МИНИСТЕРСТВО науки и высшего ОБРАЗОВАНИЯ РОссИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(национальный исследовательский университет)»

Институт №3 «Системы управления, информатика и электроэнергетика»

Кафедра № 304 «Вычислительные машины, системы и сети»

Программирование

Отчет по лабораторной работе № 4

Линейные списки

Выполнили студенты группы М3О-209Б-22

Глазов В.В., Абдурахманов М.Р.

Проверила доцент, к.т.н., Дмитриева Е.А.

Москва 2023 г.

Оглавление

[Задание 3](#_Toc148213218)

[Структурные схемы алгоритмов 4](#_Toc148213219)

[Функция main 4](#_Toc148213220)

[Функция random\_INT 6](#_Toc148213221)

[Код программы 7](#_Toc148213222)

[Тестирование программы 15](#_Toc148213223)

[Тестирование некорректных режимов 15](#_Toc148213224)

[Тестирование корректных режимов 16](#_Toc148213225)

[Графики зависимостей времени от размерностей 23](#_Toc148213226)

[Вывод 29](#_Toc148213227)

# Задание

Реализовать заданный согласно варианту задания линейный список, состоящий из 20 элементов заданного типа. Интерфейс должен включать **для всех вариантов** следующие операции**:**

- создание списка;

- вывод на экран и/или в файл значений элементов списка с их индексами (номерами);

- удаление списка,

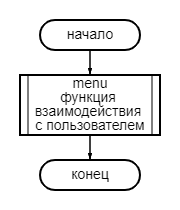
а также **некоторые из дополнительных операций**:

1. Поиск в списке наличия элемента с заданным значением с получением его номера в списке (запрещено).
2. Включение нового элемента в конец списка.
3. Удаление элемента из позиции списка с заданным в программе номером.

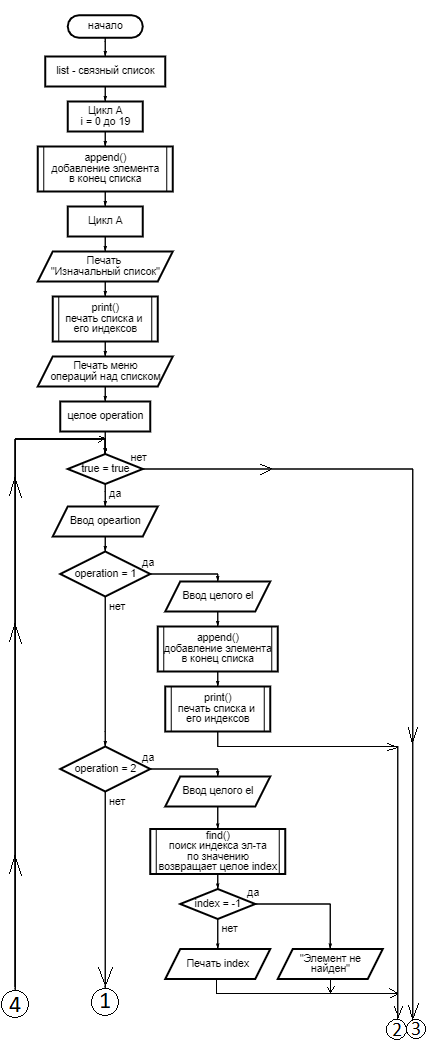
После выполнения операций включения или удаления вывести содержимое списка. Выполнение операций организовать с помощью меню.

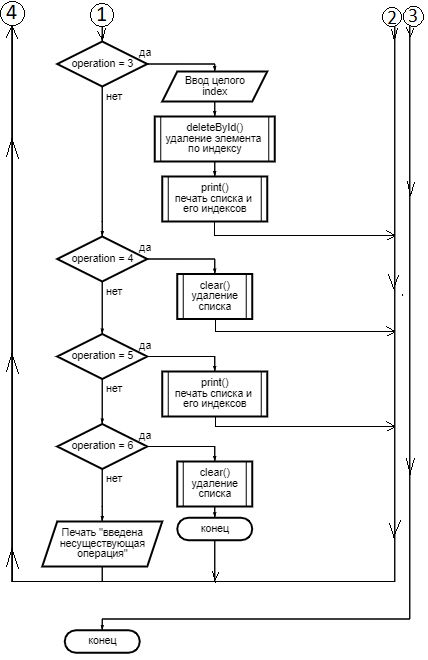
# Структурные схемы алгоритмов

## **Main**

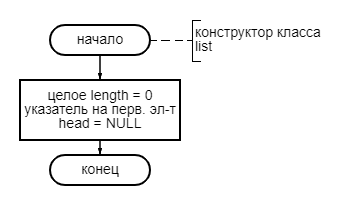


## **Menu**

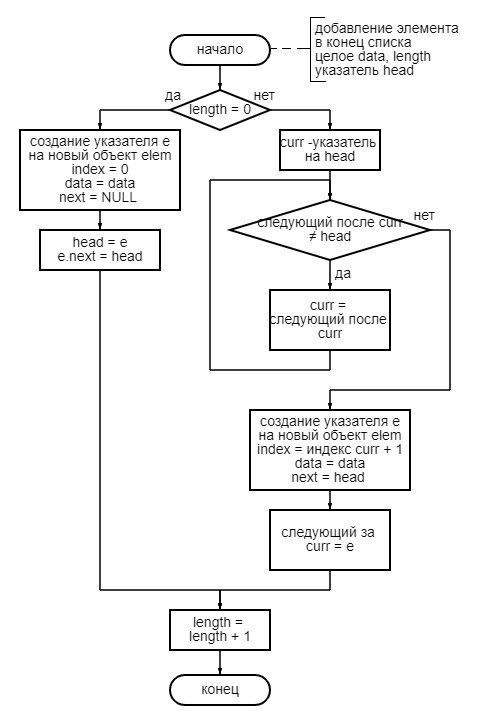




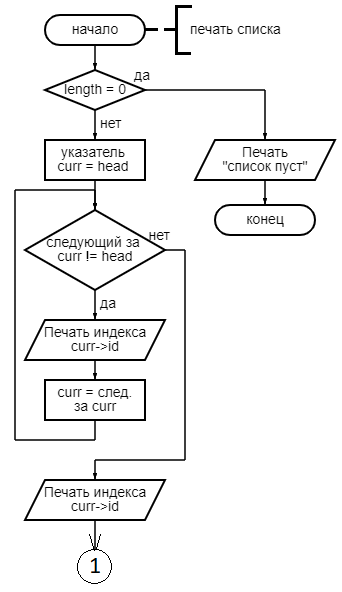
## **List**

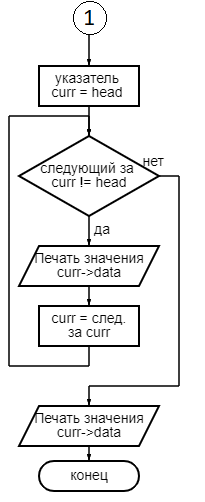


## **Append**

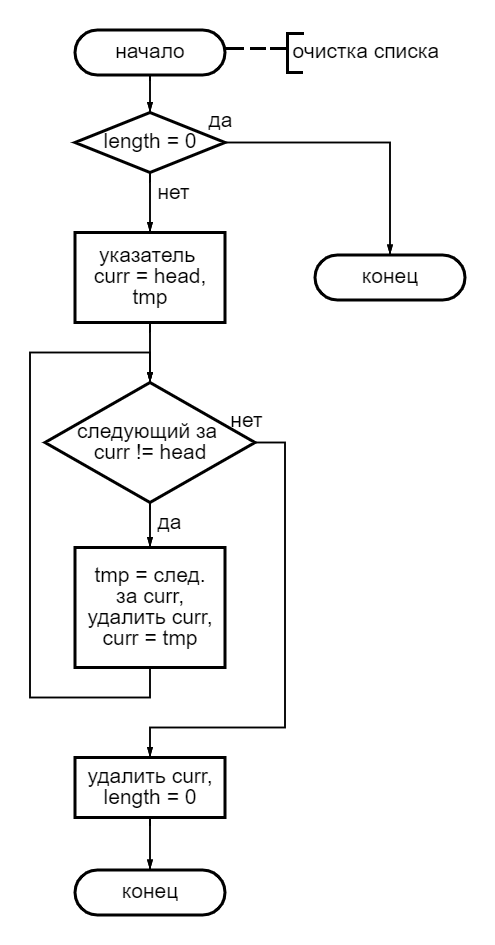


## **Print**

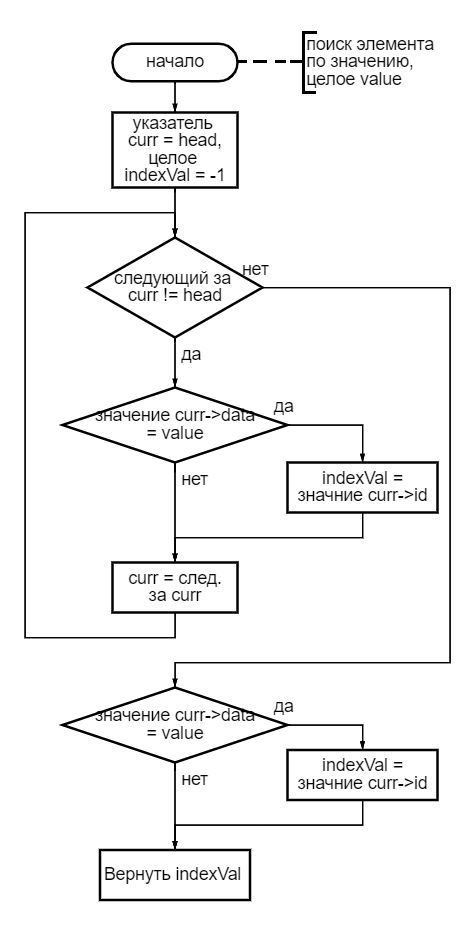




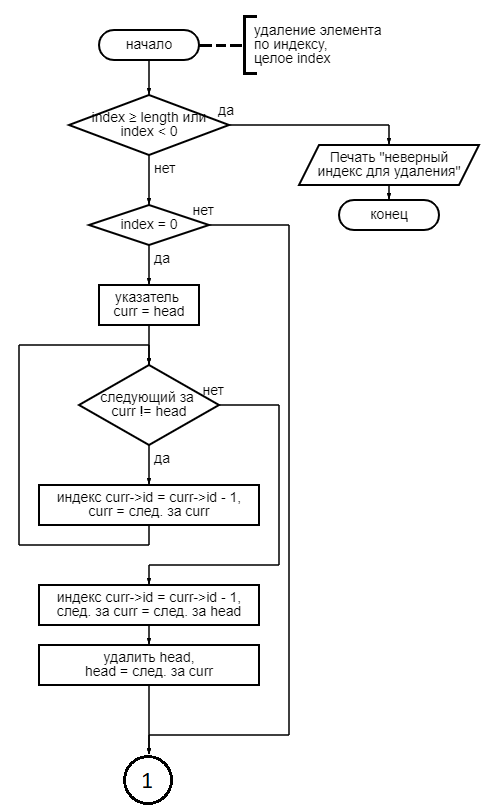
## **Clear**

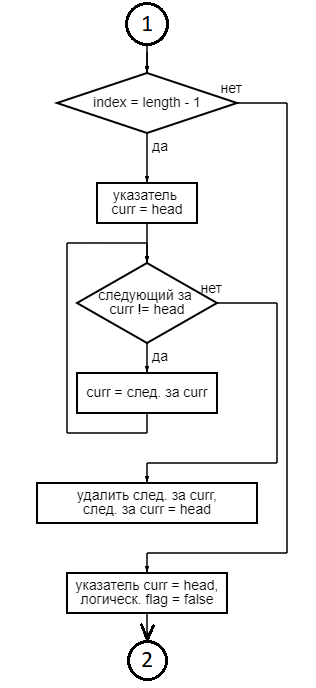


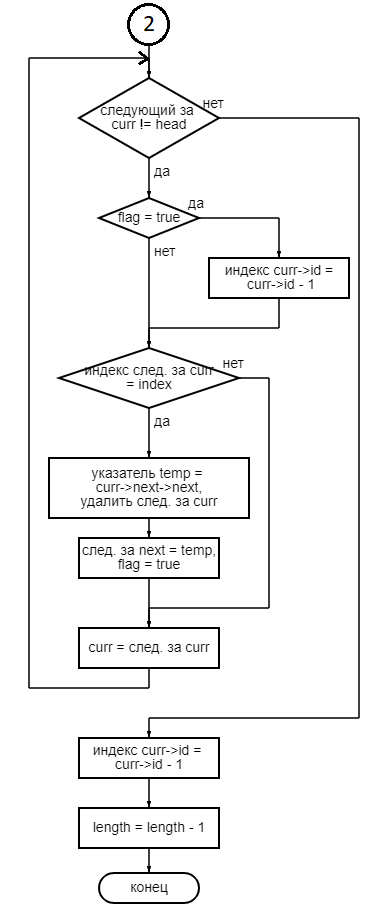
## **Find**



## **DeleteById**







# Код программы

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Программирование \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Project type :Win64 Console Application \*

\*Project name :lab4\_linked\_lists.sln \*

\*File name : lab4\_linked\_lists.cpp \*

\*Language :CPP, MSVS 2022 \*

\*Programmers : Абдурахманов М.Р., Глазов В.В., М3О-209Б-22 \*

\*Modified By : \*

\*Comment : \*

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <iostream>

#include <iomanip>

using namespace std;

//элемент списка

struct elem{

    int id;     //индекс

    int data;   //значение

    elem\* next; //следующий элемент

};

//односвязный кольцевой список

class list{

    public:

        int length;             //длина

        elem\* head;             //первый элемент

        list();                 //контруктор

        void append(int);       //добавление в конец

        void print();           //печать всего списка

        void clear();           //удаление всех элементов

        int find(int);          //поиск индекса элемента по значению

        void deleteById(int);   //удаление элемента по индексу

};

//интерфейс взаимодействия с пользователем

void menu(){

    list list;

    for (int i = 0; i < 20; i++){

        list.append(20-i);

    }

    cout << "Изначальный список\n";

    list.print();

    cout << "\tОперации со списком\n";

    cout << "\tДобавить элемент в конец - 1\n";

    cout << "\tНайти индекс элемента - 2\n";

    cout << "\tУдалить элемент по индексу - 3\n";

    cout << "\tОчистить список - 4\n";

    cout << "\tНапечатать список - 5\n";

    cout << "\tВыход - 6\n";

    int operation;

    while (true){

        cin >> operation;

        switch (operation)

        {

            case 1:{

                cout << "Введите элемент: ";

                int el;

                cin >> el;

                cout << endl;

                list.append(el);

                list.print();

                break;

            }

            case 2:{

                cout << "Введите элемент: ";

                int el;

                cin >> el;

                int index = list.find(el);

                if (index == -1){

                    cout << "Элемент не найден\n";

                } else{

                    cout << "Индекс найденного элемента: " << index << endl;

                }

                break;

            }

            case 3:{

                cout << "Введите индекс элемента: ";

                int index;

                cin >> index;

                list.deleteById(index);

                list.print();

                break;

            }

            case 4:{

                list.clear();

                cout << "Список очищен\n";

                break;

            }

            case 5:{

                list.print();

                break;

            }

            case 6:{

                list.clear();

                return;

            }

            default:{

                cout << "введена несуществующая операция\n";

                break;

            }

        }

    }

}

//главная функция

int main(){

    menu();

}

//конструктор

list::list(){

    length = 0;

    head = NULL;

}

//добавление в конец

void list::append(int data){

    if (length == 0){

        //создать элемент, закольцованный сам на себя

        elem\* e = new elem{0, data, NULL};

        head = e;

        e->next = head;

    } else{

        //добавить элемент после последнего

        //закольцевать на первый элемент

        elem\* curr = head;

        while(curr->next != head){

            curr = curr->next;

        }

        elem\* e = new elem{curr->id+1, data, head};

        curr->next = e;

    }

    length++;   //прибавить длину

}

//печать всего списка

void list::print(){

    if (length == 0){

        cerr << "list is empty\n";

        exit(1);

    }

    cout << "index: ";

    elem\* curr = head;

    while(curr->next != head){

        cout << setw(5) << curr->id << " ";

        curr = curr->next;

    }

    cout << setw(5) << curr->id << endl;

    cout << "list:  ";

    curr = head;

    while(curr->next != head){

        cout << setw(5) << curr->data << " ";

        curr = curr->next;

    }

    cout << setw(5) << curr->data << endl << endl;

}

//удаление всех элементов

void list::clear(){

    if (length == 0){

        return;

    }

    elem\* curr = head;

    elem\* tmp;

    while(curr->next != head){

        tmp = curr->next;

        delete curr;

        curr = tmp;

    }

    delete curr;

    length = 0;

}

//поиск индекса элемента по значению

int list::find(int value){

    elem\* curr = head;

    int indexVal = -1;

    while(curr->next != head){

        if (curr->data == value){

            indexVal = curr->id;

        }

        curr = curr->next;

    }

    if (curr->data == value){

        indexVal = curr->id;

    }

    return indexVal;

}

//удаление элемента по индексу

void list::deleteById(int index){

    if (index >= length || index < 0){

        cerr << "wrong index to delete\n";

        exit(1);

    }

    if (index == 0){

        elem\* curr = head->next;

        while(curr->next != head){

            curr->id--;             //понизить все индесы на 1

            curr = curr->next;

        }

        curr->id--;

        curr->next = head->next;    //закольцевать на второй элемент

        delete head;                //удалить первый элемент

        head = curr->next;          //второй теперь первый

    } else if (index == length - 1){

        elem\* curr = head;

        while(curr->next->next != head){

            curr = curr->next;

        }

        delete curr->next;          //удалить последний

        curr->next = head;          //закольцевать предпоследний на первый

    } else{

        elem\* curr = head;

        bool flag = false;          //отслеживаем удалили или нет

        while(curr->next != head){

            if (flag){              //если уже удалили

                curr->id--;         //понижаем все индексы на 1

            }

            if (curr->next->id == index){

                elem\* temp = curr->next->next;

                delete curr->next;

                curr->next = temp;

                flag = true;

            }

            curr = curr->next;

        }

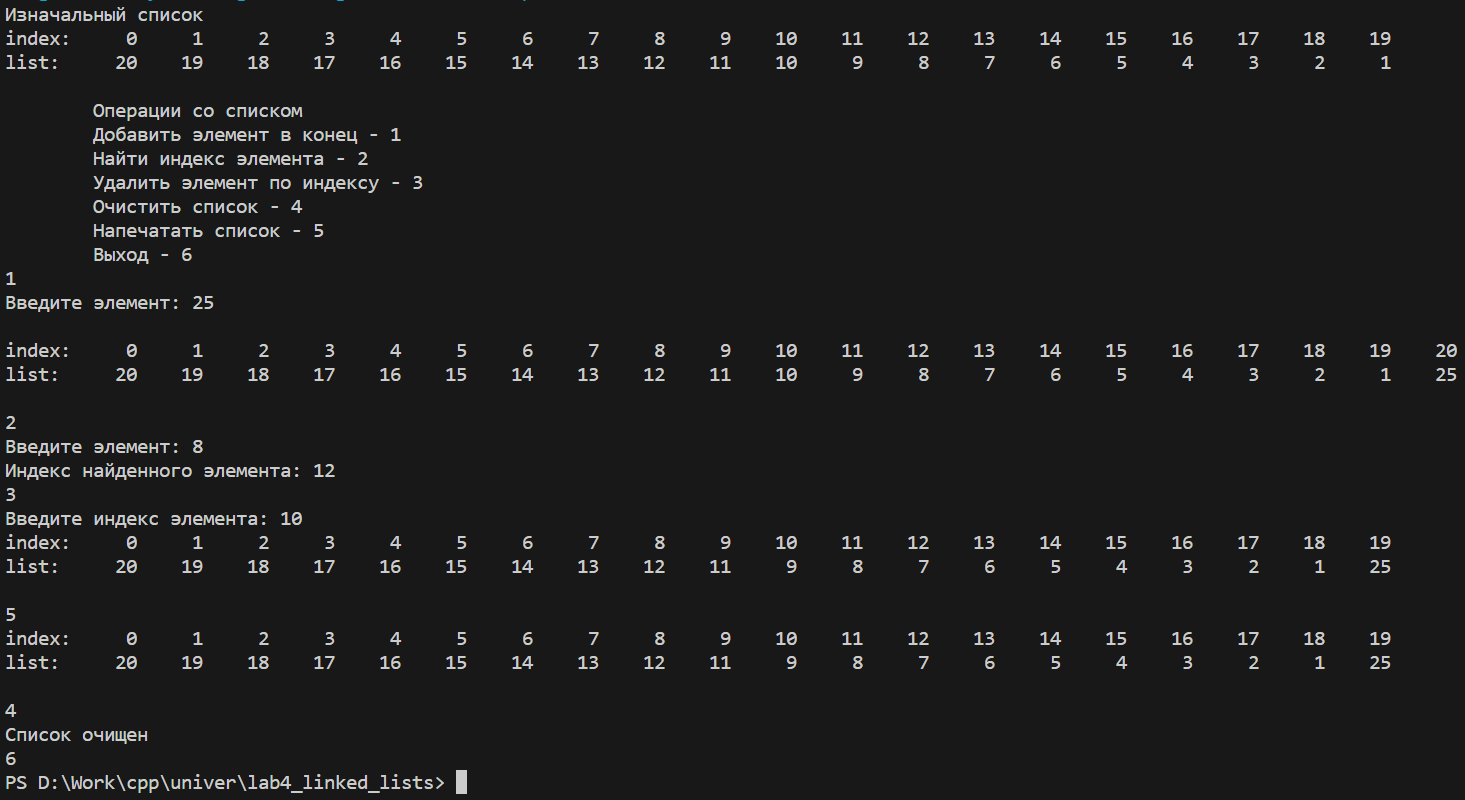
        curr->id--;

    }

    length--;

}

# Результат работы программы



**Вывод**

В данной лабораторной работе были рассмотрены и описаны односвязные кольцевые списки и их методы.

Разработка программы успешно завершена так как:

1) Полученные результаты совпадают с ожидаемыми;