//node.h

#ifndef NOD\_H

#define NOD\_H

#include<iostream>

using namespace std;

class Node

{

int data;

Node \*next;

public:

Node();

Node(int);

void setdata(int);

int getdata()const;

void setnext(Node \*);

Node\* getnext()const;

};

#endif

//node.cpp

#include"node.h"

Node::Node(int data)

{

this->data = data;

this->next = NULL;

}

void Node::setdata(int data)

{

this->data = data;

}

int Node::getdata()const

{

return this->data;

}

void Node::setnext(Node \*next)

{

this->next = next;

}

Node\* Node::getnext()const

{

return this->next;

}

//stack.h

#ifndef STK\_H

#define STK\_H

#include"node.h"

class Stack

{

Node \*top;

public:

Stack();

//bool empty();

bool push(int);

bool pop();

void Display();

};

#endif

//stack.cpp

#include"stack.h"

Stack::Stack()

{

this->top = NULL;

}

//bool Stack::empty()

//{

// if (this->top == NULL)

// return true;

//}

bool Stack::push(int data)

{

Node \*newNode = new Node(data);

if (newNode == NULL)

{

return false;

}

if (this->top == NULL)

{

this->top = newNode;

return true;

}

else

{

newNode->setnext(top);

this->top = newNode;

return true;

}

}

bool Stack::pop()

{

//cout << " Top is = " << this->top->getdata() << endl;

//return true;

if (top == NULL) {

cout << " Stack is empty ";

return false;

}

else

{ Node \*ptr = top;

cout << "Deleted = " << this->top->getdata()<<endl;

top = this->top->getnext();

delete ptr;

return true;

}

}

void Stack::Display()

{

Node \*trav = this->top;

while (trav != NULL)

{

cout << trav->getdata()<<endl;

trav = trav->getnext();

}

}

//smain.cpp

#include"node.h"

#include"stack.h"

#include<conio.h>

int menu\_list()

{

int choice;

cout << "0.Exit" << endl;

cout << "1.Push" << endl;

cout << "2.Pop" << endl;

cout << "3.Display" << endl;

cout << "Enter the choice : ";

cin >> choice;

return choice;

}

int main()

{

int choice, data;

Stack s1;

while ((choice = menu\_list()) != 0)

{

switch (choice)

{

case 0:exit(0);

case 1:

cout << "Enter the data :";

cin >> data;

s1.push(data);

break;

case 2:

s1.pop();

break;

case 3:

s1.Display();

break;

}

}

\_getch();

return 0;

}