Data Structures
UCS301

Assignment-5

Submitted by:
Name: Ananya Agarwal
Roll No: 102083036
Batch: 2CO14

Q1. Develop a menu driven program demonstrating the following operations on simple Queues: enqueue(), dequeue(), isEmpty(), isFull(), display(), and peek().

Sol-1

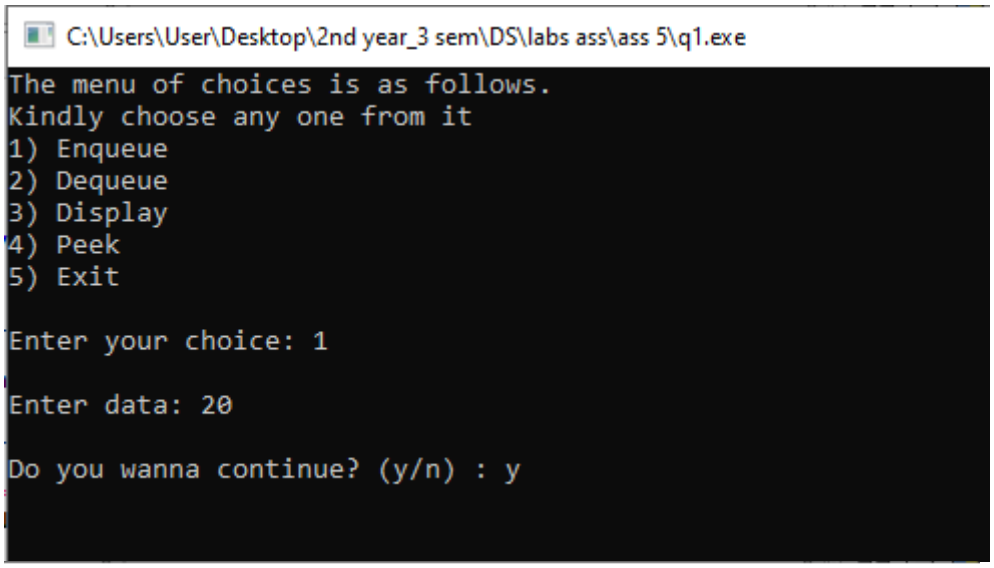
```
#include<iostream>
using namespace std;
struct node
{
 int data;
 node *next;
};
class Queue
{
 node *front,*rear;
 public:
 Queue()
 {
 front = rear = NULL;
 }
 void enqueue(int d);
 void dequeue();
 bool isEmpty();
 bool isFull();
 void display();
 int peek();
};
void Queue::enqueue(int d)
{
 if(!isFull())
 {
 node *temp = new node;
 temp->data = d;
 temp->next = NULL;
 if(front == NULL)
 {
```

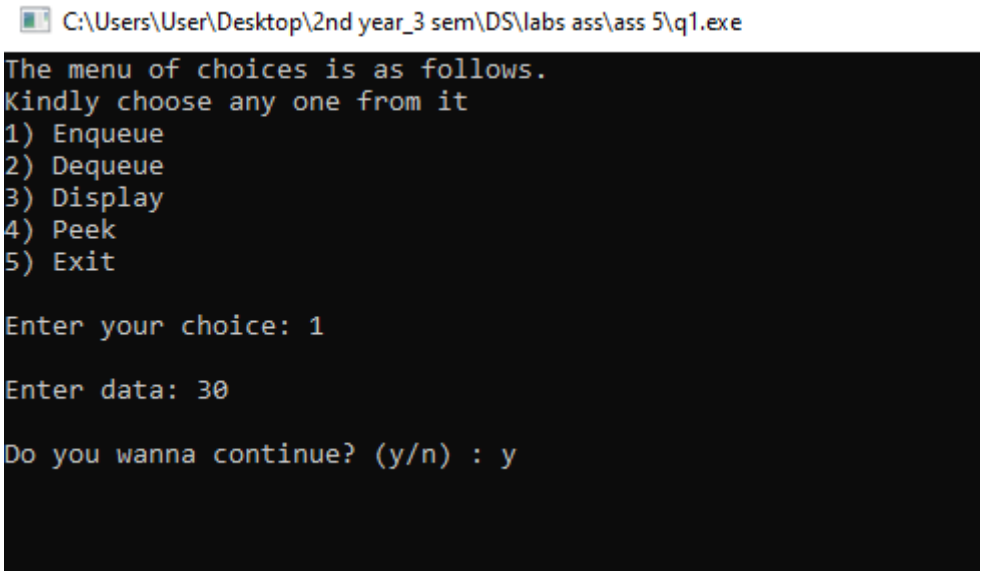
```
 front = rear = temp;
 }
 else
 {
 rear->next = temp;
 rear = temp;
 }
 }
 else
 {
 cout<<"Sorry!!! The Queue is full! \n";
 }
}
void Queue::dequeue()
{
 if(!isEmpty())
 {
 node *temp = front;
 front = front->next;
 if(front == NULL)
 {
 rear = NULL;
 }
 delete temp;
 }
 else
 {
 cout<<"Sorry!! Queue is empty! \n";
 }
}
bool Queue::isEmpty()
{
 return (front == NULL);
}
```

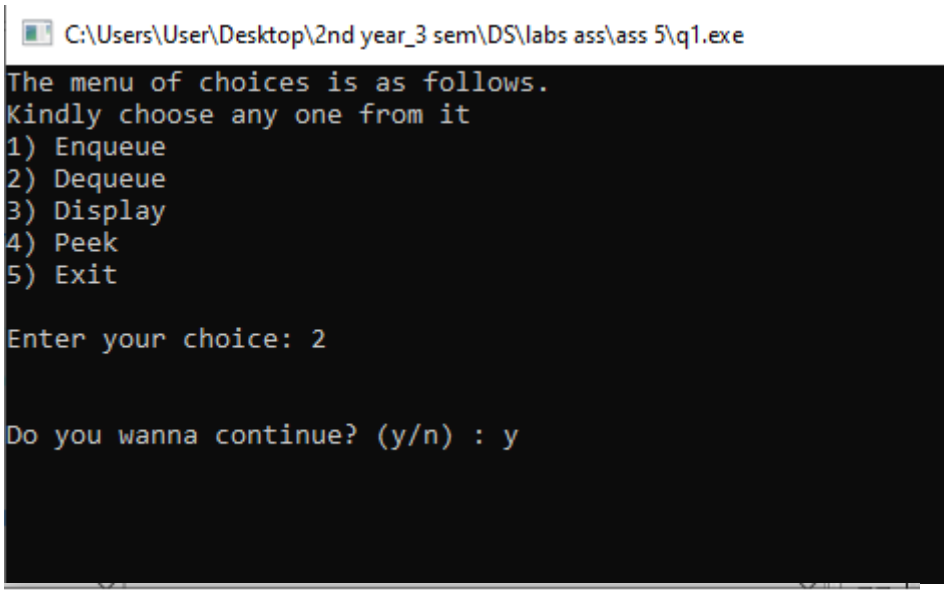
```
bool Queue::isFull()
{
 node *temp = new node;
 if(temp == NULL)
 {
 delete temp;
 return true;
 }
 delete temp;
 return false;
}
void Queue::display()
{
 if(!isEmpty())
 {
 node *temp = front;
 cout<<"The Queue elements are : \n";
 while(temp!=NULL)
 {
 cout<<temp->data<<" ";
 temp = temp->next;
 }
 cout<<endl;
 }
 else
 {
 cout<<"Sorry !!! Queue is empty! \n";
 }
 cout<<endl;
}
int Queue::peek()
{
 return front->data;
}
```

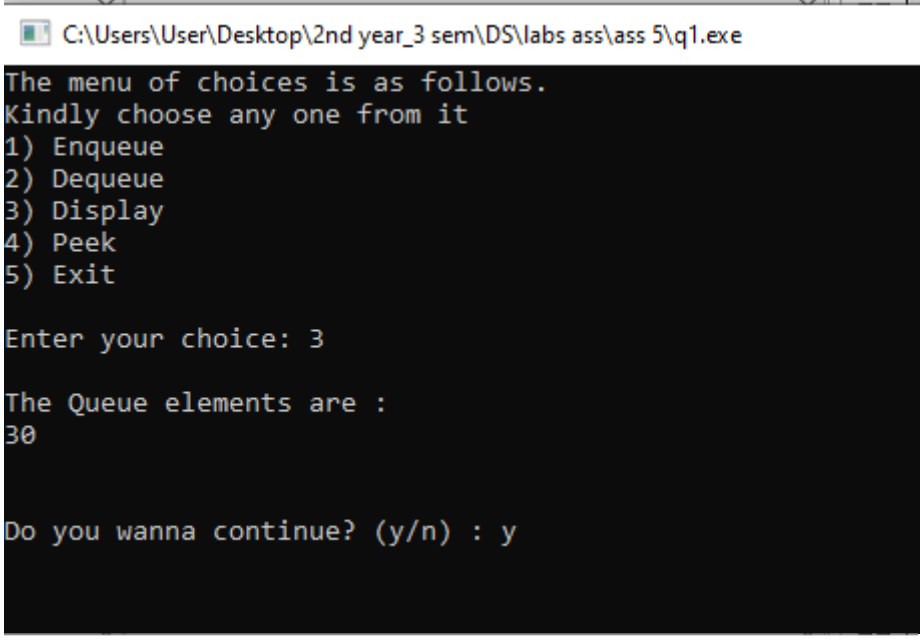
```
int main()
{
 Queue q;
 int ch,d;
 char c;
 do
 {
 cout<<"The menu of choices is as follows. \nKindly choose any one from it \n";
 cout<<"1) Enqueue \n";
 cout<<"2) Dequeue \n";
 cout<<"3) Display \n";
 cout<<"4) Peek \n";
 cout<<"5) Exit \n";
 cout<<"\nEnter your choice: ";
 cin>>ch;
 cout<<endl;
 switch(ch)
 {
 case 1:
 cout<<"Enter data: ";
 cin>>d;
q.enqueue(d);
 break;
 case 2:
 q.dequeue();
 break;
 case 3:
 q.display();
 break;
 case 4:
 cout<<"Element at front : "<<q.peek()<<endl;
 break;
 case 5:
 exit(0);
```

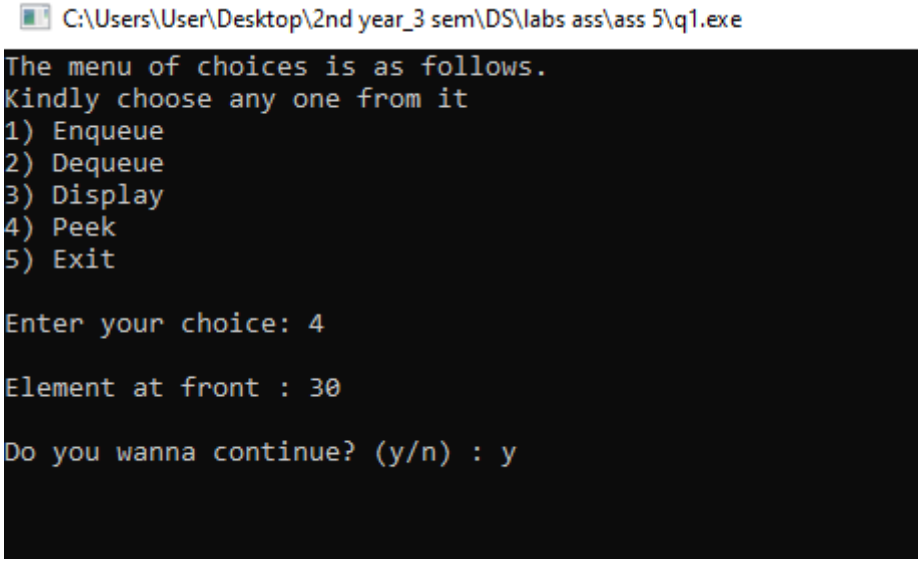
```
 default:
 cout<<"Wrong choice \n";
 }
 cout<<"\nDo you wanna continue? (y/n) : ";
 cin>>c;
 system("cls");
 }while(c == 'y' || c == 'Y');
}
```






C:\Users\User\Desktop\2nd year_3 sem\DS\labs ass\ass 5\q1.exe
The menu of choices is as follows.
Kindly choose any one from it
1) Enqueue
2) Dequeue
3) Display
4) Peek
5) Exit
Enter your choice: 2
Do you wanna continue? (y/n) : y


C:\Users\User\Desktop\2nd year_3 sem\DS\labs ass\ass 5\q1.exe
The menu of choices is as follows.
Kindly choose any one from it
1) Enqueue
2) Dequeue
3) Display
4) Peek
5) Exit
Enter your choice: 3
The Queue elements are :
30
Do you wanna continue? (y/n) : y


C:\Users\User\Desktop\2nd year_3 sem\DS\labs ass\ass 5\q1.exe
The menu of choices is as follows.
Kindly choose any one from it
1) Enqueue
2) Dequeue
3) Display
4) Peek
5) Exit
Enter your choice: 4
Element at front : 30
Do you wanna continue? (y/n) : y

C:\Users\User\Desktop\2nd year_3 sem\DS\labs ass\ass 5\q1.exe

```
The menu of choices is as follows.
Kindly choose any one from it
1) Enqueue
2) Dequeue
3) Display
4) Peek
5) Exit

Enter your choice: 5


--------------------------------
Process exited after 203.7 seconds with return value 0
Press any key to continue . . .
```

C:\Users\User\Desktop\2nd year_3 sem\DS\labs ass\ass 5\q1.exe

```
The menu of choices is as follows.
Kindly choose any one from it
1) Enqueue
2) Dequeue
3) Display
4) Peek
5) Exit

Enter your choice: 2


Do you wanna continue? (y/n) : y
```

C:\Users\User\Desktop\2nd year_3 sem\DS\labs ass\ass 5\q1.exe

```
The menu of choices is as follows.
Kindly choose any one from it
1) Enqueue
2) Dequeue
3) Display
4) Peek
5) Exit

Enter your choice: 3

Sorry !!! Queue is empty!


Do you wanna continue? (y/n) : y
```

Q2. Develop a menu driven program demonstrating the following operations on Circular Queues: enqueue(), dequeue(), isEmpty(), isFull(), display(), and peek().

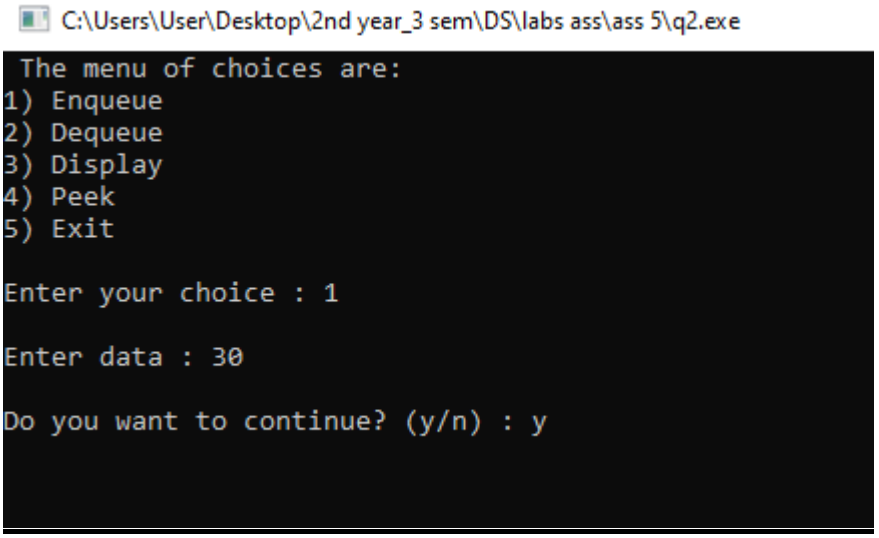
Sol-2.

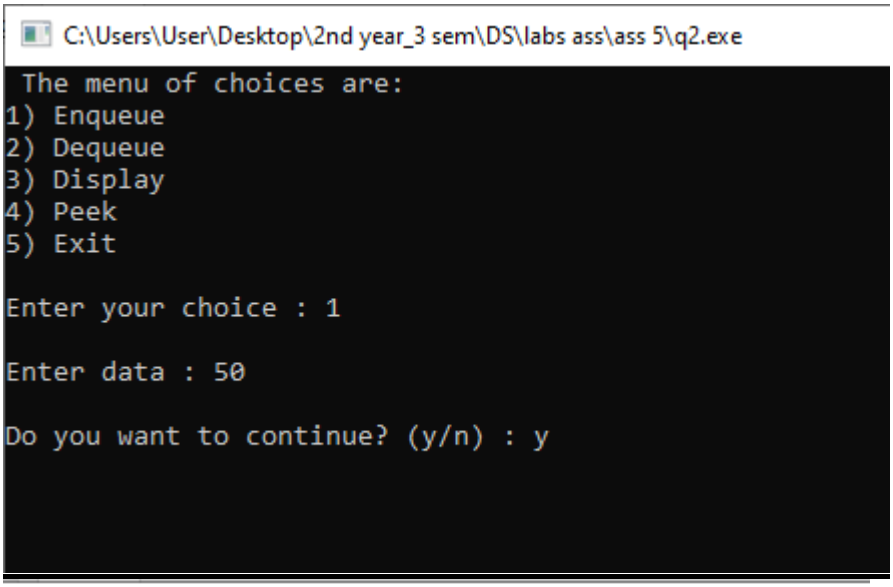
```
#include<iostream>
using namespace std;
const int size = 10;
class C_Queue
{
 int data[size];
 int front,rear;
 public:
 C_Queue()
 {
 front = rear = -1;
 }
 void enqueue(int d);
 void dequeue();
 bool isEmpty();
 bool isFull();
 void display();
 int peek();
};
void C_Queue::enqueue(int d)
{
 if(!isFull())
 {
 rear = (rear + 1)%size;
 data[rear] = d;
 if(front == -1)
 front = 0;
 }
 else
 cout<<"Sorry!!! Overflow!!! \n";
}
void C_Queue::dequeue()
{
```

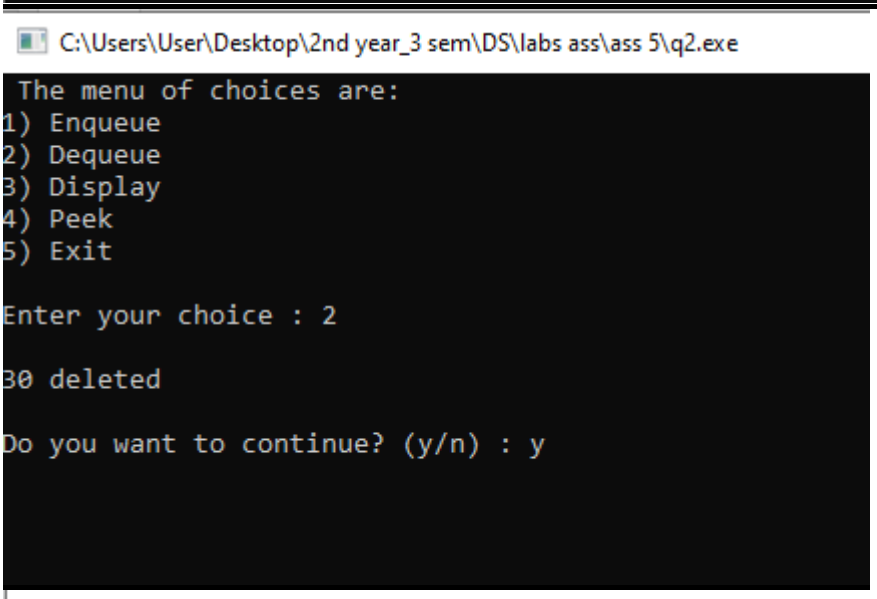
```
 if(!isEmpty())
 {
 cout<<data[front]<<" deleted \n";
 if(front == rear)
 front = rear = -1;
 else
 front = (front + 1)%size;
 }
 else
 cout<<"Sorry!! Underflow! \n";
}
bool C_Queue::isEmpty()
{
 if(rear == -1)
 return true;
 return false;
}
bool C_Queue::isFull()
{
 if(front == (rear + 1)%size)
 return true;
 return false;
}
void C_Queue::display()
{
 if(!isEmpty())
 {
 int i;
 cout<<"Circular Queue elements are : \n";
 if(front <= rear)
 {
 for(i=front;i<=rear;i++)
 cout<<data[i]<<" ";
 }
 else
 {
```

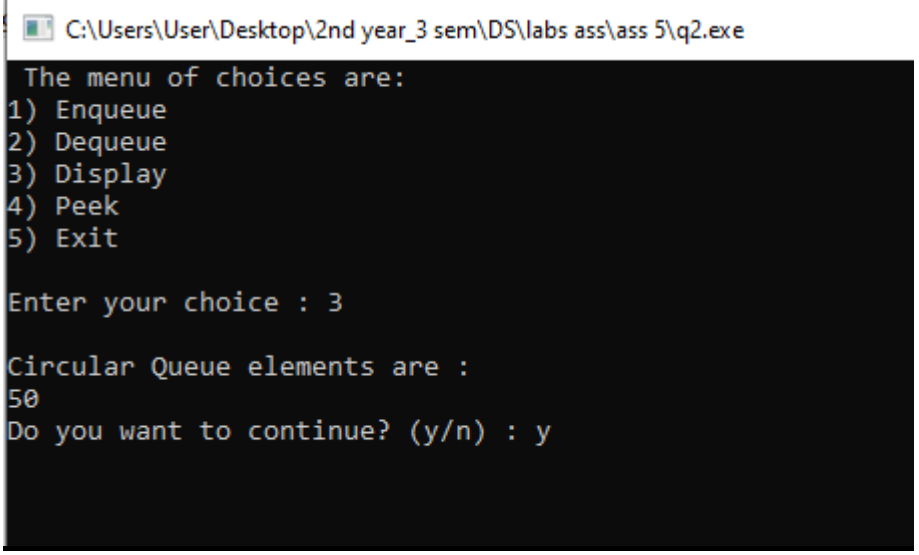
```
 for(i=front;i<size;i++)
 cout<<data[i]<<" ";
 for(i=0;i<=rear;i++)
 cout<<data[i]<<" ";
 }
 }
 else
 cout<<" Sorry!! Underflow! \n";
}
int C_Queue::peek()
{
 if(!isEmpty())
 return data[front];
}
int main()
{
 C_Queue q;
 int ch,d;
 char c;
 do
 {
 cout<<" The menu of choices are: \n";
 cout<<"1) Enqueue \n";
 cout<<"2) Dequeue \n";
 cout<<"3) Display \n";
 cout<<"4) Peek \n";
 cout<<"5) Exit \n";
 cout<<"\nEnter your choice : ";
 cin>>ch;
 cout<<endl;
 switch(ch)
 {
 case 1:
 cout<<"Enter data : ";
 cin>>d;
 q.enqueue(d);
```

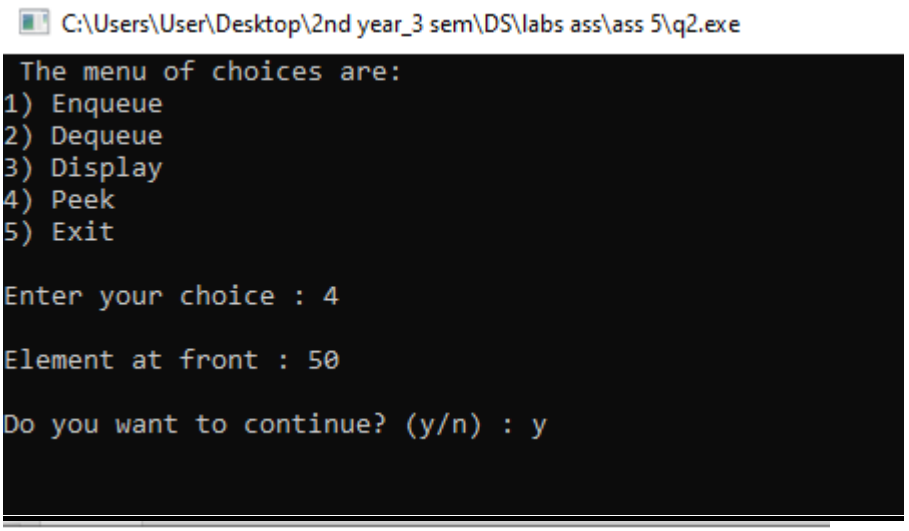
```
break;
case 2:
q.dequeue();
break;
case 3:
q.display();
break;
case 4:
cout<<"Element at front : "<<q.peek()<<endl;
break;
case 5:
exit(0);
default:
cout<<"Wrong choice \n";
}
cout<<"\nDo you want to continue? (y/n) : ";
cin>>c;
system("cls");
}
while(c == 'y' || c == 'Y');
}
```

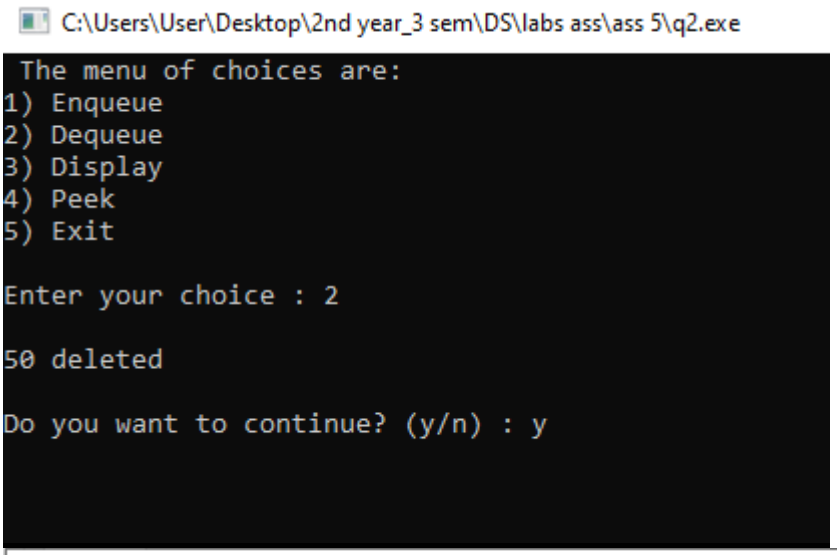


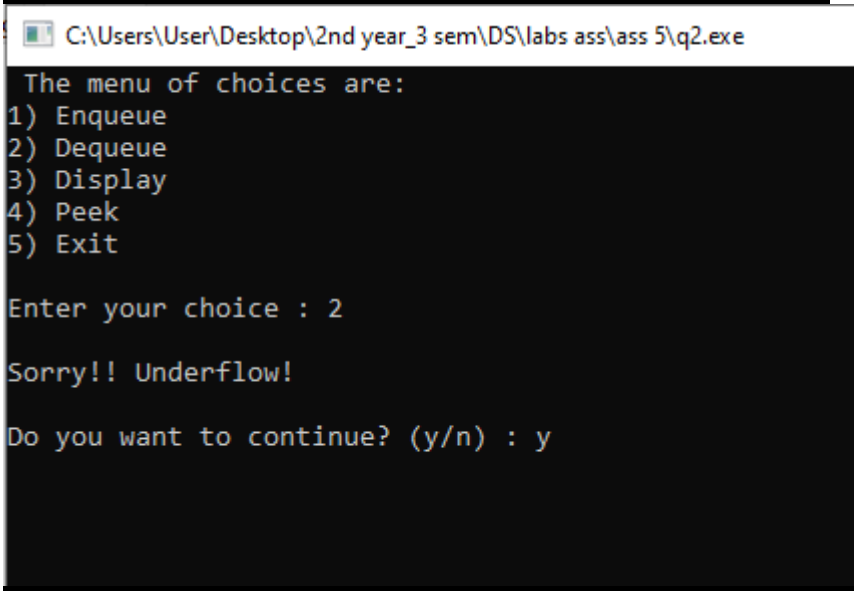

C:\Users\User\Desktop\2nd year_3 sem\DS\labs ass\ass 5\q2.exe
The menu of choices are:
1) Enqueue
2) Dequeue
3) Display
4) Peek
5) Exit
Enter your choice : 1
Enter data : 50
Do you want to continue? (y/n) : y

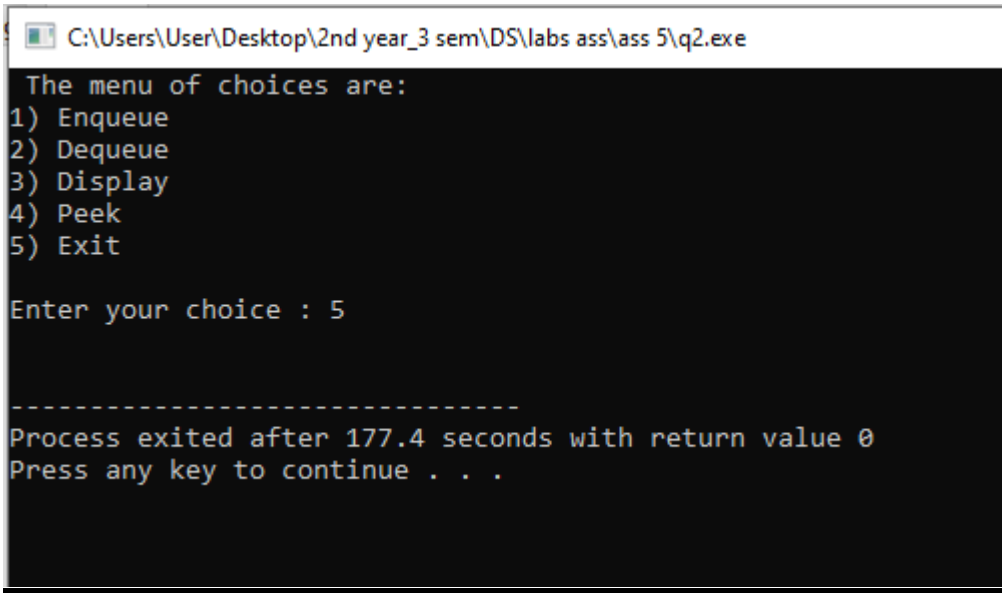

C:\Users\User\Desktop\2nd year_3 sem\DS\labs ass\ass 5\q2.exe
The menu of choices are:
1) Enqueue
2) Dequeue
3) Display
4) Peek
5) Exit
Enter your choice : 2
30 deleted
Do you want to continue? (y/n) : y


C:\Users\User\Desktop\2nd year_3 sem\DS\labs ass\ass 5\q2.exe
The menu of choices are:
1) Enqueue
2) Dequeue
3) Display
4) Peek
5) Exit
Enter your choice : 3
Circular Queue elements are :
50
Do you want to continue? (y/n) : y


C:\Users\User\Desktop\2nd year_3 sem\DS\labs ass\ass 5\q2.exe
The menu of choices are:
1) Enqueue
2) Dequeue
3) Display
4) Peek
5) Exit
Enter your choice : 4
Element at front : 50
Do you want to continue? (y/n) : y


C:\Users\User\Desktop\2nd year_3 sem\DS\labs ass\ass 5\q2.exe
The menu of choices are:
1) Enqueue
2) Dequeue
3) Display
4) Peek
5) Exit
Enter your choice : 2
50 deleted
Do you want to continue? (y/n) : y


C:\Users\User\Desktop\2nd year_3 sem\DS\labs ass\ass 5\q2.exe
The menu of choices are:
1) Enqueue
2) Dequeue
3) Display
4) Peek
5) Exit
Enter your choice : 2
Sorry!! Underflow!
Do you want to continue? (y/n) : y



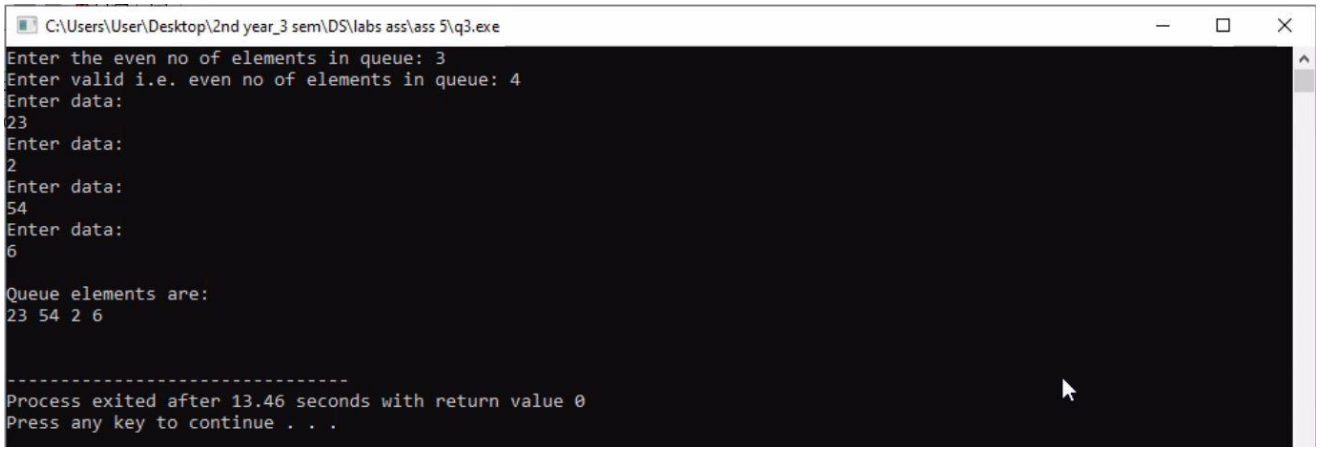
Q3. Write a program interleave the first half of the queue with second half

Sol-3.

```
#include<iostream>
using namespace std;
struct node
{
 int data;
 node *next;
};
class Queue
{
 node *front,*rear;
 public:
 Queue()
 {
 front = rear = NULL;
 }
 void enqueue(int d);
 void dequeue();
 bool isEmpty();
 bool isFull();
 void display();
 int peek();
};
void Queue::enqueue(int d)
{
 if(!isFull())
 {
 node *temp = new node;
 temp->data = d;
 temp->next = NULL;
 if(front == NULL)
 {
 front = rear = temp;
 }
 else
 {
 rear->next = temp;
 rear = temp;
```

```
 }
 }
 else
 {
 cout<<"Sorry!!! Queue is full! \n";
 }
}
void Queue::dequeue()
{
 if(!isEmpty())
 {
 node *temp = front;
 front = front->next;
 if(front == NULL)
 {
 rear = NULL;
 }
 delete temp;
 }
 else
 {
 cout<<"Sorry!!! Queue is empty! \n";
 }
}
bool Queue::isEmpty()
{
 return (front == NULL);
}
bool Queue::isFull()
{
 node *temp = new node;
 if(temp == NULL)
 {
 delete temp;
 return true;
 }
 delete temp;
 return false;
}
void Queue::display()
{
 if(!isEmpty())
 {
 node *temp = front;
 cout<<"\nQueue elements are: \n";
 while(temp!=NULL)
 {
 cout<<temp->data<<" ";
 temp = temp->next;
 }
 cout<<endl;
 }
 else
 {
 cout<<"Sorry!!! Queue is empty! \n";
 }
 cout<<endl;
}
int Queue::peek()
```

```
{
 return front->data;
}
int main()
{
 Queue q1,q2,q3;
 int n,i,d;
 cout<<"Enter the even no of elements in queue: ";
 cin>>n;
 while(n%2!=0)
 {
 cout<<"Enter valid i.e. even no of elements in queue: ";
 cin>>n;
 }
 for(i=0;i<n;i++)
 {
 cout<<"Enter data: \n";
 cin>>d;
 q1.enqueue(d);
 }
 for(i=0;i<n;i++)
 {
 if(i < n/2)
 q2.enqueue(q1.peek());
 else
 q3.enqueue(q1.peek());
 q1.dequeue();
 }
 for(i=0;i<n/2;i++)
 {
 q1.enqueue(q2.peek());
 q1.enqueue(q3.peek());
 q2.dequeue();
 q3.dequeue();
 }
 q1.display();
 return 0;
}
```



Q4. Write a program to find first non-repeating character in a string using Queue

Sol-4.

```
#include<iostream>
using namespace std;
struct node
{
 char data;
 node *next;
};
class Queue
{
 node *front,*rear;
 public:
 Queue()
 {
 front = rear = NULL;
 }
 void enqueue(char d);
 void dequeue();
 bool isEmpty();
 bool isFull();
 void display();
 char peek();
};
void Queue::enqueue(char d)
{
 if(!isFull())
 {
 node *temp = new node;
 temp->data = d;
 temp->next = NULL;
 if(front == NULL)
 {
 front = rear = temp;
 }
 else
 {
 rear->next = temp;
 rear = temp;
 }
 }
 else
 {
 cout<<"Sorry!!! Queue is full! \n";
 }
}
void Queue::dequeue()
{
 if(!isEmpty())
 {
 node *temp = front;
 front = front->next;
 if(front == NULL)
 {
 rear = NULL;
 }
```

```
 delete temp;
 }
 else
 {
 cout<<"Sorry!!! Queue is empty! \n";
 }
}
bool Queue::isEmpty()
{
 return (front == NULL);
}
bool Queue::isFull()
{
 node *temp = new node;
 if(temp == NULL)
 {
 delete temp;
 return true;
 }
 delete temp;
 return false;
}
void Queue::display()
{
 if(!isEmpty())
 {
 node *temp = front;
 cout<<"Queue elements are: \n";
 while(temp!=NULL)
 {
 cout<<temp->data<<" ";
 temp = temp->next;
 }
 cout<<endl;
 }
 else
 {
 cout<<"Sorry!! Queue is empty!!! \n";
 }
 cout<<endl;
}
char Queue::peek()
{
 return front->data;
}
int main()
{
 Queue q;
 string str;
 int i;
 int freq[26] = {0};
 cout<<"Enter the string : ";
 getline(cin,str);
 for(i=0;str[i]!='\0';i++)
 {
 if(str[i]>='a' && str[i]<='z')
 {
```

```
 int flag = 0;
 q.enqueue(str[i]);
 freq[str[i] - 'a']++;
 while(!q.isEmpty())
 {
 if(freq[q.peek() - 'a'] == 1)
 {
 cout<<q.peek()<<" ";
flag = 1;
break;
 }
 q.dequeue();
 }
 if(flag == 0)
 cout<<-1<<" ";
 }
 }
 return 0;
}
```