Министерство высшего образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет» (ПНИПУ)**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

ОТЧЁТ

Лабораторная работа № 11

"Информационные динамические структуры"

Выполнил

Студент группы РИС-22-2б

Вековшинин Д. А.

Проверила

Доцент кафедры ИТАС

Полякова О. А.

Пермь 2023

**Постановка задачи**

Написать программу, в которой создаются динамические структуры и выполнить их обработку в соответствии со своим вариантом.

Для каждого вариант разработать следующие функции:

1. Создание списка.

2. Добавление элемента в список (в соответствии со своим вариантом).

3. Удаление элемента из списка (в соответствии со своим вариантом).

4. Печать списка.

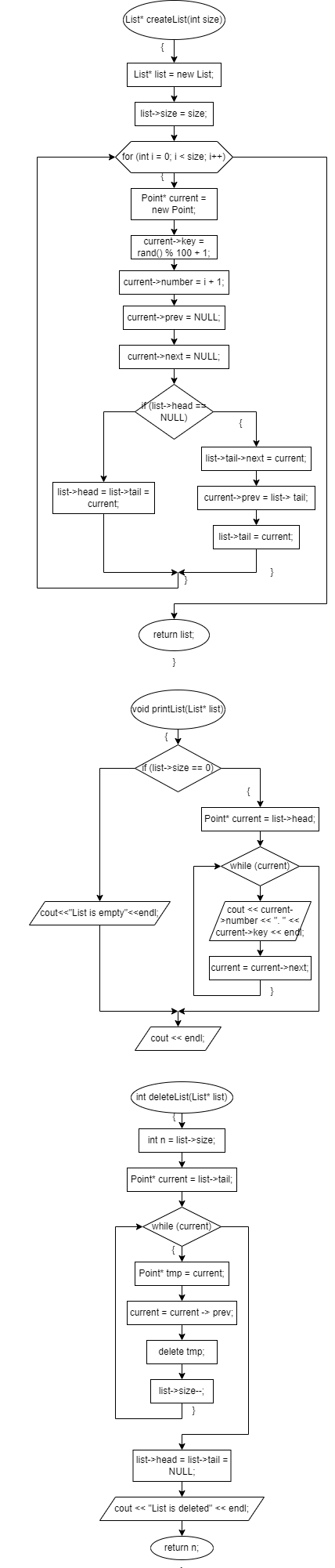
5. Запись списка в файл.

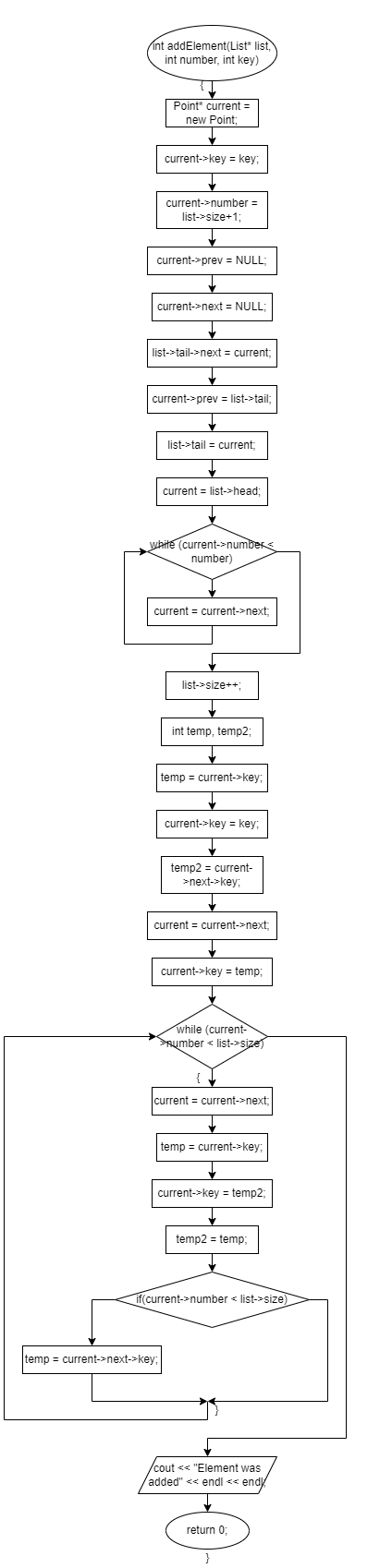
6. Уничтожение списка.

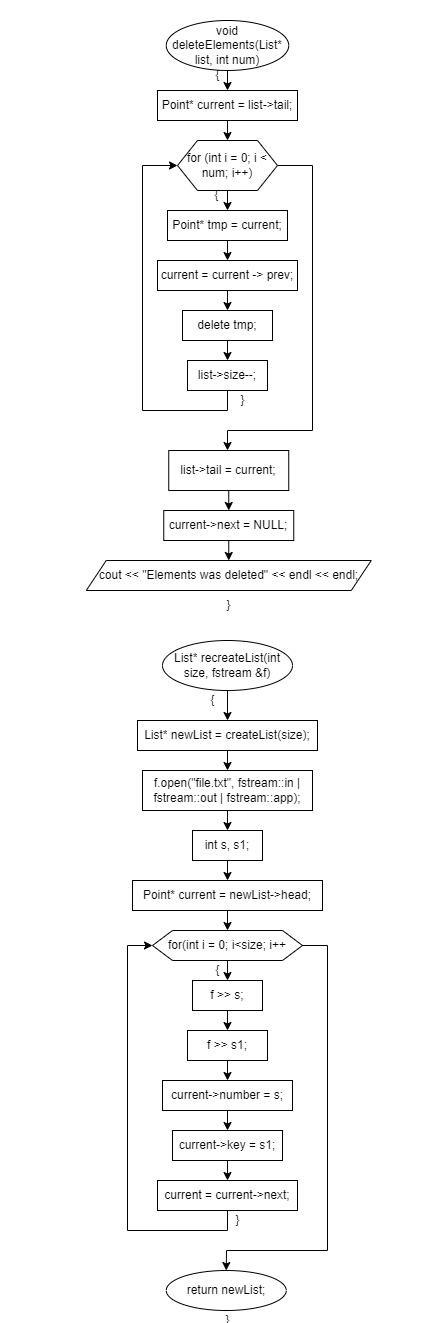
7. Восстановление списка из файла.

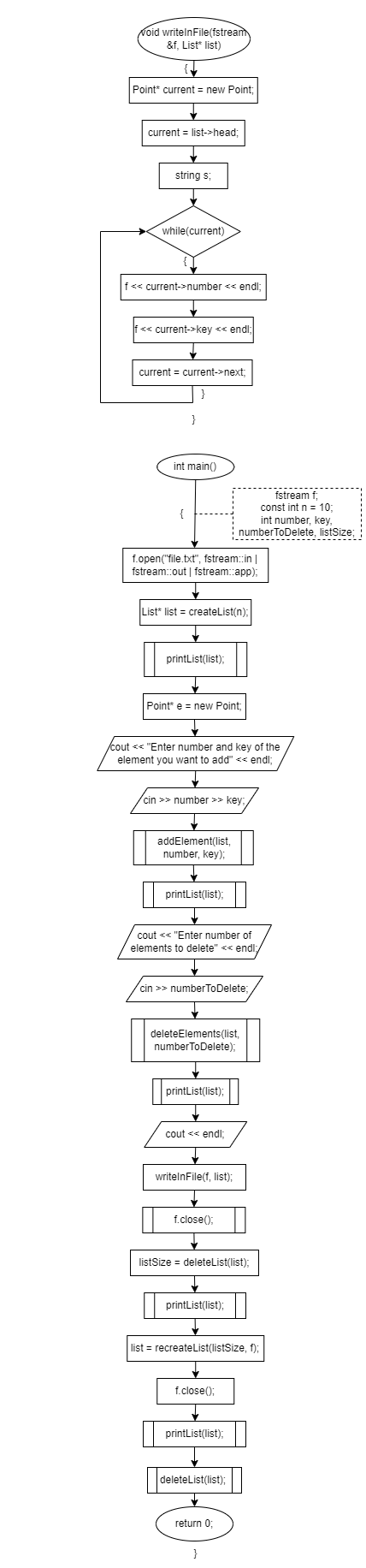
Вариант 10: Записи в линейном списке содержат ключевое поле типа int. Сформировать двунаправленный список. Добавить в него элемент с заданным номером, удалить К элементов из конца списка.

**Блок-схема**

****

****

****

****

**Код программы**

#include <iostream>

#include <string>

#include <fstream>

using namespace std;

struct Point

{

int number;

Point\* prev;

int key;

Point\* next;

};

struct List

{

size\_t size = 0;

Point\* head = NULL;

Point\* tail = NULL;

};

List\* createList(int size)

{

List\* list = new List;

list->size = size;

for (int i = 0; i < size; i++)

{

Point\* current = new Point;

current->key = rand() % 100 + 1;

current->number = i + 1;

current->prev = NULL;

current->next = NULL;

if (list->head == NULL)

list->head = list->tail = current;

else

{

list->tail->next = current;

current->prev = list-> tail;

list->tail = current;

}

}

return list;

}

void printList(List\* list)

{

if (list->size == 0)

cout<<"List is empty"<<endl;

else

{

Point\* current = list->head;

while (current)

{

cout << current->number << ". " << current->key << endl;

current = current->next;

}

}

cout << endl;

}

int deleteList(List\* list)

{

int n = list->size;

Point\* current = list->tail;

while (current)

{

Point\* tmp = current;

current = current -> prev;

delete tmp;

list->size--;

}

list->head = list->tail = NULL;

cout << "List is deleted" << endl;

return n;

}

int addElement(List\* list, int number, int key)

{

Point\* current = new Point;

current->key = key;

current->number = list->size+1;

current->prev = NULL;

current->next = NULL;

list->tail->next = current;

current->prev = list->tail;

list->tail = current;

current = list->head;

while (current->number < number)

current = current->next;

list->size++;

int temp, temp2;

temp = current->key;

current->key = key;

temp2 = current->next->key;

current = current->next;

current->key = temp;

while (current->number < list->size)

{

current = current->next;

temp = current->key;

current->key = temp2;

temp2 = temp;

if(current->number < list->size) temp = current->next->key;

}

cout << "Element was added" << endl << endl;

return 0;

}

void deleteElements(List\* list, int num)

{

Point\* current = list->tail;

for (int i = 0; i < num; i++)

{

Point\* tmp = current;

current = current->prev;

delete tmp;

list->size--;

}

list->tail = current;

current->next = NULL;

cout << "Elements was deleted" << endl << endl;

}

List\* recreateList(int size, fstream &f)

{

List\* newList = createList(size);

f.open("file.txt", fstream::in | fstream::out | fstream::app);

int s, s1;

Point\* current = newList->head;

for(int i = 0; i<size; i++)

{

f >> s;

f >> s1;

current->number = s;

current->key = s1;

current = current->next;

}

return newList;

}

void writeInFile(fstream &f, List\* list)

{

Point\* current = new Point;

current = list->head;

string s;

while(current)

{

f << current->number << endl;

f << current->key << endl;

current = current->next;

}

}

int main()

{

fstream f;

f.open("file.txt", fstream::in | fstream::out | fstream::app);

const int n = 10;

int number, key, numberToDelete, listSize;

List\* list = createList(n);

printList(list);

Point\* e = new Point;

cout << "Enter number and key of the element you want to add" << endl;

cin >> number >> key;

addElement(list, number, key);

printList(list);

cout << "Enter number of elements to delete" << endl;

cin >> numberToDelete;

deleteElements(list, numberToDelete);

printList(list);

cout << endl;

writeInFile(f, list);

f.close();

listSize = deleteList(list);

printList(list);

list = recreateList(listSize, f);

f.close();

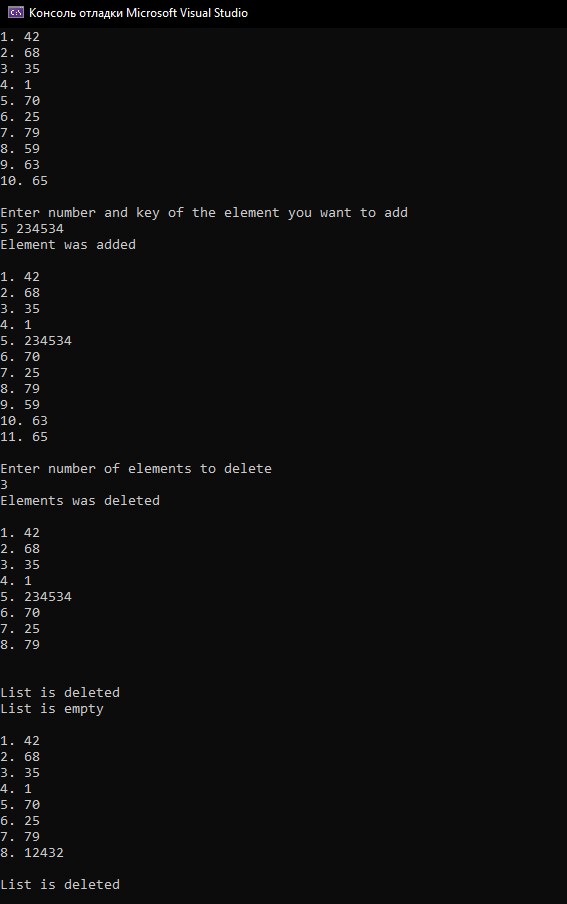
printList(list);

deleteList(list);

return 0;

}

**Результат работы программы**

****