**Lab Assignment No: 05**

**NAME:** TAKANKHAR SHUBHAM

**ROLLNO:** 54

**SUBJECT CODE**: IT8002

**SUBJECT NAME**: CPP AND JAVA

**GR NO:** 119C0054

**BATCH:** B3

**PROBLEM STATEMENT:1**:- Write cpp program to implement Queue(array) using Class.

**CODE:**

#include <iostream>

#include <string.h>

using namespace std;

#define size 5

// class queue definition

class Queue

{

public:

int rear, front;

int ele;

int q[size];

Queue()

{

rear = front = 0;

}

void Insert\_Q();

void Delete\_Q();

void Display\_Q();

};

// Inserting element in Queue

void Queue::Insert\_Q()

{

cout << "Input Queue Element:";

cin>> ele;

// rear must never be greater than size

if(rear < size)

{

rear++; //increment the rear

q[rear] = ele; // insert the new element

if(front == 0)

{

front=1;

}

else

{

cout << "Queue is Full\n";

}

}

}

// Deleting element in Queue

void Queue::Delete\_Q()

{

// Find out if queue is empty

if(front == 0)

{

cout << "Queuy is Empty. Nothing to Delete!" << endl;

return;

}

else

{

ele = q[front]; // delete the element that front is pointing to

cout << "Element Deleted :" << ele << endl;

}

if(front == rear)

{

front = 0;

rear = 0;

}

else

{

front = front + 1;

}

}

//For Displaying Queue

void Queue::Display\_Q()

{

if(front == 0)

return;

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

{

cout << " " << q[i];

}

cout << "\n\n";

}

int main()

{

Queue Q;

int k = 0,choice;

do

{

cout<"\*\*\* Queue Menu \*\*\*";

cout <<"1.Insert \n2.Delete \n3.Quit\n";

cout <<"Input the choice:"<< endl;

cin >> choice ;

switch(choice)

{

case 1:

Q.Insert\_Q();

cout <<"Queue after Insertion:";

Q.Display\_Q();

break;

case 2:

Q.Delete\_Q();

cout <<"Queue after Deletion"<< endl;

Q.Display\_Q();

break;

case 3:

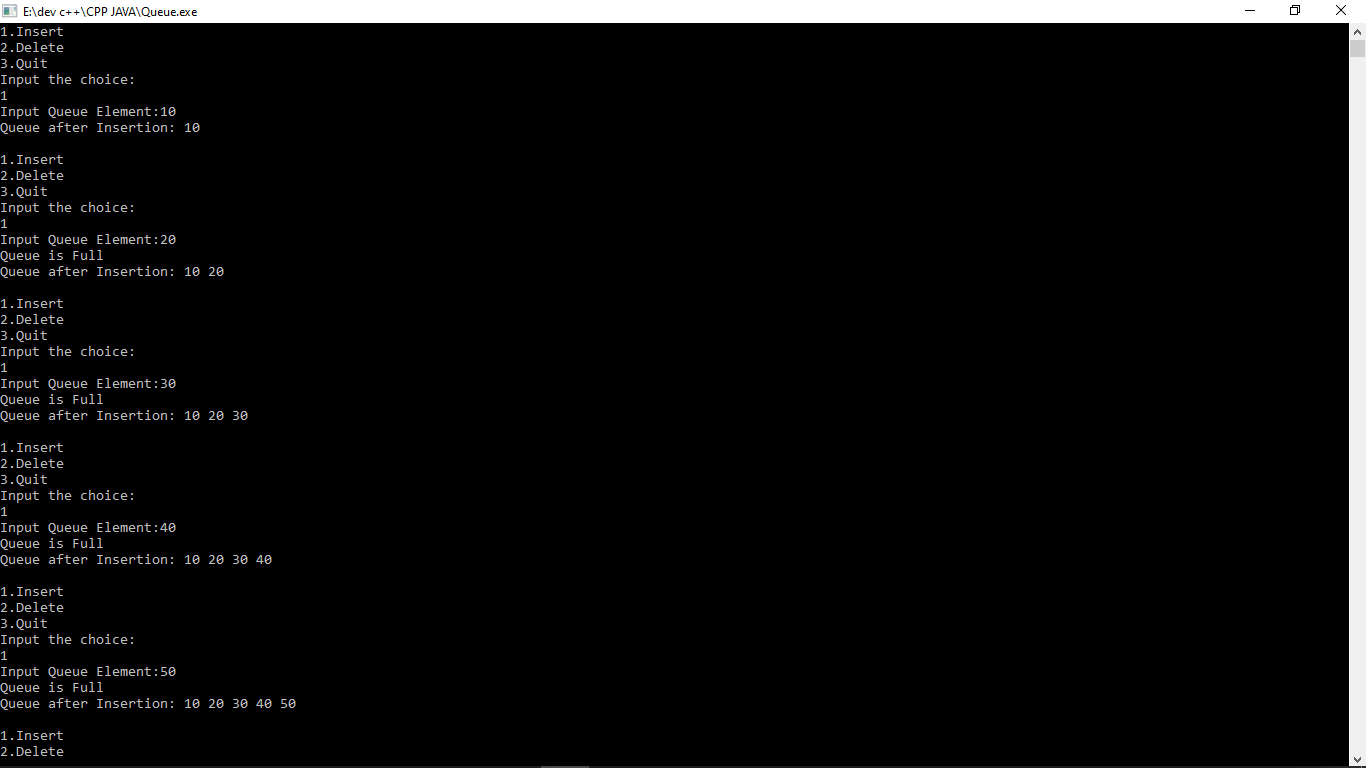
k=1;

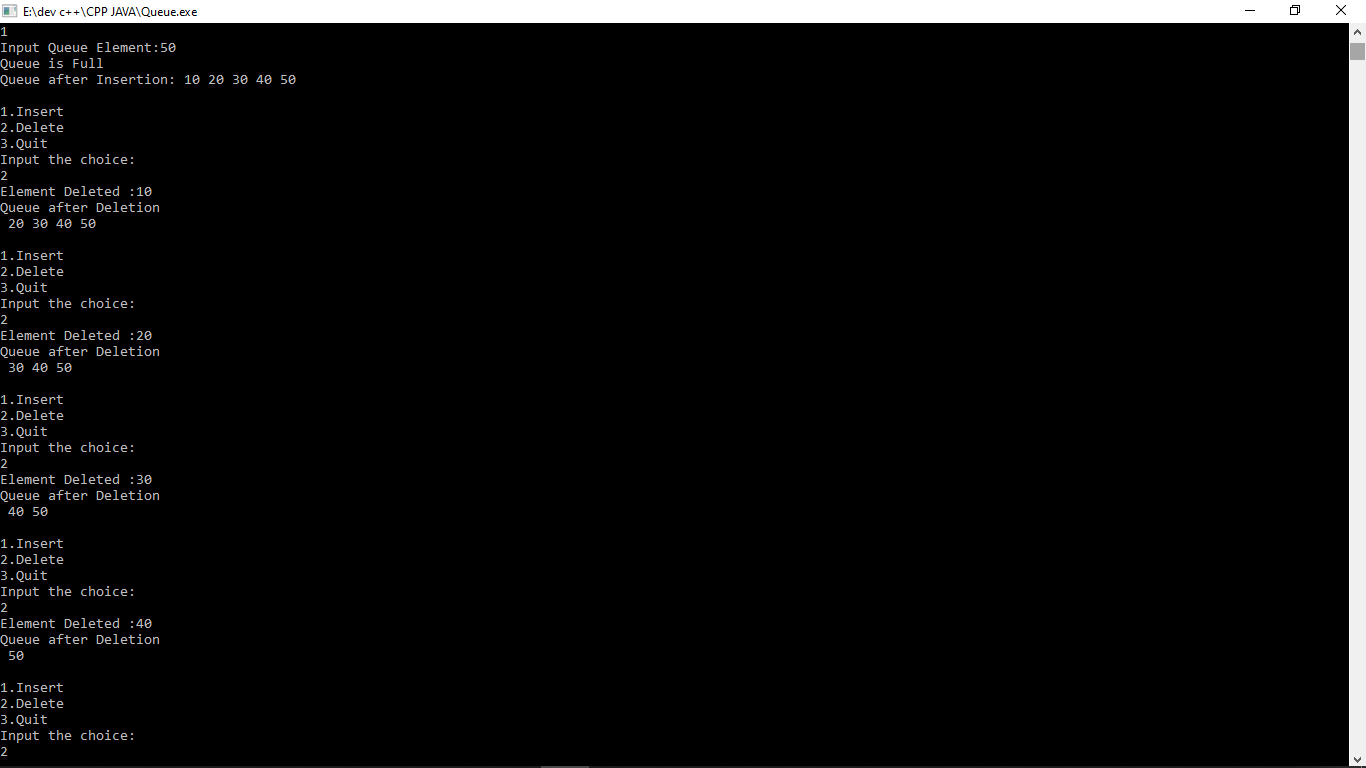
}

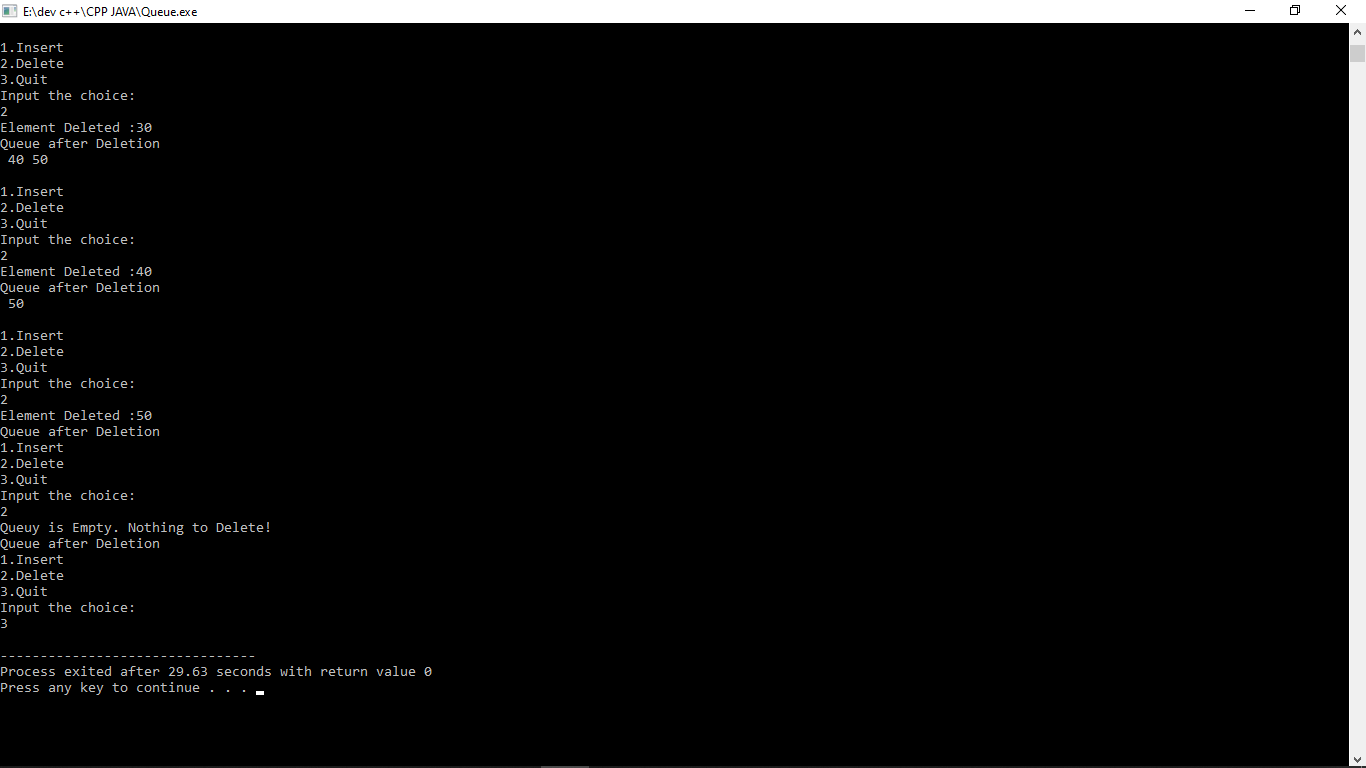
}while(!k);

}

**OUTPUT:-**





o

**PROBLEM STATEMENT:2**:- Write cpp program to implement Circular Queue(array) using Class.

**CODE:**

#include<iostream>

#define MAX 5

using namespace std;

//Class Circular Queue

class Circular\_Queue

{

private:

int \*cqueue\_arr;

int front, rear;

public:

Circular\_Queue()

{

cqueue\_arr = new int [MAX];

rear = front = -1;

}

//Insert into Circular Queue

void insert(int item)

{

if ((front == 0 && rear == MAX-1) || (front == rear+1))

{

cout<<"Queue Overflow \n";

return;

}

if (front == -1)

{

front = 0;

rear = 0;

}

else

{

if (rear == MAX - 1)

rear = 0;

else

rear = rear + 1;

}

cqueue\_arr[rear] = item ;

}

//Delete from Circular Queue

void del()

{

if (front == -1)

{

cout<<"Queue Underflow\n";

return ;

}

cout<<"Element deleted from queue is : "<<cqueue\_arr[front]<<endl;

if (front == rear)

{

front = -1;

rear = -1;

}

else

{

if (front == MAX - 1)

front = 0;

else

front = front + 1;

}

}

//Display Circular Queue

void display()

{

int front\_pos = front, rear\_pos = rear;

if (front == -1)

{

cout<<"Queue is empty\n";

return;

}

cout<<"Queue elements :\n";

if (front\_pos <= rear\_pos)

{

while (front\_pos <= rear\_pos)

{

cout<<cqueue\_arr[front\_pos]<<" ";

front\_pos++;

}

}

else

{

while (front\_pos <= MAX - 1)

{

cout<<cqueue\_arr[front\_pos]<<" ";

front\_pos++;

}

front\_pos = 0;

while (front\_pos <= rear\_pos)

{

cout<<cqueue\_arr[front\_pos]<<" ";

front\_pos++;

}

}

cout<<endl;

}

};

int main()

{

int choice, item;

Circular\_Queue cq;

do

{

cout<<"1.Insert\n";

cout<<"2.Delete\n";

cout<<"3.Display\n";

cout<<"4.Quit\n";

cout<<"Enter your choice : ";

cin>>choice;

switch(choice)

{

case 1:

cout<<"Insert the element in queue : ";

cin>>item;

cq.insert(item);

break;

case 2:

cq.del();

break;

case 3:

cq.display();

break;

case 4:

break;

default:

cout<<"Wrong choice\n";

}//End of switch

}

while(choice != 4);

return 0;

}

**OUTPUT:-**

