C++ Program To Implement Doubly Linked List

This C++ Program demonstrates operations on doubly linked list.

Here is source code of the C++ Program to demonstrate doubly single linked list. The C++ program is successfully compiled and run on a Linux system. The program output is also shown below.

*/\**

*\* C++ Program to Implement Doubly Linked List*

*\*/*

#include<iostream>

#include<cstdio>

#include<cstdlib>

*/\**

*\* Node Declaration*

*\*/*

using namespace std;

struct node

{

int info;

struct node \*next;

struct node \*prev;

}\*start;

*/\**

*Class Declaration*

*\*/*

class double\_llist

{

public:

void create\_list(int value);

void add\_begin(int value);

void add\_after(int value, int position);

void delete\_element(int value);

void search\_element(int value);

void display\_dlist();

void count();

void reverse();

double\_llist()

{

start = NULL;

}

};

*/\**

*\* Main: Conatins Menu*

*\*/*

int main()

{

int choice, element, position;

double\_llist dl;

while (1)

{

cout<<endl<<"----------------------------"<<endl;

cout<<endl<<"Operations on Doubly linked list"<<endl;

cout<<endl<<"----------------------------"<<endl;

cout<<"1.Create Node"<<endl;

cout<<"2.Add at begining"<<endl;

cout<<"3.Add after position"<<endl;

cout<<"4.Delete"<<endl;

cout<<"5.Display"<<endl;

cout<<"6.Count"<<endl;

cout<<"7.Reverse"<<endl;

cout<<"8.Quit"<<endl;

cout<<"Enter your choice : ";

cin>>choice;

switch ( choice )

{

case 1:

cout<<"Enter the element: ";

cin>>element;

dl.create\_list(element);

cout<<endl;

break;

case 2:

cout<<"Enter the element: ";

cin>>element;

dl.add\_begin(element);

cout<<endl;

break;

case 3:

cout<<"Enter the element: ";

cin>>element;

cout<<"Insert Element after postion: ";

cin>>position;

dl.add\_after(element, position);

cout<<endl;

break;

case 4:

if (start == NULL)

{

cout<<"List empty,nothing to delete"<<endl;

break;

}

cout<<"Enter the element for deletion: ";

cin>>element;

dl.delete\_element(element);

cout<<endl;

break;

case 5:

dl.display\_dlist();

cout<<endl;

break;

case 6:

dl.count();

break;

case 7:

if (start == NULL)

{

cout<<"List empty,nothing to reverse"<<endl;

break;

}

dl.reverse();

cout<<endl;

break;

case 8:

exit(1);

default:

cout<<"Wrong choice"<<endl;

}

}

return 0;

}

*/\**

*\* Create Double Link List*

*\*/*

void double\_llist::create\_list(int value)

{

struct node \*s, \*temp;

temp = new(struct node);

temp->info = value;

temp->next = NULL;

if (start == NULL)

{

temp->prev = NULL;

start = temp;

}

else

{

s = start;

while (s->next != NULL)

s = s->next;

s->next = temp;

temp->prev = s;

}

}

*/\**

*\* Insertion at the beginning*

*\*/*

void double\_llist::add\_begin(int value)

{

if (start == NULL)

{

cout<<"First Create the list."<<endl;

return;

}

struct node \*temp;

temp = new(struct node);

temp->prev = NULL;

temp->info = value;

temp->next = start;

start->prev = temp;

start = temp;

cout<<"Element Inserted"<<endl;

}

*/\**

*\* Insertion of element at a particular position*

*\*/*

void double\_llist::add\_after(int value, int pos)

{

if (start == NULL)

{

cout<<"First Create the list."<<endl;

return;

}

struct node \*tmp, \*q;

int i;

q = start;

for (i = 0;i < pos - 1;i++)

{

q = q->next;

if (q == NULL)

{

cout<<"There are less than ";

cout<<pos<<" elements."<<endl;

return;

}

}

tmp = new(struct node);

tmp->info = value;

if (q->next == NULL)

{

q->next = tmp;

tmp->next = NULL;

tmp->prev = q;

}

else

{

tmp->next = q->next;

tmp->next->prev = tmp;

q->next = tmp;

tmp->prev = q;

}

cout<<"Element Inserted"<<endl;

}

*/\**

*\* Deletion of element from the list*

*\*/*

void double\_llist::delete\_element(int value)

{

struct node \*tmp, \*q;

*/\*first element deletion\*/*

if (start->info == value)

{

tmp = start;

start = start->next;

start->prev = NULL;

cout<<"Element Deleted"<<endl;

free(tmp);

return;

}

q = start;

while (q->next->next != NULL)

{

*/\*Element deleted in between\*/*

if (q->next->info == value)

{

tmp = q->next;

q->next = tmp->next;

tmp->next->prev = q;

cout<<"Element Deleted"<<endl;

free(tmp);

return;

}

q = q->next;

}

*/\*last element deleted\*/*

if (q->next->info == value)

{

tmp = q->next;

free(tmp);

q->next = NULL;

cout<<"Element Deleted"<<endl;

return;

}

cout<<"Element "<<value<<" not found"<<endl;

}

*/\**

*\* Display elements of Doubly Link List*

*\*/*

void double\_llist::display\_dlist()

{

struct node \*q;

if (start == NULL)

{

cout<<"List empty,nothing to display"<<endl;

return;

}

q = start;

cout<<"The Doubly Link List is :"<<endl;

while (q != NULL)

{

cout<<q->info<<" <-> ";

q = q->next;

}

cout<<"NULL"<<endl;

}

*/\**

*\* Number of elements in Doubly Link List*

*\*/*

void double\_llist::count()

{

struct node \*q = start;

int cnt = 0;

while (q != NULL)

{

q = q->next;

cnt++;

}

cout<<"Number of elements are: "<<cnt<<endl;

}

*/\**

*\* Reverse Doubly Link List*

*\*/*

void double\_llist::reverse()

{

struct node \*p1, \*p2;

p1 = start;

p2 = p1->next;

p1->next = NULL;

p1->prev = p2;

while (p2 != NULL)

{

p2->prev = p2->next;

p2->next = p1;

p1 = p2;

p2 = p2->prev;

}

start = p1;

cout<<"List Reversed"<<endl;

}

$ **g++** doubly\_llist.cpp

$ a.out

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 2

Enter the element: 100

First Create the list.

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 3

Enter the element: 200

Insert Element after postion: 1

First Create the list.

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 4

List empty,nothing to delete

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 5

List empty,nothing to display

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 6

Number of elements are: 0

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 7

List empty,nothing to reverse

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 1

Enter the element: 100

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 5

The Doubly Link List is :

100 **<**-**>** NULL

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 2

Enter the element: 200

Element Inserted

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 5

The Doubly Link List is :

200 **<**-**>** 100 **<**-**>** NULL

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 3

Enter the element: 50

Insert Element after postion: 2

Element Inserted

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 5

The Doubly Link List is :

200 **<**-**>** 100 **<**-**>** 50 **<**-**>** NULL

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 3

Enter the element: 150

Insert Element after postion: 3

Element Inserted

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 5

The Doubly Link List is :

200 **<**-**>** 100 **<**-**>** 50 **<**-**>** 150 **<**-**>** NULL

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 6

Number of elements are: 4

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 4

Enter the element **for** deletion: 50

Element Deleted

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 5

The Doubly Link List is :

200 **<**-**>** 100 **<**-**>** 150 **<**-**>** NULL

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 6

Number of elements are: 3

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 7

List Reversed

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 5

The Doubly Link List is :

150 **<**-**>** 100 **<**-**>** 200 **<**-**>** NULL

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 3

Enter the element: 200

Insert Element after postion: 100

There are **less** than 100 elements.

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 4

Enter the element **for** deletion: 150

Element Deleted

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 5

The Doubly Link List is :

100 **<**-**>** 200 **<**-**>** NULL

---------------------------------

Operations on Doubly linked list

---------------------------------

1.Create Node

2.Add at begining

3.Add after

4.Delete

5.Display

6.Count

7.Reverse

8.Quit

Enter your choice : 8

------------------

**(**program exited with code: 1**)**

Press **return** to **continue**