#pragma once

#include"node.h"

#include<iostream>

using namespace std;

class list

{

int size;

node \*headnode;

node \*currentnode;

node \*lastcurrentnode;

public:

list()

{

size = 0;

headnode = nullptr;

currentnode = nullptr;

lastcurrentnode = nullptr;

}

~list()

{

}

void insertion();

void deletion();

void update();

void start();

void next();

void back();

void tail();

int find();

void get();

void length();

void display();

void exit();

};

#include "list.h"

#include"node.h"

int main()

{

list l;

int f;

int c;

do

{

cout << endl

<< "$$$ MENU $$$" << endl

<< "1: insertion" << endl

<< "2: deletion" << endl

<< "3: update" << endl

<< "4: start" << endl

<< "5: next" << endl

<< "6: back" << endl

<< "7: tail" << endl

<< "8: find" << endl

<< "9: get" << endl

<< "10: length" << endl

<< "11: display list" << endl

<< "12: exit" << endl

<< "enter your choice: ";

cin >> c;

switch (c)

{

case 1:

{

l.insertion();

break;

}

case 2:

{

l.deletion();

break;

}

case 3:

{

l.update();

break;

}

case 4:

{

l.start();

break;

}

case 5:

{

l.next();

break;

}

case 6:

{

l.back();

break;

}

case 7:

{

l.tail();

break;

}

case 8:

{

f = l.find();

if (f == 1)

cout << "value has successfully been found" << endl;

else if (f == 0)

cout << "value has not been found" << endl;

break;

}

case 9:

{

l.get();

break;

}

case 10:

{

l.length();

break;

}

case 11:

{

l.display();

break;

}

case 12:

{

l.exit();

break;

}

default:cout << "enter correct choice" << endl;

}

} while (c != 12);

system("pause");

return 0;

}

void list::insertion()

{

int x, c;

node \*newnode;

newnode = new node();

cout << "enter value: ";

cin >> x;

newnode->set(x);

newnode->setnext(nullptr);

size++;

if (headnode == nullptr)

{

headnode = newnode;

currentnode = newnode;

lastcurrentnode = newnode;

}

else

{

cout << "## MENU ##" << endl

<< "1: after" << endl

<< "2: before" << endl

<< "3: before headnode" << endl

<< "4: exit" <<endl

<< "enter your choice" << endl;

cin >> c;

switch (c)

{

case 1:

{

newnode->setnext(currentnode->getnext());

currentnode->setnext(newnode);

lastcurrentnode = currentnode;

currentnode = newnode;

break;

}

case 2:

{

if (currentnode != headnode)

{

newnode->setnext(currentnode);

lastcurrentnode->setnext(newnode);

currentnode = newnode;

}

else

{

newnode->setnext(headnode);

headnode = newnode;

lastcurrentnode = newnode;

currentnode = newnode;

}

break;

}

case 3:

{

newnode->setnext(headnode);

headnode = newnode;

lastcurrentnode = newnode;

currentnode = newnode;

break;

}

case 4:break;

default:cout << "enter the correct choice" << endl;

}

}

}

void list::deletion()

{

if (headnode != nullptr)

{

int c;

cout << "## MENU ##" << endl

<< "1: by value" << endl

<< "2: last node" << endl

<< "3: start node" << endl

<< "4:exit" << endl

<< "enter your choice" << endl;

cin >> c;

switch (c)

{

case 1:

{

if (find() == 1)

{

if ((currentnode != headnode) || (currentnode->getnext() != nullptr))

{

lastcurrentnode->setnext(currentnode->getnext());

delete currentnode;

currentnode = lastcurrentnode->getnext();

size--;

}

else if (currentnode == headnode)

{

node \*temp;

temp = headnode;

headnode = headnode->getnext();

delete temp;

size--;

}

else if (currentnode->getnext() == nullptr)

{

lastcurrentnode->setnext(currentnode->getnext());

delete currentnode;

if (currentnode == nullptr)

{

back();

}

size--;

}

}

else

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

break;

}

case 2:

{

lastcurrentnode->setnext(currentnode->getnext());

delete currentnode;

if (currentnode == nullptr)

{

back();

}

size--;

break;

}

case 3:

{

node \*temp;

temp = headnode;

headnode = headnode->getnext();

delete temp;

size--;

break;

}

case 4:break;

default:cout << "enter correct choice" << endl;

}

}

else

{

cout << "errer: plz insert a list of numbers first" << endl;

}

}

void list::update()

{

if (headnode != nullptr)

{

int x;

if (find() == 1)

{

cout << "enter new value: ";

cin >> x;

currentnode->set(x);

cout << "value has been updated" << endl;

}

else

{

cout << "value has not been found" << endl;

}

}

else

{

cout << "errer: plz insert a list of numbers first" << endl;

}

}

void list::start()

{

if (headnode != nullptr)

{

if (currentnode != headnode)

{

lastcurrentnode = headnode;

currentnode = headnode;

cout << "now you are on start node" << endl;

cout << "now the value is: " << currentnode->get() << endl;

}

else

{

cout << "you are already on the start node" << endl;

}

}

else

{

cout << "errer: plz insert a list of numbers first" << endl;

}

}

void list::next()

{

if (headnode != nullptr)

{

if (currentnode->getnext() != nullptr)

{

lastcurrentnode = currentnode;

currentnode = currentnode->getnext();

cout << "you are currently on next node" << endl;

cout << "now the value is: " << currentnode->get() << endl;

}

else

{

cout << "error: you are currently on the last node so you cannot move forward" << endl;

}

}

else

{

cout << "errer: plz insert a list of numbers first" << endl;

}

}

void list::back()

{

if (headnode != nullptr)

{

if (currentnode != headnode)

{

node \*ltemp = headnode;

node \*temp = headnode;

while (temp != lastcurrentnode)

{

ltemp = temp;

temp = temp->getnext();

}

lastcurrentnode = ltemp;

currentnode = temp;

cout << "now you are currently on back node" << endl;

cout << "now the value is: " << currentnode->get() << endl;

}

else

{

cout << "you are already on the first node" << endl;

}

}

else

{

cout << "errer: plz insert a list of numbers first" << endl;

}

}

void list::tail()

{

if (headnode != nullptr)

{

if (currentnode->getnext() != nullptr)

{

while (currentnode->getnext() != nullptr)

{

lastcurrentnode = currentnode;

currentnode = currentnode->getnext();

}

cout << "now you are currently on end node" << endl;

cout << "now the value is: " << currentnode->get() << endl;

}

else

{

cout << "you are already on the last node" << endl;

}

}

else

{

cout << "errer: plz insert a list of numbers first" << endl;

}

}

int list::find()

{

if (headnode != nullptr)

{

int x, flag = 0;

cout << "enter a value: ";

cin >> x;

node \*temp = headnode;

node \*ltemp = headnode;

while (temp != nullptr)

{

if (x == temp->get())

{

flag = 1;

lastcurrentnode = ltemp;

currentnode = temp;

break;

}

else

{

flag = 0;

ltemp = temp;

temp = temp->getnext();

}

}

return flag;

}

else

{

cout << "errer: plz insert a list of numbers first" << endl;

}

}

void list::get()

{

if (headnode != nullptr)

{

cout << "the current index is: " << currentnode << endl;

cout << "the value of current index is: " << currentnode->get() << endl;

}

else

{

cout << "errer: plz insert a list of numbers first" << endl;

}

}

void list::length()

{

if (headnode != nullptr)

{

cout << "total size of list: " << size << endl;

}

else

{

cout << "errer: plz insert a list of numbers first" << endl;

}

}

void list::display()

{

if (headnode != nullptr)

{

node \*temp;

temp = headnode;

while (temp != nullptr)

{

cout << "value is: " << temp->get() << endl;

temp = temp->getnext();

}

}

else

{

cout << "errer: plz insert a list of numbers first" << endl;

}

}

void list::exit()

{

if (headnode != nullptr)

{

cout << "program ended" << endl;

}

else

{

cout << "errer: plz insert a list of numbers first" << endl;

}

}

#pragma once

class node

{

int object;

node \*next;

public:

node()

{

object = 0;

next = nullptr;

}

~node()

{

}

int get()

{

return object;

}

void set(int object)

{

this->object = object;

}

node \*getnext()

{

return next;

}

void setnext(node \*next)

{

this->next = next;

}

};