Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №6**

**«РЕАЛИЗАЦИЯ ЭЛЕМЕНТАРНЫХ СТРУКТУР ДАННЫХ НА ОСНОВЕ СТАТИЧЕСКОЙ ПАМЯТИ»**

**ПО МДК 05.02 РАЗРАБОТКА КОДА ИНФОРМАЦИОННЫХ СИСТЕМ**

Выполнил: студент учебной группы

ИСПк-202-52-00

Щемелева Мария Андреевна

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2023

1. Цель работы: изучение принципов работы с базовыми структурами данных, получение навыков организации case-меню.
2. Скриншот задания (с вариантом)

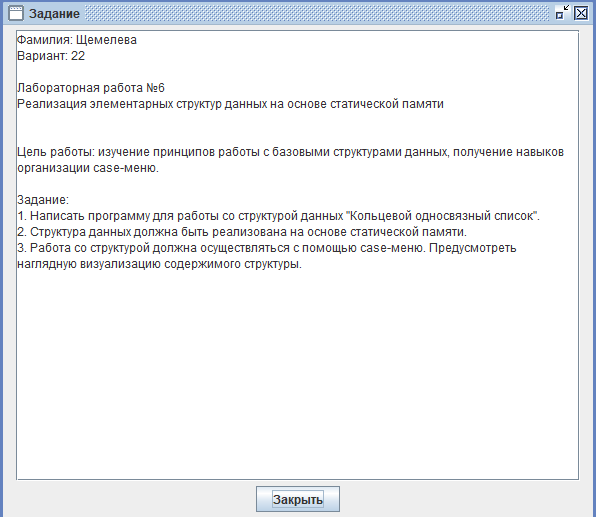


Рисунок 1 – Скриншот задания

1. Описание алгоритма

Кольцевой (циклический, замкнутый) связный список — это разновидность связного списка, при которой первый элемент указывает на последний, а последний — на первый.

Список состоит из начального узла — головы — и связанного с ним списка. Каждый элемент списка содержит информационную и ссылочную части. Т.е. каждый элемент имеет информационное поле (поля) — полезная информация — и ссылку (ссылки), то есть адрес на другой элемент списка.

Односвязный (линейный) список: структура, каждый элемент которой «знает» адрес только следующего за ним элемента.

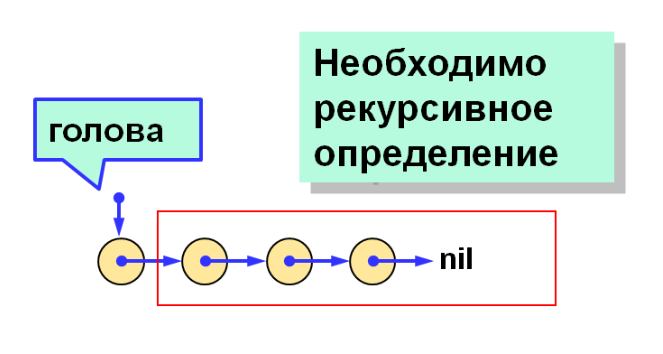


Рисунок 2 – Схема односвязного списка

1. Код программы

**uses** Crt;

**const**

MAX\_NODES = 100;

**type**

TNode = **record**

data: integer;

next: integer;

prev: integer;

**end**;

**var**

head, tail: integer;

nodeArray: **array** [1..MAX\_NODES] **of** TNode;

nodeCounter: integer;

**procedure** InsertNode(**var** head, tail: integer; data: integer);

**var**

node: integer;

**begin**

Inc(nodeCounter);

node := nodeCounter;

nodeArray[node].data := data;

nodeArray[node].next := 0;

nodeArray[node].prev := 0;

**if** head = 0 **then**

**begin**

head := node;

tail := node;

**end**

**else**

**begin**

nodeArray[tail].next := node;

nodeArray[node].prev := tail;

tail := node;

**end**;

**end**;

**procedure** DeleteNode(**var** head, tail: integer; data: integer);

**var**

node, prevNode, nextNode: integer;

**begin**

node := head;

**while** (node <> 0) **and** (nodeArray[node].data <> data) **do**

**begin**

node := nodeArray[node].next;

**end**;

**if** node <> 0 **then**

**begin**

prevNode := nodeArray[node].prev;

nextNode := nodeArray[node].next;

**if** prevNode <> 0 **then**

**begin**

nodeArray[prevNode].next := nextNode;

**end**

**else**

**begin**

head := nextNode;

**end**;

**if** nextNode <> 0 **then**

**begin**

nodeArray[nextNode].prev := prevNode;

**end**

**else**

**begin**

tail := prevNode;

**end**;

Dec(nodeCounter);

**end**;

**end**;

**procedure** PrintList(head: integer);

**var**

node: integer;

**begin**

node := head;

**while** node <> 0 **do**

**begin**

Write(nodeArray[node].data, ' ');

node := nodeArray[node].next;

**end**;

Writeln;

**end**;

**var**

choice: integer;

data: integer;

**begin**

head := 0;

tail := 0;

nodeCounter := 0;

**repeat**

Writeln('Введите число: ');

Writeln('1. Ввод данных');

Writeln('2. Удалить данные');

Writeln('3. Напечатать список данных');

Writeln('4. Выход из меню');

Readln(choice);

**case** choice **of**

1: **begin**

Write('Введите данные: ');

Readln(data);

InsertNode(head, tail, data);

**end**;

2: **begin**

Write('Введите данные: ');

Readln(data);

DeleteNode(head, tail, data);

**end**;

3: **begin**

Writeln('Список данных:');

PrintList(head);

**end**;

**end**;

**until** (choice = 4);

**end**.

5. Результат выполнения программы

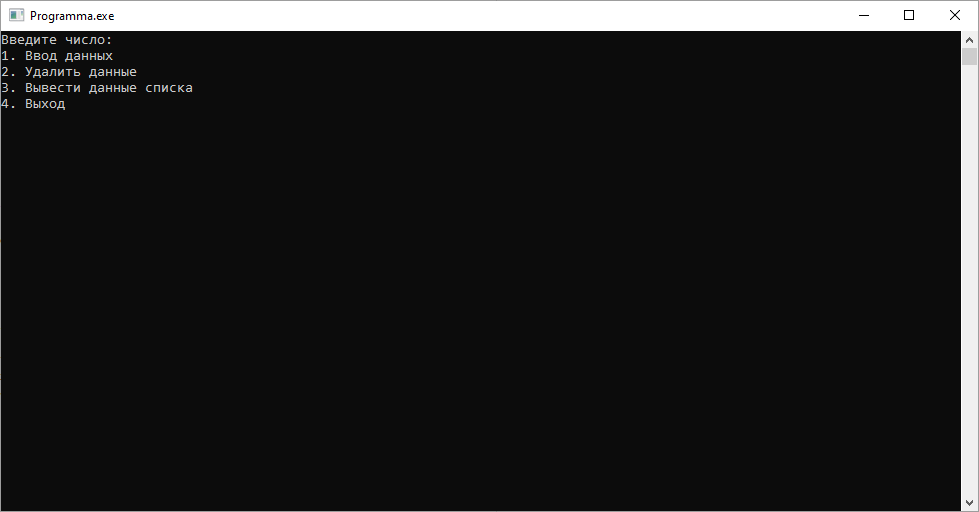


Рисунок 3 – Результат выполнения программы

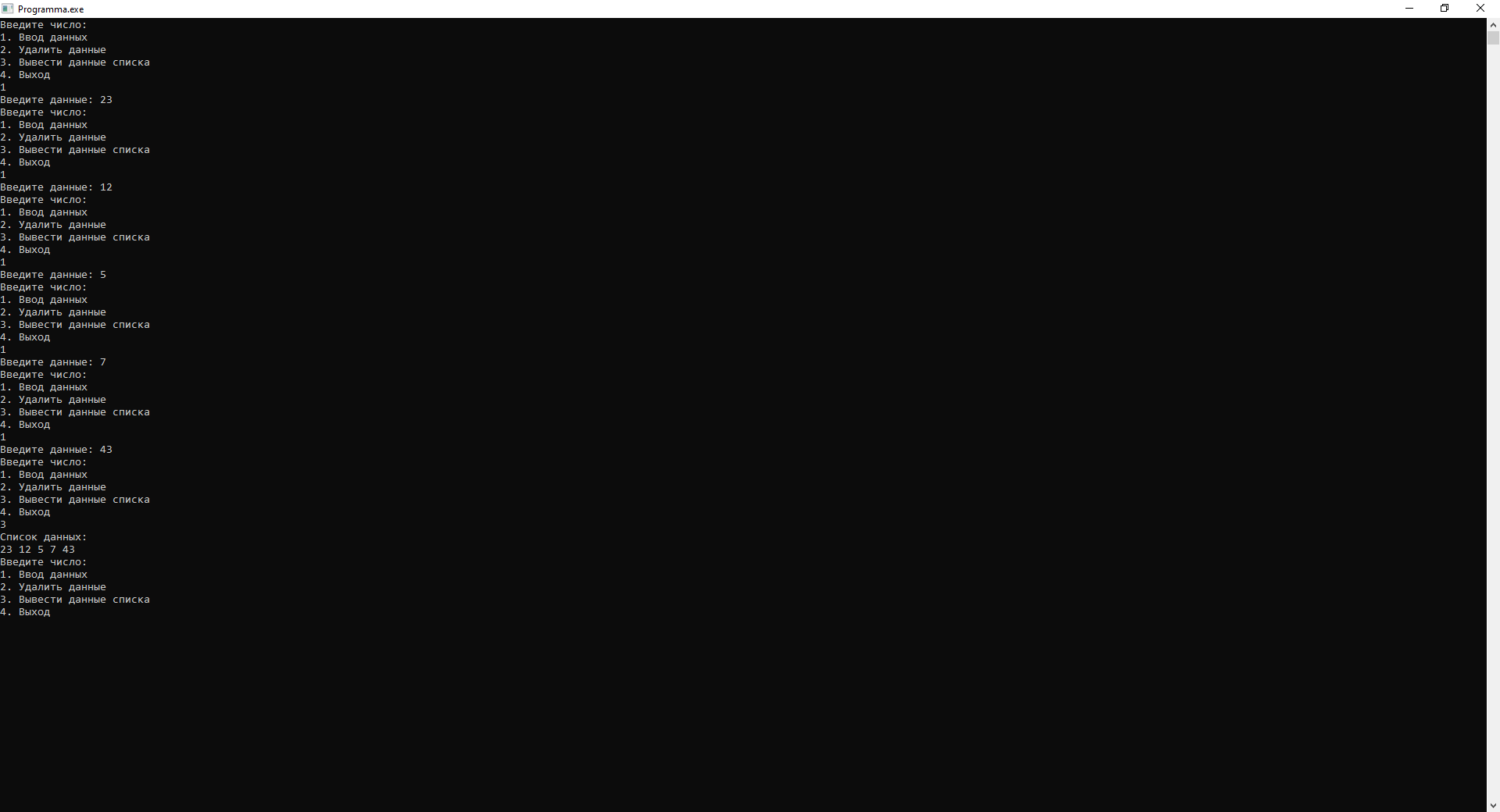


Рисунок 4 - Результат выполнения программы

1. Вывод

В ходе выполнения домашней контрольной работы №6 «Реализация элементарных структур данных на основе статической памяти». Изучили принципы работы с базовыми структурами данных, получили навыки организации case-меню и закрепили знания во время работы.

Данная программа реализует кольцевой односвязный список. Пользователь может выбрать действие из меню: вставка данных, удаление данных, вывод списка данных или выход из меню. Программа использует статическую память для создания новых узлов в списке с помощью массива. Удаление узла также реализовано с помощью статической памяти. Вывод списка данных происходит циклически, проходя по всем узлам списка и выводя значение каждого узла.

Во время работы возникали трудности, по типу не правильном выборе памяти, не статической, а динамической. Проблемы удалось решить и с работой справились в полном объеме.