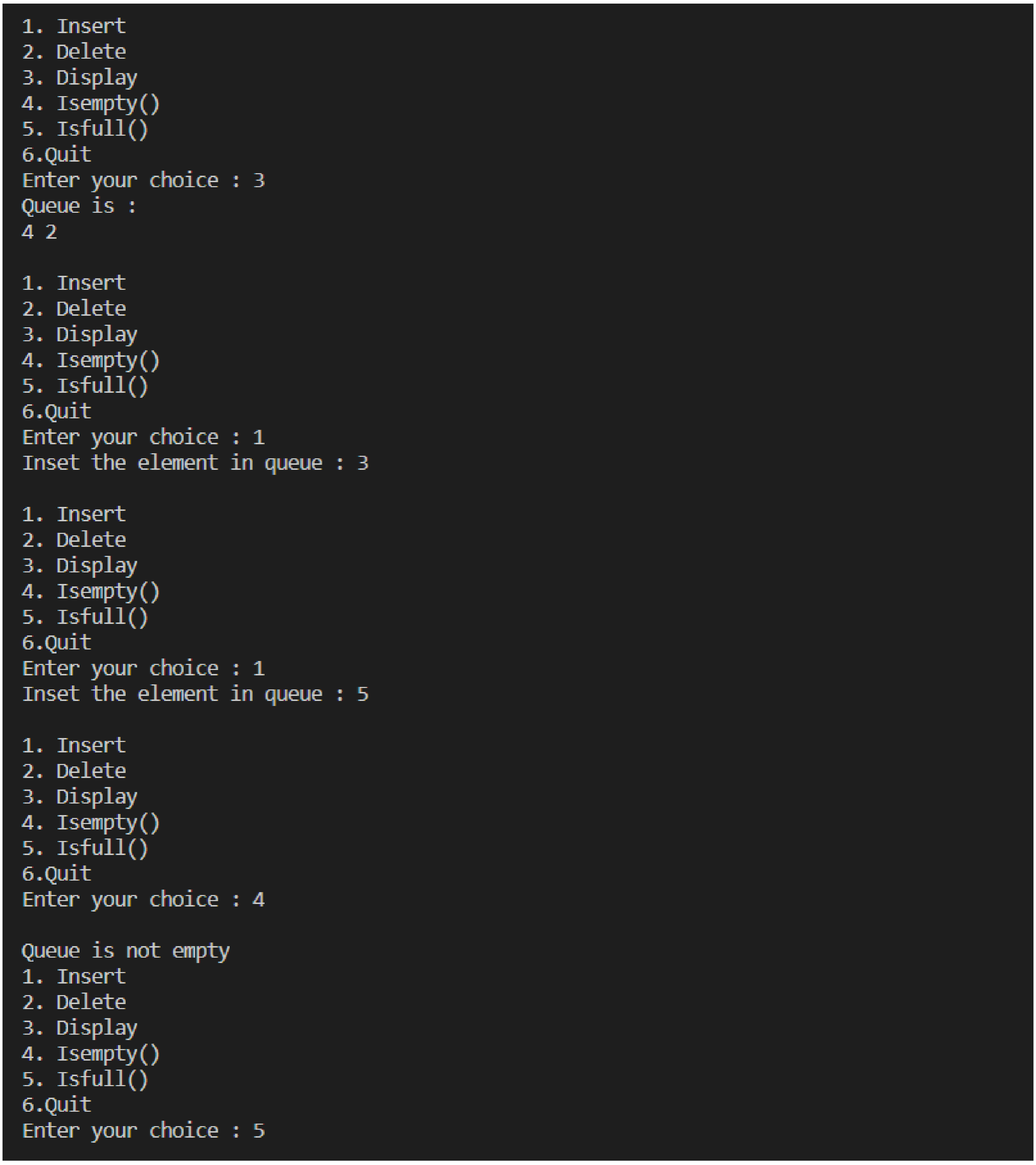
```
                printf("Wrong choice \n");
            }
        }
    }
}
```

OUTPUT

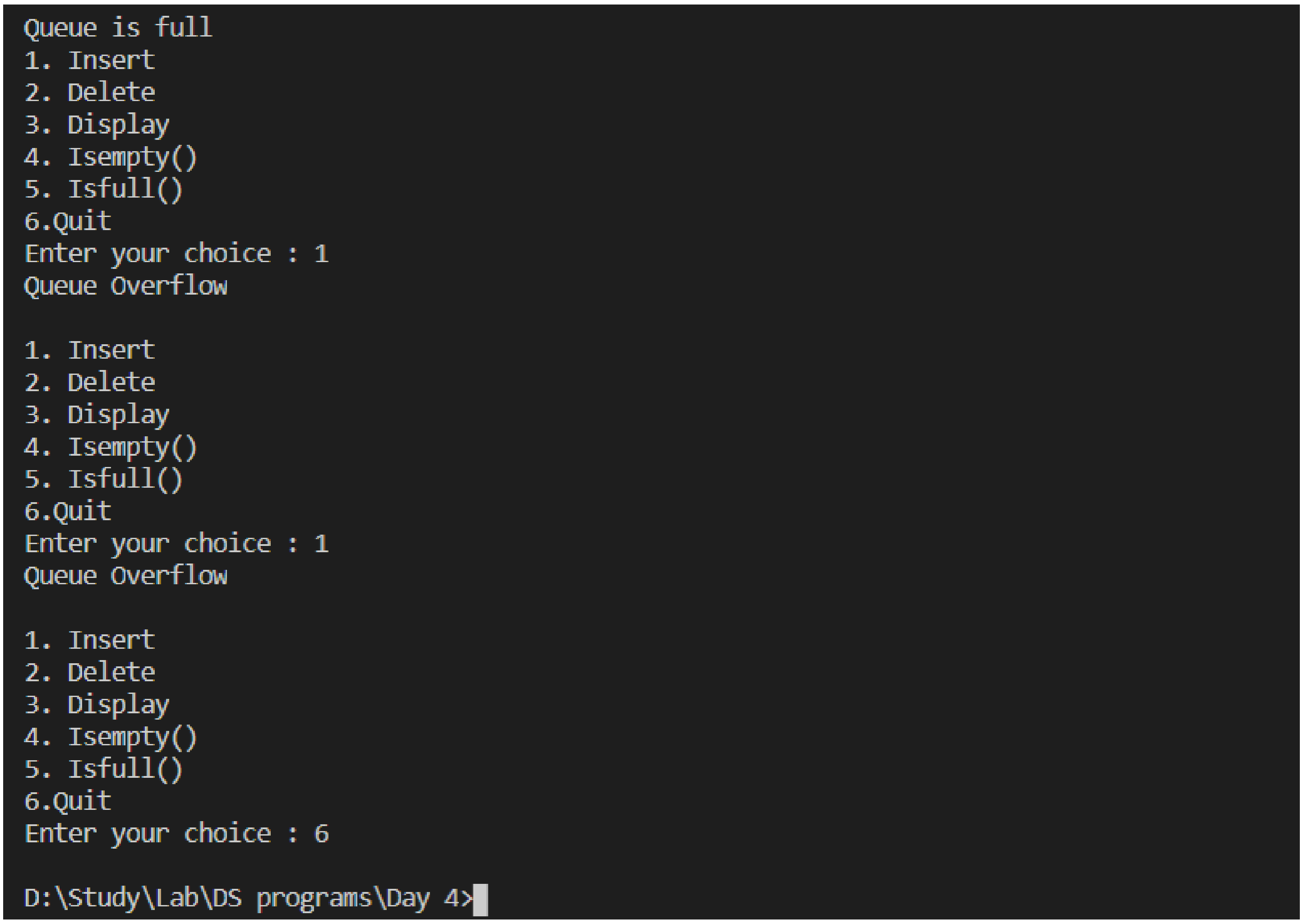
TEST CASE 1:



```
Enter the type of queue to be used:
1. Normal Queue
2. Circular queue
Your choice:1

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 1
Inset the element in queue : 5

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 1
Inset the element in queue : 4

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 1
Inset the element in queue : 2

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 2
Element deleted from queue is : 5
```

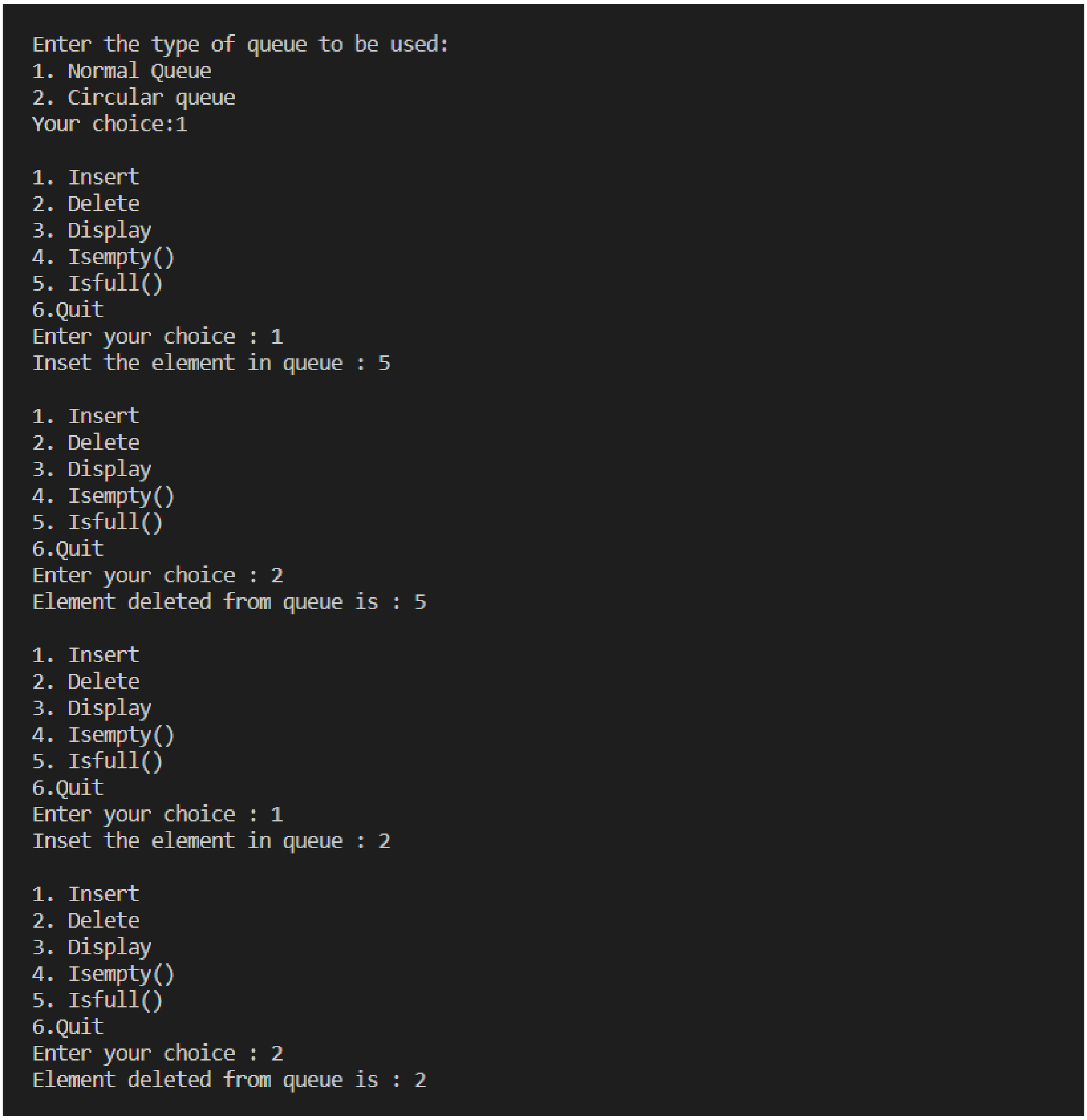


```
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 3
Queue is :
4 2

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 1
Inset the element in queue : 3

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 1
Inset the element in queue : 5

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 4

Queue is not empty
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 5
```

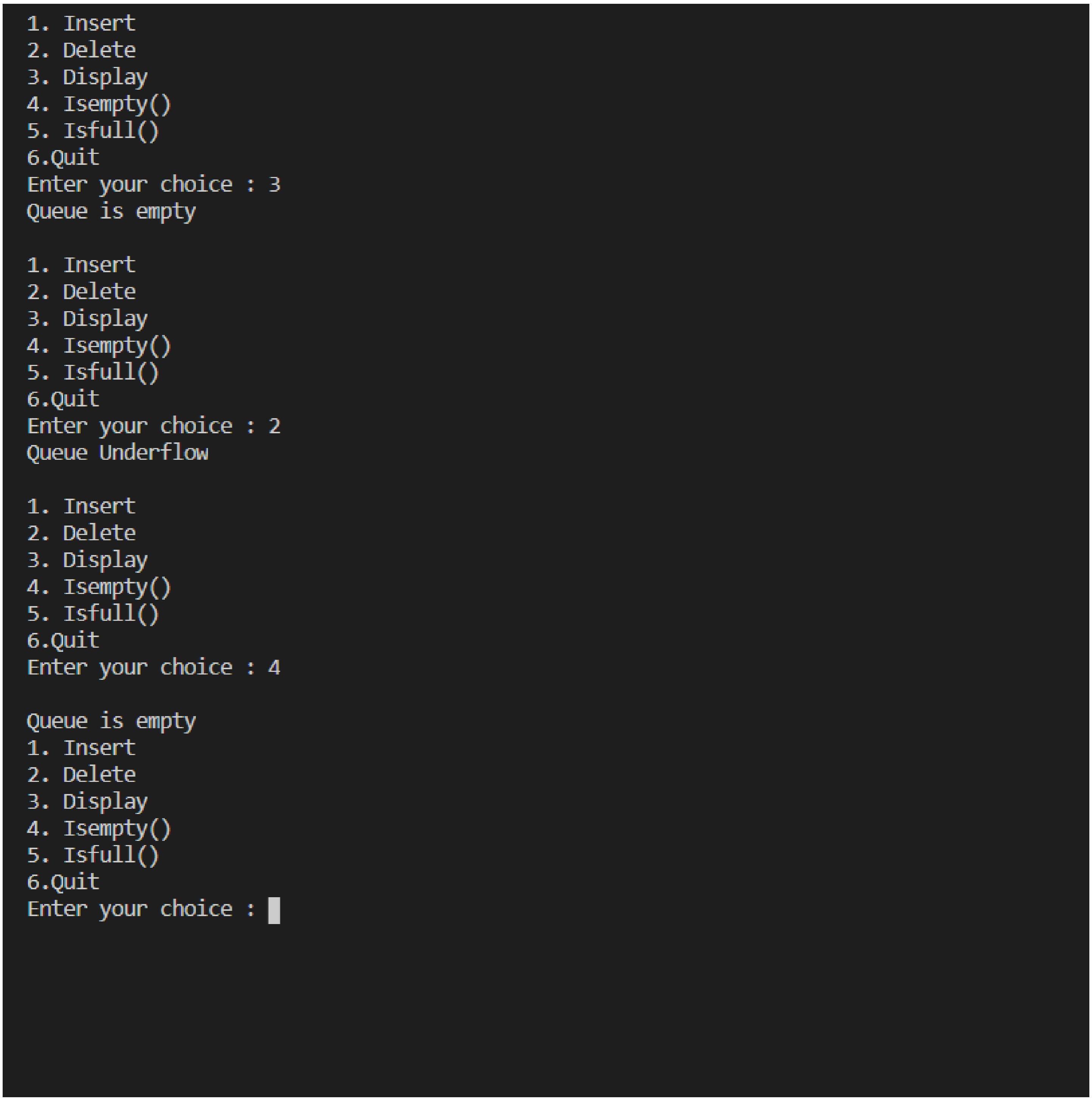


```
Queue is full
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 1
Queue Overflow

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 1
Queue Overflow

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 6

D:\Study\Lab\DS programs\Day 4>
```

TEST CASE 2:

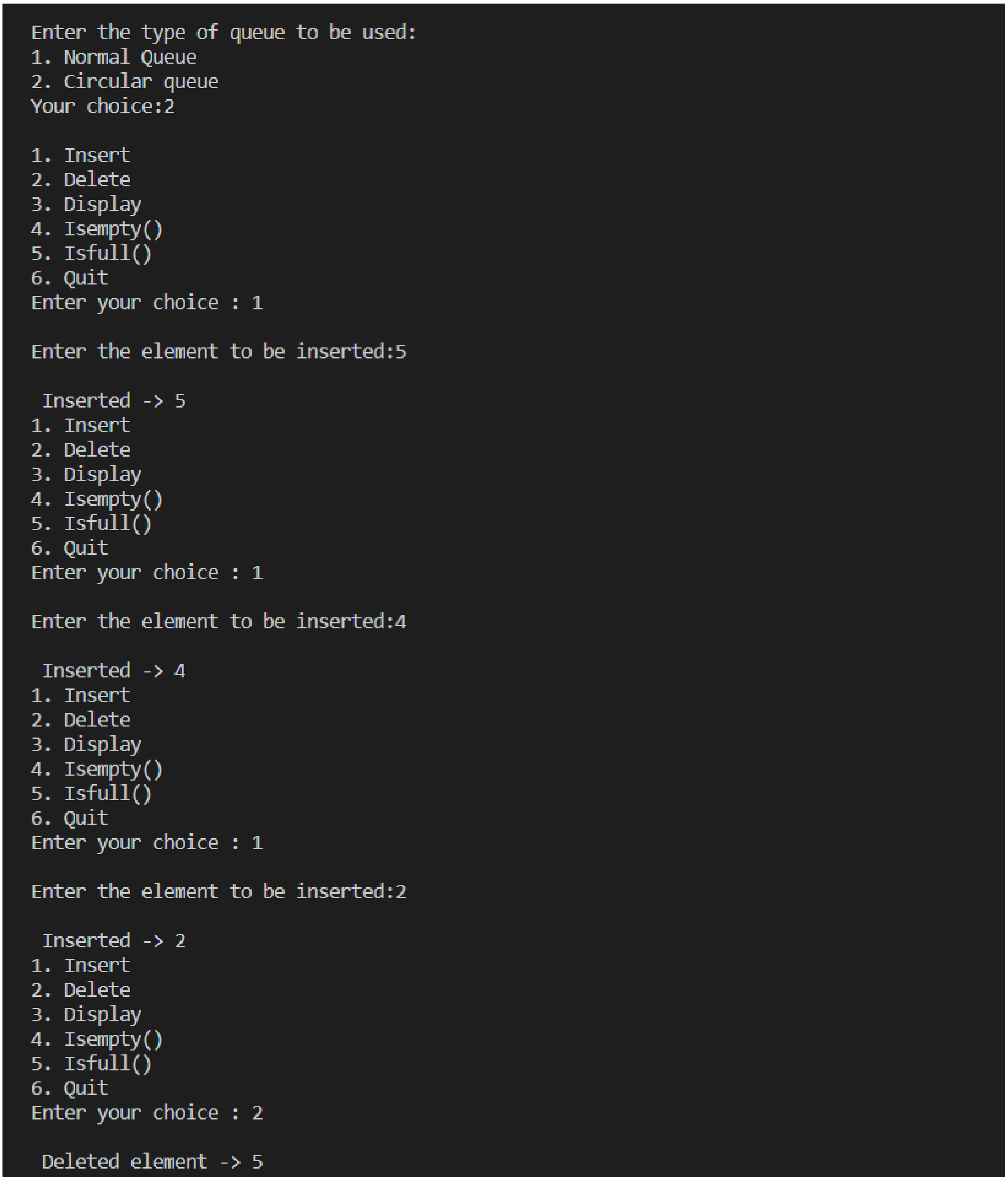


```
Enter the type of queue to be used:
1. Normal Queue
2. Circular queue
Your choice:1

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 1
Inset the element in queue : 5

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 2
Element deleted from queue is : 5

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 1
Inset the element in queue : 2

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 2
Element deleted from queue is : 2
```

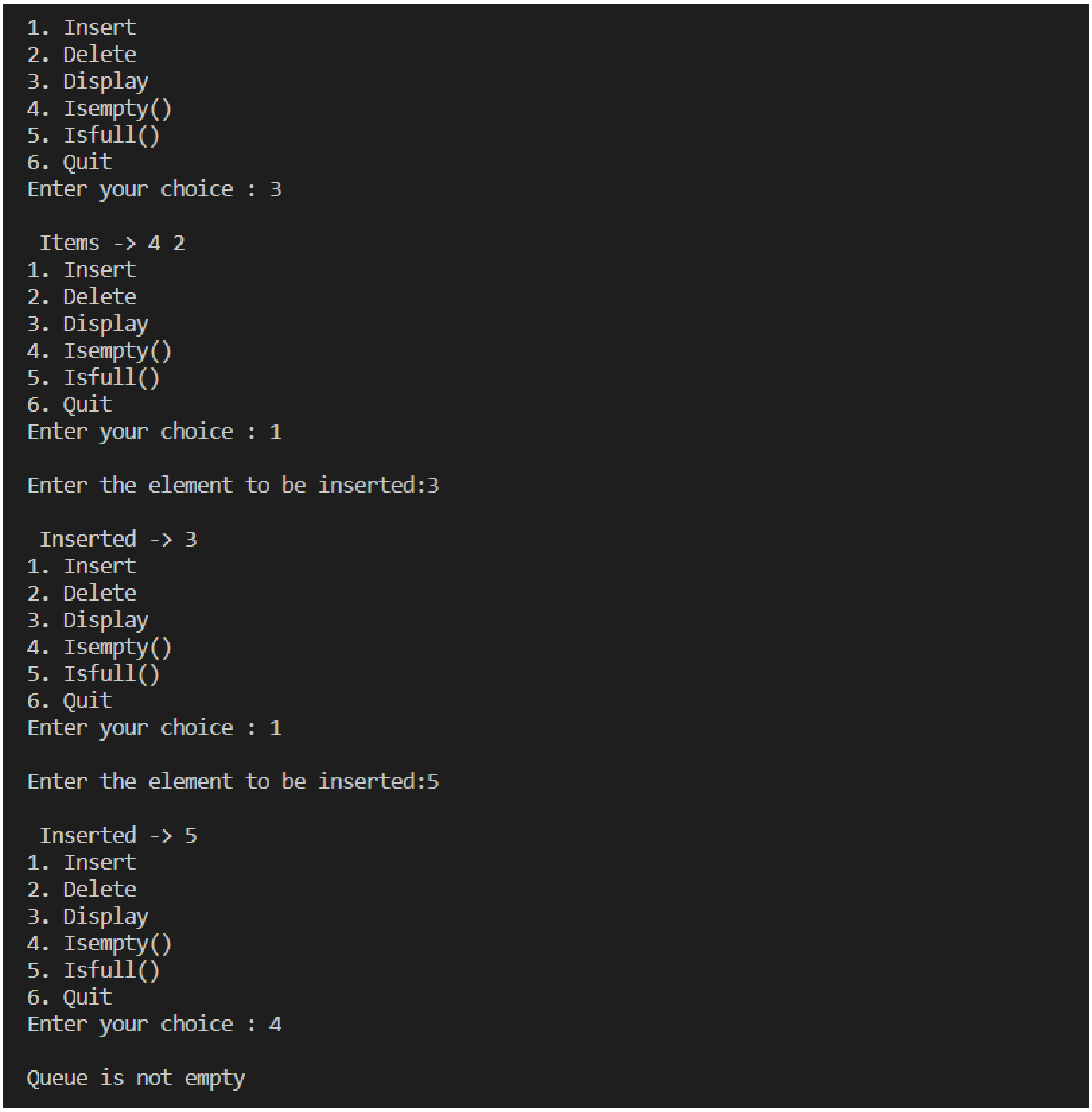


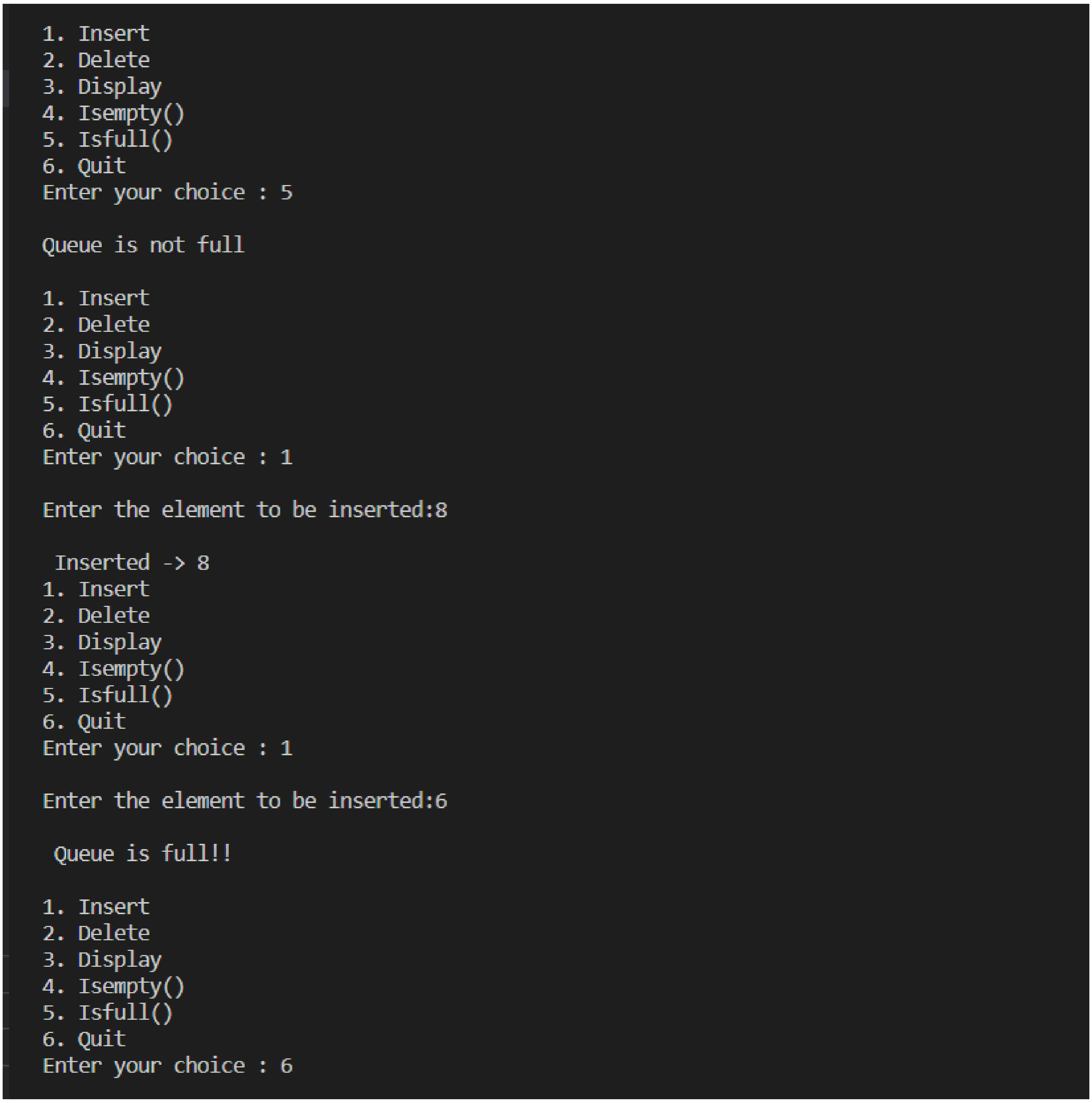
```
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 3
Queue is empty

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 2
Queue Underflow

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice : 4

Queue is empty
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6.Quit
Enter your choice :
```

TEST CASE 3:



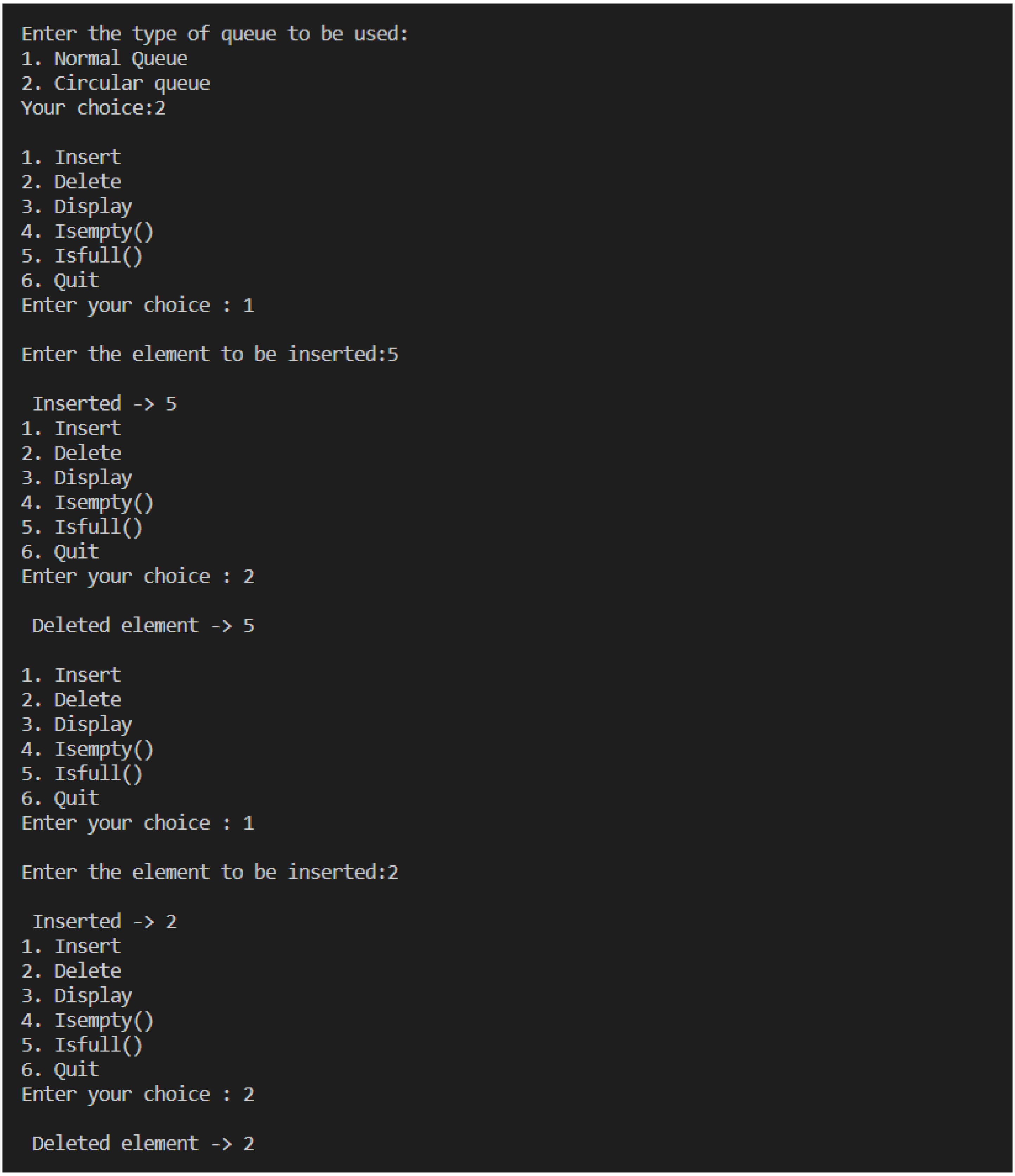
```
Enter the type of queue to be used:
1. Normal Queue
2. Circular queue
Your choice:2

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 1

Enter the element to be inserted:5

 Inserted -> 5
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 1

Enter the element to be inserted:4

 Inserted -> 4
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 1

Enter the element to be inserted:2

 Inserted -> 2
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 2

 Deleted element -> 5
```


```
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 3

 Items -> 4 2
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 1

Enter the element to be inserted:3

 Inserted -> 3
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 1

Enter the element to be inserted:5

 Inserted -> 5
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 4

Queue is not empty
```

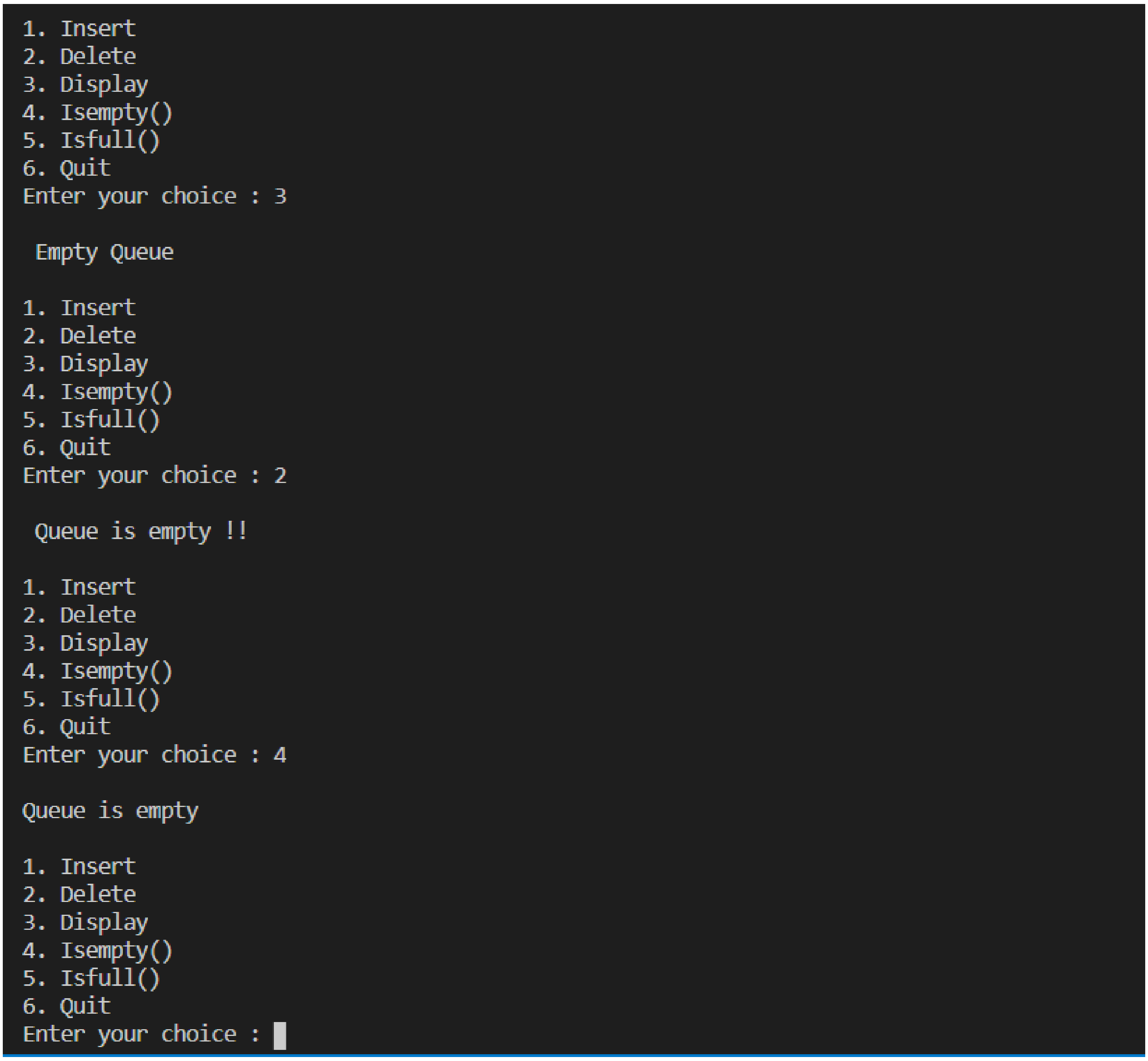


```
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 5

Queue is not full

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 1

Enter the element to be inserted:8

 Inserted -> 8
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 1

Enter the element to be inserted:6

 Queue is full!!

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 6
```

TEST CASE 4:



```
Enter the type of queue to be used:
1. Normal Queue
2. Circular queue
Your choice:2

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 1

Enter the element to be inserted:5

 Inserted -> 5
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 2

 Deleted element -> 5

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 1

Enter the element to be inserted:2

 Inserted -> 2
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 2

 Deleted element -> 2
```



```
1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 3

 Empty Queue

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 2

 Queue is empty !!

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice : 4

Queue is empty

1. Insert
2. Delete
3. Display
4. Isempty()
5. Isfull()
6. Quit
Enter your choice :
```

Write a menu driven C program to implement Priority Queues using arrays.

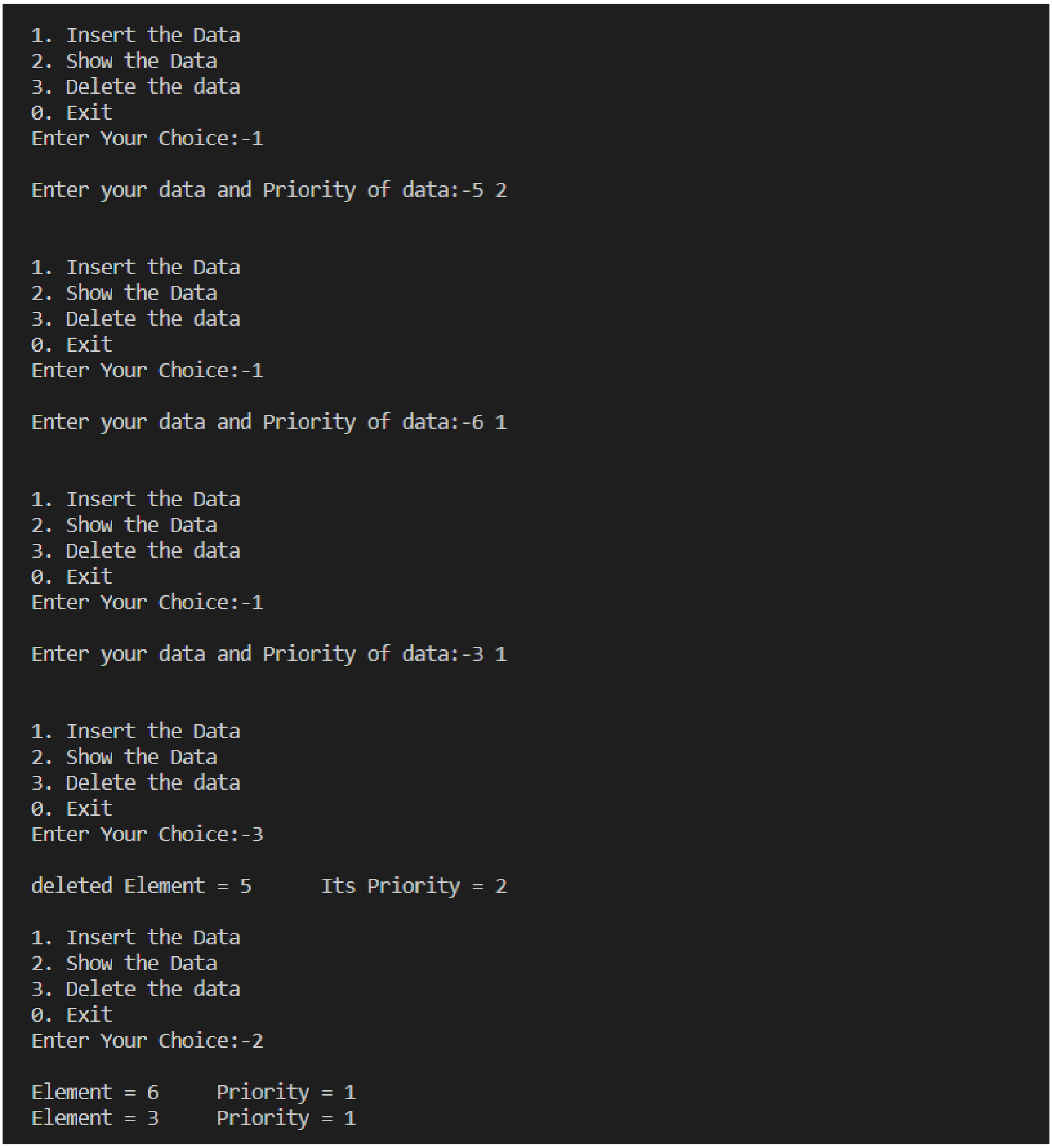
PROGRAM

```
#include<stdio.h>
#define N 5
int Q[N],Pr[N];
int r = -1,f = -1;
void enqueue(int data,int p)
{
    int i;
    if(r==N-1)
        printf("Queue is full");
    else
    {
        if(f==-1)
        {
            f = r = 0;
            Q[r] = data;
            Pr[r] = p;

        }
        else
        {
            for(i = r;i>=f;i--)
            {
                if(p>Pr[i])
                {
                    Q[i+1] = Q[i];
                    Pr[i+1] = Pr[i];
                }
                else
                    break;
            }
            Q[i+1] = data;
            Pr[i+1] = p;
            r++;
        }
    }

}
void print() //print the data of Queue
{
int i;
    for(i=f;i<=r;i++)
    {
        printf("\nElement = %d\tPriority = %d",Q[i],Pr[i]);
    }
}
int dequeue() //remove the data from front
```

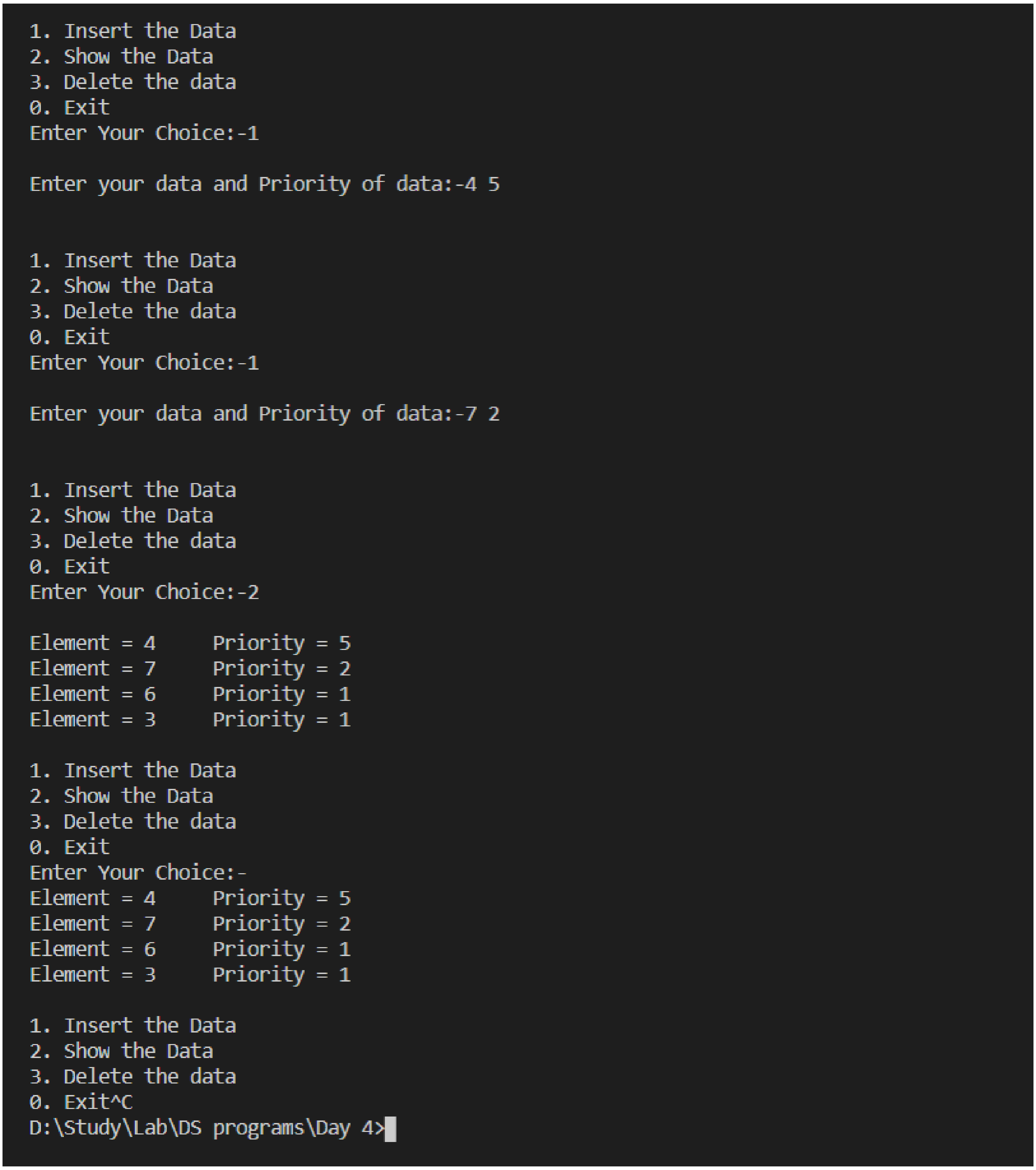
```c
{
    if(f == -1)
    {
        printf("Queue is Empty");
    }
    else
    {
        printf("\ndeleted Element = %d\t Its Priority = %d",Q[f],Pr[f]);
        if(f==r)
            f = r = -1;
        else
            f++;
    }
}
void main()
{
    int opt,data,p;
    do{
        printf("\n\n1. Insert the Data\n2. Show the Data\n3. Delete the data\n0. Exit");
        printf("\nEnter Your Choice:-");
        scanf("%d",&opt);
        switch(opt){
            case 1:
                printf("\nEnter your data and Priority of data:-");
                scanf("%d %d",&data,&p);
                enqueue(data,p);
                break;
            case 2:
                print();
                break;
            case 3:
                 dequeue();
                break;
            case 0:
                break;
            default:
                printf("\nIncorrect Choice");

        }
    }while(opt!=0);
}
```

OUTPUT

TEST CASE 1:

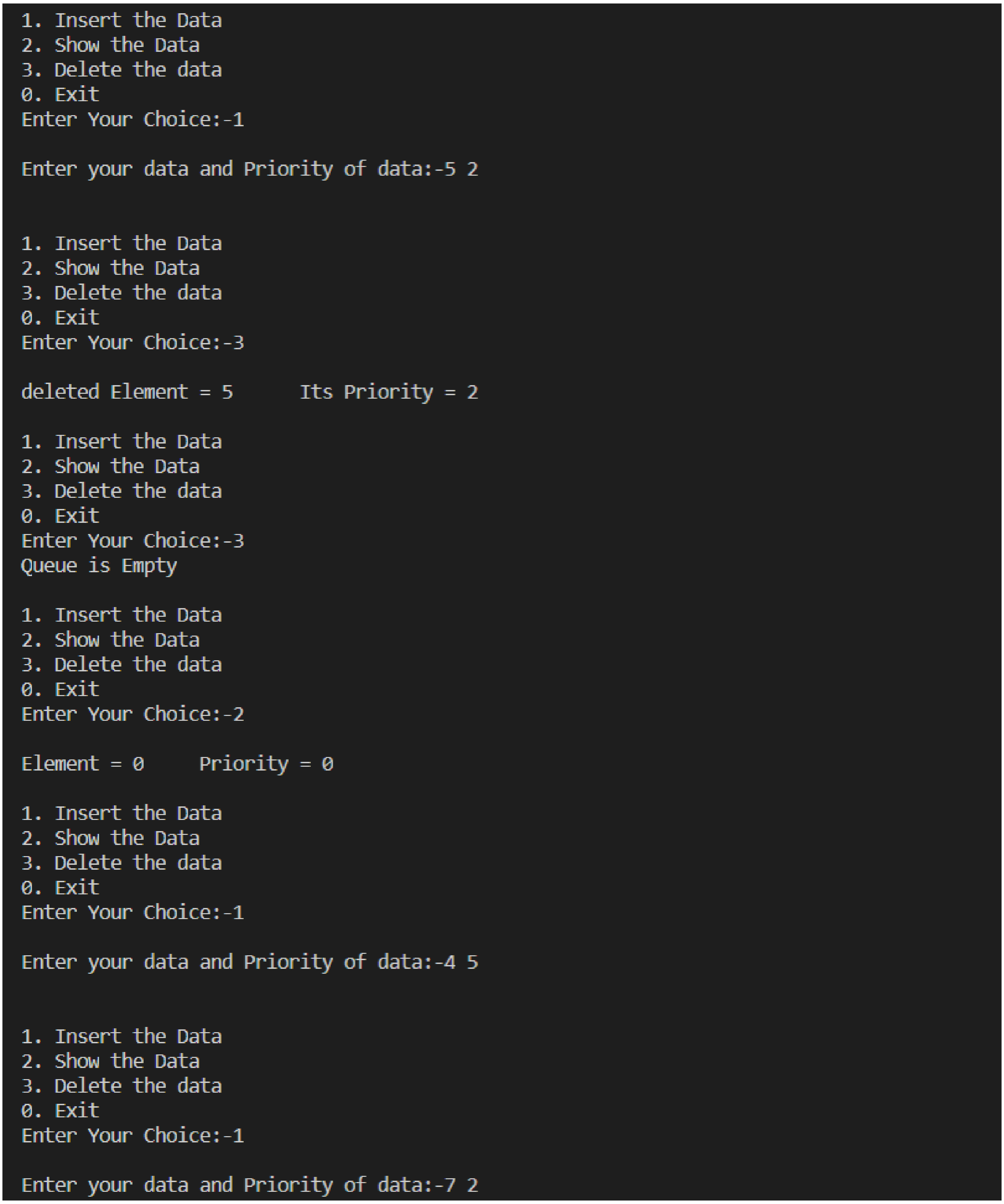


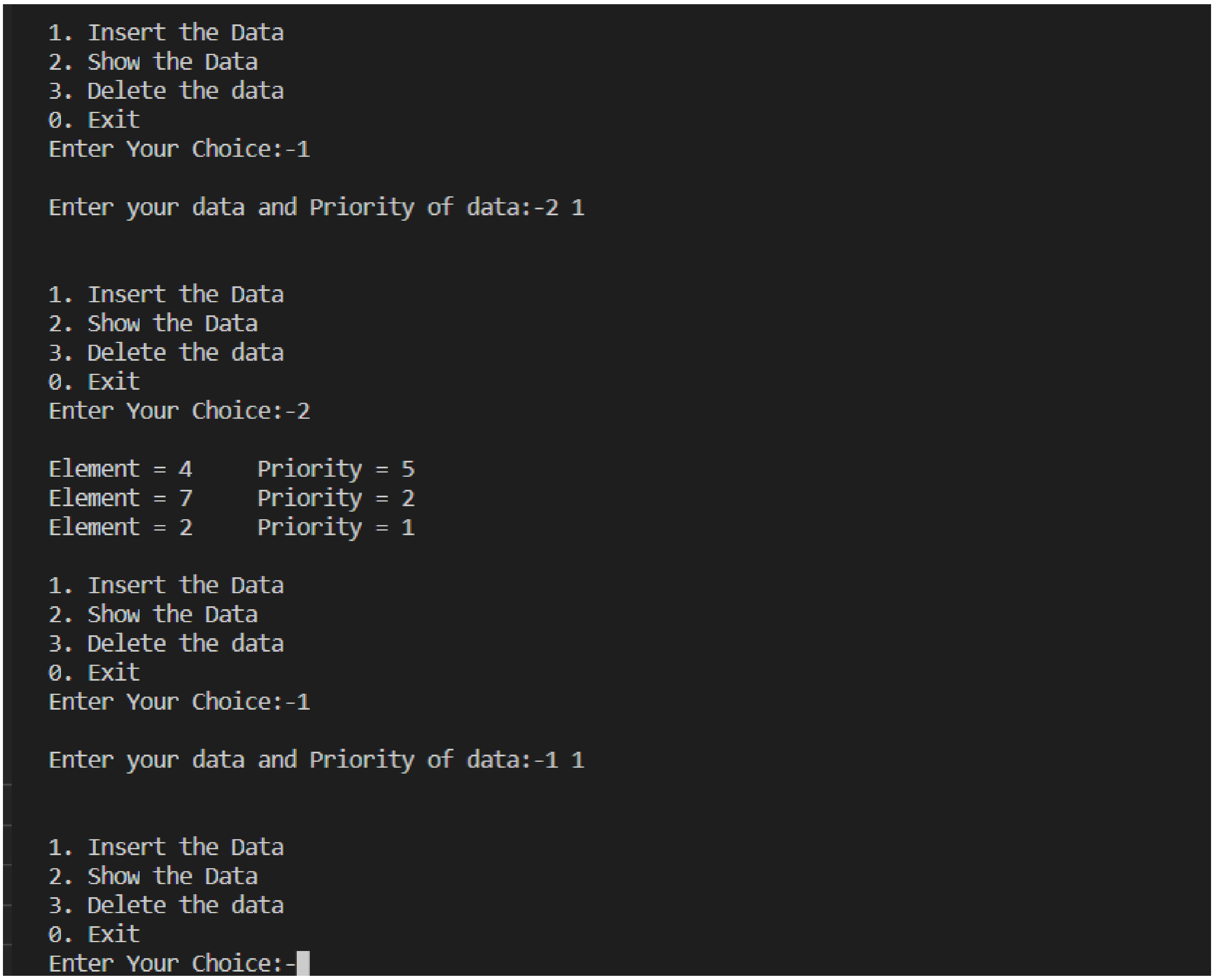
```
1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-1

Enter your data and Priority of data:-5 2


1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-1

Enter your data and Priority of data:-6 1


1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-1

Enter your data and Priority of data:-3 1


1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-3

deleted Element = 5      Its Priority = 2

1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-2

Element = 6     Priority = 1
Element = 3     Priority = 1
```



```
1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-1

Enter your data and Priority of data:-4 5


1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-1

Enter your data and Priority of data:-7 2


1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-2

Element = 4     Priority = 5
Element = 7     Priority = 2
Element = 6     Priority = 1
Element = 3     Priority = 1

1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-
Element = 4     Priority = 5
Element = 7     Priority = 2
Element = 6     Priority = 1
Element = 3     Priority = 1

1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit^C
D:\Study\Lab\DS programs\Day 4>
```

TEST CASE 2:


```
1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-1

Enter your data and Priority of data:-5 2


1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-3

deleted Element = 5      Its Priority = 2

1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-3
Queue is Empty

1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-2

Element = 0     Priority = 0

1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-1

Enter your data and Priority of data:-4 5


1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-1

Enter your data and Priority of data:-7 2
```



```
1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-1

Enter your data and Priority of data:-2 1


1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-2

Element = 4     Priority = 5
Element = 7     Priority = 2
Element = 2     Priority = 1

1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-1

Enter your data and Priority of data:-1 1


1. Insert the Data
2. Show the Data
3. Delete the data
0. Exit
Enter Your Choice:-
```