МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«МИРЭА – Российский технологический университет»

РТУ МИРЭА

Отчет по выполнению практического задания №1.5

Тема:

Однонаправленный динамический список

Дисциплина: «Структуры и алгоритмы обработки данных»

Выполнил студент: Враженко Д.О.

Группа: ИКБО-10-23

Вариант: 11

Москва – 2024

ЦЕЛЬ РАБОТЫ

Получить знания и практические навыки управления динамическим однонаправленным списком

ХОД РАБОТЫ

1. Постановка задачи

Тип информационной части узла: int.

Дан линейный однонаправленный список L, информационная часть которого содержит однозначные и двузначные числа.

1. Разработать функцию, которая создает массив A из 10 указателей на элемент списка и включает в список элемента массива с индексом i, числа списка L, которые начинаются с цифры равной i. Включение в конец списка. Однозначные числа включаются в список массива с индексом 0.

2. Разработать функцию, которая удаляет список L.

3. Разработать функцию, которая создает список L, включая в него списки массива A последовательно от списка с индексом 0 до списка с индексом 9.

2. Список операций над списком:

2.1 Структура узла:

У каждого узла (Node) будет информационное поле val со значением типа int, информационное поле next типа Node*, а также конструктор экземпляра узла, принимающий значение _val типа int.

2.2 Операции над списками:

1) Функция, которая создает массив A из 10 указателей на элемент списка и включает в список элемента массива с индексом i, числа списка L, которые начинаются с цифры равной i. Включение в конец списка. Однозначные числа включаются в список массива с индексом 0.

На рис. 1 представлена блок-схема этого алгоритма.

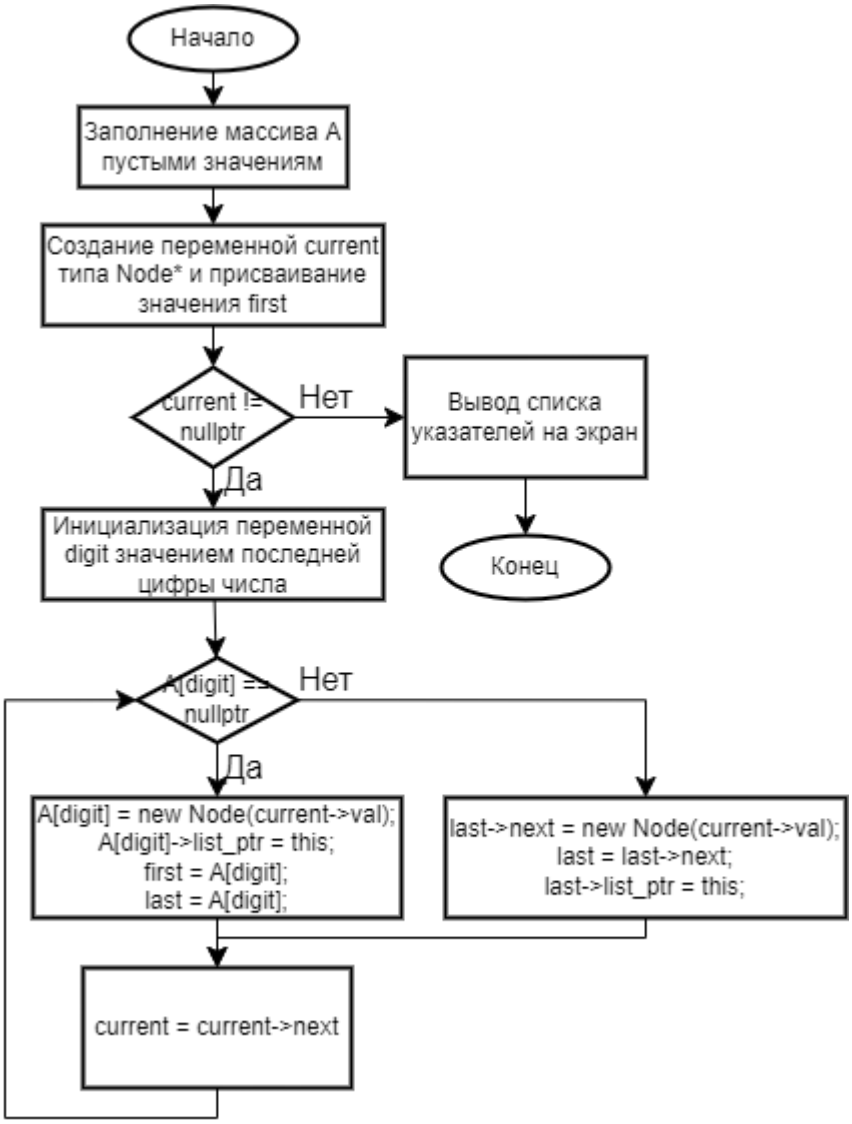


Рисунок 1 - Блок-схема алгоритма create1

2) Функция, которая удаляет список L.

На рис. 2 представлена блок-схема этого алгоритма.

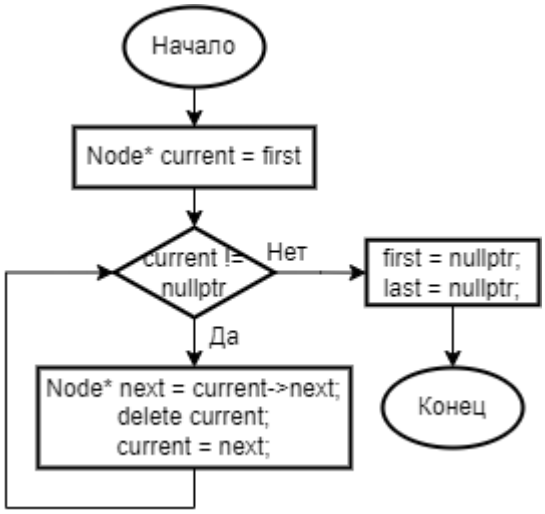


Рисунок 2 - Блок-схема алгоритма del

3) Функция, которая создает список L, включая в него списки массива A последовательно от списка с индексом 0 до списка с индексом 9.

На рис. 3 представлена блок-схема этого алгоритма.

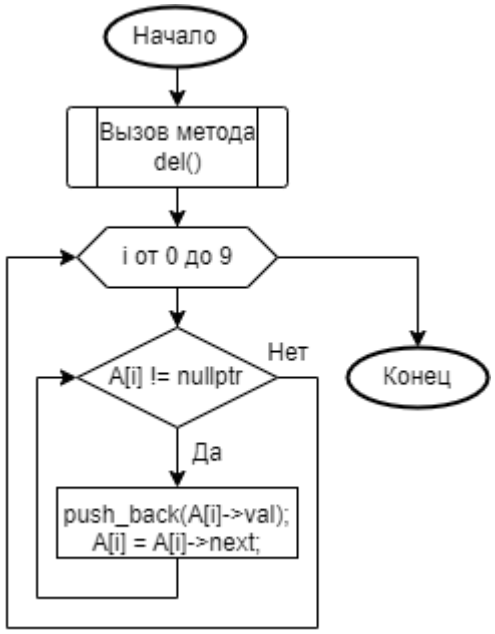
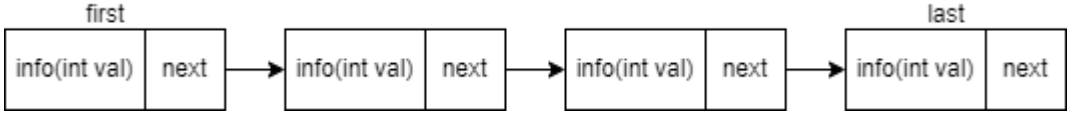


Рисунок 3 - Блок-схема алгоритма create2

2.3 Структура данных:

Ниже представлен схематичный рисунок используемой структуры.



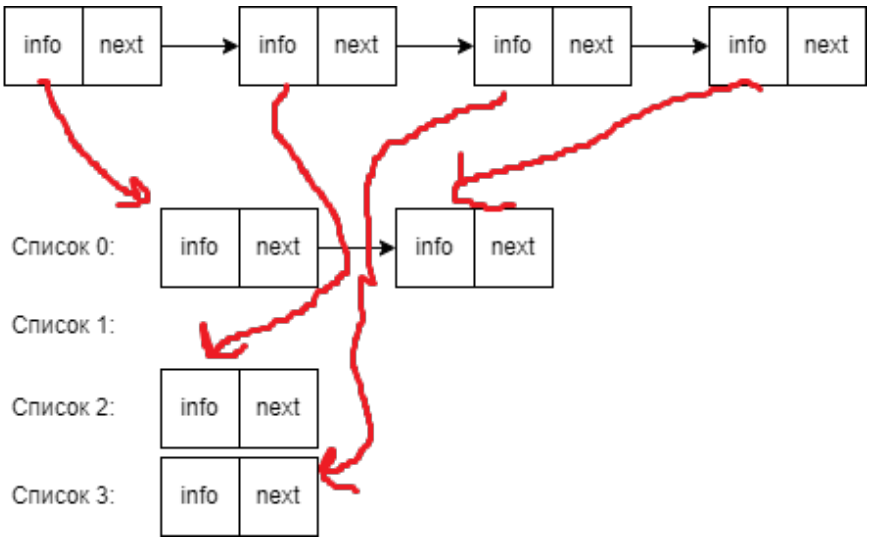


Рисунок 4 - Структура данных для метода create1()

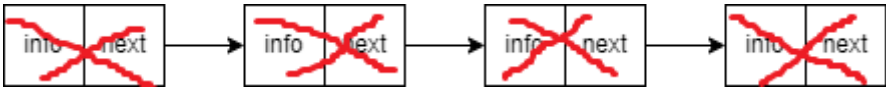


Рисунок 5 - Структура данных для метода del()

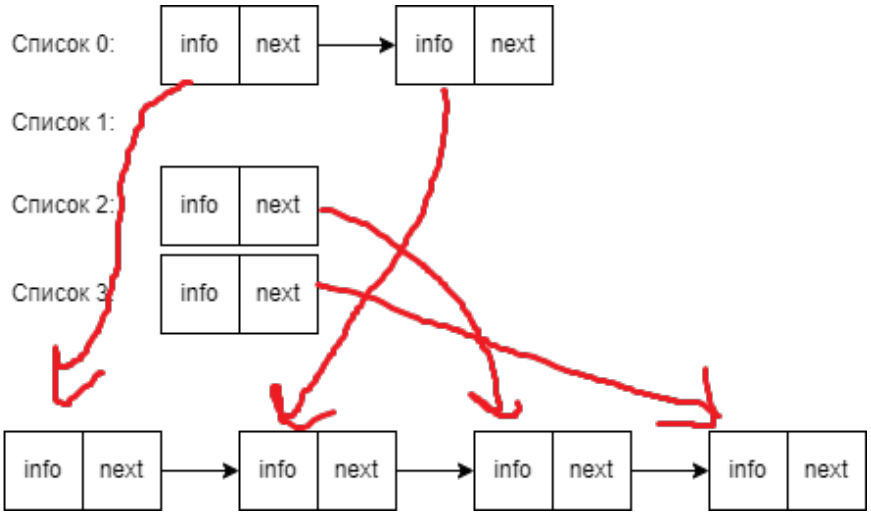


Рисунок 6 - Структура данных для метода create2()

2.4 Алгоритм выполнения операции:

1) Присваивание всем элементам списка A значений nullptr. Присваивание переменной current значение first. Пока current не равно nullptr: инициализация переменной digit значением цифры единиц числа. Если элемент списка A с индексом digit равен nullptr, то элементу списка A с индексом digit присваивается указатель на новый однонаправленный динамический список, списку элемента списка A с индексом digit присваивается указатель на данный элемент, первому элементу присваивается значение элемента списка A индексом digit, последнему

элементу присваивается значение элемента списка A индексом digit. Иначе последнему элементу создаем новый последний элемент, который указывает на новый однонаправленный динамический список, бывшему последнему элементу списка A присваивается значения нового последнего списка A, последнему списку списка A присваивается указатель на данный элемент. Присваивание переменной current следующего значения. Вывод списка указателей на экран.

2) Инициализируем переменную current типа Node* на первый элемент списка L. Пока current не равно nullptr, то инициализируем переменную next типа Node* на следующий элемент, удаляем предыдущий элемент, и current присваиваем значения следующего элемента, то есть next. first равен nullptr и last равен nullptr.

3) Вызов метода del(). Для i от 0 до 9: пока элемент списка A с индексом i не равен nullptr, то вызов метода push_back() с параметром значения элемента A с индексом i, присваивание этому элементу следующего значения.

2.5 Таблица тестов:

1)

Исходные данные	Выходные данные
12 53 34 83 62 71 75 6 9	6 9 12 34 53 62 71 75 83
1 10 20 41 52 63 65 72	1 10 20

	41 52 63 65 72
12 34 71 52 27 43 68	12 27 34 43 52 68 71

2)

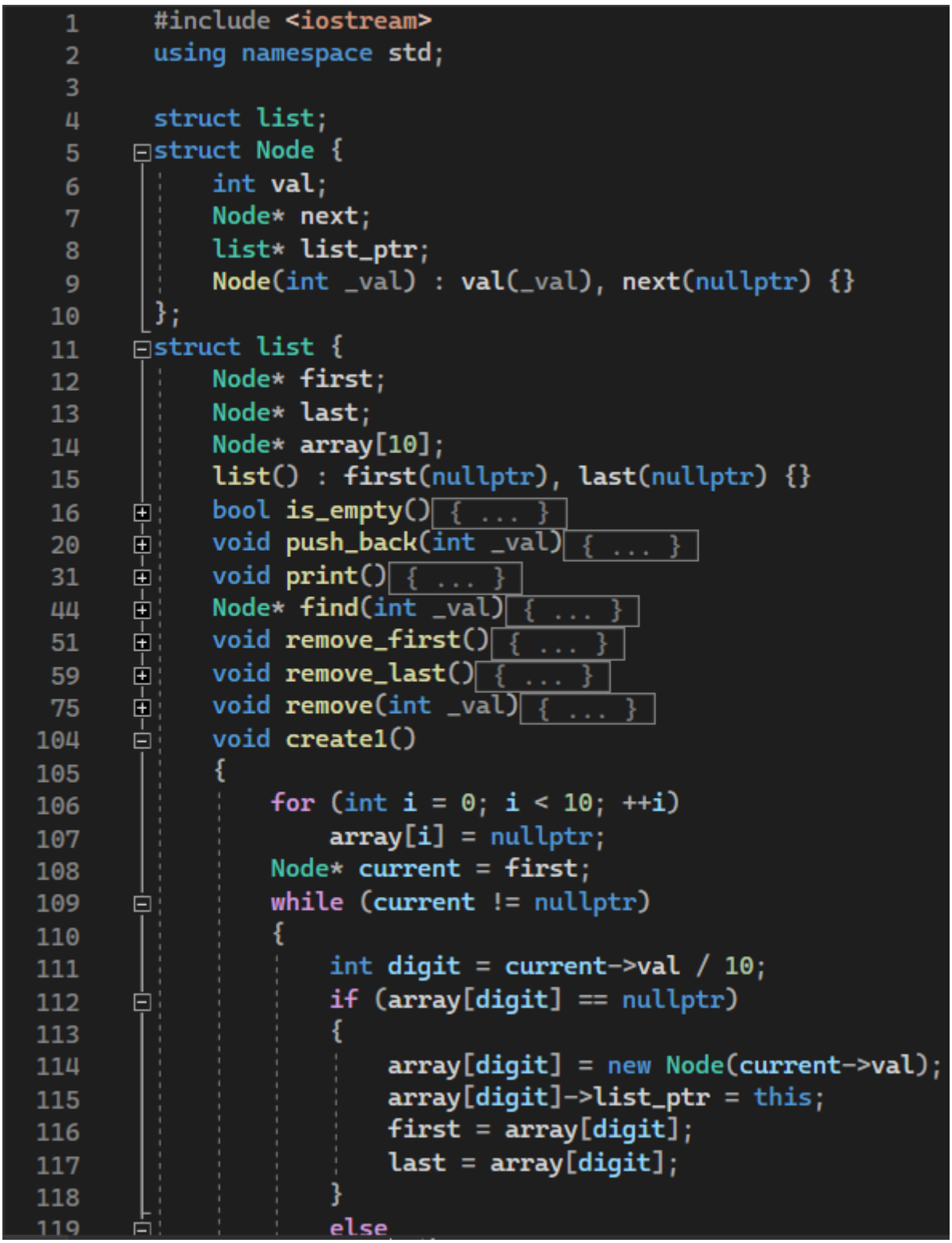
Исходные данные	Выходные данные
12 53 34 83 62 71 75 6 9	
1 10 20 41 52 63 65 72	
12 34 71 52 27 43 68	

3)

Исходные данные	Выходные данные
12 53 34 83 62 71 75 6 9	6 9 12 34 53 62 71 75 83
1 10 20 41 52 63 65 72	1 10 20 41 52 63 65 72
12 34 71 52 27 43 68	12 27 34 43 52 68 71

3. Код программы:

На рис. 7-10 представлен код программы.



```
#include <iostream>
using namespace std;

struct list;
struct Node {
    int val;
    Node* next;
    list* list_ptr;
    Node(int _val) : val(_val), next(nullptr) {}
};
struct list {
    Node* first;
    Node* last;
    Node* array[10];
    list() : first(nullptr), last(nullptr) {}
    bool is_empty() { ... }
    void push_back(int _val) { ... }
    void print() { ... }
    Node* find(int _val) { ... }
    void remove_first() { ... }
    void remove_last() { ... }
    void remove(int _val) { ... }
    void create1()
    {
        for (int i = 0; i < 10; ++i)
            array[i] = nullptr;
        Node* current = first;
        while (current != nullptr)
        {
            int digit = current->val / 10;
            if (array[digit] == nullptr)
            {
                array[digit] = new Node(current->val);
                array[digit]->list_ptr = this;
                first = array[digit];
                last = array[digit];
            }
            else
```

Рисунок 7 - 1 часть программы

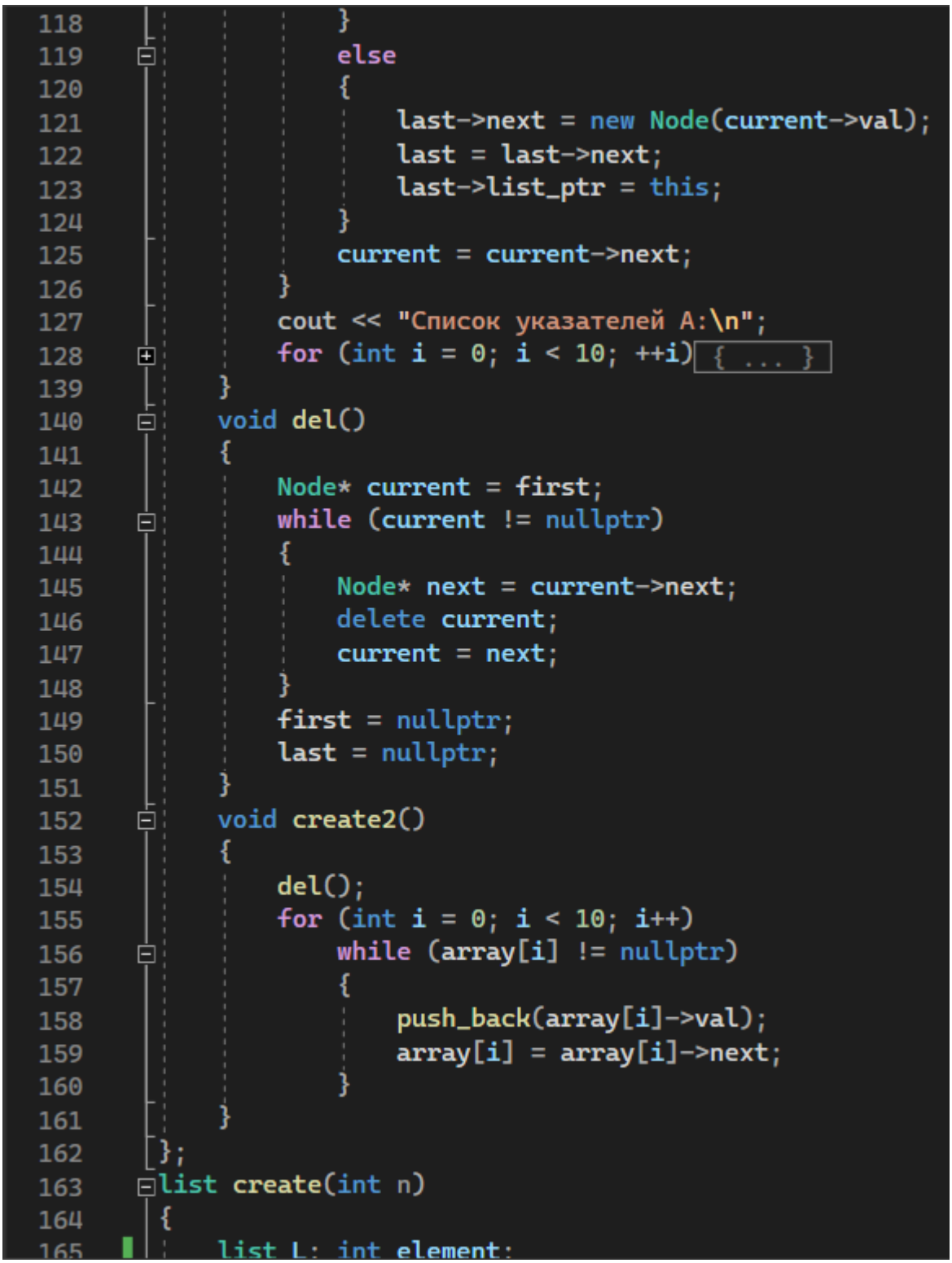
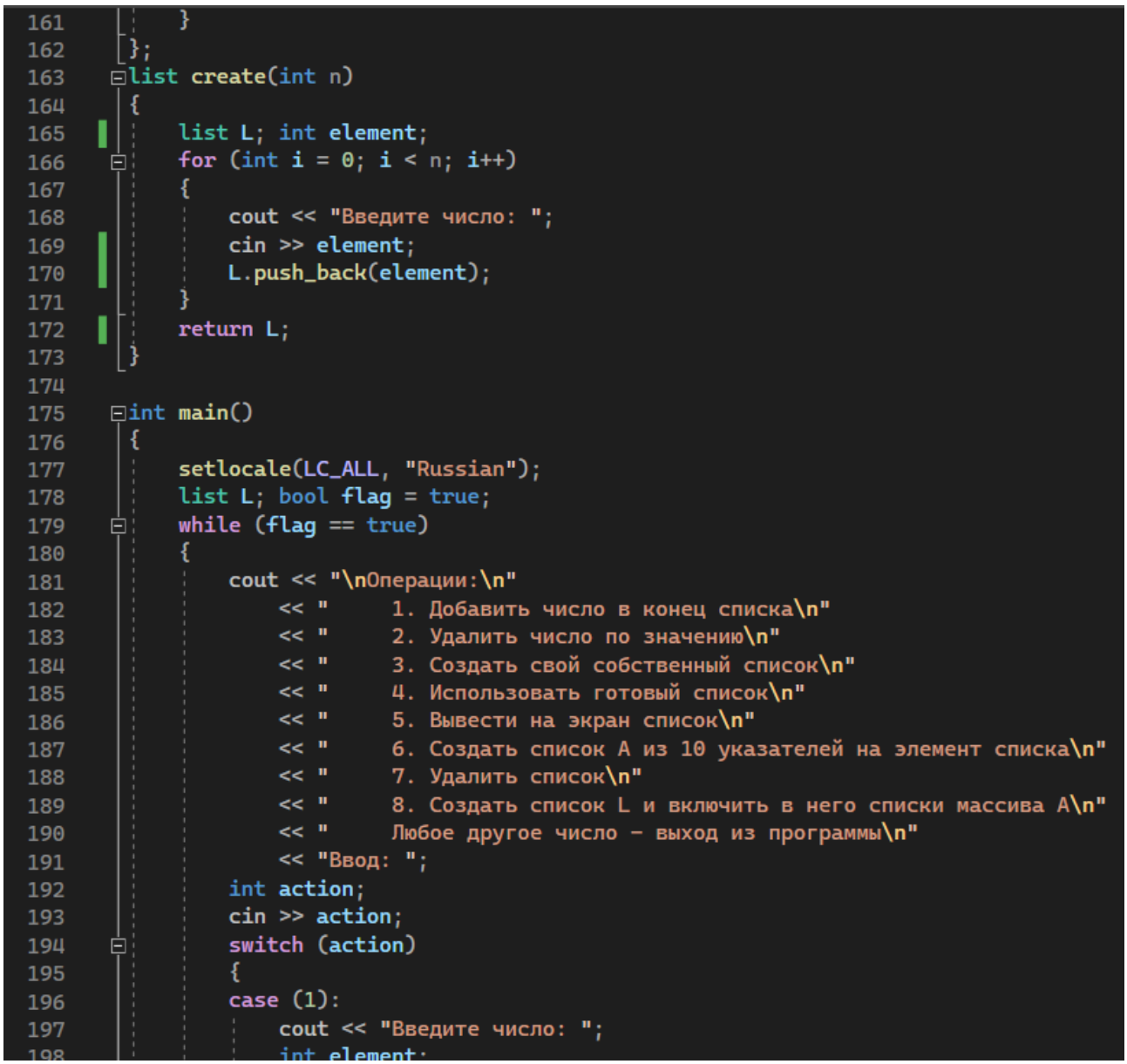


Рисунок 8 - 2 часть программы



```
161         }
162     };
163     list create(int n)
164     {
165         list L; int element;
166         for (int i = 0; i < n; i++)
167         {
168             cout << "Введите число: ";
169             cin >> element;
170             L.push_back(element);
171         }
172         return L;
173     }
174
175     int main()
176     {
177         setlocale(LC_ALL, "Russian");
178         list L; bool flag = true;
179         while (flag == true)
180         {
181             cout << "\nОперации:\n"
182                 << "     1. Добавить число в конец списка\n"
183                 << "     2. Удалить число по значению\n"
184                 << "     3. Создать свой собственный список\n"
185                 << "     4. Использовать готовый список\n"
186                 << "     5. Вывести на экран список\n"
187                 << "     6. Создать список A из 10 указателей на элемент списка\n"
188                 << "     7. Удалить список\n"
189                 << "     8. Создать список L и включить в него списки массива A\n"
190                 << "     Любое другое число - выход из программы\n"
191                 << "Ввод: ";
192             int action;
193             cin >> action;
194             switch (action)
195             {
196             case (1):
197                 cout << "Введите число: ";
198                 int element;
```

Рисунок 9 - 3 часть программы

```
switch (action)
{
case (1):
    cout << "Введите число: ";
    int element;
    cin >> element;
    L.push_back(element);
    break;
case(2):
    cout << "Введите число: ";
    cin >> element;
    L.remove(element);
    create;
    break;
case(3):
    cout << "Введите размер списка: ";
    cin >> element;
    L = create(element);
    break;
case(4):
    L = list();
    L.push_back(12); L.push_back(53); L.push_back(34); L.push_back(83); L.push_back(62); L.push_back(71); L.push_back(75); L.push_back(6); L.push_back(9);
    break;
case(5):
    L.print();
    break;
case(6):
    L.create1();
    break;
case(7):
    L.del();
    break;
case(8):
    L.create2();
    break;
default:
    flag = false;
```

Рисунок 10 - 4 часть программы

4. Результат тестирования программы:

На рис. 11-13 представлены примеры работы алгоритмов, добавленных исходя из задания индивидуального варианта.

```
Операции:
     1. Добавить число в конец списка
     2. Удалить число по значению
     3. Создать свой собственный список
     4. Использовать готовый список
     5. Вывести на экран список
     6. Создать список A из 10 указателей на элемент списка
     7. Удалить список
     8. Создать список L и включить в него списки массива A
     Любое другое число - выход из программы
Ввод: 6
Список указателей A:
Список 0: 6 9
Список 1: 12
Список 2:
Список 3: 34
Список 4:
Список 5: 53
Список 6: 62
Список 7: 71 75
Список 8: 83
Список 9:
```

Рисунок 11 - Алгоритм create1

```
Операции:
     1. Добавить число в конец списка
     2. Удалить число по значению
     3. Создать свой собственный список
     4. Использовать готовый список
     5. Вывести на экран список
     6. Создать список A из 10 указателей на элемент списка
     7. Удалить список
     8. Создать список L и включить в него списки массива A
     Любое другое число - выход из программы
Ввод: 7
```

Рисунок 12 - Алгоритм del

```
Операции:
     1. Добавить число в конец списка
     2. Удалить число по значению
     3. Создать свой собственный список
     4. Использовать готовый список
     5. Вывести на экран список
     6. Создать список A из 10 указателей на элемент списка
     7. Удалить список
     8. Создать список L и включить в него списки массива A
     Любое другое число - выход из программы
Ввод: 8
```

Рисунок 13 - Алгоритм create2

5. Вывод:

Изучена такая структура как односвязный список, а также работа с ней. Реализовано несколько функций для работы с этой структурой. Проверена корректность их работы.