

## Ordinary Queue

### Source Code:

```
/*
    Name : Robin Singh
    Rollno : 1261
    Unit : 04
    Program : Ordinary Queue
*/

#include<iostream
>
#include<conio.h>

#define MAX 4
using namespace std;
// 2. Queue Template

class OQueue
{
    int
    A[MAX];
    int front;
    int rear;

    public
    :
        OQueue()
        {
            front = -1;
            rear = -1;
        }

        void Enqueue(int
x); void
Dequeue(); void
PeekFront(); void
PeekRear(); void
Display();
int Full();
int
Empty();
};

int OQueue :: Full()
{
    if(rear == MAX-1)
    {
        return 1;
    }
    else
    {
        return 0;
    }
}
```

```
    }
}

int OQueue :: Empty()
{
    if(front == -1)
    {
        return 1;
    }
    else
    {
        return 0;
    }
}

void OQueue :: Enqueue(int x)
{
    if(Full()
    )
    {
        cout <<
        "Overflow!!!"; return;
    }
    if(front == -1)
    {
        front++;
    }
    rear++;
    A[rear]=x;
}

void OQueue :: PeekFront()
{
    if(Empty())
    {
        cout <<
        "Underflow!!"; return;
    }
    else
    {
        cout << A[front];
    }
}

void OQueue :: PeekRear()
{
    if(Empty())
    {
        cout <<
        "Underflow!!"; return;
    }
}
```

```
    }
    else
    {
        cout << A[rear];
    }
}

void OQueue :: Display()
{
    if(Empty())
    {
        cout << "Underflow";
    }
    for (int i = front ; i <= rear ; i++)
    {
        cout << A[i] << " ";
    }
}

void OQueue::Dequeue()
{
    if(Empty())
    {
        cout <<
            "Underflow!"; return;
    }
    int tmp = A[front];
    if(front== rear)
    {
        front = -1;
        rear = -1;
    }
    else
    {
        front++;
    }
    cout << tmp << " dequeued form the queue";
}

int main()
{
    OQueue s;
    int ch,
    num;
    while(1)
    {
        system("cls");
        cout << "**** Ordinary Queue ****\n\n\n";
```

```
cout << "1. Enqueue a
element\n"; cout << "2.
Dequeue a element\n"; cout <<
"3. Peek Front element\n"; cout
<< "4. Peek Last element\n";
cout << "5. Display the
queue\n"; cout << "6. Exit\n\n";

cout << "Enter Your
Choice:"; cin >> ch;

switch(ch)
{
    case 1:
        cout << "Enter the
        element:"; cin >> num;
        s.Enqueue(num); getch();
        break;
    case 2:
        s.Dequeue();
        getch();
        ;
    case 3: break;

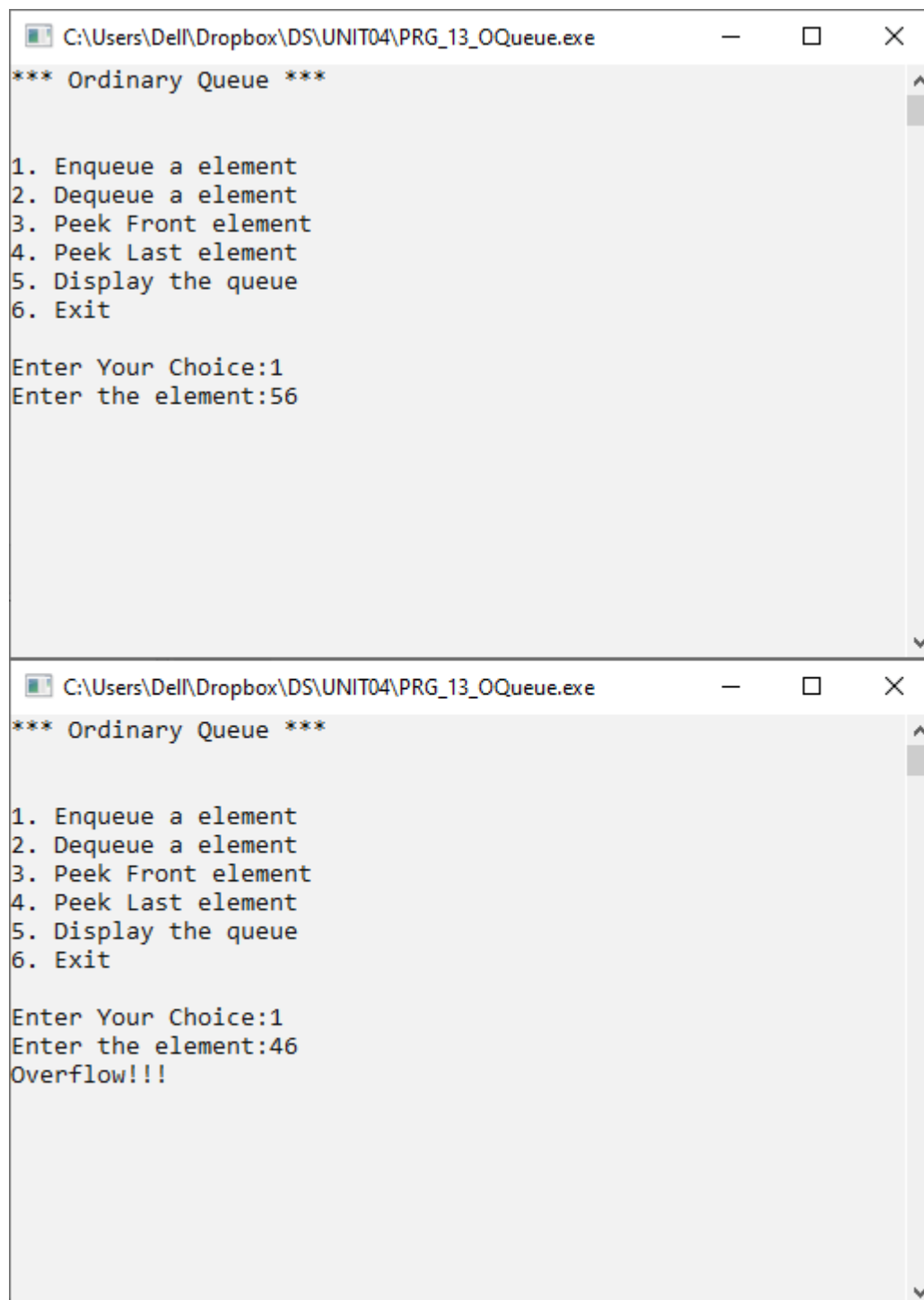
        s.PeekFront(
        ); getch();
    case 4: break;

        s.PeekRear();
        getch()
    case 5: ;
        break;

        s.Display();
    case 6: getch()
        ;
    default break;

    :      exit(1);

        cout << "Enter a valid
        option"; getch();
        break;
} //End of
switch
} //End of while
} //End of main
```

**Output:**

```
C:\Users\Dell\Dropbox\DS\UNIT04\PRG_13_OQueue.exe
*** Ordinary Queue ***

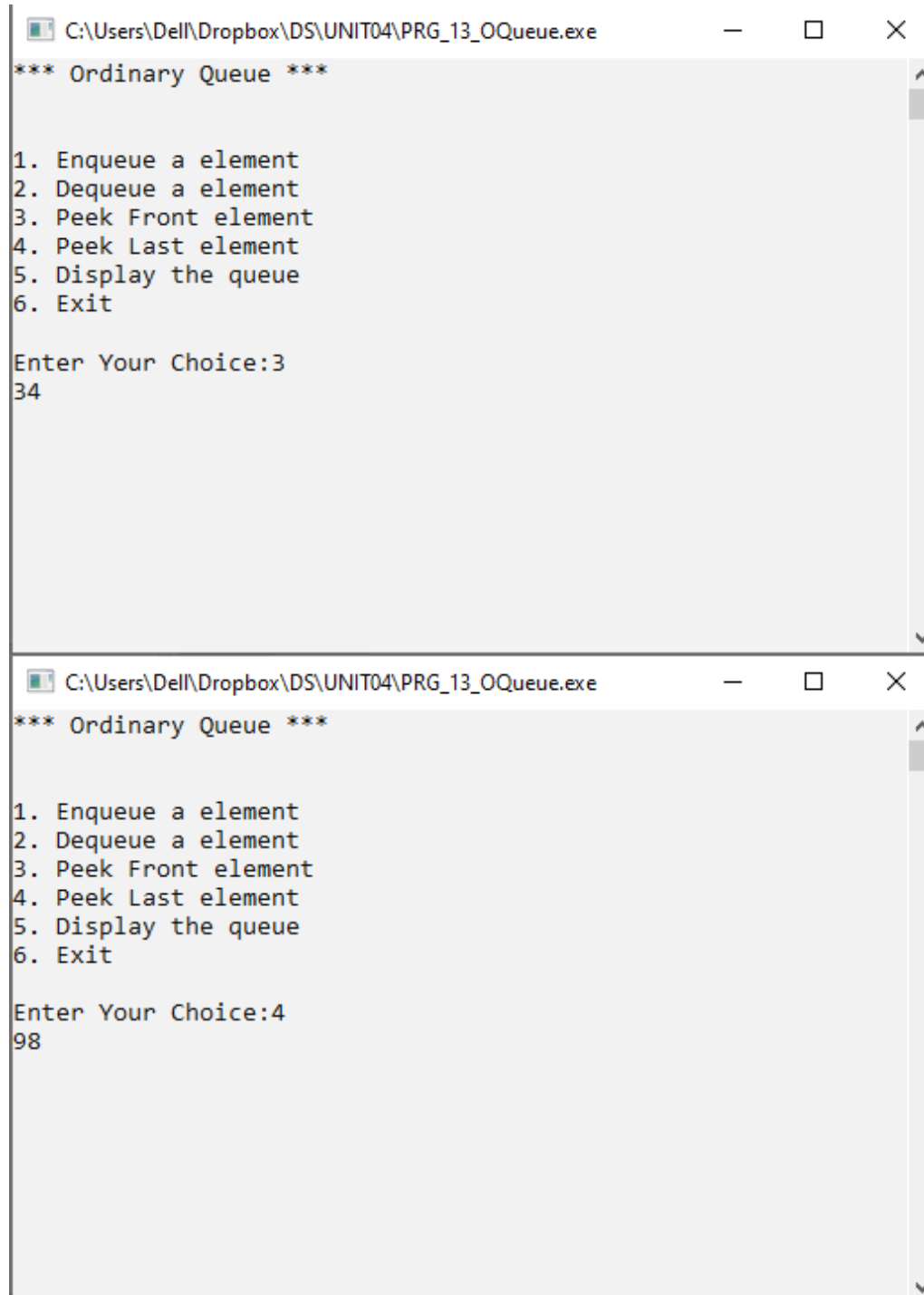
1. Enqueue a element
2. Dequeue a element
3. Peek Front element
4. Peek Last element
5. Display the queue
6. Exit

Enter Your Choice:1
Enter the element:56

C:\Users\Dell\Dropbox\DS\UNIT04\PRG_13_OQueue.exe
*** Ordinary Queue ***

1. Enqueue a element
2. Dequeue a element
3. Peek Front element
4. Peek Last element
5. Display the queue
6. Exit

Enter Your Choice:1
Enter the element:46
Overflow!!!
```



The image shows two overlapping Windows command prompt windows. Both windows have a title bar that reads 'C:\Users\Dell\Dropbox\DS\UNIT04\PRG\_13\_OQueue.exe'. The top window displays the following text:

```
*** Ordinary Queue ***  
  
1. Enqueue a element  
2. Dequeue a element  
3. Peek Front element  
4. Peek Last element  
5. Display the queue  
6. Exit  
  
Enter Your Choice:3  
34
```

The bottom window displays the following text:

```
*** Ordinary Queue ***  
  
1. Enqueue a element  
2. Dequeue a element  
3. Peek Front element  
4. Peek Last element  
5. Display the queue  
6. Exit  
  
Enter Your Choice:4  
98
```

```
C:\Users\Dell\Dropbox\DS\UNIT04\PRG_13_OQueue.exe
*** Ordinary Queue ***

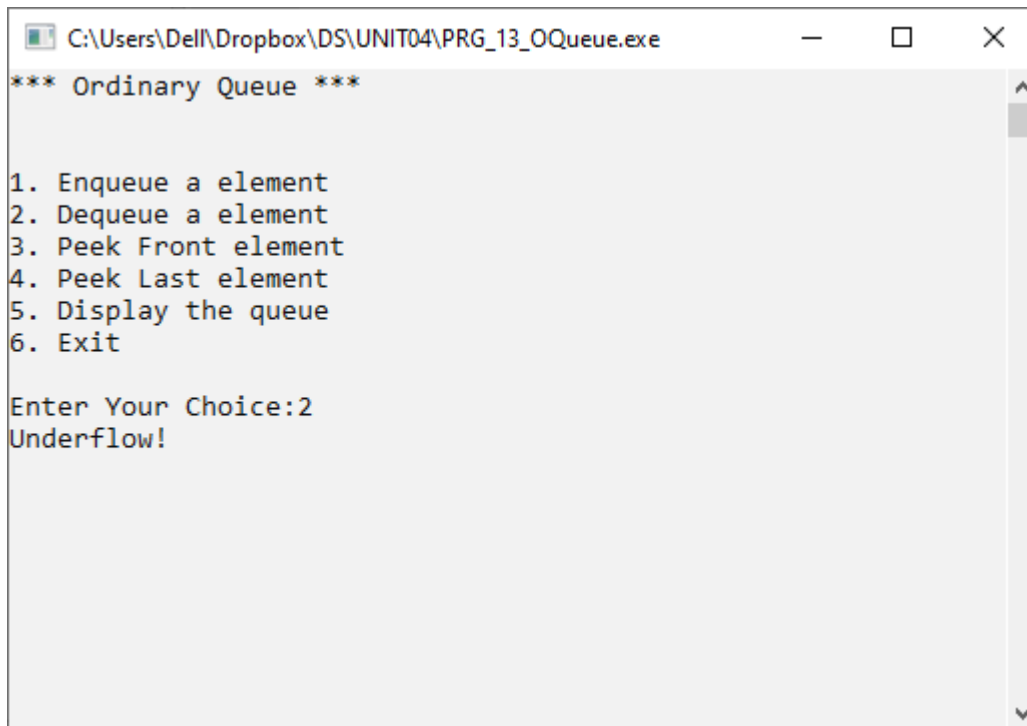
1. Enqueue a element
2. Dequeue a element
3. Peek Front element
4. Peek Last element
5. Display the queue
6. Exit

Enter Your Choice:5
34 56 75 98
```

```
C:\Users\Dell\Dropbox\DS\UNIT04\PRG_13_OQueue.exe
*** Ordinary Queue ***

1. Enqueue a element
2. Dequeue a element
3. Peek Front element
4. Peek Last element
5. Display the queue
6. Exit

Enter Your Choice:2
34 dequeued form the queue
```



```
*** Ordinary Queue ***

1. Enqueue a element
2. Dequeue a element
3. Peek Front element
4. Peek Last element
5. Display the queue
6. Exit

Enter Your Choice:2
Underflow!
```



## Circular Queue

### Source Code:

```
/*      Name : Robin
        Singh RollNo :
        1261
        Program : Circular
        Queues Unit : 04
*/

#include<iostream>
#include<conio.h>

#define MAX 4

using namespace std;

//1.Node Template
//2.class
Template class
CQueue
{
    int
    A[MAX];
    int front;
    int rear;
    int cnt;

    public
    :
        CQueue()
        {
            front = -1;
            rear = -1;
            cnt = 0;
        }

        void Enqueue(int
x); void
Dequeue(); void
PeekFront(); void
PeekRear(); void
Display();
int Full();
int
Empty();
}; //end of
class
//3.Functions
```

void CQueue :: Enqueue(int x)

```
{
    if(Full())
    {
        cout <<
        "Overflow!!\n"; return;
    }
    if(Empty())
    {
        front++;
    }
    if(rear == MAX-1)
    {
        rear = 0;
    }
    else
    {
        rear++;
    }
    A[rear] =
    x; cnt++;
} //end of
enqueue int
CQueue :: Full()
{
    if(cnt == MAX)
    {
        return 1;
    }
    else
    {
        return 0;
    }
} //end of
full
int CQueue :: Empty()
{
    if(front == -1)
    {
        return 1;
    }
    else
    {
        return 0;
    }
} //end of empty
```

```
void CQueue :: PeekFront()
```

```
{
    if(Empty())
    {
        cout <<
        "Underflow!!!"; return;
    }

    cout << "Element at the front is : " << A[front];
} //end of peekfront
```

```
void CQueue :: PeekRear()
```

```
{
    if(Empty())
    {
        cout <<
        "Underflow!!!"; return;
    }

    cout << "Element is at the rear is : " << A[rear];
} //end of rear
```

```
void CQueue::Dequeue()
```

```
{
    if(Empty())
    {
        cout<<"Underflow"
        ; return;
    }

    int tmp=A[front];
    if(front==rear)//single
    element
    {
        front=-1;//no
        elements rear=-1;
    }
    else
    {
        if(front==MAX-1)//if front is at last element
        {
            front=0;//set front =0
        }
        else
        {
            front++; //sets next element as front
        }
    }
}
```

```

        }
    }
    cout<<"Dequeue:
"<<tmp; cnt--;

} //end of dequeue
void
CQueue::Display()
{
    if(Empty())
    {
        cout<<"underflow";
    }
    int j=front;
    for(int i=1;i<=cnt;i++)
    {
        cout<<A[j]<<"
"; if(j==MAX-1)
        {
            j=0;
        }
        else
        {
            j++;
        } //end of else
    } //end of for
} //end of display

//4.Menu

int

main()
{
    CQueue
    c; int
    num,ch;

    while(1)
    {
        system("cls");

        cout << "*****Circular Queues*****\n";

        cout << "1.Enqueue an Element\n";
        cout << "2.Dequeue an Element\n";
        cout << "3.Peek at the front in

```

```
Queue\n"; cout << "4.Peek at the rear  
in Queue\n";
```

```
cout << "5.Display the
Queue\n"; cout << "6.Exit\n\n";

cout << "Enter Your Choice
: "; cin >> ch;

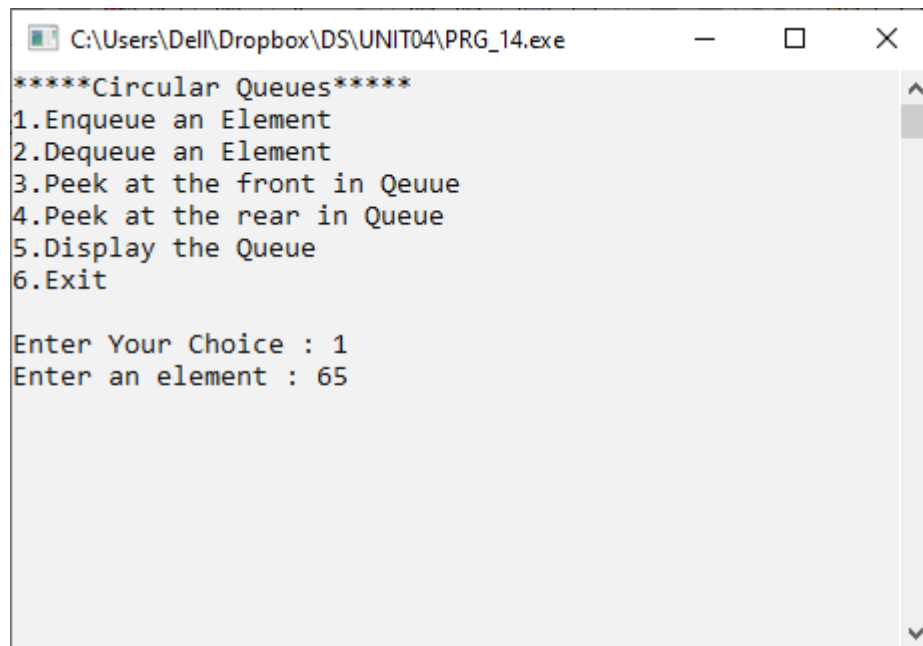
switch(ch)
{
    case 1:
        cout << "Enter an element :
        "; cin >> num;
        c.Enqueue(num
        ); getch();
        break;
    case 2:
        c.Dequeue();
        getch()
        ;
    case 3: break;

        c.PeekFront(
        ); getch();
    case 4: break;

        c.PeekRear();
        getch()
    case 5: ;
        break;

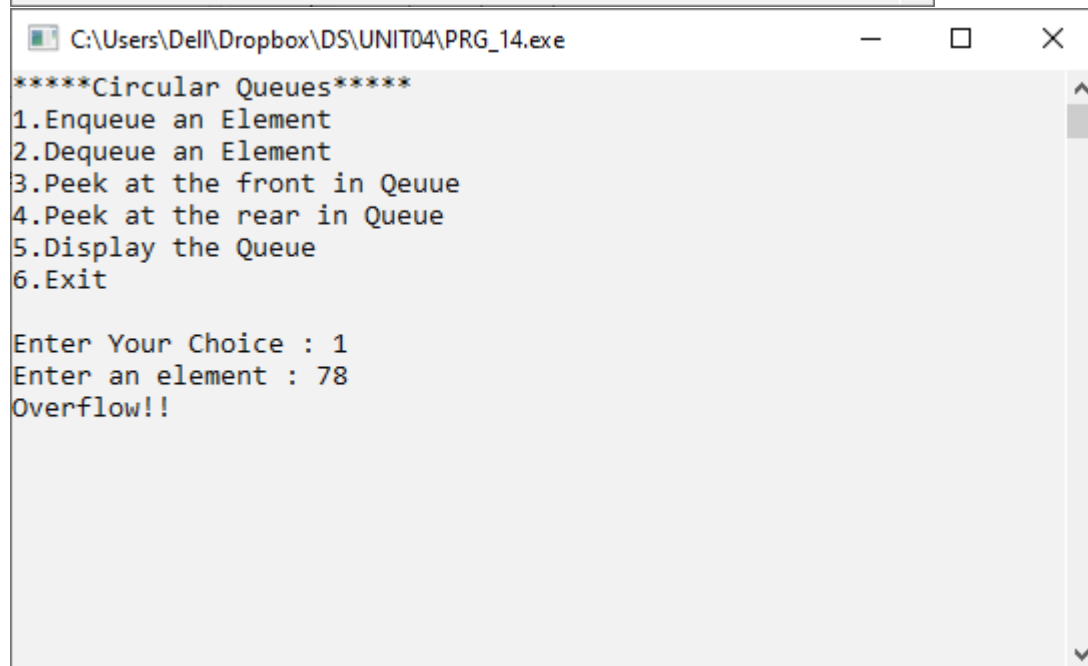
        c.Display();
    case 6: getch()
        ;
        break;

        exit(1);
    default:
        cout << "Incorrect
        Choice."; getch();
        break;
}
}
} //end of menu
```

**Output:**

```
C:\Users\Del\Dropbox\DS\UNIT04\PRG_14.exe
*****Circular Queues*****
1.Enqueue an Element
2.Dequeue an Element
3.Peek at the front in Queue
4.Peek at the rear in Queue
5.Display the Queue
6.Exit

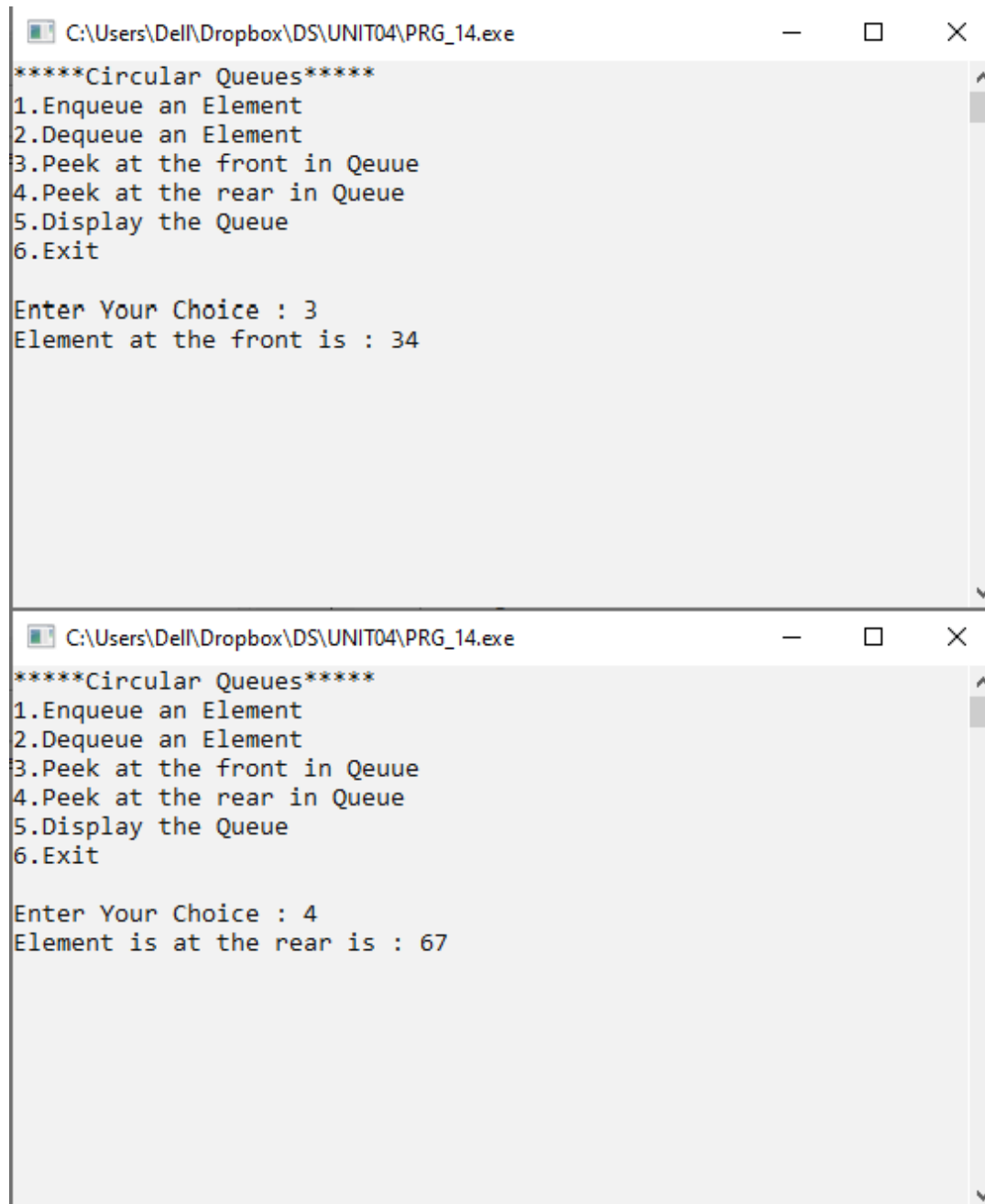
Enter Your Choice : 1
Enter an element : 65
```



```
C:\Users\Del\Dropbox\DS\UNIT04\PRG_14.exe
*****Circular Queues*****
1.Enqueue an Element
2.Dequeue an Element
3.Peek at the front in Queue
4.Peek at the rear in Queue
5.Display the Queue
6.Exit

Enter Your Choice : 1
Enter an element : 78
Overflow!!
```



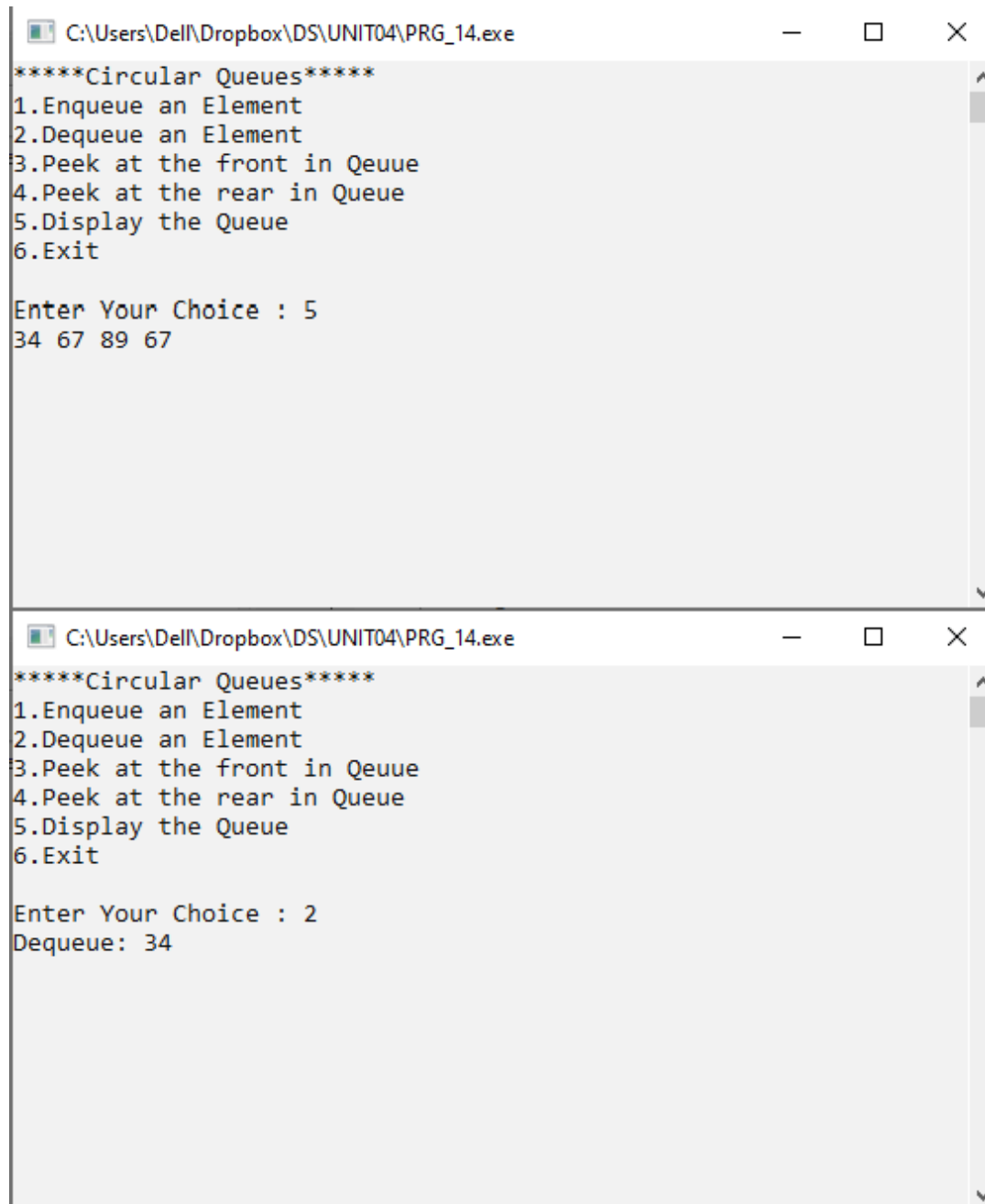


The image shows two screenshots of a Windows command prompt window titled "C:\Users\Dell\Dropbox\DS\UNIT04\PRG\_14.exe". Both windows display the same menu for "Circular Queues":

```
*****Circular Queues*****  
1.Enqueue an Element  
2.Dequeue an Element  
3.Peek at the front in Queue  
4.Peek at the rear in Queue  
5.Display the Queue  
6.Exit
```

In the top screenshot, the user enters "3" and the output is "Element at the front is : 34".

In the bottom screenshot, the user enters "4" and the output is "Element is at the rear is : 67".



The image shows two screenshots of a Windows command prompt window titled "C:\Users\Dell\Dropbox\DS\UNIT04\PRG\_14.exe".

The top screenshot shows the program's menu and the user's input:

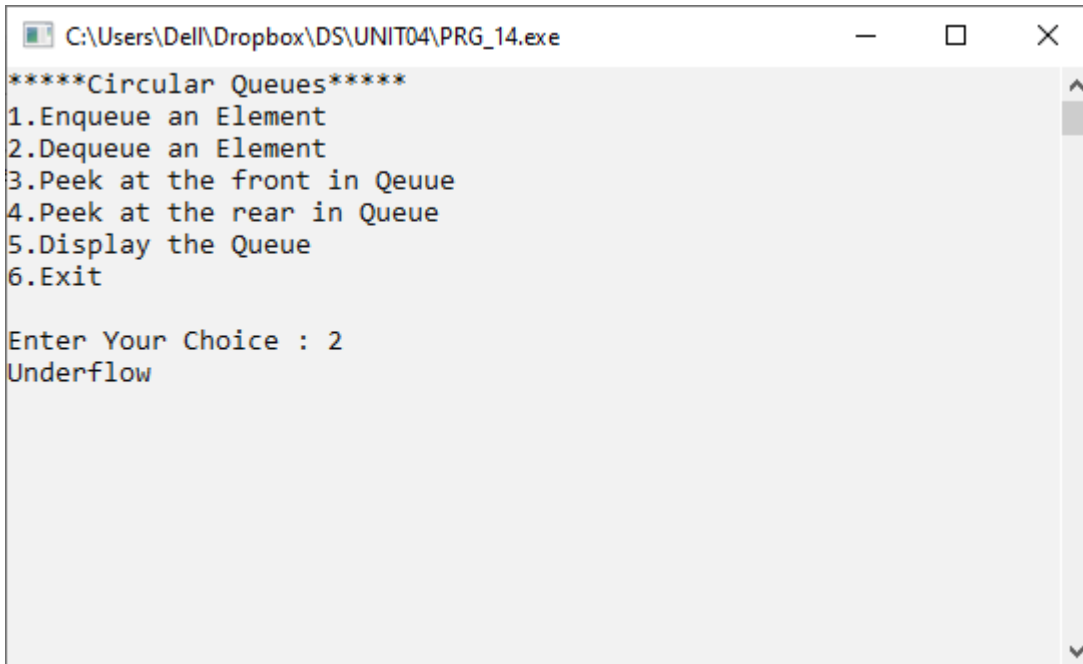
```
*****Circular Queues*****
1.Enqueue an Element
2.Dequeue an Element
3.Peek at the front in Queue
4.Peek at the rear in Queue
5.Display the Queue
6.Exit

Enter Your Choice : 5
34 67 89 67
```

The bottom screenshot shows the program's menu and the user's input, with the output of the dequeue operation:

```
*****Circular Queues*****
1.Enqueue an Element
2.Dequeue an Element
3.Peek at the front in Queue
4.Peek at the rear in Queue
5.Display the Queue
6.Exit

Enter Your Choice : 2
Dequeue: 34
```



```
C:\Users\ DELL\Dropbox\DS\UNIT04\PRG_14.exe
*****Circular Queues*****
1.Enqueue an Element
2.Dequeue an Element
3.Peek at the front in Queue
4.Peek at the rear in Queue
5.Display the Queue
6.Exit

Enter Your Choice : 2
Underflow
```

## Double Ended Queue

### Source Code:

```
/*
    Name : Robin Singh
    Rollno : 1261
    Unit : 04
    Program : Double Ended Queue
*/

#include<iostream>
using namespace std;

// 1. Node
Template class
DQNode
{
    public
    :
        int data;
        DQNode
        *right;
        DQNode *left;
};

// 2. Queue Template

class DQueue
{
    DQNode
    *front;
    DQNode
    *rear;

    public
    :
        DQueue()
        {
            front =
            NULL; rear
            = NULL;
        }

        void EnqueueFront(int
        x ); void
        EnqueueRear(int x);
};
```

```
void DequeueFront();  
void DequeueRear();  
void PeekFront();
```

```

        void
        PeekLast();
        void Display();
};        int Empty();

```

// 3.  
Functions

```
void DQueue::EnqueueFront(int x)
```

```

{
    //make a new node t
    DQNode *t = new
    DQNode(); t->data = x;
    t->right =
    NULL; t->left
    = NULL;
    //First Node in the
    DQNode if(front ==
    NULL)
    {
        front =
        t; rear =
    }    t;
    else
    {
        t->right =
        front; front->
        left=t; front =
    }    t;
}

```

```
void DQueue::EnqueueRear(int x)
```

```

{
    DQNode *t = new
    DQNode(); t->data = x;
    t->right =
    NULL; t->left
    = NULL;

    if(front == NULL) // can use Empty as well
    {
        front =
        t; rear =
    }    t;
    else
    {

```

```
r
e
a
r
-
>
r
i
g
h
t

=

t
;

t
-
>
l
e
f
t

=

r
e
a
r
;
```

```
        rear = t;
    }
}

int DQueue::Empty()
{
    if(front == NULL)
    {
        return 1;
    }
    else
    {
        return 0;
    }
}

void DQueue::DequeueFront()
{
    if(Empty())
    {
        cout <<
            "Underflow!!!"; return;
    }
    DQNode *tmp = front;

    if(front == rear)
    {
        front =
            NULL; rear
            = NULL;
    }
    else
    {
        front = front->right; front->left
            = NULL;
    }
    cout << "Element removed " << tmp->data ; delete tmp;
}

void DQueue::DequeueRear()
{
    if(Empty())
    {
        cout << "Underflow!!!";
```



```

        return;
    }
    DQNode *tmp =
    rear; if(front ==
    rear)
    {
        front =
        NULL; rear
    } = NULL;
    else
    {
        rear = rear->left;
        rear->right =
        NULL;
        cout << "Element removed " << tmp->data;
    }
}

void DQueue::Display()
{
    if(Empty())
    {
        cout <<
        "Underflow!!!"; return;
    }
    DQNode
    *tmp=front; cout <<
    ("Queue :");
    while(tmp != NULL)
    {
        cout << tmp->data << "<-
        >"; tmp = tmp->right;
    }
    cout << "End";
}

void DQueue::PeekFront()
{
    if(Empty())
    {
        cout <<
        "Underflow!!!"; return;
    }
    cout << front->data;
}

void DQueue::PeekLast()
{

```

```
        if(Empty())
        {
            cout <<
            "Underflow!!!"; return;
        }
        cout << rear->data;
    }

int main()
{
    int ch,
    num;
    DQueue
    dq;

    while(1)
    {
        system("cls");
        cout << "*** Double Ended Queue
        ***\n\n\n"; cout << "1. Enqueue Element in
        front\n";
        cout << "2. Enqueue Element in
        Rear\n"; cout << "3. Dequeue from
        front\n";
        cout << "4. Dequeue from rear\n";
        cout << "5. Peek the front
        element\n"; cout << "6. Peek the
        rear element\n"; cout << "7.
        Display the Queue\n"; cout << "8.
        Exit\n\n\n";

        cout << "Enter your
        choice:"; cin >> ch;

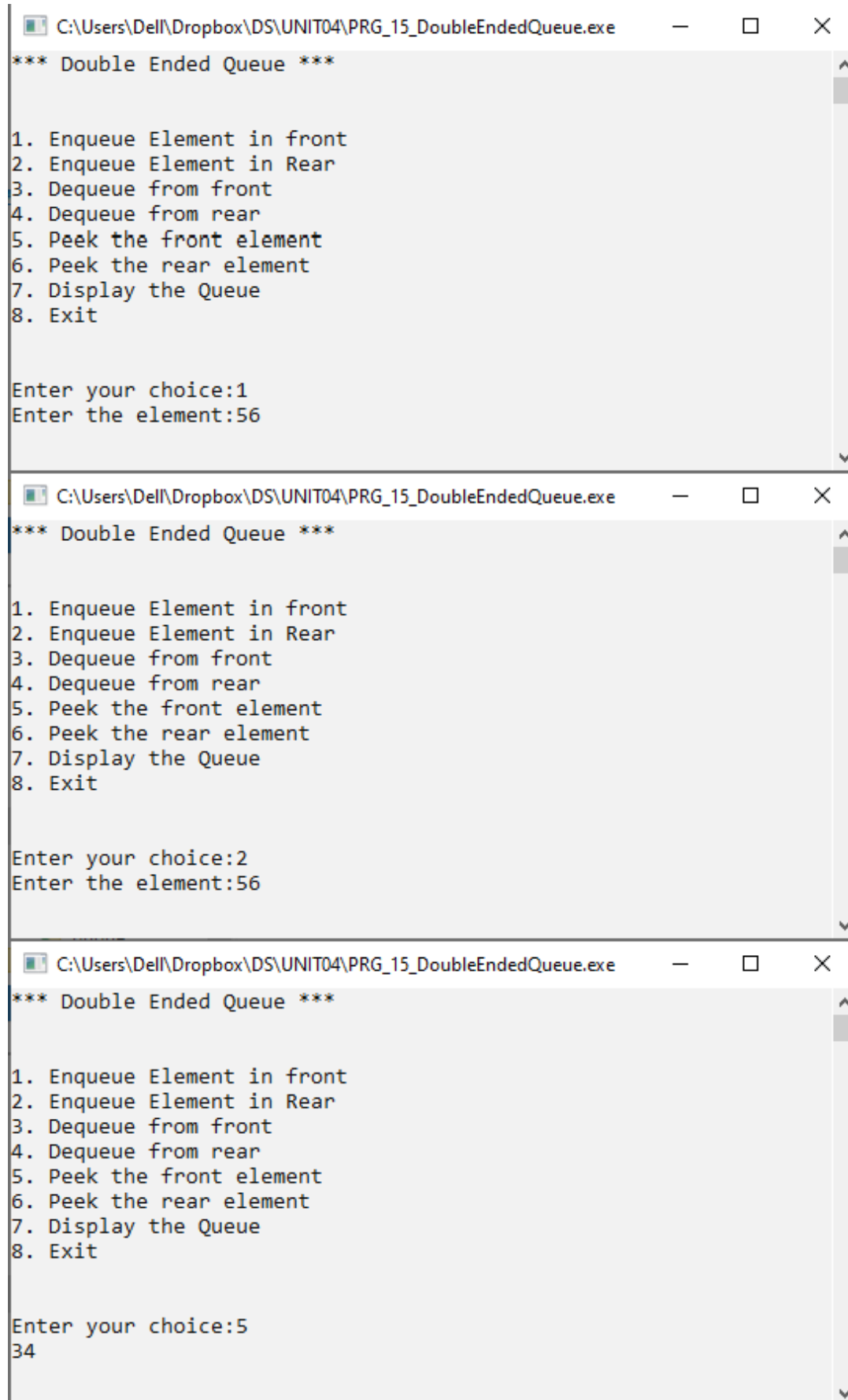
        switch(ch)
        {
            case 1:
                cout << "Enter the
                element:"; cin >> num;
                dq.EnqueueFront(num);
                getch();
                break;

            case 2:
                cout << "Enter the
                element:"; cin >> num;
                dq.EnqueueRear(num);
                getch();
```

break;

```
        case 3:
            dq.DequeueFront(
            ); getch();
            break;
        case 4:
            dq.DequeueRear
            (); getch();
            break;
        case 5:
            dq.PeekFront(
            ); getch();
            break;
        case 6:
            dq.PeekLast
            (); getch();
            break;
        case 7:
            dq.Display();
            getch()
            ;
        case 8: break;

            exit(1);
        default:
            cout << "Enter a valid
            choice!!!"; getch();
            break;
        }
    }
}
```

**Output:**

```
*** Double Ended Queue ***

1. Enqueue Element in front
2. Enqueue Element in Rear
3. Dequeue from front
4. Dequeue from rear
5. Peek the front element
6. Peek the rear element
7. Display the Queue
8. Exit

Enter your choice:1
Enter the element:56

*** Double Ended Queue ***

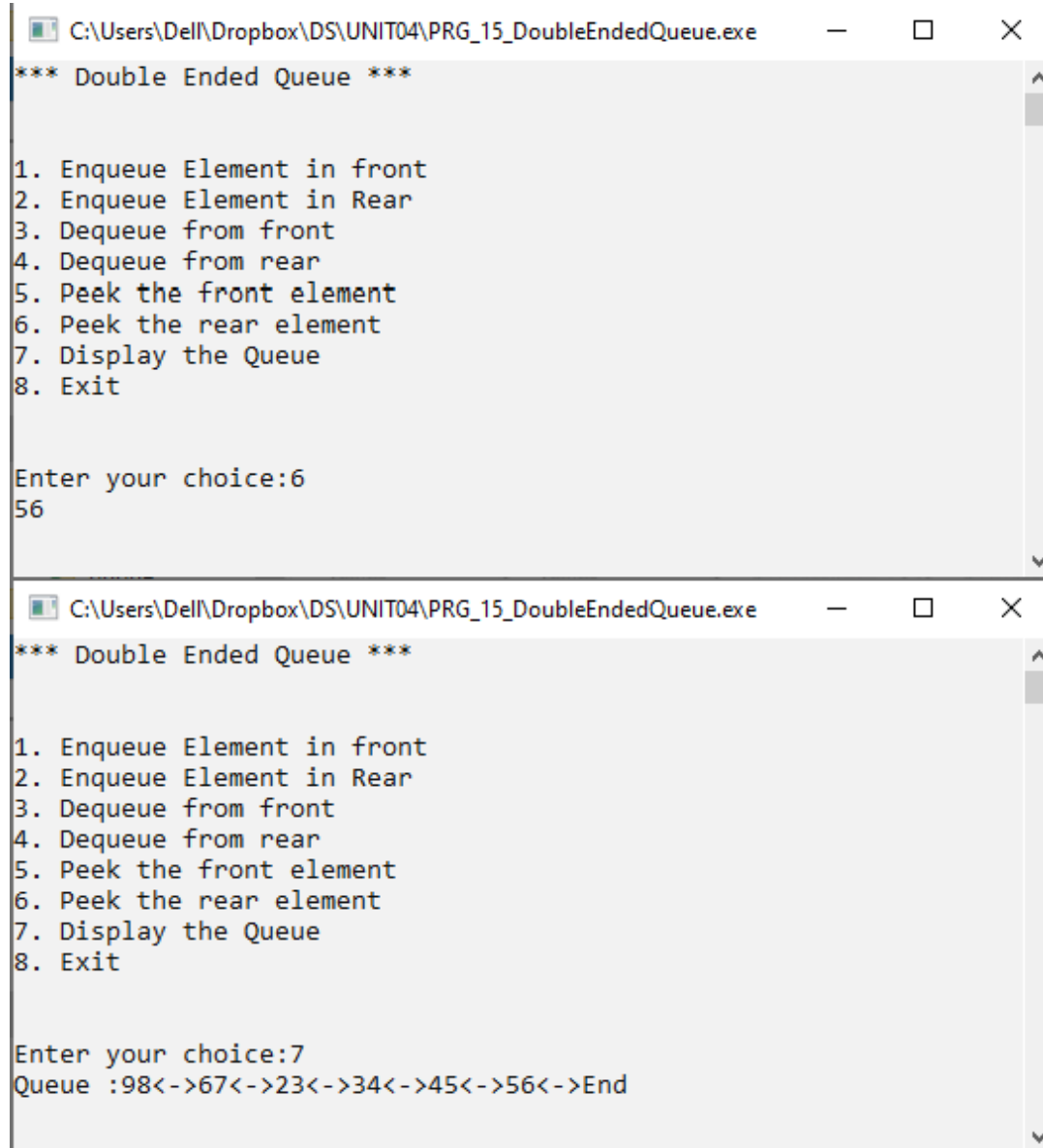
1. Enqueue Element in front
2. Enqueue Element in Rear
3. Dequeue from front
4. Dequeue from rear
5. Peek the front element
6. Peek the rear element
7. Display the Queue
8. Exit

Enter your choice:2
Enter the element:56

*** Double Ended Queue ***

1. Enqueue Element in front
2. Enqueue Element in Rear
3. Dequeue from front
4. Dequeue from rear
5. Peek the front element
6. Peek the rear element
7. Display the Queue
8. Exit

Enter your choice:5
34
```



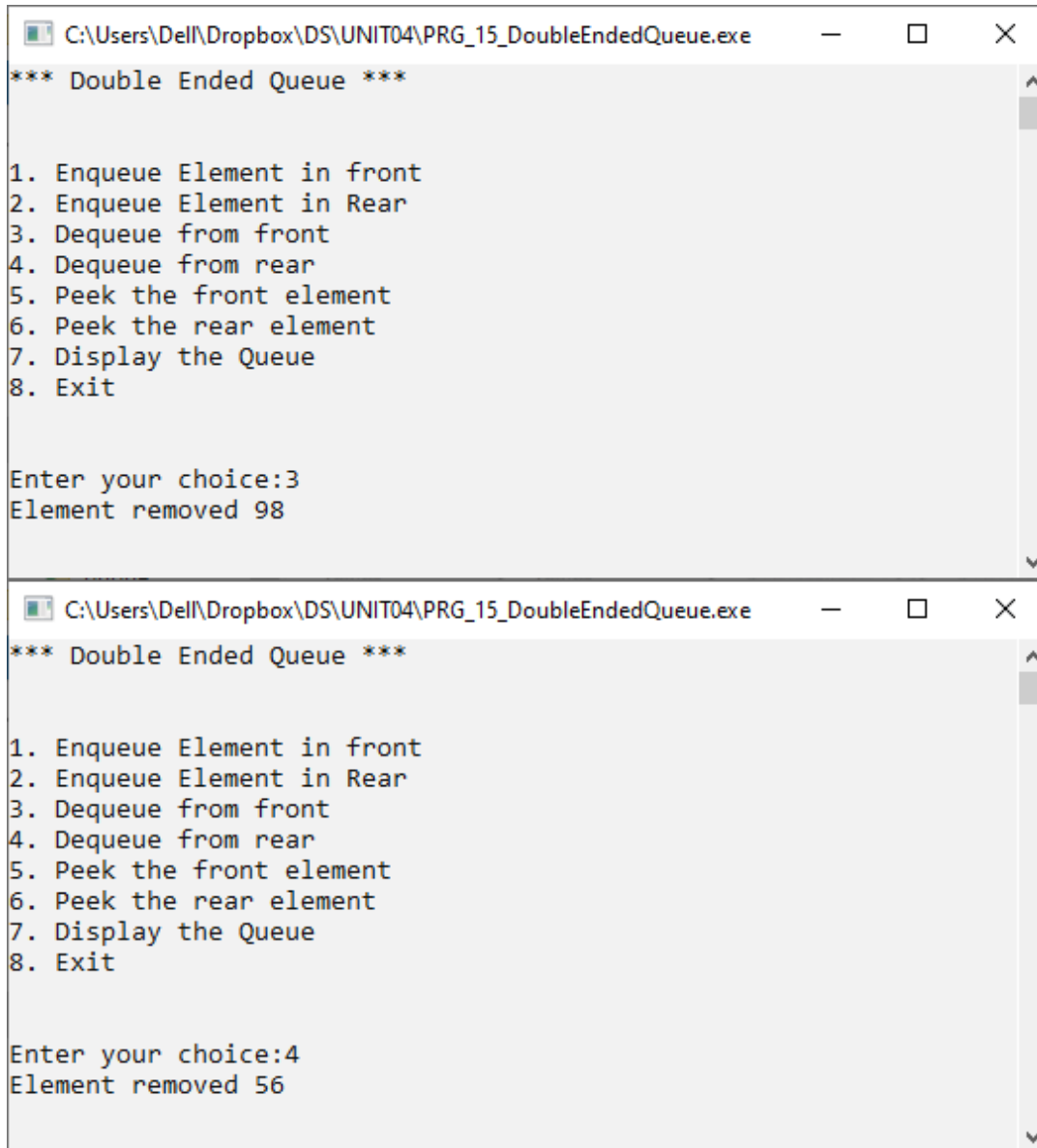
The image shows two screenshots of a Windows command prompt window titled "C:\Users\Dell\Dropbox\DS\UNIT04\PRG\_15\_DoubleEndedQueue.exe".

The first screenshot shows the program's menu:

```
*** Double Ended Queue ***  
  
1. Enqueue Element in front  
2. Enqueue Element in Rear  
3. Dequeue from front  
4. Dequeue from rear  
5. Peek the front element  
6. Peek the rear element  
7. Display the Queue  
8. Exit  
  
Enter your choice:6  
56
```

The second screenshot shows the program after choice 7 was selected:

```
*** Double Ended Queue ***  
  
1. Enqueue Element in front  
2. Enqueue Element in Rear  
3. Dequeue from front  
4. Dequeue from rear  
5. Peek the front element  
6. Peek the rear element  
7. Display the Queue  
8. Exit  
  
Enter your choice:7  
Queue :98<->67<->23<->34<->45<->56<->End
```



The image shows two screenshots of a Windows command prompt window titled "C:\Users\Dell\Dropbox\DS\UNIT04\PRG\_15\_DoubleEndedQueue.exe".

The top screenshot displays the menu:

```
*** Double Ended Queue ***  
  
1. Enqueue Element in front  
2. Enqueue Element in Rear  
3. Dequeue from front  
4. Dequeue from rear  
5. Peek the front element  
6. Peek the rear element  
7. Display the Queue  
8. Exit
```

The user entered choice 3, and the output was:

```
Enter your choice:3  
Element removed 98
```

The bottom screenshot shows the same menu. The user entered choice 4, and the output was:

```
Enter your choice:4  
Element removed 56
```

## Priority Queue

### Source Code:

```
/*
Name : Robin Singh
Roll number : 1261
unit : 04
program : Priority Queue
*/
#include<iostream>
using namespace std;
class PQNode
{
    public
    :       int data;
           int priority;
           PQNode *next;
};

class PQueue
{
    PQNode
    *front;
    PQNode
    *rear;

    public
    :       PQueue()
            {
                front = NULL;
            }
            void Enqueue(int x, int
            p); void Dequeue();
            void
            Display(); int
            Empty();
};

/*functions */
/*Enqueue function*/
```



---

```
void PQueue :: Enqueue(int x,int p)
```

```

{
    //make new node
    PQNode * t =new
    PQNode(); t->data=x;
    t->priority=p;
    t-
    >next=NULL
    ;

    //first node in
    pqueue if(front ==
    NULL)
    {
        front =
        t;
        return;
    }

    PQNode
    *tmp=front;
    PQNode
    *prev=NULL;
    while(tmp && tmp->priority<t->priority)
    {
        prev = tmp;
        tmp = tmp->next;
    }
    if(tmp==NULL)//insert at the rear
    {
        prev->next=t;
    }
    else if(prev==NULL)//t inserted at the front
    {
        t-
        >next=front;
        front=t;
    }
    else//t inserted in the middle
    {
        prev-
        >next=t; t-
        >next=tmp;
    }
}
/*Dequeue function*/
void PQueue ::
Dequeue()
{

```

---

```
if(front==NULL)
{
    cout<<"Underflow . ";
    return;
}
```

```

PQNode
*tmp=front;
if(front==rear)
{
    front=NUL
    L;
}    rear=NULL
els    ;
e
{

}    front=front->next;
cout<<"Dequeued element is "<<tmp->data<<" with priority : "<<tmp-
>priority; delete tmp;
}

```

```

/*Display function*/
void PQueue ::
Display()
{
    if(front==NULL)
    {
        cout<<"Underflow . ";
        return;
    }
    els
    e
    {
        PQNode * tmp =
        front; while(tmp)
        {
            cout<< tmp->data<<","<<tmp->priority<<"-
            >"; tmp = tmp -> next;
        }
        cout << "end";
    }
}

```

```

/*menu*/
int main()
{
    PQueue p1;
    int
    num,ch,p;

```

while(1)

```
{
    system("cls");
    cout<<"-----Priority Queue -----"<<endl;
    cout<<"1. Enqueue "<<endl;
    cout<<"2. Dequeue "<<endl;
    cout<<"3. Display "<<endl;
    cout<<"4.Exit "<<endl;
    cout<<"Enter your choice :
"<<endl; cin>>ch;
    switch(ch)
    {
        case 1:
            cout<<"Enter a number
:\n"; cin>>num;
            cout<<"Enter priority :
"; cin>>p;
            p1.Enqueue(num,p);

            getch()
            ;
            break;

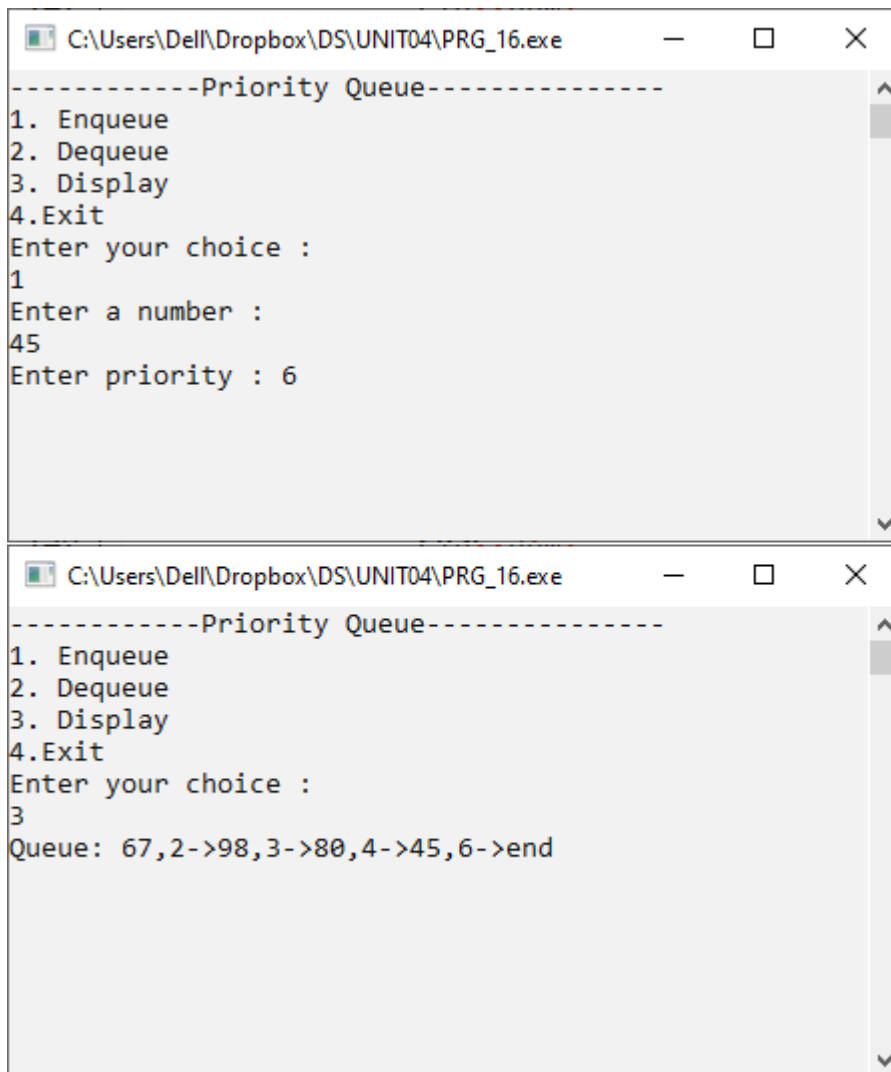
        case 2:
            cout<<"Dequeue
operation"<<endl; p1.Dequeue();
            getch()
            ;
            break;

        case 3:
            cout<<"Queue:
"; p1.Display();

            getch()
            ;
            break;

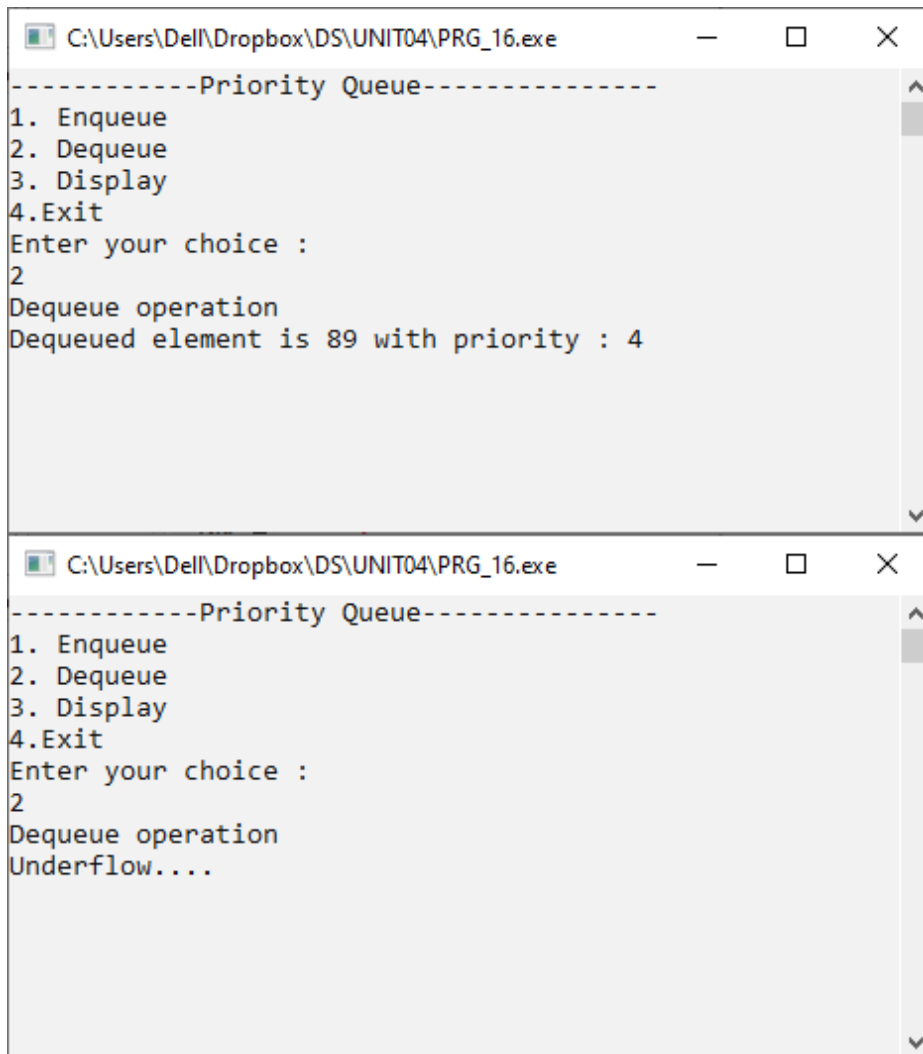
        case
        4:
            exit(1)
            ;

        default:
            cout<<"Invalid
",
            ;
    }
}
```

**Output:**

```
-----Priority Queue-----
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice :
1
Enter a number :
45
Enter priority : 6

-----Priority Queue-----
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice :
3
Queue: 67,2->98,3->80,4->45,6->end
```



```
-----Priority Queue-----
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice :
2
Dequeue operation
Dequeued element is 89 with priority : 4

-----Priority Queue-----
1. Enqueue
2. Dequeue
3. Display
4. Exit
Enter your choice :
2
Dequeue operation
Underflow....
```