Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«**Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №11**

**Вариант 7**

Дисциплина: «Основы алгоритмизации и программирования»

Семестр 2

Выполнил работу

Студент группы РИС-22-1Б

Бреднев Максим Павлович

Проверил

Доцент кафедры ИТАС

Полякова Ольга Андреевна

Г. Пермь-2023

**Постановка задачи**

Сформировать двунаправленный список. Удалить из него первый элемент, добавить элемент в конец списка.

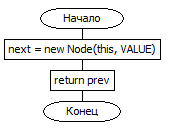
**Анализ решения**

Каждый узел двунаправленного списка содержит поля для хранения указателей на следующий и предыдущий узел.

В головном узле в поле для хранения указателя на предыдущий узел хранится нулевой указатель. В хвостовом узле в поле для хранения указателя на следующий узел хранится нулевой указатель.

**Блок-схемы**

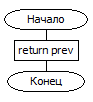
Node\* Add(int VALUE)



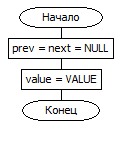
Node\* GetNext()



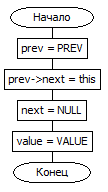
Node\* GetPrev()



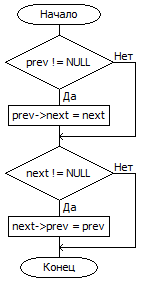
Node(int VALUE)



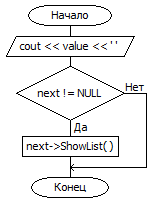
Node(Node\* PREV, int VALUE)



~Node()



void ShowList()



**Код программы**

#include <iostream>

#include <fstream>

using namespace std;

class Node

{

int value;

Node\* prev, \* next;

public:

Node(int VALUE)

{

prev = next = NULL;

value = VALUE;

}

Node(Node\* PREV, int VALUE)

{

prev = PREV;

prev->next = this;

next = NULL;

value = VALUE;

}

~Node()

{

if (prev != NULL)

prev->next = next;

if (next != NULL)

next->prev = prev;

}

Node\* Add(int VALUE)

{

next = new Node(this, VALUE);

return prev;

}

void ShowList()

{

cout << value << ' ';

if (next != NULL)

next->ShowList();

}

Node\* GetPrev()

{

return prev;

}

Node\* GetNext()

{

return next;

}

};

int main()

{

setlocale(0, "");

cout << "Вывести файл(1 или 0)\n";

int a;

cin >> a;

if (a == 1) {

ifstream fout("file.txt");

int nam;

cout << "\n";

int cnt = 0;

while (!fout.eof())

{

fout >> nam;

if (nam != 0) {

cout << nam << " ";

}

nam = 0;

}

cout << "\n";

fout.close();

}

ofstream fout("file.txt", ios::trunc);

Node\* Begin = NULL, \* End = NULL, \* Temp;

char option;

int i, value;

int n[10] = {};

int k = 0, j = 0;

do

{

if (Begin != NULL)

Begin->ShowList();

cout << "\n1 - Add to end\n2 - Delete from begin\n3 - Exit\nMy choice: ";

cin >> option;

switch (option)

{

case '1':

{

cout << "Input value: ";

cin >> value;

if (End != NULL)

{

Temp = new Node(End, value);

End = Temp;

}

else

Begin = End = new Node(value);

n[k] = value;

k++;

}break;

case '2':

{

if (Begin != NULL)

{

Temp = Begin->GetNext();

delete Begin;

Begin = Temp;

if (Begin == NULL)

End = NULL;

}

n[j] = 0;

j++;

}break;

}

} while (option != '3');

for (int i = 0; i < k; i++) {

fout << n[i] << " ";

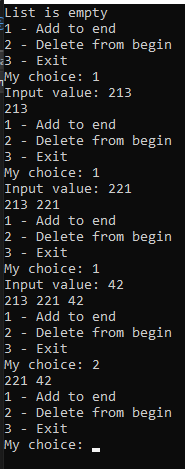
}

fout.close();

return 0;

}

**Результат программы**

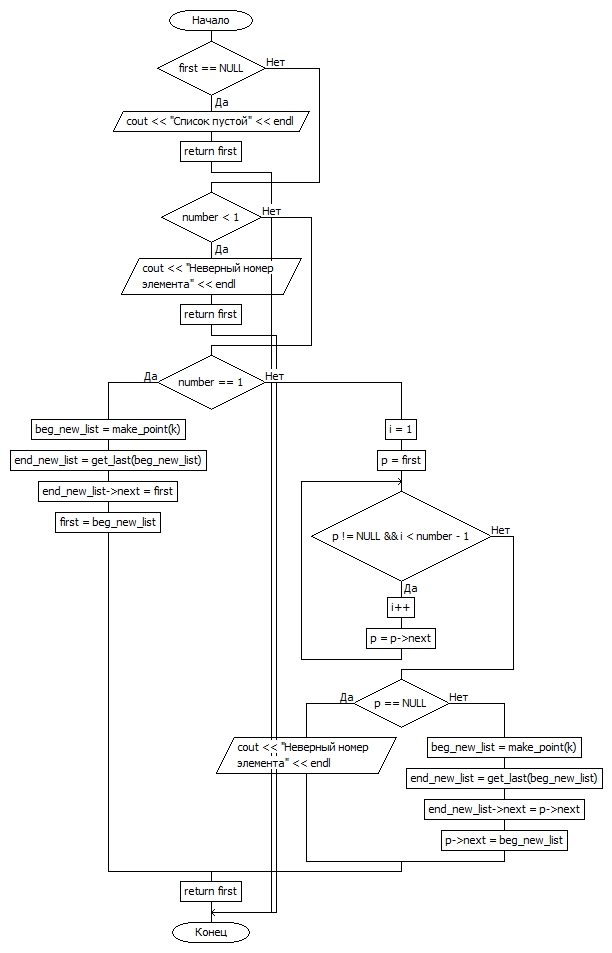
****

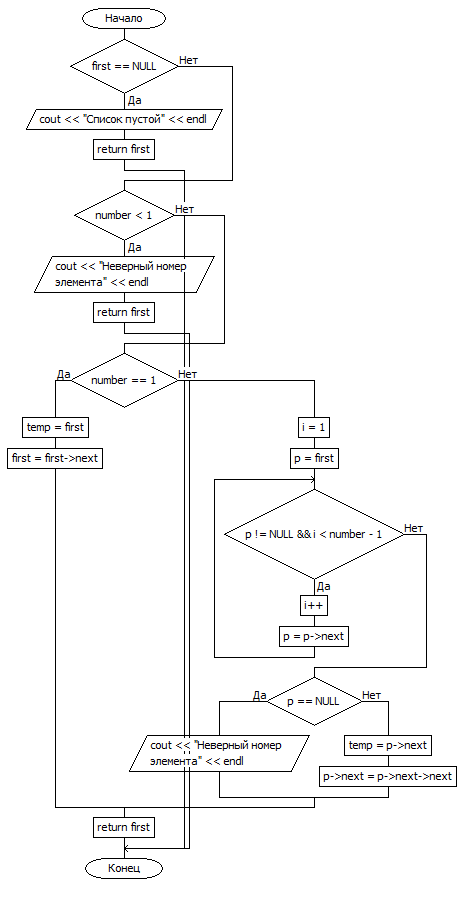
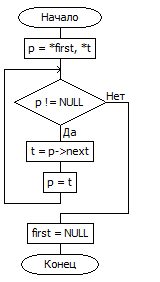
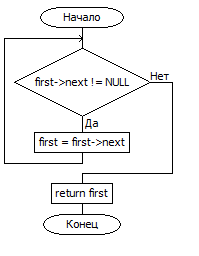
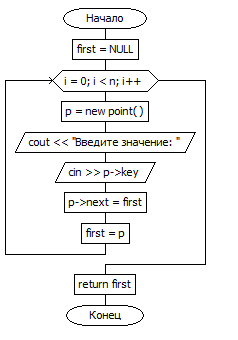
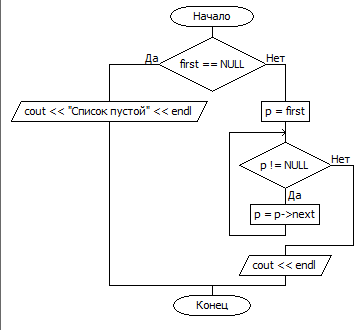
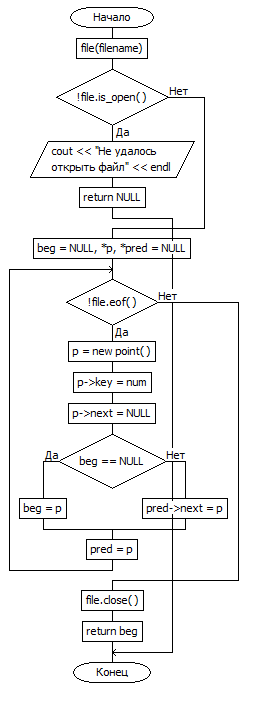
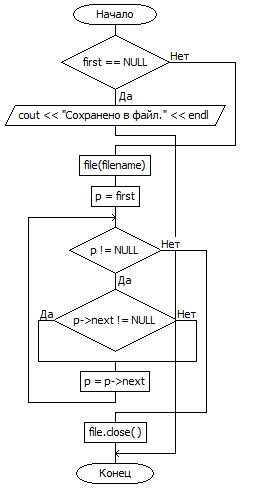
**Анализ результатов**

Как можем заметить, на первом скриншоте, программа выполняет задание.

Однонаправленный список:

**Блок схема**



**Код программы**

#include <iostream>

#include <fstream>

using namespace std;

struct point

{

int key;

point\* next;

};

point\* make\_point(int n)

{

point\* first, \* p;

first = NULL;

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

{

p = new point();

cout << "Введите значение: ";

cin >> p->key;

p->next = first;

first = p;

}

return first;

}

void print(point\* first)

{

if (first == NULL)

cout << "Список пустой" << endl;

else

{

point\* p = first;

while (p != NULL)

{

cout << p->key << " ";

p = p->next;

}

cout << endl;

}

}

point\* get\_last(point\* first)

{

while (first->next != NULL)

{

first = first->next;

}

return first;

}

point\* delete\_element(point\* first, int number)

{

if (first == NULL)

{

cout << "Список пустой" << endl;

return first;

}

if (number < 1)

{

cout << "Неверный номер элемента" << endl;

return first;

}

if (number == 1)

{

point\* temp = first;

first = first->next;

delete temp;

}

else

{

int i = 1;

point\* p = first;

while (p != NULL && i < number - 1)

{

i++;

p = p->next;

}

if (p == NULL)

cout << "Неверный номер элемента" << endl;

else

{

point\* temp = p->next;

p->next = p->next->next;

delete temp;

}

}

return first;

}

point\* add\_elements(point\* first, int number, int k)

{

if (first == NULL)

{

cout << "Список пустой" << endl;

return first;

}

if (number < 1)

{

cout << "Неверный номер элемента" << endl;

return first;

}

if (number == 1)

{

point\* beg\_new\_list = make\_point(k);

point\* end\_new\_list = get\_last(beg\_new\_list);

end\_new\_list->next = first;

first = beg\_new\_list;

}

else

{

int i = 1;

point\* p = first;

while (p != NULL && i < number - 1)

{

i++;

p = p->next;

}

if (p == NULL)

cout << "Неверный номер элемента" << endl;

else

{

point\* beg\_new\_list = make\_point(k);

point\* end\_new\_list = get\_last(beg\_new\_list);

end\_new\_list->next = p->next;

p->next = beg\_new\_list;

}

}

return first;

}

void free\_list(point\*\* first)

{

point\* p = \*first, \* t;

while (p != NULL)

{

t = p->next;

delete p;

p = t;

}

\*first = NULL;

}

void save\_to\_file(point\* first, const char\* filename)

{

if (first == NULL)

{

cout << "Сохранено в файл." << endl;

return;

}

ofstream file(filename);

point\* p = first;

while (p != NULL)

{

file << p->key;

if (p->next != NULL)

file << " ";

p = p->next;

}

file.close();

}

point\* read\_file(const char\* filename)

{

ifstream file(filename);

if (!file.is\_open())

{

cout << "Не удалось открыть файл" << endl;

return NULL;

}

point\* beg = NULL, \* p, \* pred = NULL;

int num;

while (!file.eof())

{

file >> num;

p = new point();

p->key = num;

p->next = NULL;

if (beg == NULL)

beg = p;

else

pred->next = p;

pred = p;

}

file.close();

return beg;

}

int main()

{

setlocale(LC\_ALL, "rus");

point\* beg = NULL;

int oper;

do

{

system("cls");

cout << "1.Создать список" << endl;

cout << "2.Удалить элемент с заданным номером" << endl;

cout << "3.Вставка элементов" << endl;

cout << "4.Вывод списка" << endl;

cout << "5.Сохранение в файл" << endl;

cout << "6.Загрузка из файла" << endl;

cout << "0.Выход" << endl;

cin >> oper;

switch (oper)

{

case 1:

{

int n;

cout << "Кол-во эл-тов списка = ";

cin >> n;

beg = make\_point(n);

print(beg);

break;

}

case 2:

{

print(beg);

int number;

cout << "Введите номер: ";

cin >> number;

beg = delete\_element(beg, number);

print(beg);

break;

}

case 3:

{

print(beg);

int number;

cout << "Введите номер: ";

cin >> number;

int k;

cout << "Сколько элементов добавить: ";

cin >> k;

beg = add\_elements(beg, number, k);

print(beg);

break;

}

case 4:

{

print(beg);

break;

}

case 5:

{

char filename[200];

cout << "Введите название файла: ";

cin >> filename;

save\_to\_file(beg, filename);

free\_list(&beg);

print(beg);

break;

}

case 6:

{

char filename[200];

cout << "Введите название файла: ";

cin >> filename;

beg = read\_file(filename);

print(beg);

break;

}

default:

break;

}

if (oper != 0)

system("pause");

} while (oper != 0);

free\_list(&beg);

}