МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМ. И. С. ТУРГЕНЕВА»

Кафедра «Программная инженерия»

**ОТЧЁТ**

по лабораторной работе № 6

по дисциплине: «Объектно-ориентированное программирование

на С++»

Выполнил: Шорин В. Д. Шифр: 171406

Институт приборостроения, автоматизации и информационных технологий

Направление: 09.03.04 «Программная инженерия»

Группа: 71-ПГ

Проверил:

Отметка о зачете:

Дата: «\_\_» \_\_\_\_\_\_\_\_ 2018г.

Орёл. 2018

**Задание**

**Вариант 25**

Разработать класс List (однонаправленный список). В качестве стандартных типов использовать целочисленный, вещественный и символьный типы. Для пользовательского типа использовать класс из лабораторной работы № 3. Перегрузить следующие операции:1) добавление элемента в начало списка (list + item);2) удаление элемента из начала списка (--list);3) проверка на неравенство списков (!=).

**Код:**

#include <iostream>#include <iomanip>#include <string>

#include <windows.h>#include <cctype>#include <vector>

using namespace std;

template <class T>

class List{

struct Node{

T someValue; //Значение

Node\* m\_next; //Указатель на следующий узел

Node(): m\_next(NULL){};

Node(const T& t) : someValue(t), m\_next(NULL){};

};

Node\* m\_head; // Голова односвязного списка

public:

List(); ~List();

void Add( const T& t );

void Remove();

size\_t MySize() const;

void cmp(List, List);

class Iterator {

private:

Node\* m\_node;

public:

Iterator (Node\* node): m\_node(node){};

bool operator==( const Iterator& other ) const {

if( this == &other ) {return true;}return m\_node == other.m\_node; }

bool operator!=( const Iterator& other ) const {return !operator==( other );}

T operator\*() const { if( m\_node ) {return m\_node->m\_t;}

cout << "Конец списка" << endl; return T(); }

void operator++() {

if( m\_node ){m\_node = m\_node->m\_next;}cout << "Конец списка" << endl;}

};

void ShowInfo(){

Node \*temp = m\_head;

while(temp != NULL){

cout << "Some value - " << temp->someValue << endl;

temp = temp->m\_next;

}

}

void operator+(const T &t){

if(Node\* node = new Node(t)){node->m\_next = m\_head;m\_head = node }

}

bool operator !=(List &t){

Node \*temp = this->m\_head;

Node \*temp2 = t.m\_head;

int c = 0;

if(t.MySize() != this->MySize()){return 0;}

else {

for( Node\* n = m\_head; n != NULL; n = n->m\_next ) {

if(temp->someValue == temp2->someValue){c++;}

}

}

if(c == t.MySize()){return 1;} else {return 0;}

}

List& operator--(){

if(m\_head){Node\* newHead = m\_head->m\_next;

delete m\_head; m\_head = newHead

} else cout << "Список пуст!" << endl;

}

};

template <typename T>List <T>:: List(): m\_head(NULL){};

template <typename T>

void List <T>:: Add(const T &t){

if(Node\* node = new Node(t)){ node->m\_next = m\_head; m\_head =node;}}

template <typename T>

void List <T>:: Remove(){

if(m\_head){Node\*newHead = m\_head->m\_next;delete m\_head;m\_head = newHead;} else cout << "Список пуст!" << endl;

}

template< typename T >

List< T >::~List() {while( m\_head ) { Remove(); }}

template< typename T >

size\_t List< T >::MySize() const {

size\_t s = 0;for( Node\* n = m\_head; n != NULL; n = n->m\_next ) { ++s;}return s;

}

class Product{

int id; string name; string department; int cost; string information;

public:

Product(){};Product(const Product &some\_obj);

string getDep(){return department;} string getName(){return name;}

string getInfo(){return information;}

char FirstSymbolOfName(){return name[0];}

int getCost(){return cost;}

int getId(){return id;}

void ShowInfo(){ cout << "|Номер - " << id << endl; cout << "|Название - " << name << endl; cout << "|Отдел - " << department << endl;

cout << "|Цена - " << cost << endl; cout << "|Информация о товаре - " << information << endl;

}

void SetInfo(){

int i; string n,d, inf; int c;

cout << "Введите номер - "; cin >> i; id = i;

cout << "Введите название - "; cin >> n; name = n;

cout << "Введите отдел - "; cin >> d; department = d;

cout << "Введите цену - "; cin >> c; cost = c;

cout << "Введите информацию о товаре - "; cin >> inf; information = inf;

}

~Product(){cout << "Все очищено!" << endl;}

};

int main(){

SetConsoleCP(1251); SetConsoleOutputCP(1251); int exit = 1; char userChoise;

List <int> IntList; List <int> IntList2; List <double> DoubleList; List <char> CharList; List <Product> ProductList;

IntList2 + 123; IntList2 + 456; IntList2 + 789;

while(exit) { system("cls"); cout << "Выберите действие: " << endl;

cout << ">>Добавление элемента в начало списка - 1" << endl;

cout << ">>Удаление элемента из начала списка - 2" << endl;

cout << ">>Проверка на неравенство списков - 3" << endl << endl;

cout << ">>Вывод списка - 4" << endl << endl; cout << ">>Выход - 0" << endl << endl;

cout << "Ваш выбор: "; cin >> userChoise; switch(userChoise) {

case'1': {

system("cls"); char curChoise; cout << "Выберите действие: " << endl;

cout << ">>Добавление элемента в начало списка INT - 1" << endl;

cout << ">>Добавление элемента в начало списка DOUBLE - 2" << endl;

cout << ">>Добавление элемента в начало списка CHAR - 3" << endl;

cout << ">>Выход - 0" << endl << endl; cout << "Ваш выбор: ";

cin >> curChoise; switch(curChoise) {

case '1': {

system("cls");int a; cout << "Введите число: "; cin >> a;

IntList + a; IntList.ShowInfo();system("pause");break; }

case '2': {

system("cls"); double a; cout << "Введите число: "; cin >> a;

DoubleList + a;DoubleList.ShowInfo();system("pause");break;}

case '3': {

system("cls");char a; cout << "Введите символ: "; cin >> a;

CharList + a;CharList.ShowInfo();system("pause"); break; }

case '0':{break;}

}

break;

}

case'2': {

system("cls"); char curChoise; cout << "Выберите действие: " << endl;

cout << ">>Удаление элемента из начала списка INT - 1" << endl;

cout << ">>Удаление элемента из начала списка DOUBLE - 2" << endl;

cout << ">>Удаление элемента из начала списка CHAR - 3" << endl;

cout << ">>Выход - 0" << endl << endl; cout << "Ваш выбор: "; cin >> curChoise;

switch(curChoise) { case '1':{--IntList;IntList.ShowInfo();system("pause");break;}

case '2':{--DoubleList;DoubleList.ShowInfo();system("pause");break;}

case '3':{--CharList;CharList.ShowInfo();system("pause");break;} case '0':{break;}

} break; }

case'3': {

system("cls"); cout << "Содержимое первого листа:" << endl;

IntList + 123; IntList + 456; IntList + 789;

IntList.ShowInfo();cout << endl;cout << "Содержимое второго листа:" << endl;

IntList2.ShowInfo();cout << endl;

if ((IntList != IntList2) == 0){cout << "Списки не равны!" << endl;}

else if ((IntList != IntList2) == 1){cout << "Списки равны!" << endl;}

system("pause"); break; }

case '4': {

system("cls"); char curChoise; cout << "Выберите действие: " << endl;

cout << ">>Просмотр списка INT - 1" << endl; cout << ">>Просмотр списка DOUBLE - 2" << endl;

cout << ">>Просмотр списка CHAR - 3" << endl; cout << ">>Выход - 0" << endl << endl;

cout << "Ваш выбор: "; cin >> curChoise; switch(curChoise) {

case '1':{IntList.ShowInfo();system("pause");break;}

case '2':{DoubleList.ShowInfo();system("pause");break;}

case '3':{CharList.ShowInfo();system("pause");break;} case '0':{break;}

} break; }

case'0':{exit = 0;break;}

}

}

return 0;

}