| Program No | 5 |
| --- | --- |
| Roll No | 1313 |
| Topic | Queue |
| Unit | 4 |
| Title of Program | Ordianary Queue |

**Source Code: Ordinary Queue**

**/\* Name : Yash Ishan Gohil**

**Roll No: 1314**

**Unit 4: Queues**

**Program: Ordinary Queue**

**\*/**

**#include<iostream>**

**#include<conio.h>**

**#define MAX 5**

**using namespace std;**

**//1.Node Template - Not Required**

**//2.Queue Template**

**class OQueue**

**{**

**int a[MAX];**

**int front , rear;**

**public:**

**OQueue()**

**{**

**front =-1;**

**rear=-1;**

**}**

**void Enqueue(int x);**

**void Dequeue();**

**void PeekFront();**

**void PeekRear();**

**void Display();**

**int Full();**

**int Empty();**

**};**

**//3.Functions**

**void OQueue :: Enqueue(int x)**

**{**

**if(Full())**

**{**

**cout << "Queue Overflow!"<< endl;**

**return;**

**}**

**if( front == -1)**

**{**

**front ++;**

**}**

**rear++;**

**a[rear] = x;**

**}**

**int OQueue :: Full()**

**{**

**return (rear == MAX -1 ? 1:0);**

**}**

**void OQueue :: Dequeue()**

**{**

**if(Empty())**

**{**

**cout << "Queue Underflow!";**

**}**

**int t = a[front];**

**if(front == rear) //Single element**

**{**

**front = -1;**

**rear = -1;**

**}**

**else**

**{**

**front++;**

**}**

**cout << "Element Removed is:" << t;**

**}**

**int OQueue :: Empty()**

**{**

**if(front == -1)**

**{**

**return 1;**

**}**

**else**

**{**

**return 0;**

**}**

**}**

**void OQueue :: Display()**

**{**

**if(Empty())**

**{**

**cout << "Underflow";**

**return;**

**}**

**for(int i=front; i<=rear; i++)**

**{**

**cout << a[i] << " ";**

**}**

**}**

**void OQueue :: PeekFront()**

**{**

**if(Empty())**

**{**

**cout << "Underflow";**

**return;**

**}**

**cout << "Element at Front is:" << a[front];**

**}**

**void OQueue :: PeekRear()**

**{**

**if(Empty())**

**{**

**cout << "Underflow";**

**return;**

**}**

**cout << "Element at rear is:" << a[rear];**

**}**

**//4.Menu**

**int main()**

**{**

**int ch, num;**

**OQueue q;**

**while(1)**

**{**

**system("cls");**

**cout << "\*\*\*Ordinary Queue \*\*\*\n\n";**

**cout << "1.Enqueue an element" << endl;**

**cout << "2.Dequeue" << endl;**

**cout << "3.Peek Front" << endl;**

**cout << "4.Peek Rear" << endl;**

**cout << "5.Display the queue" << endl;**

**cout << "6.exit" <<endl <<endl;**

**cout << "Enter your choice:";**

**cin>> ch;**

**switch(ch)**

**{**

**case 1:**

**cout << "Enter an element: ";**

**cin >> num;**

**q.Enqueue(num);**

**getch();**

**break;**

**case 2:**

**q.Dequeue();**

**getch();**

**break;**

**case 3:**

**q.PeekFront();**

**getch();**

**break;**

**case 4:**

**q.PeekRear();**

**getch();**

**break;**

**case 5:**

**q.Display();**

**getch();**

**break;**

**case 6:**

**exit(1);**

**default :**

**cout << "Incorrect Choice:";**

**break;**

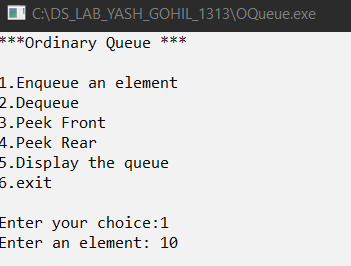
**}**

**}//end of switch**

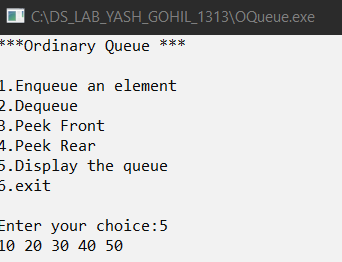
**}//end of while**

**OUTPUT:**

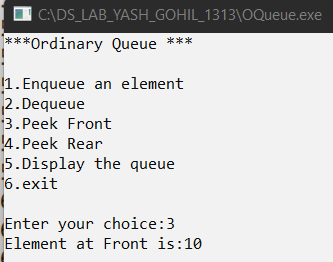
1. **Enqueue an element:**

****

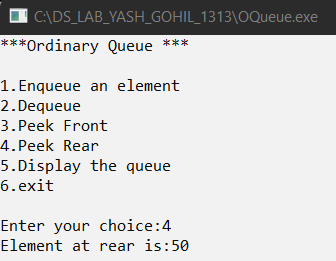
1. **Display the queue:**

****

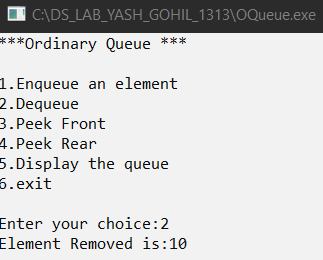
1. **Peek front:**

****

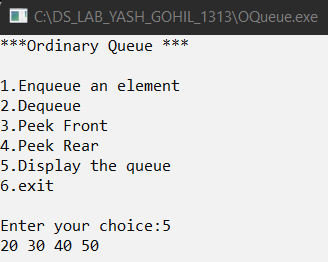
1. **Peek rear:**

****

1. **Dequeue an element:**

****

1. **Display the Dequeued queue:**

****

| Program No | 5 |
| --- | --- |
| Roll No | 1313 |
| Topic | Queue |
| Unit | 4 |
| Title of Program | Circular Queue |

**Source Code:**

**/\***

**Name: Yash Ishan Gohil**

**Roll No: 1313**

**Unit 4: Queues**

**Program: Circular Queues \*/**

**#include<iostream>**

**#include<conio.h>**

**#define MAX 5**

**using namespace std;**

**//1.Node Template - Not Required**

**//2.Queue Template**

**class CQueue**

**{**

**int a[MAX];**

**int front , 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();**

**};**

**//3.Functions**

**void CQueue :: Enqueue(int x)**

**{**

**if(Full())**

**{**

**cout << "Queue Overflow";**

**return;**

**}**

**if(front == -1)**

**{**

**front ++;**

**}**

**rear = (rear+1) % MAX;**

**a[rear] = x;**

**cnt++;**

**}**

**int CQueue :: Full()**

**{**

**if(cnt == MAX )**

**{**

**return 1;**

**}**

**else**

**{**

**return 0;**

**}**

**}**

**void CQueue :: Dequeue()**

**{**

**if (Empty())**

**{**

**cout << "Queue Underflow!";**

**return;**

**}**

**int t = a[front];**

**if(front == rear) //Single Element**

**{**

**front = -1;**

**rear = -1;**

**}**

**else**

**{**

**front = (front+1)%MAX;**

**}**

**cout << "Element removed is:" << t;**

**cnt--;**

**}**

**int CQueue :: Empty()**

**{**

**if(cnt==0)**

**{**

**return 1;**

**}**

**else**

**{**

**return 0;**

**}**

**}**

**void CQueue :: Display()**

**{**

**if(Empty())**

**{**

**cout << "Queue Undeflow";**

**return;**

**}**

**int x, i = front;**

**for(x=1 ; x<=cnt ; x++)**

**{**

**cout << a[i] << " ";**

**i = (i+1) % MAX;**

**}**

**}**

**void CQueue :: PeekFront()**

**{**

**if(Empty())**

**{**

**cout << "Underflow";**

**return;**

**}**

**cout << "Element at front is:" << a[front];**

**}**

**void CQueue :: PeekRear()**

**{**

**if(Empty())**

**{**

**cout << "Underflow";**

**return;**

**}**

**cout << "Element at rear is:" << a[rear];**

**}**

**//4.Menu**

**int main()**

**{**

**int ch, num;**

**CQueue c;**

**while(1)**

**{**

**system("cls");**

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

**cout << "1.Enqueue an element" << endl;**

**cout << "2.Dequeue" << endl;**

**cout << "3.Peek Front" << endl;**

**cout << "4.Peek Rear" << endl;**

**cout << "5.Display the queue" << endl;**

**cout << "6.exit" <<endl <<endl;**

**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();**

**break;**

**case 3:**

**c.PeekFront();**

**getch();**

**break;**

**case 4:**

**c.PeekRear();**

**getch();**

**break;**

**case 5:**

**c.Display();**

**getch();**

**break;**

**case 6:**

**exit(1);**

**default :**

**cout << "Incorrect Choice:";**

**getch();**

**break;**

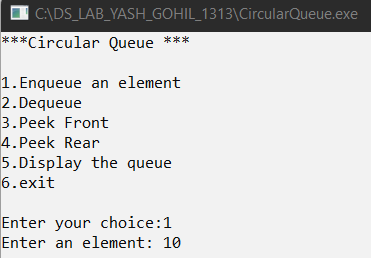
**}**

**}//end of switch**

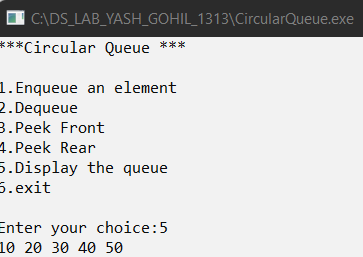
**}//end of while**

**Output:**

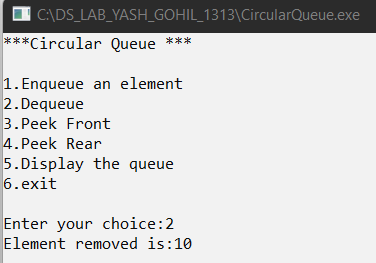
**1)Enqueue an element:**

****

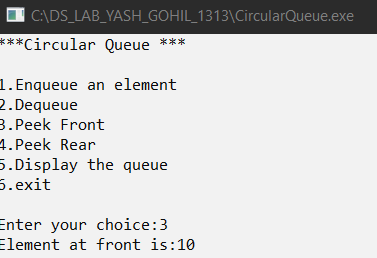
**2)Display the queue:**

****

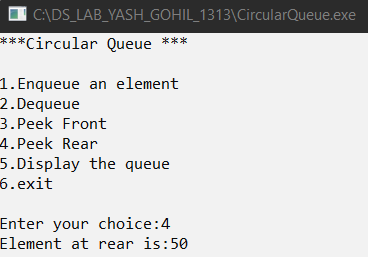
**3)Dequeue an element:**

****

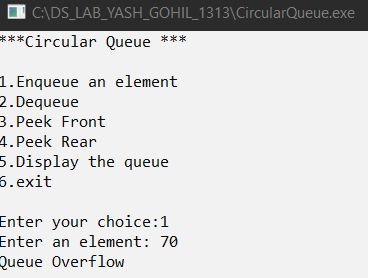
**4)Peekfront:**

****

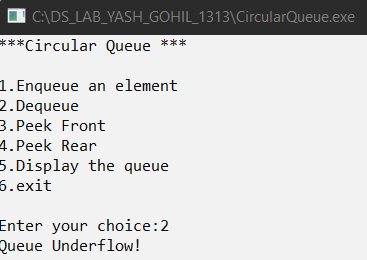
**5)PeekRear:**

****

**6)Overflow Condition:**

****

**7)Underflow Condition:**

****

| Program No | 5 |
| --- | --- |
| Roll No | 1313 |
| Topic | Queue |
| Unit | 4 |
| Title of Program | Double Ended - Queue |

**Source Code:**

**/\***

**Name: Yash Ishan Gohil**

**Roll No:1313**

**UNIT 4:Queues**

**Program:Double Ended-Queue**

**\*/**

**#include<iostream>**

**#include<conio.h>**

**using namespace std;**

**/\* 1. Node Template\*/**

**class DQNode**

**{**

**public:**

**int data;**

**DQNode \*right;**

**DQNode \*left;**

**};**

**/\* 2. Double Ended-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 PeekRear();**

**void Display();**

**};**

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

**{**

**//Make a new Node**

**DQNode \*t = new DQNode();**

**t->data = x;**

**t->right = NULL;**

**t->left = NULL;**

**//first node in the queue**

**if(front==NULL)**

**{**

**front = t;**

**rear = t;**

**}**

**else //Any other node**

**{**

**t->right = front;**

**front->left = t;**

**front = t;**

**}**

**}**

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

**{**

**// Make a node**

**DQNode \*t = new DQNode();**

**t->data = x;**

**t->right = NULL;**

**t->left = NULL;**

**// special case first case**

**if(front==NULL)**

**{**

**front = t;**

**rear = t;**

**}**

**else//any other node**

**{**

**rear->right = t;**

**t->left = rear;**

**rear = t;**

**}**

**}**

**void DQueue :: Display()**

**{**

**if (front == NULL)**

**{**

**cout << "Queue Overflow";**

**return;**

**}**

**DQNode \*tmp = front;**

**while(tmp != NULL)**

**{**

**cout << tmp->data << "<->";**

**tmp = tmp->right;**

**}**

**cout << "NULL";**

**}**

**void DQueue :: DequeueFront()**

**{**

**//Empty Queue**

**if(front== NULL)**

**{**

**cout << "Queue Undeflow";**

**return;**

**}**

**DQNode \*tmp = front;**

**if(front == rear)//single node deletion**

**{**

**front = NULL;**

**rear = NULL;**

**}**

**else**

**{**

**front = front->right;**

**front->left = NULL;**

**}**

**cout << "Element removed:" <<tmp->data;**

**delete tmp;**

**}**

**void DQueue :: DequeueRear()**

**{**

**//Empty Queue**

**if(front == NULL)**

**{**

**cout << "Queue Underflow";**

**return;**

**}**

**DQNode \*tmp = rear;**

**if(front == rear)//single node deletion**

**{**

**front = NULL;**

**rear = NULL;**

**}**

**else**

**{**

**rear = rear->left;**

**rear->right = NULL;**

**}**

**cout <<"Element removed:" <<tmp->data;**

**delete tmp;**

**}**

**void DQueue :: PeekFront()**

**{**

**if(front == NULL)**

**{**

**cout <<"Underflow";**

**return;**

**}**

**cout << "Element at front is:" <<front->data;**

**}**

**void DQueue :: PeekRear()**

**{**

**if(front==NULL)**

**{**

**cout <<"Queue overflow";**

**return;**

**}**

**cout << "Element at the rear is: "<< rear->data;**

**}**

**/\* 4. Menu\*/**

**int main()**

**{**

**int ch, num;**

**DQueue d;**

**while(1)**

**{**

**system("cls");**

**cout << "\*\*\* Double Ended-Queue \*\*\*\n\n";**

**cout << "1.Enqueue Front" << endl;**

**cout << "2.Enqueue Rear" << endl;**

**cout << "3.Dequeue Front" << endl;**

**cout << "4.Dequeue Rear" << endl;**

**cout << "5.Peek Front" << endl;**

**cout << "6.Peek Rear" <<endl;**

**cout << "7.Dispaly the Queue" <<endl;**

**cout << "8.Exit" <<endl <<endl;**

**cout << "Enter your choice:";**

**cin>> ch;**

**switch(ch)**

**{**

**case 1:**

**cout << "Enter an element: ";**

**cin >> num;**

**d.EnqueueFront(num);**

**getch();**

**break;**

**case 2:**

**cout << " Enter element at rear:";**

**cin >> num;**

**d.EnqueueRear(num);**

**getch();**

**break;**

**case 3:**

**d.DequeueFront();**

**getch();**

**break;**

**case 4:**

**d.DequeueRear();**

**getch();**

**break;**

**case 5:**

**d.PeekFront();**

**getch();**

**break;**

**case 6:**

**d.PeekRear();**

**getch();**

**break;**

**case 7:**

**d.Display();**

**getch();**

**break;**

**case 8:**

**exit(1);**

**default :**

**cout << "Incorrect Choice:";**

**break;**

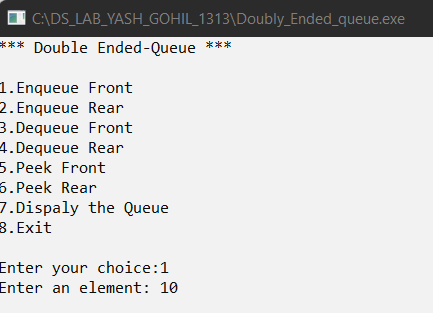
**}//end of switch**

**}//end of while**

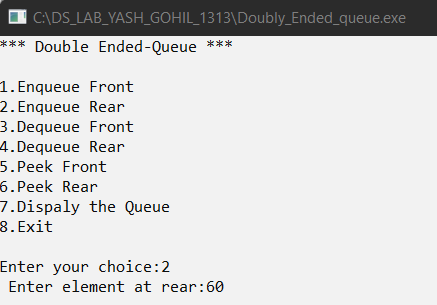
**}//end of main**

**Output:**

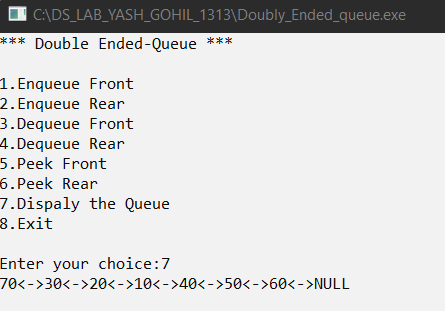
**1)Enqueue at Front:**

****

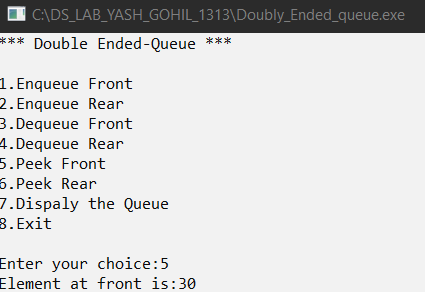
**2)Enqueue at Rear:**

****

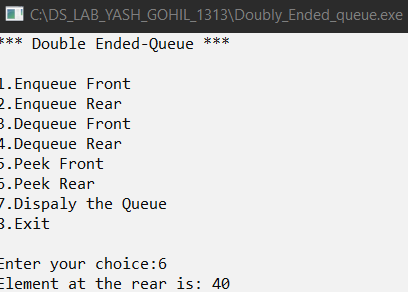
**3)Display the queue:**

****

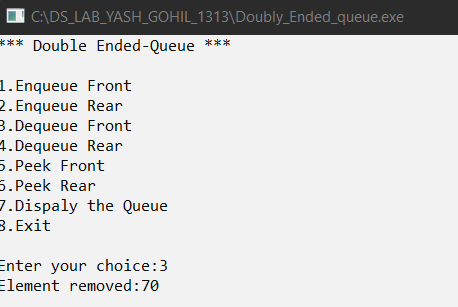
**4)Peek Front:**

****

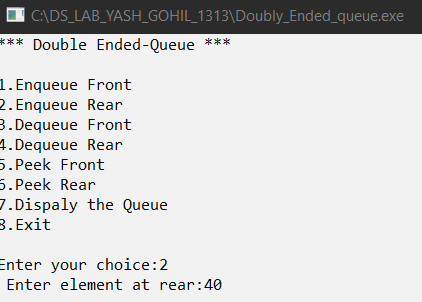
**5)Peek Rear:**

****

**6)Dequeue Front:**

****

**7)Dequeue Rear:**

****

| Program No | 5 |
| --- | --- |
| Roll No | 1313 |
| Topic | Queue |
| Unit | 4 |
| Title of Program | Priority Queue |

**Source Code:**

**/\***

**Name: Yash Ishan Gohil**

**Roll No:1313**

**Unit 4:Queues**

**Program:Priority Queue**

**\*/**

**#include<iostream>**

**#include<conio.h>**

**using namespace std;**

**/\* 1.node template \*/**

**class PQNode**

**{**

**public:**

**int data;**

**int priority;**

**PQNode \*next;**

**};**

**/\*2. Queue template\*/**

**class PQueue**

**{**

**PQNode \*front;**

**public:**

**PQueue()**

**{**

**front = NULL;**

**}**

**void Enqueue(int x, int p);**

**void Dequeue();**

**void PeekFront();**

**void PeekRear();**

**void Display();**

**};**

**/\*3.Functions\*/**

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

**{**

**//Make a new Node**

**PQNode \*t = new PQNode();**

**t->data = x;**

**t->priority = p;**

**t->next = NULL;**

**//First Node in the List**

**if(front == NULL)**

**{**

**front = t;**

**return;**

**}**

**//Ordered Traversal**

**PQNode \*tmp = front;**

**PQNode \*prev;**

**while(tmp!= NULL && tmp->priority<p)**

**{**

**prev = tmp;**

**tmp = tmp->next;**

**}**

**//Insert t in queue in order of priority**

**if(tmp == front)**

**{**

**t->next = front;**

**front = t;**

**}**

**else if(tmp == NULL)**

**{**

**prev->next = t;**

**}**

**else**

**{**

**prev->next = t;**

**t->next = tmp;**

**}**

**}**

**void PQueue :: Display()**

**{**

**if(front == NULL)**

**{**

**cout << "Queue Underflow!";**

**return;**

**}**

**PQNode \*tmp = front;**

**while(tmp!= NULL)**

**{**

**cout << "Data: " << tmp->data << "Priority: " << tmp->priority <<"->"<<endl;**

**tmp = tmp->next;**

**}**

**}**

**void PQueue :: Dequeue()**

**{**

**if(front == NULL)**

**{**

**cout << "Queue underflow";**

**return;**

**}**

**PQNode \*tmp = front;**

**if(front->next == NULL)**

**{**

**front = NULL;**

**}**

**else**

**{**

**front = front->next;**

**}**

**cout << "Element Dequeued :" <<tmp->data;**

**cout << "with Priority: " <<tmp->priority;**

**delete tmp;**

**}**

**void PQueue :: PeekFront()**

**{**

**if(front == NULL)**

**{**

**cout <<"Queue Underflow";**

**return;**

**}**

**cout << "Element at front is: " <<front->data << "And its Priority: " << front->priority ;**

**}**

**void PQueue :: PeekRear()**

**{**

**if(front==NULL)**

**{**

**cout <<"Queue overflow";**

**return;**

**}**

**//Traverse till last node**

**PQNode \*tmp = front;**

**while(tmp->next != NULL)**

**{**

**tmp = tmp->next;**

**}**

**cout << "Element at rear is:" << tmp->data;**

**cout << "with Priority:" << tmp->priority<<endl;**

**}**

**/\*4.Menu\*/**

**int main()**

**{**

**int ch, num, pri;**

**PQueue p;**

**while(1)**

**{**

**system("cls");**

**cout << "\*\*\* Priority Queue \*\*\*\n\n";**

**cout << "1.Enqueue Front" << endl;**

**cout << "2.Dequeue Front" << endl;**

**cout << "3.Peek Front" << endl;**

**cout << "4.Peek Rear" <<endl;**

**cout << "5.Dispaly the Queue" <<endl;**

**cout << "6.Exit" <<endl <<endl;**

**cout << "Enter your choice:";**

**cin>> ch;**

**switch(ch)**

**{**

**case 1:**

**cout << "Enter an element: ";**

**cin >> num;**

**cout << "Enter and its Priority:";**

**cin >> pri;**

**p.Enqueue(num,pri);**

**getch();**

**break;**

**case 2:**

**p.Dequeue();**

**getch();**

**break;**

**case 3:**

**p.PeekFront();**

**getch();**

**break;**

**case 4:**

**p.PeekRear();**

**getch();**

**break;**

**case 5:**

**p.Display();**

**getch();**

**break;**

**case 6:**

**exit(1);**

**default :**

**cout << "Incorrect Choice:";**

**break;**

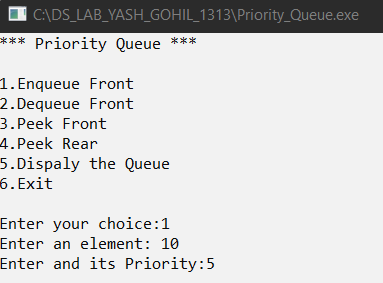
**}//end of switch**

**}//end of while**

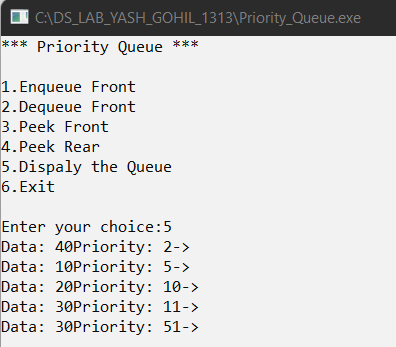
**}//end of main**

**Output:**

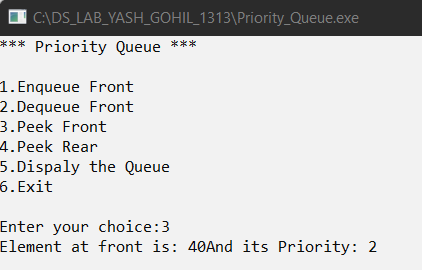
**1)Enqueue element with priority:**

****

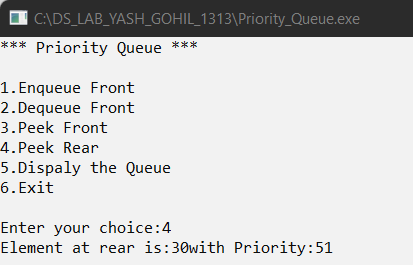
**2)Display Queue:**

****

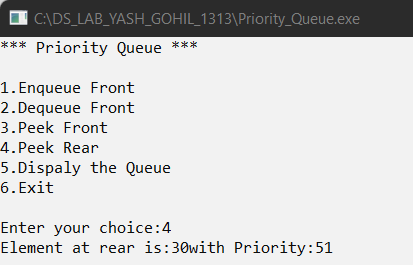
**3)PeekFront:**

****

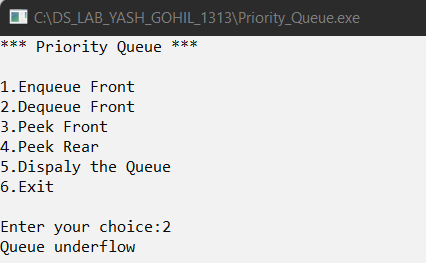
**4)PeekRear:**

****

**5)Dequeue an Element:**

****

**6)Underflow Condition:**

****