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

Ананьин Кирилл Сергеевич

Преподаватель:

Сергеева Елизавета Григорьевна

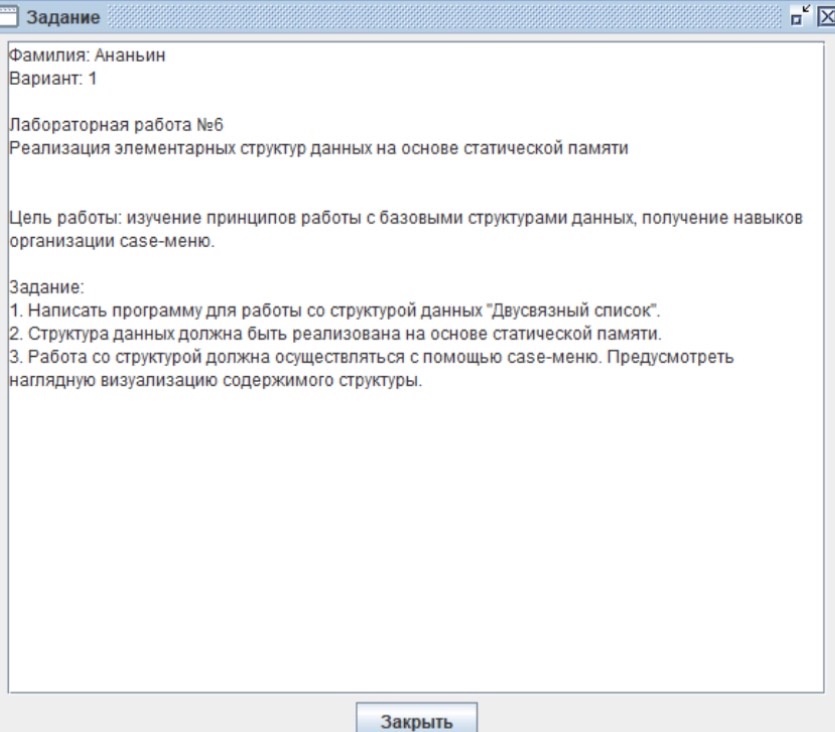
Киров

2023

1.Цель работы

Изучение принципов работы с базовыми структурами данных, получение навыков организации case-menu.

2. Скриншот задания (с вариантом)



3. Описание алгоритма

В данной программе используются такие функции как: dobavitdo (добавляет число до выбранного элемента), dobavitposle (добавляет число после выбранного элемента), udalenie(Удаление записей из списка), poisk (поиск элемента списка) для работы с двухсвязным списком. Так же в ней используется case-menu исходя из условии выполнения задания. Она упрощает работу с программой.

4.Код программы

**uses** Crt;

**const**

MAX\_NODES = 100;

**type**

TNode = **record**

data: integer;

next: ^TNode;

prev: ^TNode;

**end**;

**var**

head, tail: ^TNode;

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

nodeCounter: integer;

**procedure** CreateNode(**var** node: ^TNode; data: integer);

**begin**

**new**(node);

node^.data := data;

node^.next := nil;

node^.prev := nil;

**end**;

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

**var**

node: ^TNode;

**begin**

Inc(nodeCounter);

CreateNode(node, data);

**if** head = nil **then**

**begin**

head := node;

tail := node;

**end**

**else**

**begin**

tail^.next := node;

node^.prev := tail;

tail := node;

**end**;

**end**;

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

**var**

node, prevNode, nextNode: ^TNode;

**begin**

node := head;

**while** (node <> nil) **and** (node^.data <> data) **do**

**begin**

node := node^.next;

**end**;

**if** node <> nil **then**

**begin**

prevNode := node^.prev;

nextNode := node^.next;

**if** prevNode <> nil **then**

**begin**

prevNode^.next := nextNode;

**end**

**else**

**begin**

head := nextNode;

**end**;

**if** nextNode <> nil **then**

**begin**

nextNode^.prev := prevNode;

**end**

**else**

**begin**

tail := prevNode;

**end**;

Dec(nodeCounter);

Dispose(node);

**end**;

**end**;

**procedure** PrintList(head: ^TNode);

**var**

node: ^TNode;

**begin**

node := head;

**while** node <> nil **do**

**begin**

Write(node^.data, ' ');

node := node^.next;

**end**;

Writeln;

**end**;

**var**

choice: integer;

data: integer;

**begin**

head := nil;

tail := nil;

nodeCounter := 0;

**repeat**

Writeln('Введите число: ');

Writeln('1. Ввод данных');

Writeln('2. Удаление данных');

Writeln('3. Вывести данные');

Writeln('4. Выйти из программы');

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 выполнение программы

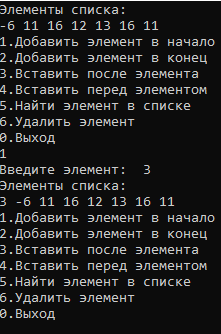


Рисунок 1 – Результат выполнения программы

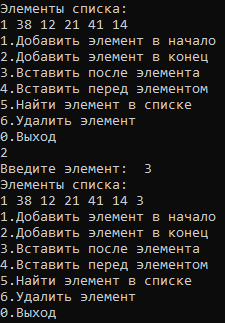


Рисунок 2 – Результат выполнения программы

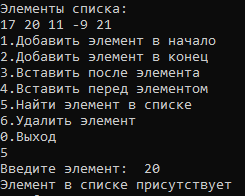


Рисунок 3 – Результат выполнения программы

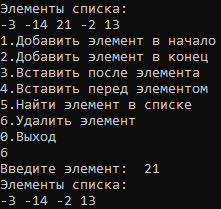


Рисунок 4 – Результат выполнения программы

6. Вывод

В ходе выполнения данной домашней контрольной работы, с помощью доп. Материала, интернета и помощью учителя, мы получили знания о принципах работы с базовыми структурами данных, таких как двухсвязный список и навыки организации case-menu.