Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа № 6

По дисциплине «Основы алгоритмизации и программирования»

На тему «Динамические структуры данных. Односвязные списки»

Выполнил:

Студент 1 курса 10 группы

Жамойдо Артём Игоревич

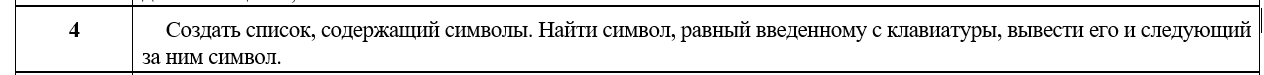
Преподаватель: асс. Андронова М.В.

Минск, 2024

В соответствии со своим вариантом разработать программу с использованием ***односвязного списка*** по данным, представленным в таблице ниже.

Программа должна содержать меню с пунктами: добавление элемента, удаление элемента, поиск элемента, вывод списка в консольное окно, запись списка в файл, считывание списка из файла.

**Вариант 4**



#include <iostream>

#include <fstream>

#include <Windows.h>

using namespace std;

// Структура односвязного списка

struct Node

{

char data;

Node\* next;

};

// Прототипы функций

void insert(Node\*& head, char value);

void remove(Node\*& head, char value);

void searchAndPrintNext(Node\* head, char value);

void printList(Node\* head);

void toFile(Node\* head);

void fromFile(Node\*& head);

void menu();

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Node\* head = nullptr; // Обнуление указателей

int choice;

char value;

menu(); // Вывод меню выбора действий

cout << "Ваш выбор: ";

cin >> choice;

// Цикл обработки выбора действий

while (choice != 0)

{

switch (choice)

{

case 1:

cout << "Введите символ: ";

cin >> value;

insert(head, value); // Вызов функции для вставки элемента

printList(head); // Вывод списка после вставки

break;

case 2:

cout << "Введите символ для удаления: ";

cin >> value;

remove(head, value); // Вызов функции для удаления элемента

printList(head); // Вывод списка после удаления

break;

case 3:

cout << "Введите символ для поиска и вывода следующего: ";

cin >> value;

searchAndPrintNext(head, value); // Вызов функции для поиска и печати следующего элемента

break;

case 4:

printList(head); // Вывод текущего списка

break;

case 5:

toFile(head); // Запись списка в файл

break;

case 6:

fromFile(head); // Чтение списка из файла

break;

default:

cout << "Неправильный выбор" << endl;

menu(); // Вывод меню выбора действий

break;

}

cout << "Ваш выбор: ";

cin >> choice;

}

return 0;

}

// Функция для вставки элемента в начало списка

void insert(Node\*& head, char value)

{

Node\* newNode = new Node;

newNode->data = value;

newNode->next = head;

head = newNode;

}

// Функция для удаления элемента из списка

void remove(Node\*& head, char value)

{

Node\* current = head;

Node\* previous = nullptr; // Обнуление указателей

while (current != nullptr)

{

if (current->data == value)

{

if (previous == nullptr)

{

head = current->next;

}

else

{

previous->next = current->next;

}

delete current;

return;

}

previous = current;

current = current->next;

}

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

}

// Функция для поиска и печати следующего элемента списка

void searchAndPrintNext(Node\* head, char value)

{

Node\* current = head;

while (current != nullptr)

{

if (current->data == value && current->next != nullptr)

{

cout << "Найденный символ: " << current->data << ", Следующий символ: " << current->next->data << endl;

return;

}

current = current->next;

}

cout << "Символ не найден или отсутствует следующий символ" << endl;

}

// Функция для вывода элементов списка

void printList(Node\* head)

{

if (head == nullptr)

{

cout << "Список пуст" << endl;

}

else

{

cout << "Список:" << endl;

Node\* current = head;

while (current != nullptr)

{

cout << "-->" << current->data;

current = current->next;

}

cout << "-->NULL" << endl;

}

}

// Функция для записи списка в файл

void toFile(Node\* head)

{

ofstream file("symbol.txt");

if (file.is\_open())

{

Node\* current = head;

while (current != nullptr)

{

file << current->data << " ";

current = current->next;

}

file.close();

cout << "Список записан в файл symbol.txt" << endl;

}

else

{

cout << "Ошибка открытия файла" << endl;

}

}

// Функция для чтения списка из файла

void fromFile(Node\*& head)

{

ifstream file("symbol.txt");

if (file.is\_open())

{

char value;

while (file >> value)

{

insert(head, value);

}

file.close();

cout << "Список считан из файла symbol.txt" << endl;

}

else

{

cout << "Ошибка открытия файла" << endl;

}

}

// Функция для вывода меню выбора действий

void menu()

{

cout << "Сделайте выбор:" << endl;

cout << "1 - Добавить элемент" << endl;

cout << "2 - Удалить элемент" << endl;

cout << "3 - Поиск и вывод следующего элемента" << endl;

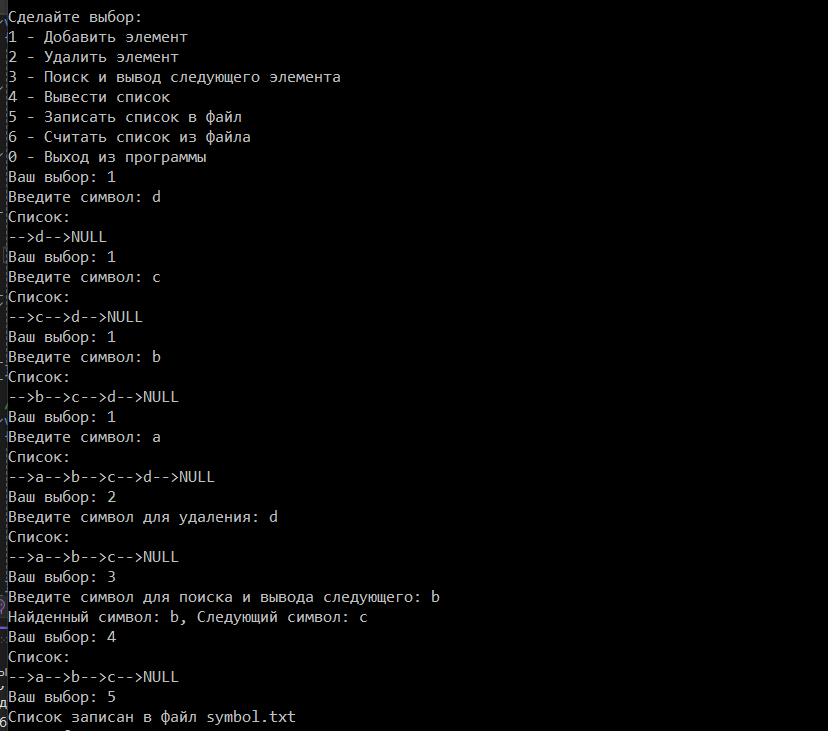
cout << "4 - Вывести список" << endl;

cout << "5 - Записать список в файл" << endl;

cout << "6 - Считать список из файла" << endl;

cout << "0 - Выход из программы" << endl;

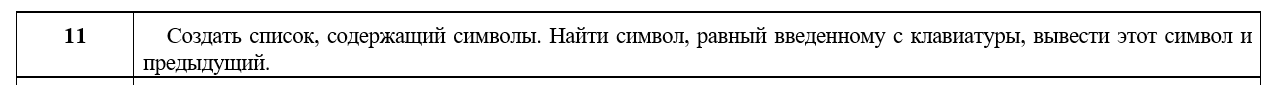
}





Доп. задачи

**Вариант 11**

****

#include <iostream>

#include <fstream>

#include <Windows.h>

using namespace std;

// Структура односвязного списка

struct Node

{

char data;

Node\* next;

};

// Прототипы функций

void insert(Node\*& head, char value);

void remove(Node\*& head, char value);

void searchAndPrintPrev(Node\* head, char value);

void printList(Node\* head);

void toFile(Node\* head);

void fromFile(Node\*& head);

void menu();

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

Node\* head = nullptr; // Обнуление указателей

int choice;

char value;

menu(); // Вывод меню выбора действий

cout << "Ваш выбор: ";

cin >> choice;

// Цикл обработки выбора действий

while (choice != 0)

{

switch (choice)

{

case 1:

cout << "Введите символ: ";

cin >> value;

insert(head, value); // Вызов функции для вставки элемента

printList(head); // Вывод списка после вставки

break;

case 2:

cout << "Введите символ для удаления: ";

cin >> value;

remove(head, value); // Вызов функции для удаления элемента

printList(head); // Вывод списка после удаления

break;

case 3:

cout << "Введите символ для поиска и вывода предыдущего: ";

cin >> value;

searchAndPrintPrev(head, value); // Вызов функции для поиска и печати предыдущего элемента

break;

case 4:

printList(head); // Вывод текущего списка

break;

case 5:

toFile(head); // Запись списка в файл

break;

case 6:

fromFile(head); // Чтение списка из файла

break;

default:

cout << "Неправильный выбор" << endl;

menu(); // Вывод меню выбора действий

break;

}

cout << "Ваш выбор: ";

cin >> choice;

}

return 0;

}

// Функция для вставки элемента в начало списка

void insert(Node\*& head, char value)

{

Node\* newNode = new Node;

newNode->data = value;

newNode->next = head;

head = newNode;

}

// Функция для удаления элемента из списка

void remove(Node\*& head, char value)

{

Node\* current = head;

Node\* previous = nullptr; // Обнуление указателей

while (current != nullptr)

{

if (current->data == value)

{

if (previous == nullptr)

{

head = current->next;

}

else

{

previous->next = current->next;

}

delete current;

return;

}

previous = current;

current = current->next;

}

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

}

// Функция для поиска и печати предыдущего элемента списка

void searchAndPrintPrev(Node\* head, char value)

{

Node\* current = head;

if (current == nullptr || current->data == value) {

cout << "Символ не найден или предыдущего символа нет" << endl;

return;

}

while (current->next != nullptr)

{

if (current->next->data == value)

{

cout << "Найденный символ: " << current->next->data << ", Предыдущий символ: " << current->data << endl;

return;

}

current = current->next;

}

cout << "Символ не найден или предыдущего символа нет" << endl;

}

// Функция для вывода элементов списка

void printList(Node\* head)

{

if (head == nullptr)

{

cout << "Список пуст" << endl;

}

else

{

cout << "Список:" << endl;

Node\* current = head;

while (current != nullptr)

{

cout << "-->" << current->data;

current = current->next;

}

cout << "-->NULL" << endl;

}

}

// Функция для записи списка в файл

void toFile(Node\* head)

{

ofstream file("word.txt");

if (file.is\_open())

{

Node\* current = head;

while (current != nullptr)

{

file << current->data << " ";

current = current->next;

}

file.close();

cout << "Список записан в файл word.txt" << endl;

}

else

{

cout << "Ошибка открытия файла" << endl;

}

}

// Функция для чтения списка из файла

void fromFile(Node\*& head)

{

ifstream file("word.txt");

if (file.is\_open())

{

char value;

while (file >> value)

{

insert(head, value);

}

file.close();

cout << "Список считан из файла word.txt" << endl;

}

else

{

cout << "Ошибка открытия файла" << endl;

}

}

// Функция для вывода меню выбора действий

void menu()

{

cout << "Сделайте выбор:" << endl;

cout << "1 - Добавить элемент" << endl;

cout << "2 - Удалить элемент" << endl;

cout << "3 - Поиск и вывод предыдущего элемента" << endl;

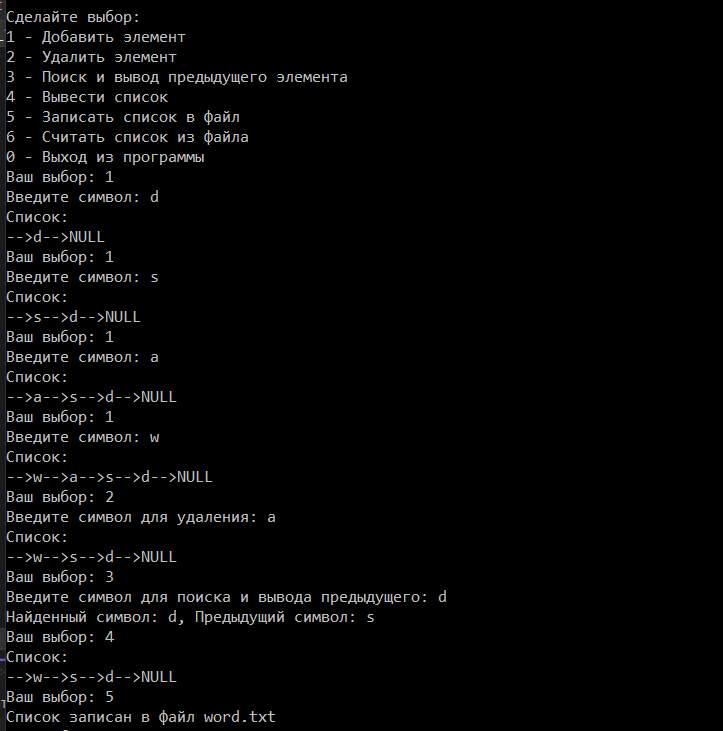
cout << "4 - Вывести список" << endl;

cout << "5 - Записать список в файл" << endl;

cout << "6 - Считать список из файла" << endl;

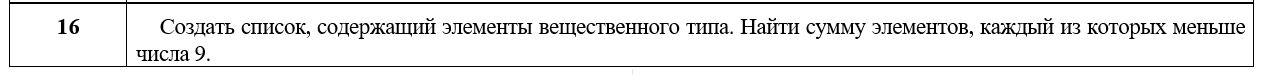
cout << "0 - Выход из программы" << endl;

}

****

****

**Вариант 16**

****

#include <iostream>

#include <fstream>

using namespace std;

// Структура односвязного списка

struct Node

{

float number;

Node\* next;

};

// Прототипы функций

void insert(Node\*& head, float value);

float del(Node\*& head, float value);

int isEmpty(Node\* head);

void printList(Node\* head);

void toFile(Node\* head);

void fromFile(Node\*& head);

void sum9(Node\* head);

void menu();

int main()

{

setlocale(0, "ru");

Node\* head = nullptr; // Обнуление указателей

int choice;

float value;

menu(); // Вывод меню выбора действий

cout << "Ваш выбор: ";

cin >> choice;

// Цикл обработки выбора действий

while (choice != 0)

{

switch (choice)

{

case 1:

cout << "Введите число для добавления: ";

cin >> value;

insert(head, value); // Вызов функции для вставки элемента

printList(head); // Вывод списка после вставки

break;

case 2:

if (!isEmpty(head))

{

cout << "Введите число для удаления: ";

cin >> value;

float deletedValue = del(head, value); // Вызов функции для удаления элемента

if (deletedValue != 0)

{

cout << "Удалено число " << deletedValue << endl;

}

else

{

cout << "Число не найдено" << endl;

}

printList(head); // Вывод списка после удаления

}

else

{

cout << "Список пуст" << endl;

}

break;

case 3:

sum9(head); // Вызов функции для подсчета суммы элементов, меньших 9

break;

case 4:

printList(head); // Вызов функции для вывода списка на экран

break;

case 5:

toFile(head); // Запись списка в файл

break;

case 6:

fromFile(head); // Считывание списка из файла

break;

default:

cout << "Неправильный выбор" << endl;

menu();

break;

}

cout << "Ваш выбор: ";

cin >> choice;

}

return 0;

}

// Функция добавления элемента в начало списка

void insert(Node\*& head, float value)

{

Node\* newNode = new Node;

newNode->number = value;

newNode->next = head;

head = newNode;

}

// Функция удаления элемента из списка

float del(Node\*& head, float value)

{

Node\* previous = nullptr;

Node\* current = head;

if (current != nullptr && current->number == value)

{

head = current->next;

delete current;

return value;

}

while (current != nullptr && current->number != value)

{

previous = current;

current = current->next;

}

if (current != nullptr)

{

previous->next = current->next;

float deletedValue = current->number;

delete current;

return deletedValue;

}

return 0;

}

// Функция проверки списка

int isEmpty(Node\* head)

{

return head == nullptr;

}

// Функция вывода списка на экран

void printList(Node\* head)

{

if (head == nullptr)

{

cout << "Список пуст" << endl;

}

else

{

cout << "Список:" << endl;

Node\* current = head;

while (current != nullptr)

{

cout << "-->" << current->number;

current = current->next;

}

cout << "-->NULL" << endl;

}

}

// Функция записи списка в файл

void toFile(Node\* head)

{

ofstream file("number.txt");

if (file.is\_open())

{

Node\* current = head;

while (current != nullptr)

{

file << current->number << " ";

current = current->next;

}

file.close();

cout << "Список записан в файл number.txt" << endl;

}

else

{

cout << "Ошибка открытия файла" << endl;

}

}

// Функция чтения списка из файла

void fromFile(Node\*& head)

{

ifstream file("number.txt");

if (file.is\_open())

{

float value;

while (file >> value)

{

insert(head, value);

}

file.close();

cout << "Список считан из файла number.txt" << endl;

}

else

{

cout << "Ошибка открытия файла" << endl;

}

}

// Функция нахождения суммы элементов, меньших 9

void sum9(Node\* head)

{

float sum = 0;

Node\* current = head;

while (current != nullptr)

{

if (current->number < 9)

{

sum += current->number;

}

current = current->next;

}

cout << "Сумма элементов, меньших 9: " << sum << endl;

}

// Функция вывода пользовательского меню

void menu()

{

cout << "Сделайте выбор:" << endl;

cout << "1 - Добавить элемент" << endl;

cout << "2 - Удалить элемент" << endl;

cout << "3 - Найти сумму элементов, меньших 9" << endl;

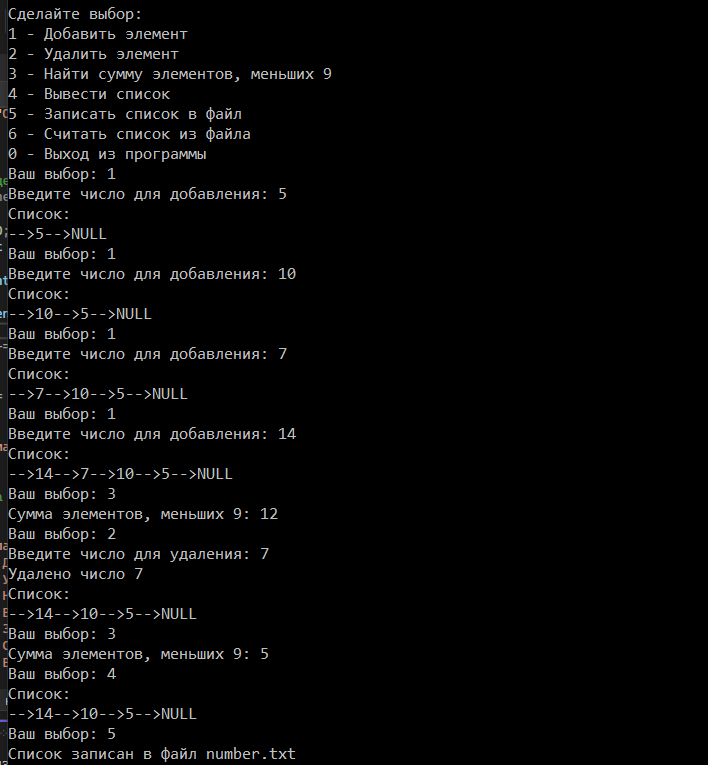
cout << "4 - Вывести список" << endl;

cout << "5 - Записать список в файл" << endl;

cout << "6 - Считать список из файла" << endl;

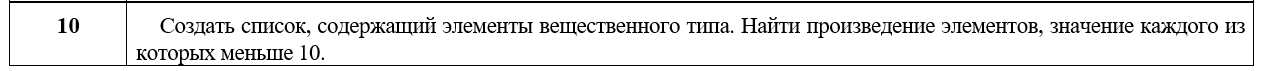
cout << "0 - Выход из программы" << endl;

}

****

****

**Вариант 10**

****

#include <iostream>

#include <fstream>

using namespace std;

// Структура односвязного списка

struct Node

{

float number;

Node\* next;

};

// Прототипы функций

void insert(Node\*& head, float value);

float del(Node\*& head, float value);

int isEmpty(Node\* head);

void printList(Node\* head);

void toFile(Node\* head);

void fromFile(Node\*& head);

void multi10(Node\* head);

void menu();

int main()

{

setlocale(0, "ru");

Node\* head = nullptr; // Обнуление указателей

int choice;

float value;

menu(); // Вывод меню выбора действий

cout << "Ваш выбор: ";

cin >> choice;

// Цикл обработки выбора действий

while (choice != 0)

{

switch (choice)

{

case 1:

cout << "Введите число для добавления: ";

cin >> value;

insert(head, value); // Вызов функции для вставки элемента

printList(head); // Вывод списка после вставки

break;

case 2:

if (!isEmpty(head))

{

cout << "Введите число для удаления: ";

cin >> value;

float deletedValue = del(head, value); // Вызов функции для удаления элемента

if (deletedValue != 0)

{

cout << "Удалено число " << deletedValue << endl;

}

else

{

cout << "Число не найдено" << endl;

}

printList(head); // Вывод списка после удаления

}

else

{

cout << "Список пуст" << endl;

}

break;

case 3:

multi10(head); // Вызов функции для подсчета произведения элементов, меньших 10

break;

case 4:

printList(head); // Вызов функции для вывода списка на экран

break;

case 5:

toFile(head); // Запись списка в файл

break;

case 6:

fromFile(head); // Считывание списка из файла

break;

default:

cout << "Неправильный выбор" << endl;

menu();

break;

}

cout << "Ваш выбор: ";

cin >> choice;

}

return 0;

}

// Функция добавления элемента в начало списка

void insert(Node\*& head, float value)

{

Node\* newNode = new Node;

newNode->number = value;

newNode->next = head;

head = newNode;

}

// Функция удаления элемента из списка

float del(Node\*& head, float value)

{

Node\* previous = nullptr;

Node\* current = head;

if (current != nullptr && current->number == value)

{

head = current->next;

delete current;

return value;

}

while (current != nullptr && current->number != value)

{

previous = current;

current = current->next;

}

if (current != nullptr)

{

previous->next = current->next;

float deletedValue = current->number;

delete current;

return deletedValue;

}

return 0;

}

// Функция проверки списка

int isEmpty(Node\* head)

{

return head == nullptr;

}

// Функция вывода списка на экран

void printList(Node\* head)

{

if (head == nullptr)

{

cout << "Список пуст" << endl;

}

else

{

cout << "Список:" << endl;

Node\* current = head;

while (current != nullptr)

{

cout << "-->" << current->number;

current = current->next;

}

cout << "-->NULL" << endl;

}

}

// Функция записи списка в файл

void toFile(Node\* head)

{

ofstream file("count.txt");

if (file.is\_open())

{

Node\* current = head;

while (current != nullptr)

{

file << current->number << " ";

current = current->next;

}

file.close();

cout << "Список записан в файл count.txt" << endl;

}

else

{

cout << "Ошибка открытия файла" << endl;

}

}

// Функция чтения списка из файла

void fromFile(Node\*& head)

{

ifstream file("count.txt");

if (file.is\_open())

{

float value;

while (file >> value)

{

insert(head, value);

}

file.close();

cout << "Список считан из файла count.txt" << endl;

}

else

{

cout << "Ошибка открытия файла" << endl;

}

}

// Функция нахождения произведения элементов, меньших 10

void multi10(Node\* head)

{

float multi = 1;

Node\* current = head;

while (current != nullptr)

{

if (current->number < 10)

{

multi \*= current->number;

}

current = current->next;

}

cout << "Произведение элементов, меньших 10: " << multi << endl;

}

// Функция вывода пользовательского меню

void menu()

{

cout << "Сделайте выбор:" << endl;

cout << "1 - Добавить элемент" << endl;

cout << "2 - Удалить элемент" << endl;

cout << "3 - Найти произведение элементов, меньших 10" << endl;

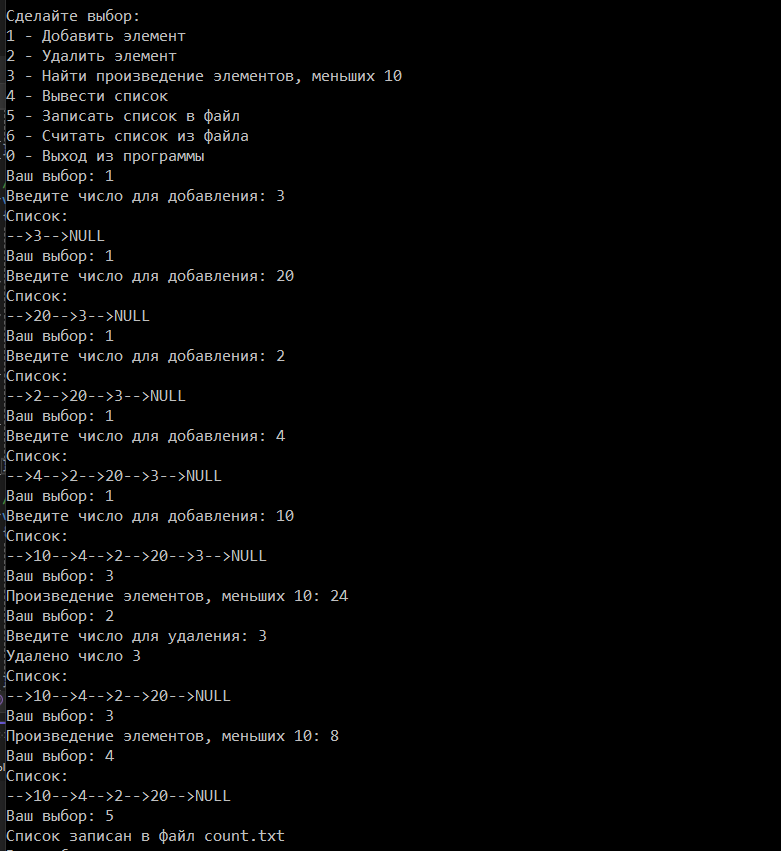
cout << "4 - Вывести список" << endl;

cout << "5 - Записать список в файл" << endl;

cout << "6 - Считать список из файла" << endl;

cout << "0 - Выход из программы" << endl;

}

****

****