***//Assignment-3 Data Structures Lab***

***//Implement Linked List using templates. Include functions for insertion, deletion and search of a number, reverse the list and concatenate two linked lists (include a function and also overload operator +)***

#include<iostream>

using namespace std;

#include<conio.h>

template<class T>

class node

{

public:

T data;

node<T> \*link;

};

template<class T>

class list:public node<T>

{

private:

node<T> \*first;

public:

list();

~list();

void createlist();

void deletion();

void insertion();

void display();

void search();

void reverse();

};

template<class T>

list<T>::list()

{

first=NULL;

}

template<class T>

list<T>::~list()

{

node<T> \*next;

while(first)

{

next=first->link;

delete first;

first=next;

}

}

template<class T>

void list<T>::createlist()

{

T a;

node<T> \*cur,\*ptr;

cout<<"\nEnter data Of New Node (Enter 0 To Have other options):";

cin>>a;

while(a)

{

cur=new node<T>;

cur->data=a;

cur->link=NULL;

if(first==NULL)

first=cur;

else

ptr->link=cur;

ptr=cur;

cout<<"\nEnter data Of New Node (Enter 0 To Have other options):";

cin>>a;

}

}

template<class T>

void list<T>::insertion()

{

node<T> \*cur,\*ptr;

T ele;

char ch;

ptr=first;

cur=new node<T>;

cout<<"\nEnter data Of New Node:";

cin>>cur->data;

cur->link=NULL;

cout<<"\n Do u wish to insert at the start [y/n]:";

cin>>ch;

if(ch=='Y'||ch=='y')

{

cur->link=first;

first=cur;

}

else

{

cout<<"\n Specify after which element do u want to insert :";

cin>>ele;

while(ptr!=NULL)

{

if(ptr->data==ele)

{

cur->link=ptr->link;

ptr->link=cur;

break;

}

else

{

ptr=ptr->link;

}

}

}

}

template<class T>

void list<T>::deletion()

{

T ele;

char ch;

node<T> \*ptr,\*ptr1;

if(first==NULL)

{

cout<<"\nSorry list is empty.";

}

else

{

ptr=first;

cout<<"\nDo u want to delete first element? [y/n]:";

cin>>ch;

if(ch=='y'||ch=='Y')

{

first=first->link;

delete ptr;

}

else

{

cout<<"\nSpecify which element do u want to delete :";

cin>>ele;

while(ptr!=NULL)

{

if(ptr->link->data==ele)

{

ptr1=ptr->link;

ptr->link=ptr1->link;

delete ptr1;

return;

}

else

{

ptr=ptr->link;

}

}

}

}

}

template<class T>

void list<T>::display()

{

node<T> \*ptr;

if(first==NULL)

{

cout<<"\n Sorry list is empty..";

}

else

{

ptr=first;

while(ptr!=NULL)

{

cout<<ptr->data<<" ";

ptr=ptr->link;

}

}

}

template<class T>

void list<T>::search()

{

T value;

int pos = 0;

bool flag = false;

if (first == NULL)

{

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

return;

}

cout<<"Enter the value to be searched: ";

cin>>value;

node<T> \*s;

s = first;

while (s != NULL)

{

pos++;

if (s->data == value)

{

flag = true;

cout<<"Element "<<value<<" is found at position "<<pos<<endl;

}

s = s->link;

}

if (!flag)

cout<<"Element "<<value<<" not found in the list"<<endl;

}

template<class T>

void list<T>::reverse()

{

node<T> \*ptr1, \*ptr2, \*ptr3;

if (first == NULL)

{

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

return;

}

if (first->link== NULL)

{

return;

}

ptr1 = first;

ptr2 = ptr1->link;

ptr3 = ptr2->link;

ptr1->link = NULL;

ptr2->link = ptr1;

while (ptr3 != NULL)

{

ptr1 = ptr2;

ptr2 = ptr3;

ptr3 = ptr3->link;

ptr2->link = ptr1;

}

first = ptr2;

cout<<"List is reversed";

}

int main()

{

int n;

list <int> l;

l.createlist();

do

{

cout<<"\n 1.Insertion \n2.Deletion \n3.Print List\n4.Search an element\n5.Reverse the list \n6.Exit \n";

cout<<"\n Enter your option : ";

cin>>n;

switch(n)

{

case 1: l.insertion();

break;

case 2: l.deletion();

break;

case 3: l.display();

break;

case 4: l.search();

break;

case 5: l.reverse();

break;

case 6:

exit(0);

break;

}

}while(n<=6);

getch();

return 0;

}

**OUTPUT:**

Enter data Of New Node (Enter 0 To Have other options):1

Enter data Of New Node (Enter 0 To Have other options):2

Enter data Of New Node (Enter 0 To Have other options):3

Enter data Of New Node (Enter 0 To Have other options):0

1.Insertion

2.Deletion

3.Print List

4.Search an element

5.Reverse the list

6.Exit

Enter your option : 1

Enter data Of New Node:0

Do u wish to insert at the start [y/n]:y

1.Insertion

2.Deletion

3.Print List

4.Search an element

5.Reverse the list

6.Exit

Enter your option : 3

0 1 2 3

1.Insertion

2.Deletion

3.Print List

4.Search an element

5.Reverse the list

6.Exit

Enter your option : 2

Do u want to delete first element? [y/n]:n

Specify which element do u want to delete :3

1.Insertion

2.Deletion

3.Print List

4.Search an element

5.Reverse the list

6.Exit

Enter your option : 3

0 1 2

1.Insertion

2.Deletion

3.Print List

4.Search an element

5.Reverse the list

6.Exit

Enter your option : 4

Enter the value to be searched: 2

Element 2 is found at position 3

1.Insertion

2.Deletion

3.Print List

4.Search an element

5.Reverse the list

6.Exit

Enter your option : 5

List is reversed

1.Insertion

2.Deletion

3.Print List

4.Search an element

5.Reverse the list

6.Exit

Enter your option : 3

2 1 0

1.Insertion

2.Deletion

3.Print List

4.Search an element

5.Reverse the list

6.Exit

Enter your option :