Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное образовательное учреждение высшего образования

**Пермский национальный исследовательский политехнический университет**

# Факультет Электротехнический Кафедра ИТАС Специальность Промышленная робототехника

ОТЧЁТ

**о лабораторной работе №11**

# Информационные динамические структуры

Выполнил: Студент группы ПРТ-21-1Б

# Степанова И.В.

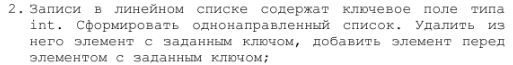
Проверил:

# Доцент кафедры ИТАС Полякова О.А.

**Пермь 2022**

## Цель лабораторной работы: применение динамических информационных структур, с использованием одно- и двунаправленых списков.

Постановка задачи: написать программу, в которой информационные создаются динамические структуры с одно- и двунаправленными списками и выполнить их обработку в соответствии со своим вариантом.



## Текст программы:

#include <iostream>

#include <cstdlib>

#include <locale.h>

#pragma warning(disable: 4996)

using namespace std;

struct Elem { //создаем список

int key;

Elem\* pnext;

static unsigned siz;

};

unsigned Elem::siz = 0;

int first\_elem(Elem\*\* pbeg, Elem\*\* pend, int d) {

Elem\* ph = new Elem;

ph->key = d;

ph->pnext = 0; //работа с первым элементом

\*pbeg = ph;

\*pend = ph;

++Elem::siz; //"размер" списка

return 0;

}

int add\_elem(Elem\*\* pend, int d) {

Elem\* ph = new Elem;

ph->key = d;

ph->pnext = 0;

(\*pend)->pnext = ph;

\*pend = ph;

++Elem::siz;

return 0;

}

int del\_elem(Elem\* pbeg, int k) { //ставит перед удаляемым элементом другой элемент(вводишь сам значение)

int n;

cout << "Введите ключ нового элемента списка: "; cin >> n;

while (pbeg) {

if (pbeg->key == k) {

Elem\* ph = new Elem;

ph->pnext = pbeg->pnext;

ph->key = pbeg->key;

pbeg->pnext = ph;

pbeg->key = n;

break;

}

pbeg = pbeg->pnext;

}

return 0;

}

int del\_elem(Elem\*\* pbeg, Elem\*\* pend, int k) { // удаление элемента, который задали удалить

Elem\* tmp = \*pbeg;

if (!\*pbeg) {

cout << "Ошибка: список пуст" << endl;

return 0;

}

if ((\*pbeg)->key == k) {

if (\*pbeg == \*pend) {

\*pbeg = 0;

\*pend = 0;

delete tmp;

--Elem::siz;

}

else {

\*pbeg = (\*pbeg)->pnext;

delete tmp;

--Elem::siz;

}

}

else {

while (tmp->pnext && tmp->pnext->key != k) {

tmp = tmp->pnext;

}

if (tmp->pnext) {

Elem\* tmp1 = tmp->pnext->pnext;

delete tmp->pnext;

tmp->pnext = tmp1;

--Elem::siz;

if (!tmp->pnext) {

\*pend = tmp;

}

}

else {

return 0;

}

}

return 0;

}

void del\_list(Elem\* pbeg) { //удаляем список

if (pbeg != NULL) {

del\_list(pbeg->pnext);

delete pbeg;

--Elem::siz;

}

}

void print\_elem(Elem\* pbeg) { //вывод списка

Elem\* ph = pbeg;

while (ph) {

cout << ph->key << ' ';

ph = ph->pnext;

}

cout << endl;

}

int main() { //сама прога

setlocale(LC\_ALL, "Russian");

Elem\* pbeg = 0;

Elem\* pend = 0;

int d;

int a;

int k;

int g = 0;

int z;

char ch;

bool exit = false;

do {

system("cls");

cout << "1 - добавить элемент/создать\n2 - удалить один\n3 - просмотр\n4 - выход" << endl;

cin >> ch;

switch (ch) {

case '1':

if (!pbeg) {

cout << "Введите 1 элемент списка: " << endl;

cin >> d;

first\_elem(&pbeg, &pend, d);

g++;

print\_elem(pbeg);

break;

}

else {

cout << "Введите новый элемент списка: " << endl;

cin >> a;

add\_elem(&pend, a);

print\_elem(pbeg);

break;

}

case '2':

cout << "Введите значение элемента для удаления: " << endl;

cin >> k;

if (del\_elem(pbeg, k) == 0) {

cout << "Элемент для удаления не найден" << endl;

}

else {

if (del\_elem(&pbeg, &pend, k) == 0) {

cout << "Элемент для удаления не найден" << endl;

}

if (g > 0) {

cout << "Список всех элементов" << endl;

print\_elem(pbeg);

}

else

cout << "Список пуст" << endl;

break;

}

break;

case '3':

if (g > 0) {

cout << "Список всех элементов" << endl;

print\_elem(pbeg);

}

else

cout << "Список пуст" << endl;

break;

case '4':

exit = true;

break;

}

\_flushall;

system("Pause");

} while (!exit);

if (pbeg != NULL) {

del\_list(pbeg->pnext);

delete pbeg;

Elem::siz = 0;

}

return 0;

## }

## Ответ для варианта №2

## 