/\*Write a C++ program to create a double linked list and display them

note

1. Creation

Create a list with required number of nodes by accessing data elements,

i.e Read the data & create the linked list with required number of nodes until zero is pressed

2. Display elements from left to right

If no nodes are present then print "List is empty".

Otherwise display all elements in list.

3. Display elements from right to left

If no nodes are present then print "List is empty".

Otherwise display all elements in list.

4. print "Exiting" and come out of the program

sample

input & output:

1. Create

2. Display from left to right

3. Display from right to left

4. Exit

Enter your choice:1

Enter data to create nodes & press '0' to exit: 10

20

30

40

0

1. Create

2. Display from left to right

3. Display from right to left

4. Exit

Enter your choice:2

10 20 30 40

1. Create

2. Display from left to right

3. Display from right to left

4. Exit

Enter your choice:3

40 30 20 10

1. Create

2. Display from left to right

3. Display from right to left

4. Exit

Enter your choice:4

Exiting

\*/

#include<iostream>

#include<cstdlib>

using namespace std;

class Node {

public:

int data;

Node \*next;

Node \*prev;

public:

Node(int x) {

data = x;

next = NULL;

prev = NULL;

}

};

class Doublell

{

public:

Node \*head,\*tail;

int count;

Doublell()

{

head = NULL;

tail = NULL;

count=0;

}

void create(int num);

void display\_left();

void display\_right();

};

void Doublell::create(int num)

{

Node \*nn;

if(head==NULL)

{

head=new Node(num);

tail=head;

count++;

}

else

{

nn=new Node(num);

tail->next=nn;

nn->prev=tail;

tail=nn;

count++;

}

}

void Doublell::display\_left()

{

Node \*temp;

if(head==NULL)

{

cout<<"List is empty"<<endl;

}

else

{

for(temp=head;temp!=NULL;temp=temp->next)

cout<<temp->data<<" ";

cout<<endl;

}

}

void Doublell::display\_right()

{

Node \*temp;

if(head==NULL)

{

cout<<"List is empty"<<endl;

}

else

{

for(temp=tail;temp!=NULL;temp=temp->prev)

cout<<temp->data<<" ";

cout<<endl;

}

}

int main()

{

Doublell l;

int num,ch;

do

{

cout<<"\n1. Create"<<endl;

cout<<"2. Display from left to right"<<endl;

cout<<"3. Display from right to left"<<endl;

cout<<"4. Exit"<<endl;

cout<<"Enter your choice:";

cin>>ch;

switch(ch)

{

case 1:

cout<<"Enter data to create nodes & press '0' to exit: ";

do{

cin>>num;

if(num==0)

break;

l.create(num);

}while(1);

break;

case 2:

l.display\_left();

break;

case 3:

l.display\_right();

break;

case 4:

cout<<"Exiting\n";

exit(0);

break;

default:

cout<<"Wrong Choice\n";

}

}while(1);

}

/\*Write a C++ program to create a double linked list, insert elements into it and display them

note

1. Creation

Create a list with required number of nodes by accessing data elements,

i.e Read the data & create the linked list with required number of nodes until zero is pressed

2. For insertion read position and data

a. if position is 1 & when list is empty - insert element at position 1

and print statement "Creating the list with head node"

b. if position value is within number of elements in list then write code to insert element after required position in the list

c. if position > count or position<=0 then print "Position not found"

3. Display elements from left to right

If no nodes are present then print "List is empty".

Otherwise display all elements in list.

4. Display elements from right to left

If no nodes are present then print "List is empty".

Otherwise display all elements in list.

5. print "Exiting" and come out of the program

sample

input & output:

case = 1

1. Create

2. Insert after required position

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:1

Enter data to create nodes & press '0' to exit: 10

20

30

0

1. Create

2. Insert after required position

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:2

Enter pos: 1

Enter data: 11

1. Create

2. Insert after required position

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:3

10 11 20 30

1. Create

2. Insert after required position

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:2

Enter pos: 4

Enter data: 44

1. Create

2. Insert after required position

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:3

10 11 20 30 44

1. Create

2. Insert after required position

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:2

Enter pos: 3

Enter data: 55

1. Create

2. Insert after required position

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:3

10 11 20 55 30 44

1. Create

2. Insert after required position

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:4

44 30 55 20 11 10

1. Create

2. Insert after required position

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:5

Exiting

case = 2

1. Create

2. Insert after required position

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:3

List is empty

1. Create

2. Insert after required position

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:4

List is empty

1. Create

2. Insert after required position

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:2

Enter pos: 0

Enter data: 10

Position not found

1. Create

2. Insert after required position

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:2

Enter pos: 1

Enter data: 10

Creating the list with head Node

1. Create

2. Insert after required position

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:2

Enter pos: 3

Enter data: 20

Position not found

1. Create

2. Insert after required position

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:5

Exiting

\*/

#include<iostream>

#include<cstdlib>

using namespace std;

class Node {

public:

int data;

Node \*next;

Node \*prev;

public:

Node() {

data = 0;

next = NULL;

prev = NULL;

}

Node(int x) {

data = x;

next = NULL;

prev = NULL;

}

};

class Doublell

{

public:

Node \*head,\*tail;

int count;

Doublell()

{

head = NULL;

tail = NULL;

count=0;

}

void create(int num);

void insertAfterPos(int num,int pos);

void display\_left();

void display\_right();

};

void Doublell::create(int num)

{

Node \*nn;

if(head==NULL)

{

head=new Node(num);

tail=head;

count++;

}

else

{

nn=new Node(num);

tail->next=nn;

nn->prev=tail;

tail=nn;

count++;

}

}

void Doublell::insertAfterPos(int num,int pos)

{

Node \*temp1,\*nn;

int i;

if(pos==1 && head==NULL)

{

head= new Node(num);

tail=head;

cout<<"Creating the list with head Node"<<endl;

count=1;

return;

}

if(pos>count || pos<=0)

{

cout<<"Position not found"<<endl;

return;

}

else

{

nn=new Node(num);

for(i=1,temp1=head;i<pos;i++,temp1=temp1->next);

if(temp1->next!=NULL)

{

nn->next=temp1->next;

nn->next->prev=nn;

nn->prev=temp1;

temp1->next=nn;

count++;

}

else

{

nn->prev=temp1;

temp1->next=nn;

tail=nn;

count++;

}

}

}

void Doublell::display\_left()

{

Node \*temp;

if(head==NULL)

{

cout<<"List is empty"<<endl;

}

else

{

for(temp=head;temp!=NULL;temp=temp->next)

cout<<temp->data<<" ";

cout<<endl;

}

}

void Doublell::display\_right()

{

Node \*temp;

if(head==NULL)

{

cout<<"List is empty"<<endl;

}

else

{

for(temp=tail;temp!=NULL;temp=temp->prev)

cout<<temp->data<<" ";

cout<<endl;

}

}

int main()

{

Doublell l;

int num,ch,pos;

do

{

cout<<"\n1. Create"<<endl;

cout<<"2. Insert after required position"<<endl;

cout<<"3. Display from left to right"<<endl;

cout<<"4. Display from right to left"<<endl;

cout<<"5. Exit"<<endl;

cout<<"Enter your choice:";

cin>>ch;

switch(ch)

{

case 1:

cout<<"Enter data to create nodes & press '0' to exit: ";

do{

cin>>num;

if(num==0)

break;

l.create(num);

}while(1);

break;

case 2:

cout<<"Enter pos: ";

cin>>pos;

cout<<"Enter data: ";

cin>>num;

l.insertAfterPos(num,pos);

break;

case 3:

l.display\_left();

break;

case 4:

l.display\_right();

break;

case 5:

cout<<"Exiting\n";

exit(0);

break;

default:

cout<<"Wrong Choice\n";

}

}while(1);

}

/\*Write a C++ program to create a double linked list, delete node from it and display them

note

1. Creation

- Create a list with required number of nodes by accessing data elements,

- i.e Read the data & create the linked list with required number of nodes until zero is pressed

2. Deleting particular node

- Read data, then if list is empty then print "List is empty"

- Read data, if element is present (i.e it includes single node list or

in group of nodes beginning, ending or middle node)then delete it & print eg: "10 is deleted"

- Read data, if element is not present then print "10 not found"

3. Display elements from left to right

- If no nodes are present then print "List is empty".

- Otherwise display all elements in list.

4. Display elements from right to left

- If no nodes are present then print "List is empty".

- Otherwise display all elements in list.

5. print "Exiting" and come out of the program

sample

input & output:

1. Create

2. Delete a particular node

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:1

Enter data to create nodes & press '0' to exit: 10

20

30

40

50

0

1. Create

2. Delete a particular node

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:3

10 20 30 40 50

1. Create

2. Delete a particular node

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:4

50 40 30 20 10

1. Create

2. Delete a particular node

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:2

Enter data: 10

10 is deleted

1. Create

2. Delete a particular node

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:2

Enter data: 50

50 is deleted

1. Create

2. Delete a particular node

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:2

Enter data: 30

30 is deleted

1. Create

2. Delete a particular node

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:3

20 40

1. Create

2. Delete a particular node

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:2

Enter data: 60

60 not found

1. Create

2. Delete a particular node

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:2

Enter data: 20

20 is deleted

1. Create

2. Delete a particular node

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:2

Enter data: 40

40 is deleted

1. Create

2. Delete a particular node

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:2

Enter data: 55

List is empty

1. Create

2. Delete a particular node

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:3

List is empty

1. Create

2. Delete a particular node

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:4

List is empty

1. Create

2. Delete a particular node

3. Display from left to right

4. Display from right to left

5. Exit

Enter your choice:5

Exiting

\*/

#include<iostream>

#include<cstdlib>

using namespace std;

class Node {

public:

int data;

Node \*next;

Node \*prev;

public:

Node() {

data = 0;

next = NULL;

prev = NULL;

}

Node(int x) {

data = x;

next = NULL;

prev = NULL;

}

};

class Doublell

{

public:

Node \*head,\*tail;

int count;

Doublell()

{

head = NULL;

tail = NULL;

count=0;

}

void create(int num);

void deleteNode(int x);

void display\_LtoR();

void display\_RtoL();

};

void Doublell::create(int num)

{

Node \*nn;

if(head==NULL)

{

head=new Node(num);

tail=head;

count++;

}

else

{

nn=new Node(num);

tail->next=nn;

nn->prev=tail;

tail=nn;

count++;

}

}

void Doublell :: deleteNode(int x)

{

Node \*temp,\*ptr;

if(head==NULL)

{

cout<<"List is empty"<<endl;

return;

}

if(head==tail && head->data==x)

{

cout<<head->data<<" is deleted"<<endl;

head=tail=NULL;

count--;

return;

}

if(head->data==x)

{

temp=head;

head= head->next;

head->prev=NULL;

cout<<temp->data<<" is deleted"<<endl;

delete temp;

count--;

return;

}

if(tail->data==x)

{

temp=tail;

tail=tail->prev;

tail->next=NULL;

cout<<temp->data<<" is deleted"<<endl;

delete temp;

count--;

return;

}

for(temp=head;temp->next!=NULL && temp->next->data!=x;temp=temp->next);

if(temp->next==NULL)

{

cout<<x<<" not found"<<endl;

return;

}

ptr=temp->next;

temp->next=ptr->next;

ptr->next->prev=temp;

cout<<ptr->data<<" is deleted"<<endl;

delete ptr;

count--;

}

void Doublell::display\_LtoR()

{

Node \*temp;

if(head==NULL)

{

cout<<"List is empty"<<endl;

}

else

{

for(temp=head;temp!=NULL;temp=temp->next)

cout<<temp->data<<" ";

cout<<endl;

}

}

void Doublell::display\_RtoL()

{

Node \*temp;

if(head==NULL)

{

cout<<"List is empty"<<endl;

}

else

{

for(temp=tail;temp!=NULL;temp=temp->prev)

cout<<temp->data<<" ";

cout<<endl;

}

}

int main()

{

Doublell l;

int num,ch,pos;

do

{

cout<<"\n1. Create"<<endl;

cout<<"2. Delete a particular node"<<endl;

cout<<"3. Display from left to right"<<endl;

cout<<"4. Display from right to left"<<endl;

cout<<"5. Exit"<<endl;

cout<<"Enter your choice:";

cin>>ch;

switch(ch)

{

case 1:

cout<<"Enter data to create nodes & press '0' to exit: ";

do{

cin>>num;

if(num==0)

break;

l.create(num);

}while(1);

break;

case 2:

cout<<"Enter data: ";

cin>>num;

l.deleteNode(num);

break;

case 3:

l.display\_LtoR();

break;

case 4:

l.display\_RtoL();

break;

case 5:

cout<<"Exiting\n";

exit(0);

break;

default:

cout<<"Wrong Choice\n";

}

}while(1);

}