Ex. No. 6	Linked list Implementation of Stack and Queue
Date of Exercise	23-08-2017

Aim:

To simulate the working of a queue and stack of integers using a linked list.

Algorithm:

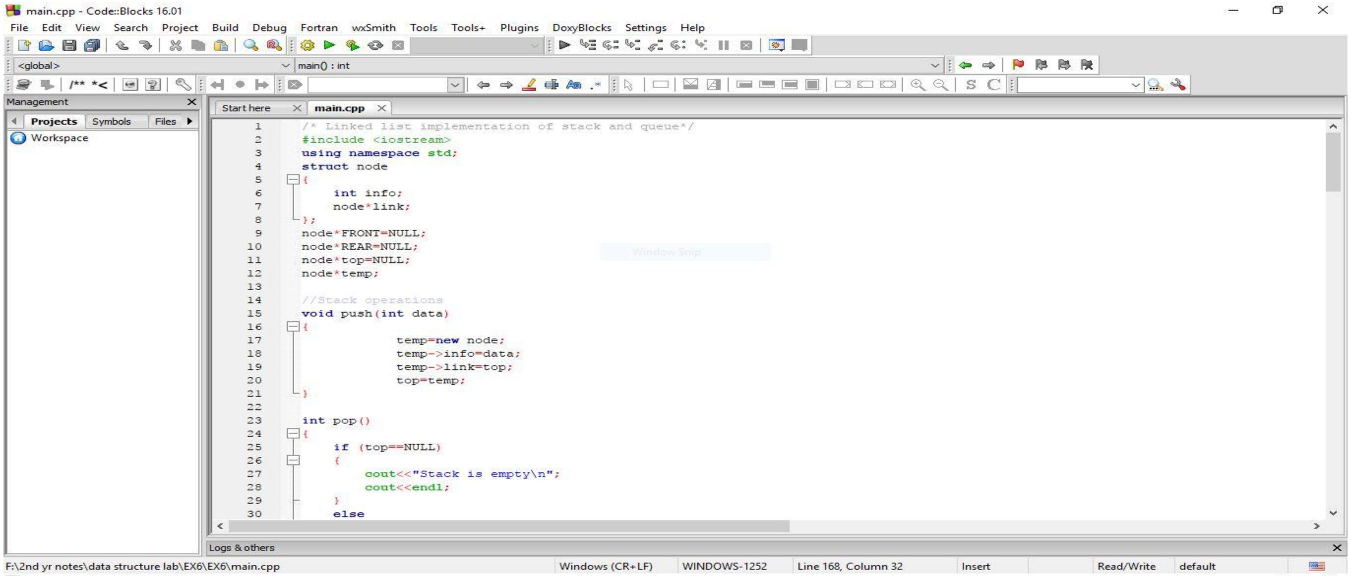
1. Start.
2. For stack operations,
3. Create struct node, int info and node*link
4. Node*front=null; node*rear= null; node*temp;
5. Create the push function with necessary steps.
6. Create the pop function with necessary steps.
7. Create the display function with necessary steps.
8. Inside the main function we are having switch case for 4 inputs
9. And the corresponding functions are called
10. For queue operations
11. Create struct node, int info and node*link
12. Node*top=null; node*temp;
13. Create the add queue function with necessary steps.
14. Create the delete queue function with necessary steps.
15. Create the display function with necessary steps.
16. Inside the main function we are having switch case for 4 inputs.
17. And the corresponding functions are called.
18. End

Source Code:

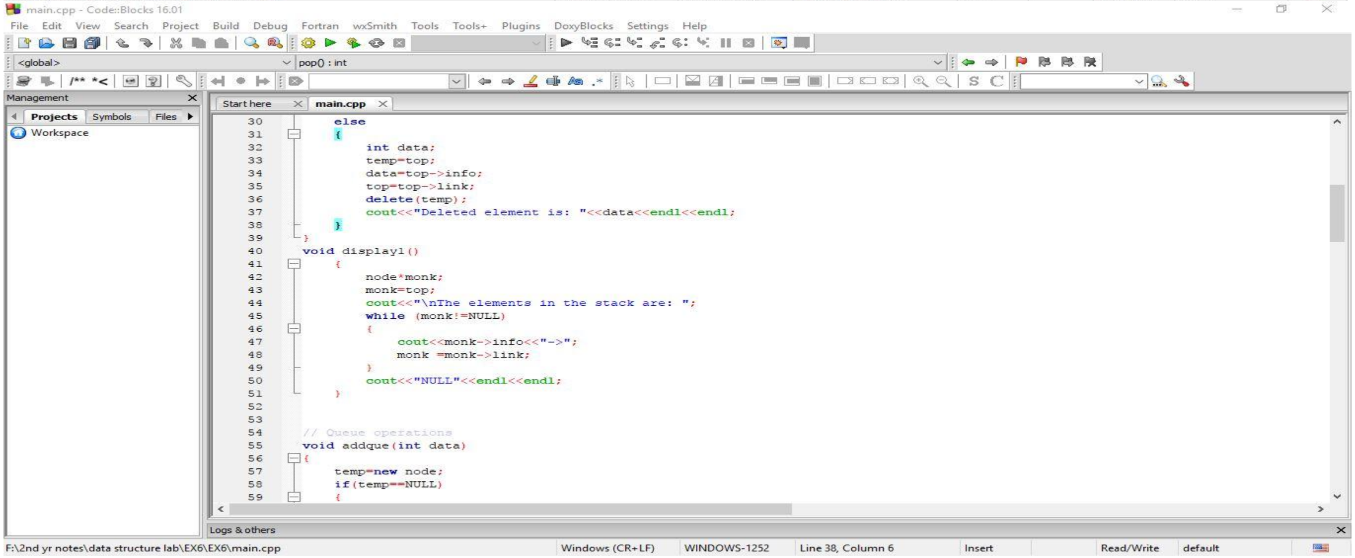
```
/* Linked list implementation of stack and queue*/
#include <iostream>
using namespace std;
struct node
{
   int info;
   node*link;
};
node*FRONT=NULL;
node*REAR=NULL;
node*top=NULL;
```

```
node*temp;

//Stack operations
void push(int data)
{
        temp=new node;
        temp->info=data;
        temp->link=top;
        top=temp;
}

int pop()
{
   if (top==NULL)
   {
      cout<<"Stack is empty\n";
      cout<<endl;
   }
   else
   {
      int data;
      temp=top;
      data=top->info;
      top=top->link;
      delete(temp);
      cout<<"Deleted element is: "<<data<<endl<<endl;
   }
}
void display1()
   {
      node*monk;
      monk=top;
      cout<<"\n The elements in the stack are: ";
      while (monk!=NULL)
      {
         cout<<monk->info<<"->";
         monk =monk->link;
      }
      cout<<"NULL"<<endl<<endl;
   }


// Queue operations
void addque(int data)
{
   temp=new node;
   if(temp==NULL)
   {
      cout<<"\n Queue is full";
      return;
   }
   else
      {
          temp->info=data;
          temp->link=NULL;
      }
   if(FRONT==NULL)
      {
         FRONT=REAR=temp;
         return;
      }
      else
      {
         REAR->link=temp;
         REAR=temp;
         return;
      }
}
int delq()
{
   if (FRONT==NULL)
   {
      cout<<"\nQueue is empty\n"<<endl;
   }
   else
   {
      int data;
      temp=FRONT;
      data=FRONT->info;
      FRONT=FRONT->link;
      cout<<"\nThe deleted element is: "<<data<<endl<<endl;
      delete(temp);
      return data;
   }
}
void display()
   {
      node*monk;
      monk=FRONT;
      cout<<"\nThe elements in the queue are: ";
      while (monk!=NULL)
      {
            cout<<monk->info<<"->";
            monk =monk->link;
        }
        cout<<"NULL\n"<<endl;
    }

int main()
{
    int op;
    cout<<"\n Choices to do.....\n";
    cout<<"\n 1.Stack operations";
    cout<<"\n 2.Queue operations";
    cout<<"\n 3.Exit";
    cout<<"\n\n Enter your choice: ";
    cin>>op;
    cout<<"\n";
    switch(op)
    {
    case 1:
        int c;
        while(1)
            {
                cout<<"Functions to perform:\n1.PUSH\n2.POP\n3.DISPLAY\n4.EXIT\n";
                cout<<"\nEnter your choice: ";
                cin>>c;
                switch(c)
                {
                    case 1:
                        cout<<"Enter a number: ";
                        int n;
                        cin>>n;
                        cout<<"\n";
                        push(n);
                        break;
                    case 2:
                        pop();
                        break;

                    case 3:
                        display1();
                        break;
                    case 4:
                        return 0;
                        break;
                }
            }
    case 2:
        int opt;
        while(1)
            {
                cout<<"Functions to 
perform:\n1.ENQUEUE\n2.DEQUEUE\n3.DISPLAY\n4.EXIT\n";
                cout<<"\nEnter your choice:";
                cin>>opt;
                switch(opt)
                {
                    case 1:
                        cout<<"\nEnter a number: ";
                        int n;
                        cin>>n;
                        cout<<"\n";
                        addque(n);
                        break;

                    case 2:
                        delq();
                        break;

                    case 3:
                        display();
                        break;
                    case 4:
                        return 0;
                        break;
                }
            }
    case 3:
        return 0;

    }
}
```

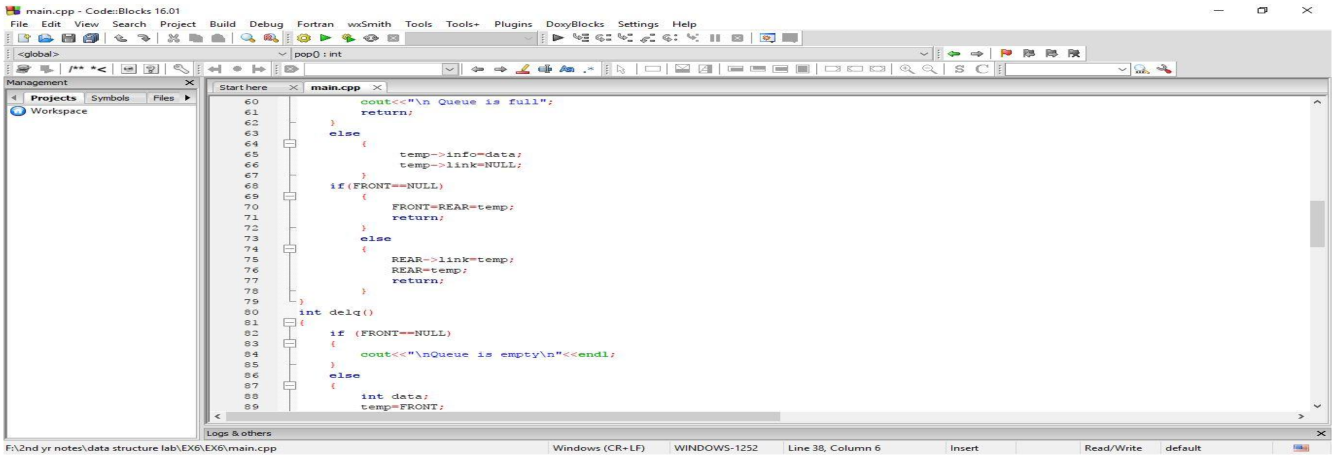
Output:


main.cpp - Code::Blocks 16.01
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global>
main() : int
Management
Projects Symbols Files
Workspace
Start here
main.cpp
Logs & others
F:\2nd yr notes\data structure lab\EX6\EX6\main.cpp
Windows (CR+LF)
WINDOWS-1252
Line 168, Column 32
Insert
Read/Write
default

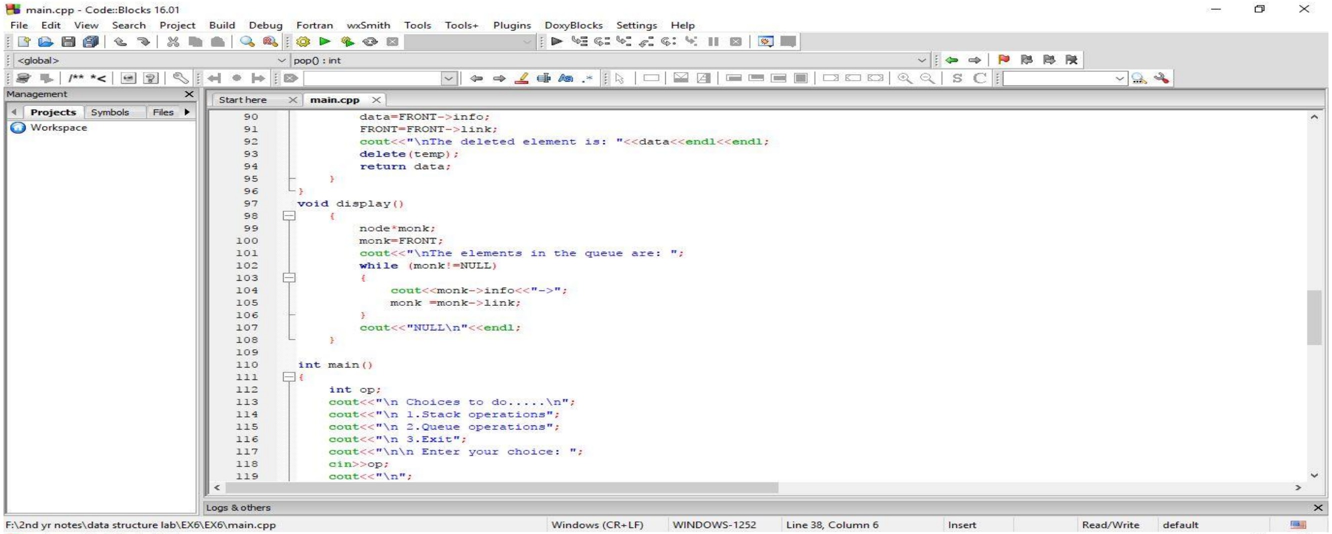
```
/* Linked list implementation of stack and queue*/
#include <iostream>
using namespace std;
struct node
{
    int info;
    node*link;
};
node*FRONT=NULL;
node*REAR=NULL;
node*top=NULL;
node*temp;

//Stack operations
void push(int data)
{
            temp=new node;
            temp->info=data;
            temp->link=top;
            top=temp;
}

int pop()
{
    if (top==NULL)
    {
        cout<<"Stack is empty\n";
        cout<<endl;
    }
    else
```


main.cpp - Code::Blocks 16.01
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global>
pop() : int
Management
Projects Symbols Files
Workspace
Start here
main.cpp
Logs & others
F:\2nd yr notes\data structure lab\EX6\EX6\main.cpp
Windows (CR+LF)
WINDOWS-1252
Line 38, Column 6
Insert
Read/Write
default

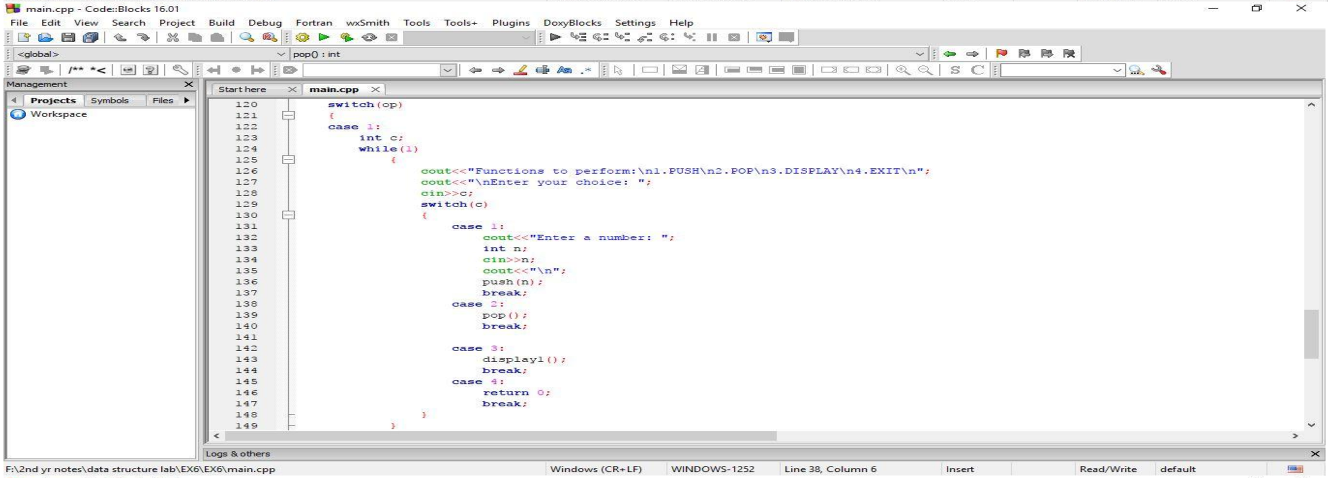
```
    else
    {
        int data;
        temp=top;
        data=top->info;
        top=top->link;
        delete(temp);
        cout<<"Deleted element is: "<<data<<endl<<endl;
    }
}
void display1()
    {
        node*monk;
        monk=top;
        cout<<"\nThe elements in the stack are: ";
        while (monk!=NULL)
        {
            cout<<monk->info<<"->";
            monk =monk->link;
        }
        cout<<"NULL"<<endl<<endl;
    }


// Queue operations
void addque(int data)
{
    temp=new node;
    if(temp==NULL)
    {
```


main.cpp - Code::Blocks 16.01
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global>
pop() : int
Management
Projects Symbols Files
Workspace
Start here
main.cpp
Logs & others
F:\2nd yr notes\data structure lab\EX6\EX6\main.cpp
Windows (CR+LF)
WINDOWS-1252
Line 38, Column 6
Insert
Read/Write
default

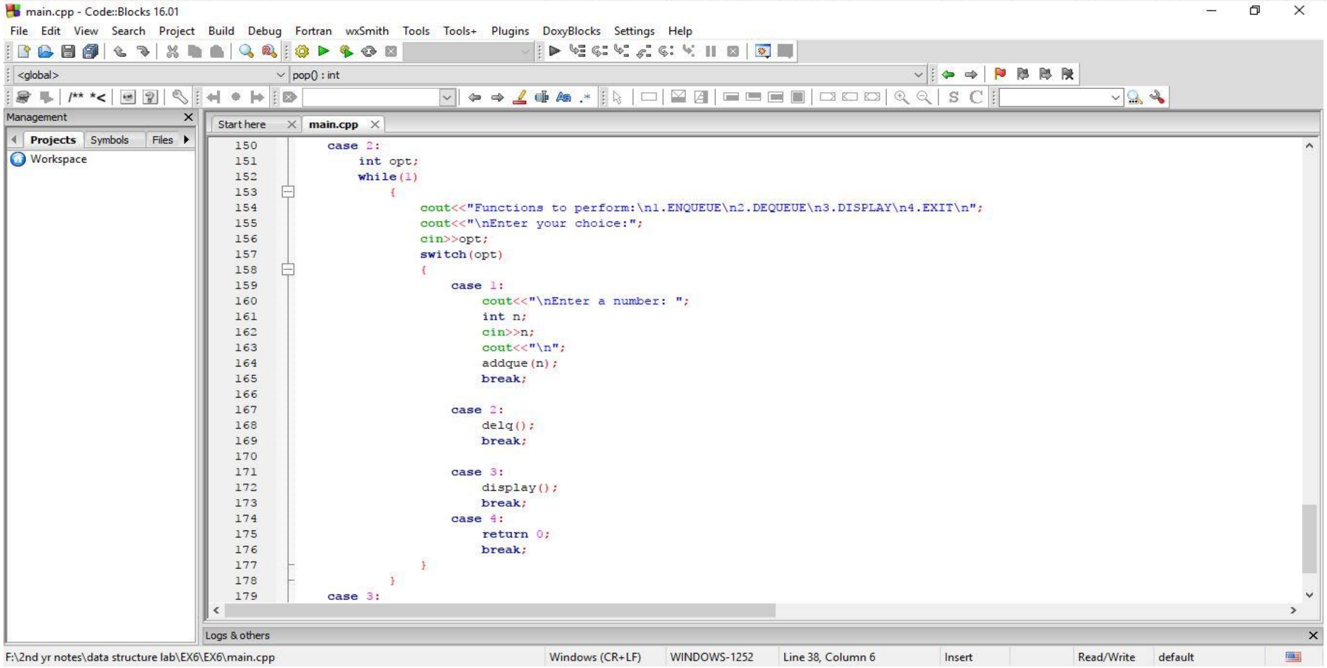
```
        cout<<"\n Queue is full";
        return;
    }
    else
        {
            temp->info=data;
            temp->link=NULL;
        }
    if(FRONT==NULL)
        {
            FRONT=REAR=temp;
            return;
        }
        else
        {
            REAR->link=temp;
            REAR=temp;
            return;
        }
}
int delq()
{
    if (FRONT==NULL)
    {
        cout<<"\nQueue is empty\n"<<endl;
    }
    else
    {
        int data;
        temp=FRONT;
```


main.cpp - Code::Blocks 16.01
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global>
pop() : int
Management
Projects Symbols Files
Workspace
Start here
main.cpp
Logs & others
F:\2nd yr notes\data structure lab\EX6\EX6\main.cpp
Windows (CR+LF)
WINDOWS-1252
Line 38, Column 6
Insert
Read/Write
default

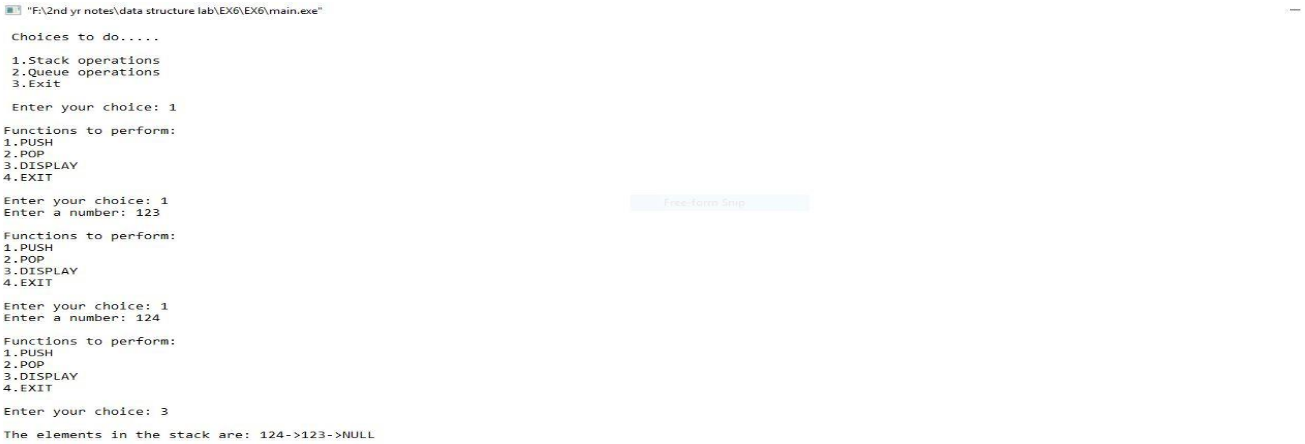
```
        data=FRONT->info;
        FRONT=FRONT->link;
        cout<<"\nThe deleted element is: "<<data<<endl<<endl;
        delete(temp);
        return data;
    }
}
void display()
    {
        node*monk;
        monk=FRONT;
        cout<<"\nThe elements in the queue are: ";
        while (monk!=NULL)
        {
            cout<<monk->info<<"->";
            monk =monk->link;
        }
        cout<<"NULL\n"<<endl;
    }

int main()
{
    int op;
    cout<<"\n Choices to do.....\n";
    cout<<"\n 1.Stack operations";
    cout<<"\n 2.Queue operations";
    cout<<"\n 3.Exit";
    cout<<"\n\n Enter your choice: ";
    cin>>op;
    cout<<"\n";
```


main.cpp - Code::Blocks 16.01
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global>
pop() : int
Management
Projects Symbols Files
Workspace
Start here
main.cpp
Logs & others
F:\2nd yr notes\data structure lab\EX6\EX6\main.cpp
Windows (CR+LF)
WINDOWS-1252
Line 38, Column 6
Insert
Read/Write
default

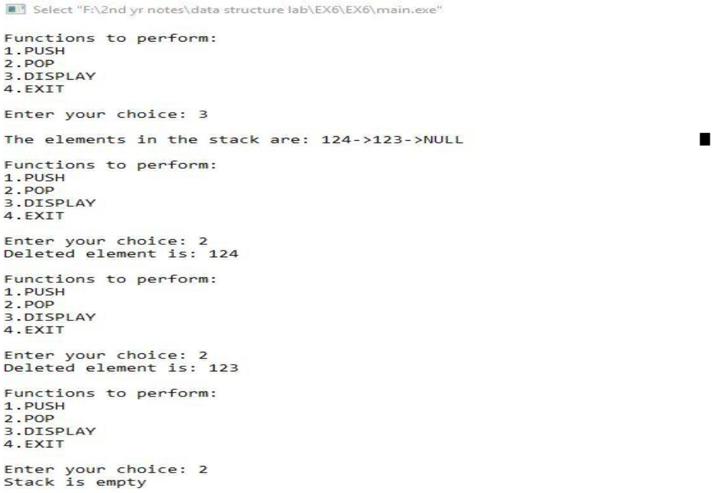
```
    switch(op)
    {
    case 1:
        int c;
        while(1)
            {
                cout<<"Functions to perform:\n1.PUSH\n2.POP\n3.DISPLAY\n4.EXIT\n";
                cout<<"\nEnter your choice: ";
                cin>>c;
                switch(c)
                {
                    case 1:
                        cout<<"Enter a number: ";
                        int n;
                        cin>>n;
                        cout<<"\n";
                        push(n);
                        break;
                    case 2:
                        pop();
                        break;

                    case 3:
                        display1();
                        break;
                    case 4:
                        return 0;
                        break;
                }
            }
```


main.cpp - Code::Blocks 16.01
File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
<global>
pop() : int
Management
Projects Symbols Files
Workspace
Start here
main.cpp
Logs & others
F:\2nd yr notes\data structure lab\EX6\EX6\main.cpp
Windows (CR+LF)
WINDOWS-1252
Line 38, Column 6
Insert
Read/Write
default

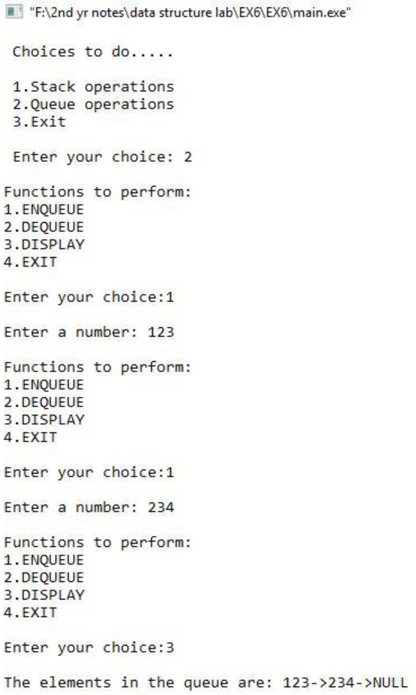
```
    case 2:
        int opt;
        while(1)
            {
                cout<<"Functions to perform:\n1.ENQUEUE\n2.DEQUEUE\n3.DISPLAY\n4.EXIT\n";
                cout<<"\nEnter your choice:";
                cin>>opt;
                switch(opt)
                {
                    case 1:
                        cout<<"\nEnter a number: ";
                        int n;
                        cin>>n;
                        cout<<"\n";
                        addque(n);
                        break;

                    case 2:
                        delq();
                        break;

                    case 3:
                        display();
                        break;
                    case 4:
                        return 0;
                        break;
                }
            }
    case 3:
```



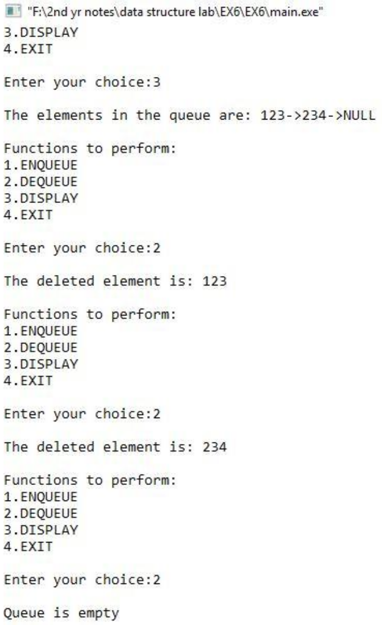
```
 Choices to do.....

 1.Stack operations
 2.Queue operations
 3.Exit

 Enter your choice: 1

Functions to perform:
1.PUSH
2.POP
3.DISPLAY
4.EXIT

Enter your choice: 1
Enter a number: 123

Functions to perform:
1.PUSH
2.POP
3.DISPLAY
4.EXIT

Enter your choice: 1
Enter a number: 124

Functions to perform:
1.PUSH
2.POP
3.DISPLAY
4.EXIT

Enter your choice: 3

The elements in the stack are: 124->123->NULL
```


Select "F:\2nd yr notes\data structure lab\EX6\EX6\main.exe"

```
Functions to perform:
1.PUSH
2.POP
3.DISPLAY
4.EXIT

Enter your choice: 3

The elements in the stack are: 124->123->NULL

Functions to perform:
1.PUSH
2.POP
3.DISPLAY
4.EXIT

Enter your choice: 2
Deleted element is: 124

Functions to perform:
1.PUSH
2.POP
3.DISPLAY
4.EXIT

Enter your choice: 2
Deleted element is: 123

Functions to perform:
1.PUSH
2.POP
3.DISPLAY
4.EXIT

Enter your choice: 2
Stack is empty
```



```
 Choices to do.....

 1.Stack operations
 2.Queue operations
 3.Exit

 Enter your choice: 2

Functions to perform:
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT

Enter your choice:1

Enter a number: 123

Functions to perform:
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT

Enter your choice:1

Enter a number: 234

Functions to perform:
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT

Enter your choice:3

The elements in the queue are: 123->234->NULL
```



```
3.DISPLAY
4.EXIT

Enter your choice:3

The elements in the queue are: 123->234->NULL

Functions to perform:
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT

Enter your choice:2

The deleted element is: 123

Functions to perform:
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT

Enter your choice:2

The deleted element is: 234

Functions to perform:
1.ENQUEUE
2.DEQUEUE
3.DISPLAY
4.EXIT

Enter your choice:2

Queue is empty
```

Video URL:

https://youtu.be/KjENb7sTBL4

Result:

The program of linked implementation of stack and queue is implemented successfully and the output is verified.