Ex. No. 5	IMPLEMENTATION OF SINGLY LINKED LIST
16-08-2017	

Question:

Implement the following operations in a singly linked list: Insertion operation at the front of the list, Deletion of an element from the list, displaying all elements in the list.

Algorithm:

1. Create structure node and declare info and the pointer node*link.
2. Assign pointer head for structure.
3. For the empty function: Create a new node and assign temp ->info=data.
4. Also temp->link=null and assign head to temp.
5. For adding at the beginning: Create new node temp and assign temp-> info = data.
6. Assign temp->link=head and assign head=temp.
7. For the delete function: create two pointers *p and *temp of type node.
8. Assign p=head.
9. If head ==null, then display" List is empty".
10. Check the condition: If head->info=data and assign temp=p.
11. Make head=head->link and delete temp. The element is removed.
12. Use while loop and check if p->link! =null.
13. Also, if p->link->info=data, temp=p->link and assign p->link=temp->link.

14. Delete the element at temp->info.
15. For display function: If head==null, display” List is empty”.
16. If else, display the elements.
17. Use switch case for the operations.
18. End.

Program:

```
/*To Implement the following operations in a singly linked list: Insertion operation
at front of the list, Deletion of an element from the list, displaying all elements in
the list. */

#include <iostream>

#include<stdlib.h>

using namespace std;


struct node

{

   int info;

   node*link;

} *head;


void addAtEmpty(int data)

{

   node *temp;
```

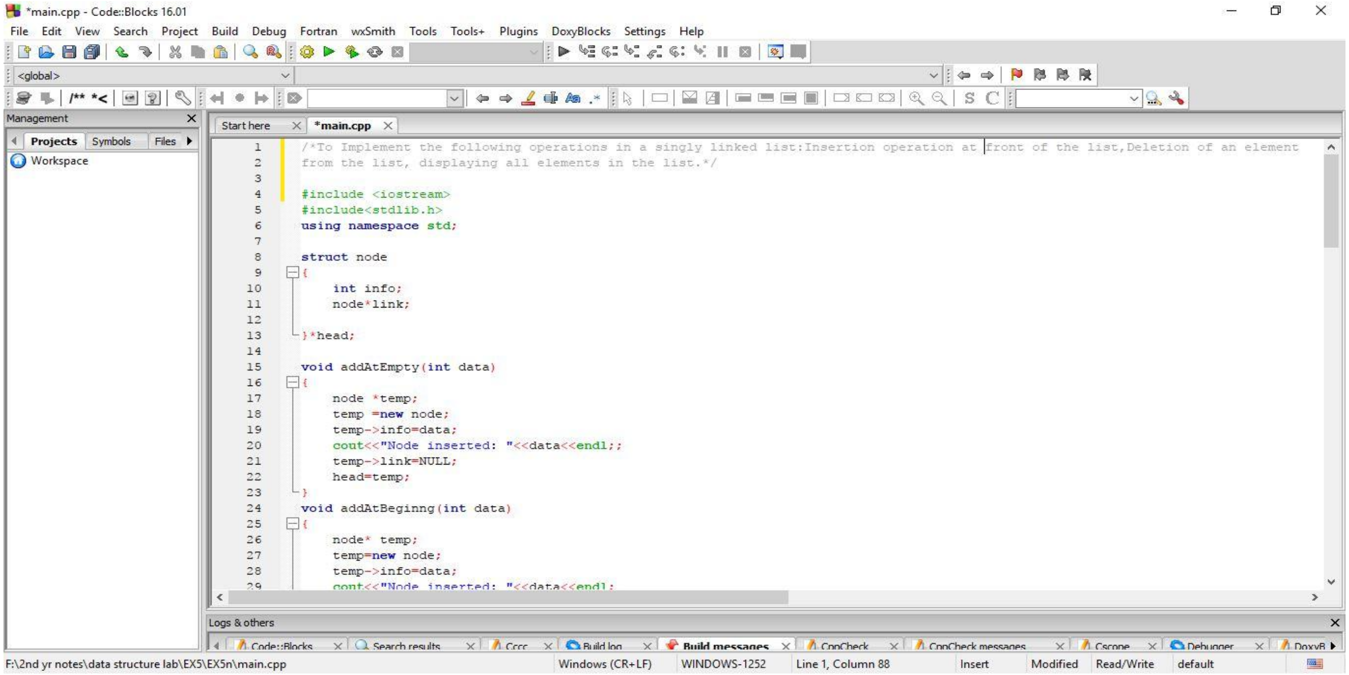
```
    temp =new node;

    temp->info=data;

    cout<<"Node inserted: "<<data<<endl;

    temp->link=NULL;

    head=temp;

}

void addAtBeginng(int data)

{

    node* temp;

    temp=new node;

    temp->info=data;

    cout<<"Node inserted: "<<data<<endl;

    temp->link=head;

    head=temp;

}

void display()

{

    node *p;

    p=head;

    if(head==NULL)

        cout<<"\n Linked List is Empty\n";

    cout<<"The elements in the linked list are: ";
    while(p!=NULL)
    {
      cout<<p->info<<" -> ";
      p=p->link;
    }
    cout<<"NULL\n\n";
}
int del(int data)
{
    node*temp,*p;
    p=head;
if(head==NULL)
{
    cout<<"\n Linked List is empty\n";
    return 0;
}
    if(head->info==data)
    {
       temp=p;
       head=head->link;
       delete temp;
       return 0;}
```

```
   while(p->link!=NULL)
   {
   if(p->link->info==data)
   {
     temp=p->link;
     p->link=temp->link;
     cout<<"Deleted node: "<<temp->info<<endl;
     delete(temp);
     return 0;
   }
   p=p->link;
 }
cout<<"Element not found";
}
int main()
{
   int opt,data,pos;
   while(1)
   {
       cout<<"\n Choices to do ";
       cout<<"\n --------------";
  cout<<"\n1.Add to Empty\n2.Add at Beginning\n3.delete \n 4.Display \n
5.Exit\n" ;
```

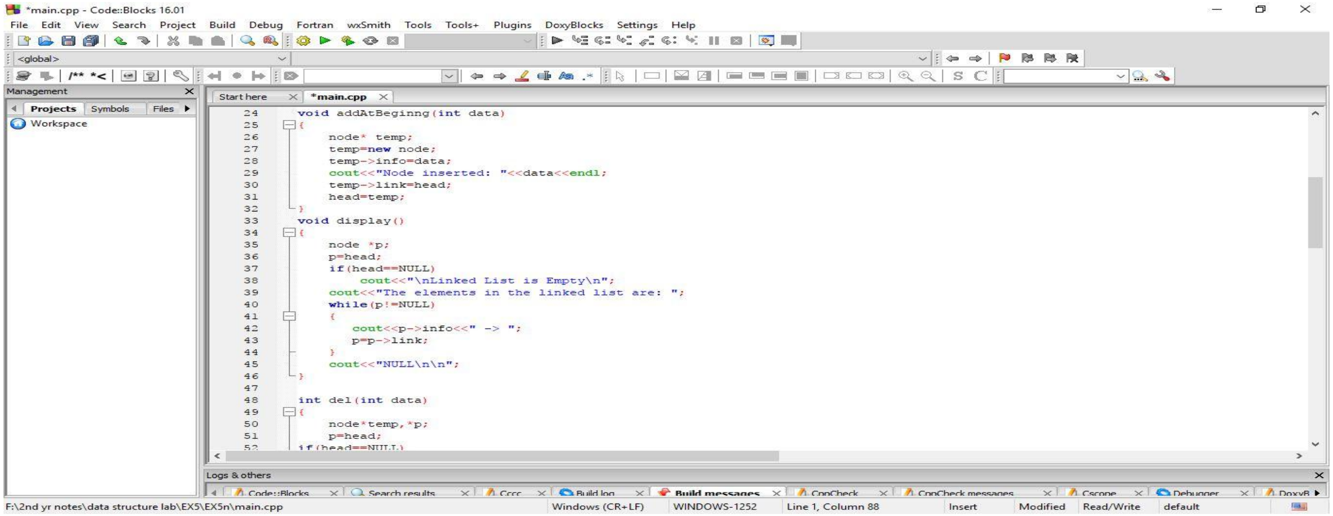
```
cout<<"\nEnter your Option: ";
cin>>opt;
switch(opt)
{
case 1:
   cout<<"Enter the Data: ";
   cin>>data;
   addAtEmpty(data);
   break;
case 2:
   cout<<"Enter the Data: ";
   cin>>data;
   addAtBeginng(data);
   break;
case 3:
   cout<<"Enter the Data to be deleted: ";
   cin>>data;
   del(data);

   break;
case 4:
   display();
```

```
        break;
    case 5:
        exit(0);
    default:
        cout<<"Sorry wrong option";
        break;
        }}
    return 0;
}
```

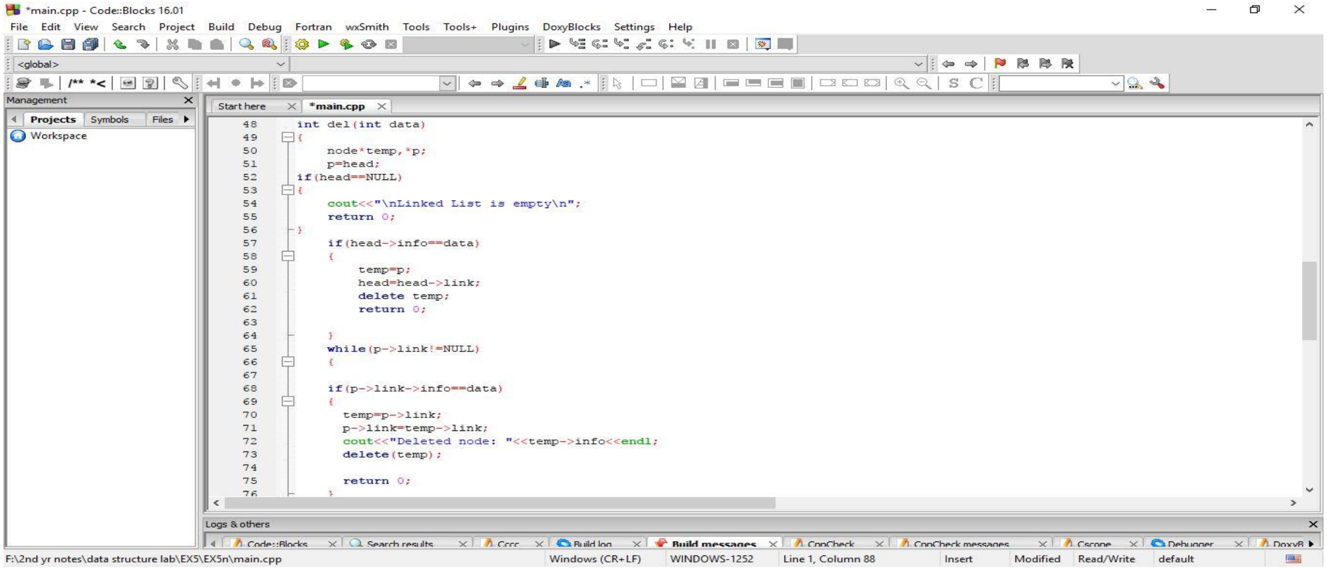
Output:



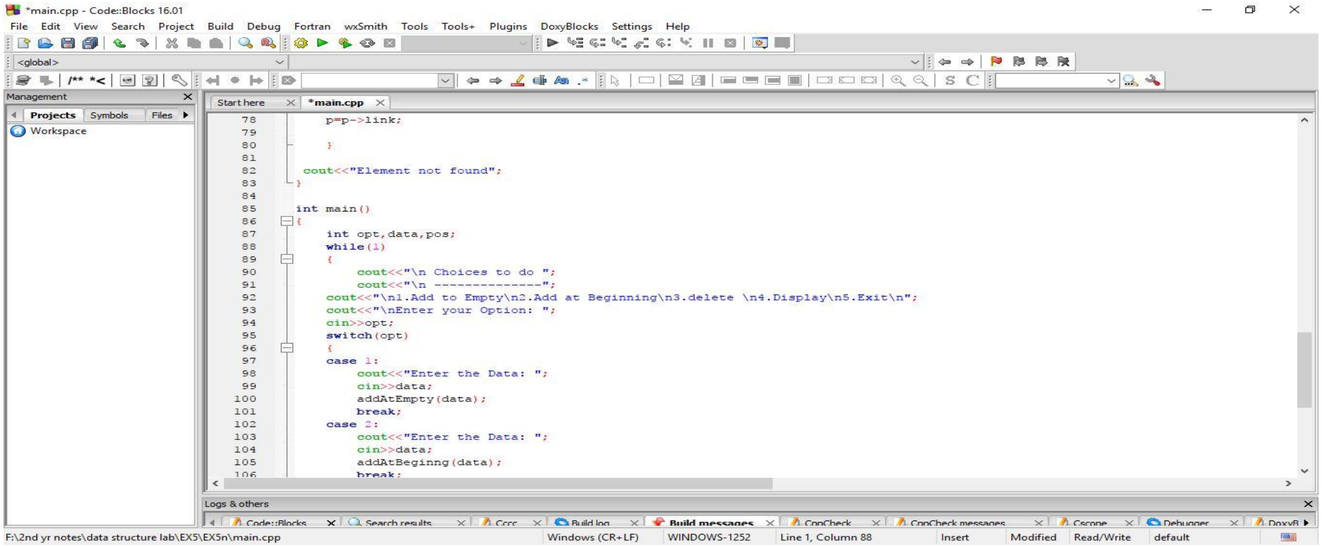
```
/*To Implement the following operations in a singly linked list:Insertion operation at front of the list,Deletion of an element
from the list, displaying all elements in the list.*/

#include <iostream>
#include<stdlib.h>
using namespace std;

struct node
{
    int info;
    node*link;

}*head;

void addAtEmpty(int data)
{
    node *temp;
    temp =new node;
    temp->info=data;
    cout<<"Node inserted: "<<data<<endl;;
    temp->link=NULL;
    head=temp;
}
void addAtBeginng(int data)
{
    node* temp;
    temp=new node;
    temp->info=data;
    cout<<"Node inserted: "<<data<<endl;
```



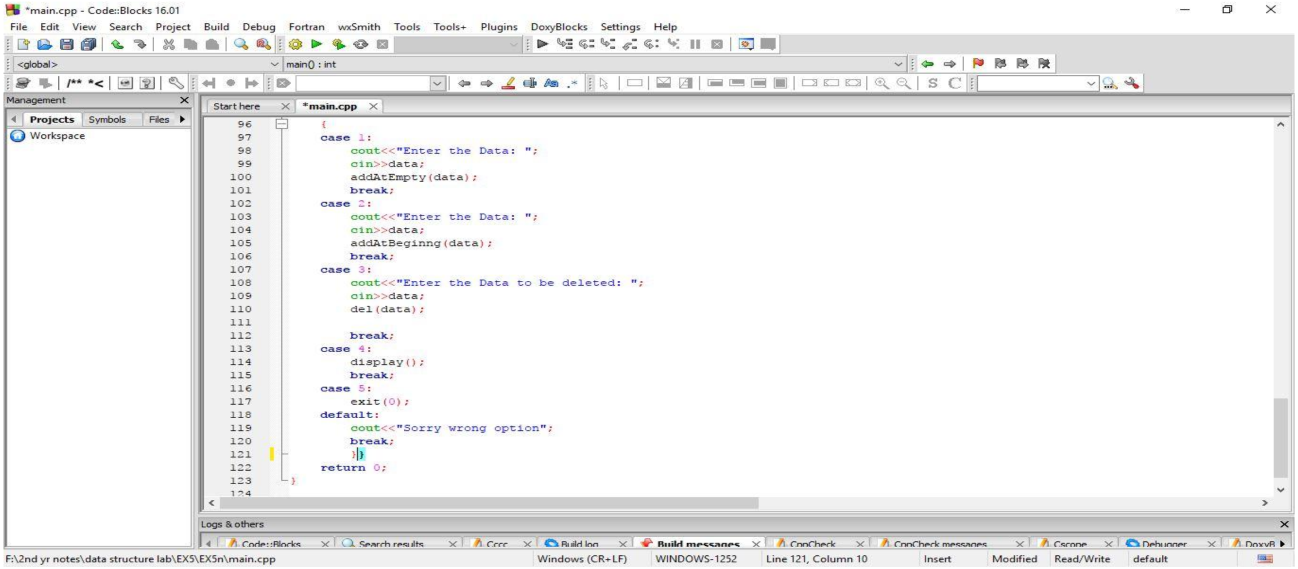
```
void addAtBeginng(int data)
{
    node* temp;
    temp=new node;
    temp->info=data;
    cout<<"Node inserted: "<<data<<endl;
    temp->link=head;
    head=temp;
}
void display()
{
    node *p;
    p=head;
    if(head==NULL)
        cout<<"\nLinked List is Empty\n";
    cout<<"The elements in the linked list are: ";
    while(p!=NULL)
    {
       cout<<p->info<<" -> ";
       p=p->link;
    }
    cout<<"NULL\n\n";
}

int del(int data)
{
    node*temp,*p;
    p=head;
if(head==NULL)
```



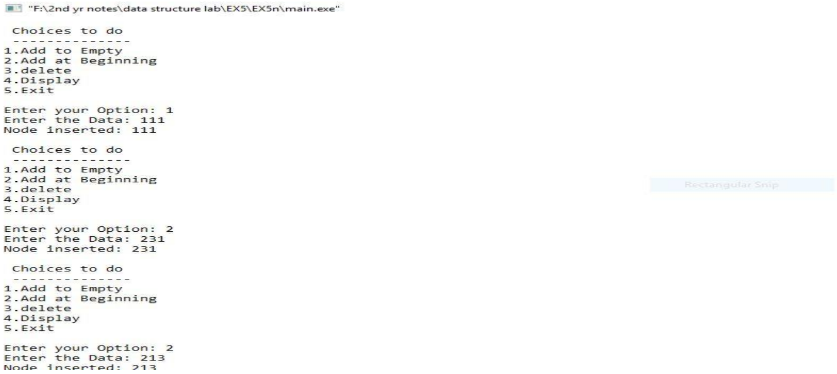
```
int del(int data)
{
    node*temp,*p;
    p=head;
if(head==NULL)
{
    cout<<"\nLinked List is empty\n";
    return 0;
}
    if(head->info==data)
    {
        temp=p;
        head=head->link;
        delete temp;
        return 0;

    }
    while(p->link!=NULL)
    {

    if(p->link->info==data)
    {
      temp=p->link;
      p->link=temp->link;
      cout<<"Deleted node: "<<temp->info<<endl;
      delete(temp);

      return 0;
    }
```



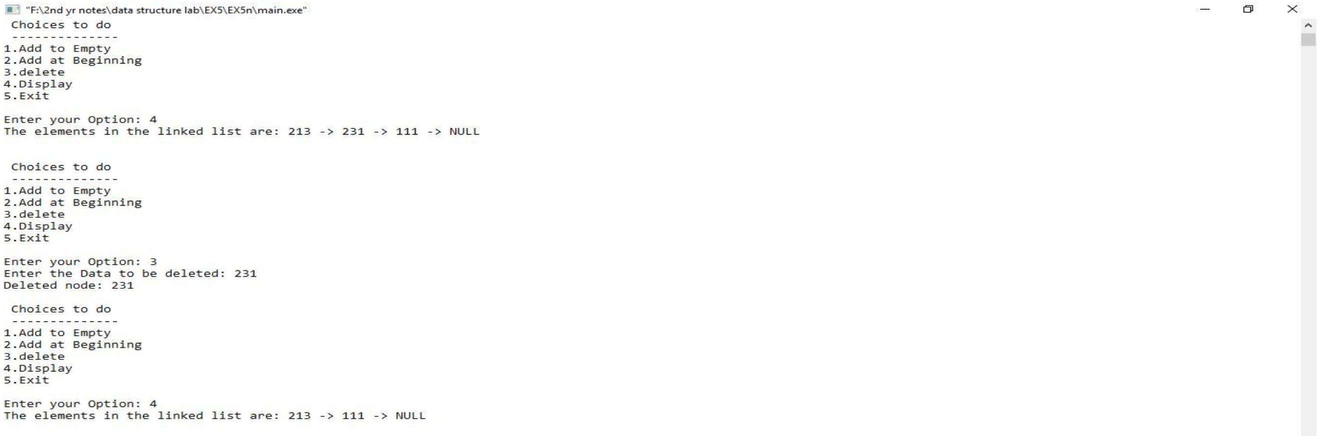
```
    p=p->link;

    }

 cout<<"Element not found";
}

int main()
{
    int opt,data,pos;
    while(1)
    {
        cout<<"\n Choices to do ";
        cout<<"\n --------------";
    cout<<"\n1.Add to Empty\n2.Add at Beginning\n3.delete \n4.Display\n5.Exit\n";
    cout<<"\nEnter your Option: ";
    cin>>opt;
    switch(opt)
    {
    case 1:
        cout<<"Enter the Data: ";
        cin>>data;
        addAtEmpty(data);
        break;
    case 2:
        cout<<"Enter the Data: ";
        cin>>data;
        addAtBeginng(data);
        break;
```


*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
Code::Blocks Search results Cccc Build log Build messages CppCheck CppCheck messages Cscope Debugger DoxyB
F:\2nd yr notes\data structure lab\EX5\EX5n\main.cpp
Windows (CR+LF) WINDOWS-1252 Line 121, Column 10 Insert Modified Read/Write default

```
    {
    case 1:
        cout<<"Enter the Data: ";
        cin>>data;
        addAtEmpty(data);
        break;
    case 2:
        cout<<"Enter the Data: ";
        cin>>data;
        addAtBeginng(data);
        break;
    case 3:
        cout<<"Enter the Data to be deleted: ";
        cin>>data;
        del(data);

        break;
    case 4:
        display();
        break;
    case 5:
        exit(0);
    default:
        cout<<"Sorry wrong option";
        break;
        }}
    return 0;
}
```



```
 Choices to do
 --------------
1.Add to Empty
2.Add at Beginning
3.delete
4.Display
5.Exit

Enter your Option: 1
Enter the Data: 111
Node inserted: 111

 Choices to do
 --------------
1.Add to Empty
2.Add at Beginning
3.delete
4.Display
5.Exit

Enter your Option: 2
Enter the Data: 231
Node inserted: 231

 Choices to do
 --------------
1.Add to Empty
2.Add at Beginning
3.delete
4.Display
5.Exit

Enter your Option: 2
Enter the Data: 213
Node inserted: 213
```



```
 Choices to do
 --------------
1.Add to Empty
2.Add at Beginning
3.delete
4.Display
5.Exit

Enter your Option: 4
The elements in the linked list are: 213 -> 231 -> 111 -> NULL


 Choices to do
 --------------
1.Add to Empty
2.Add at Beginning
3.delete
4.Display
5.Exit

Enter your Option: 3
Enter the Data to be deleted: 231
Deleted node: 231

 Choices to do
 --------------
1.Add to Empty
2.Add at Beginning
3.delete
4.Display
5.Exit

Enter your Option: 4
The elements in the linked list are: 213 -> 111 -> NULL
```

VIDEO URL:

https://youtu.be/TyYIbDbNnFs

RESULT:

The program to implement the following operations in a singly linked list: Insertion operation at the front of the list, Deletion of an element from the list, displaying all elements in the list is implemented successfully and the output is verified.