//NC AMEER HAMZA ALI RANA

//DE 40 MTS/B

//program starts here

#include <iostream>

using namespace std;

struct node //creating struct with name node

{

int data ;

node \*next;//pointer is of node type as it is pointing toward another node

};

class list //creating a class of name list

{

private ://cannott be accessed by public

node \*head, \*tail;

public :

list ();// default constructor

void createnode (int value);

void display ();//display all data of nodes in a synchronized way

//MEMBER FUNCTIONS

void insert\_start(int value);

void insert\_position (int pos, int value);

void delete\_position(int pos);

void delete\_first ();

void delete\_end();

};

int main()

{

list obj;//creating object of class list

obj.createnode(25);

obj.createnode(50);

obj.createnode(90);

obj.createnode(40);

cout<<"the member of the list are "<<endl;

obj.display();

cout<<"-----------------The end--------------"<<endl;

cout<<"write any number to enter it in linked list from start"<<endl;

int number;

cin>>number;

cout<<"the list is"<<endl;

obj.insert\_start(number);

obj.display();

cout<<"-----------------The end--------------"<<endl;

cout<<"Where do you want to add the number? "<<endl;

int position,no;

cout<<"postion= ";

cin>>position;

cout<<"no. = ";

cin>>no;

cout<<"the list is"<<endl;

obj.insert\_position(position,no);

obj.display();

cout<<"-----------------The end--------------"<<endl;

int n;

cout<<" what position do you want to delete"<<endl;

cout<<"position= ";

cin>>n;

obj.delete\_position(n);

cout<<"the list is"<<endl;

obj.display();

cout<<"-------------------The end--------------"<<endl;

obj.delete\_first();

cout<<"The list is after deleting firt node:"<<endl;

obj.display();

cout<<"-----------------The end--------------"<<endl;

obj.delete\_end();

cout<<"The list is after deleting end node :"<<endl;

obj.display();

cout<<"-------------------The end--------------"<<endl;

system ("pause");

return 0;

obj.display();

}

list :: list()//default constructor

{

head= NULL;

tail= NULL;

}

void list::createnode(int value)

{

node \*temp =new node;

temp->data=value;

temp->next=NULL;

if(head==NULL)

{

head=temp;

tail=temp;

temp=NULL;

}

else

{

tail->next=temp;

tail=temp;

}

}

void list::display()//if the definition of function is outside then we will have to use the scope resolution operator ::

{

node \*temp=new node ;

temp=head;

while(temp!=NULL)

{

cout<<temp->data<<endl;

temp=temp->next;

}

}

void list::insert\_start(int value)

{

node \*temp=new node;

temp->data=value;

temp->next=head;

head=temp;

}

void list :: insert\_position (int pos,int value)

{

node \*pre=new node;

node \*cur=new node;

node \*temp=new node;

cur=head;

for(int i=1;i<pos;i++)

{

pre=cur;

cur=cur->next;

}

temp->data=value;

pre->next=temp;

temp->next=cur;

}

void list::delete\_position(int pos)

{

node\*temp1;

temp1 = head;

if (pos == 1)

{

head = head->next;

delete temp1;

return;

}

node\*temp2;

temp2 = head;

for (int i = 0; i < pos - 2; i++)

{

temp2 = temp2->next;

}

if (temp2->next == tail)

{

temp1 = tail;

delete temp1;

tail = temp2;

temp2->next = NULL;

return;

}

temp1 = temp2->next;

temp2->next = temp1->next;

delete temp1;

}

void list::delete\_end()

{

node\*temp;

temp = head;

while (temp->next != tail)

{

temp = temp->next;

}

delete temp->next;

tail = temp;

tail->next = NULL;

}

void list::delete\_first()

{

node\*temp;

temp = head;

head = head->next;

delete temp;

}