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-204-52-00

Алембаев Артём Сергеевич

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2024

1. Цель работы

Цель работы: изучение принципов работы с базовыми структурами данных, получение навыков организации case-меню.

2. Формулировка задания.

Вариант 1.

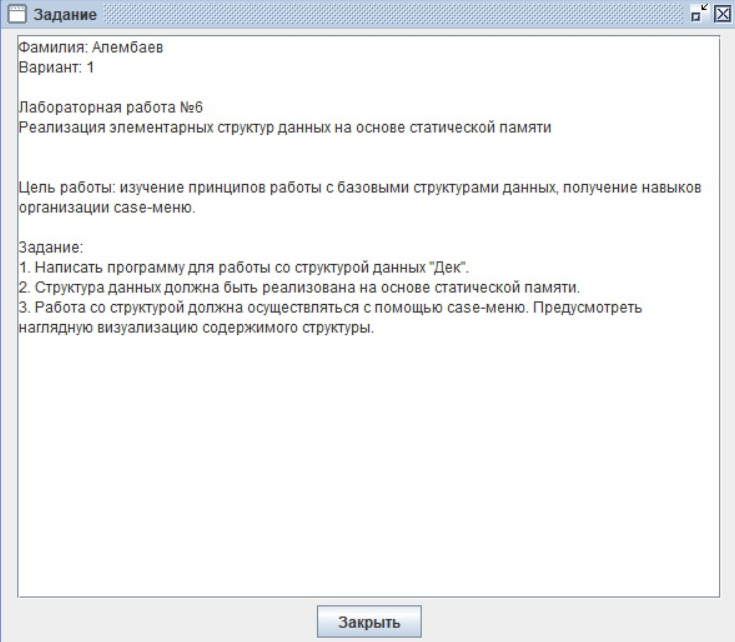


Рисунок 1 – Вариант работы

3. Описание алгоритма.

Дек может быть одного из двух типов: статическим или динамическим. Пользователь сначала выбирает тип дека и задает максимальную длину (для статического дека).

Далее программа предоставляет пользователю следующие возможности для работы с деком:

* Добавление элемента в дек: пользователь указывает, куда добавить элемент (в начало или в конец) и вводит символ.
* Удаление элемента из дека: пользователь указывает, из какого конца удалить элемент.
* Проверка, пуст ли дек.
* Просмотр содержимого дека.

Пользователь может взаимодействовать с деком, выбирая одну из приведенных выше операций. Программа продолжает работу, пока пользователь не завершит ее, выбрав соответствующую опцию.

4. Схема.

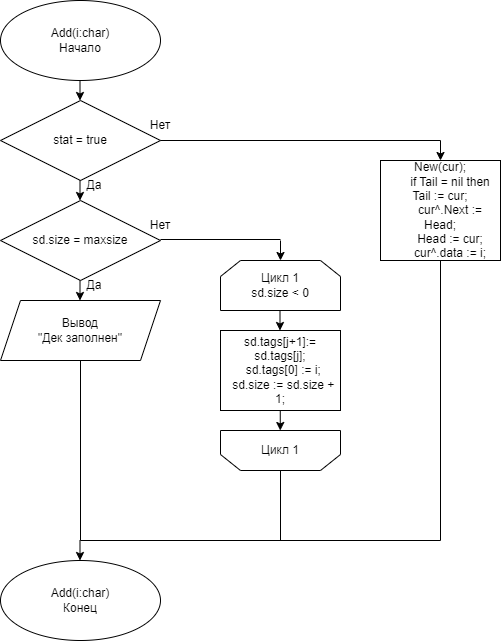


Рисунок 2 – Процедура Add

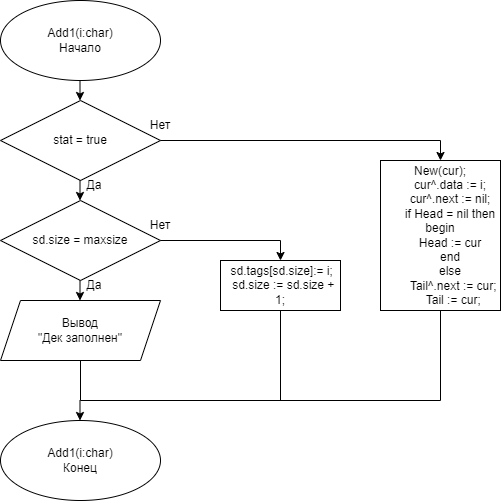


Рисунок 3 – Процедура Add1

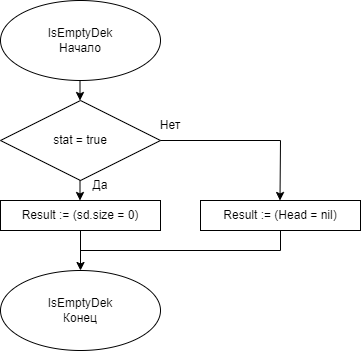


Рисунок 4 – Процедура IsEmptyDek

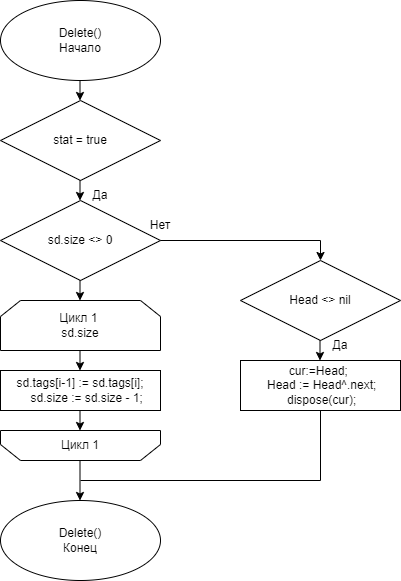


Рисунок 5 – Процедура Delete

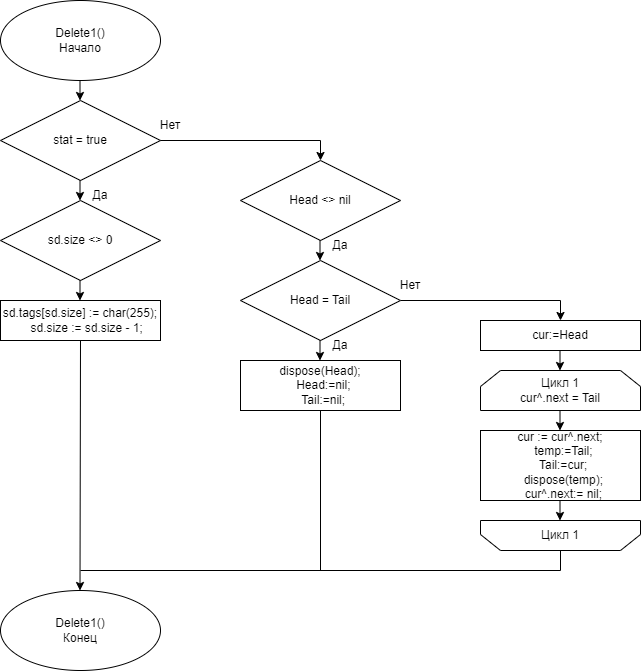


Рисунок 6 – Процедура Delete1

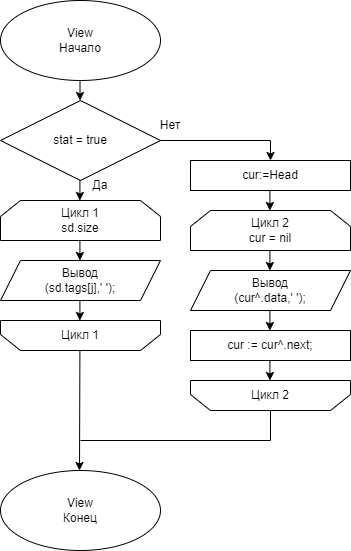


Рисунок 7 – Процедура View

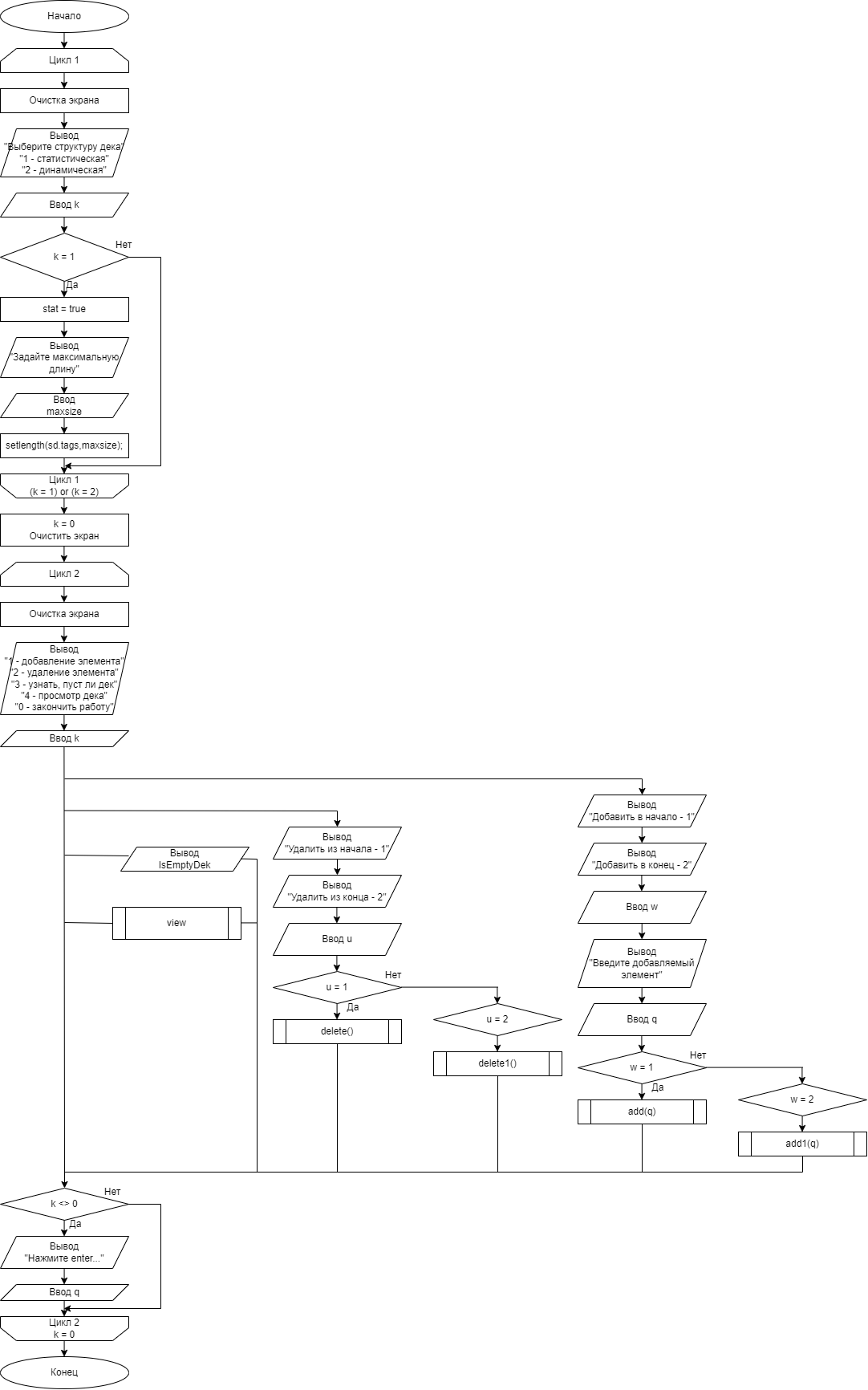


Рисунок 8 – Основная программа

5. Код программы.

**uses** crt;

**type** StatDek = **record**

tags: **array of** char;

size: integer;

**end**;

**type** DinDek = ^Node;

Node = **record**

data: char;

next: DinDek;

**end**;

**var**

sd:StatDek;

Head,Tail:DinDek;

q:char;

w,u:byte;

maxsize,k:integer;

stat:boolean;

**procedure** Add(i: char);

**var** j:integer; cur:DinDek;

**begin**

**if** stat = true **then begin**

**if** sd.size = maxsize **then**

**begin**

writeln('Дэк заполнен');

**exit**;

**end**

**else**

**begin**

**for** j:=sd.size-1 **downto** 0 **do**

sd.tags[j+1]:= sd.tags[j];

sd.tags[0] := i;

sd.size := sd.size + 1;

**end**;

**end**

**else**

**begin**

**New**(cur);

**if** Tail = nil **then** Tail := cur;

cur^.Next := Head;

Head := cur;

cur^.data := i;

**end**;

**end**;

**procedure** Add1(i: char);

**var** cur:DinDek;

**begin**

**if** stat = true **then begin**

**if** sd.size = maxsize **then**

**begin**

writeln('Дэк заполнен');

**exit**;

**end**

**else**

**begin**

sd.tags[sd.size]:= i;

sd.size := sd.size + 1;

**end**;

**end**

**else**

**begin**

**New**(cur);

cur^.data := i;

cur^.next := nil;

**if** Head = nil **then begin**

Head := cur

**end**

**else**

Tail^.next := cur;

Tail := cur;

**end**;

**end**;

**procedure** Delete();

**var** i:integer; cur: DinDek;

**begin**

**if** stat = true **then begin**

**if** sd.size <> 0 **then**

**begin**

**for** i:=1 **to** sd.size-1 **do**

sd.tags[i-1] := sd.tags[i];

sd.size := sd.size - 1;

**end**;

**end**

**else**

**begin**

**if** Head <> nil **then begin**

cur:=Head;

Head := Head^.next;

dispose(cur);

**end**;

**end**;

**end**;

**procedure** Delete1();

**var** cur,temp:DinDek;

**begin**

**if** stat = true **then begin**

**if** sd.size <> 0 **then**

**begin**

sd.tags[sd.size] := char(255);

sd.size := sd.size - 1;

**end**;

**end**

**else**

**if** Head <> nil **then**

**begin**

**if** Head = Tail **then**

**begin**

dispose(Head);

Head:=nil;

Tail:=nil;

**end**

**else**

**begin**

cur:=Head;

**while** cur^.next <> Tail **do**

cur := cur^.next;

temp:=Tail;

Tail:=cur;

dispose(temp);

cur^.next:= nil;

**end**;

**end**;

**end**;

**function** IsEmptyDek: Boolean;

**begin**

**if** stat = true **then**

Result := (sd.size = 0)

**else**

Result := (Head = nil);

**end**;

**procedure** View();

**var** j:integer; cur:DinDek;

**begin**

**if** stat = true **then begin**

**for** j:=0 **to** sd.size-1 **do**

write(sd.tags[j],' ');

**end**

**else**

cur:=Head;

**while** cur <> nil **do begin**

write(cur^.data,' ');

cur := cur^.next;

**end**;

writeln;

**end**;

**begin**

**repeat**

writeln('Выбирите структуру дека:');

writeln('1 - статическая');

writeln('2 - динамическая');

readln(k);

**if** k = 1 **then begin**

stat := True;

writeln('Задайте максимальную длину');

readln(maxsize);

setlength(sd.tags,maxsize);

**end**;

**until** (k = 1) **or** (k =2);

k := 0;

clrscr;

**repeat**

clrscr;

writeln('1 - добавление элемента');

writeln('2 - удаление элемента');

writeln('3 - узнать, пуст ли дек');

writeln('4 - просмотр дека');

writeln('0 - закончить работу');

readln(k);

**case** k **of**

1: **begin**

writeln('Добавить в начало - 1'); writeln('Добавить в конец - 2');

readln(w);

writeln('Введите добавляемый символ');

readln(q);

**if** w = 1 **then** add(q) **else if** w = 2 **then** add1(q);

**end**;

2: **begin**

writeln('Удалить из начала - 1'); writeln('Удалить из конца - 2');

readln(u);

**if** u = 1 **then** delete() **else if** u = 2 **then** delete1();

**end**;

3: writeln(IsEmptyDek);

4: view;

**end**;

writeln('Нажмите enter...');

readln(q);

**until** k = 0;

**end**.

6. Результат выполнения программы.

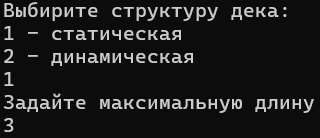


Рисунок 9

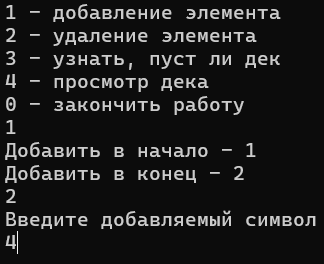


Рисунок 10

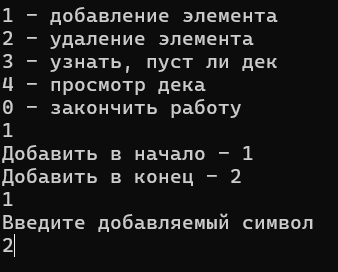


Рисунок 11

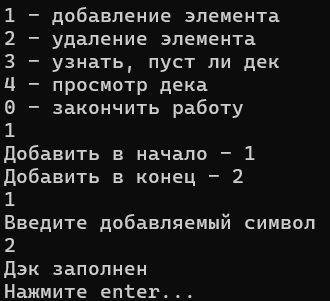


Рисунок 12

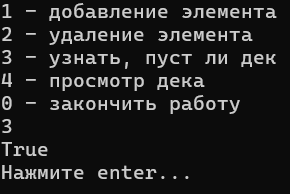


Рисунок 13

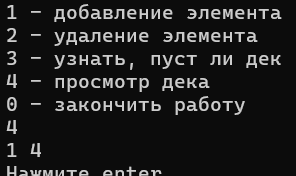


Рисунок 14

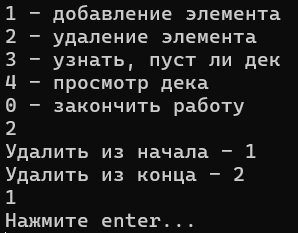


Рисунок 15

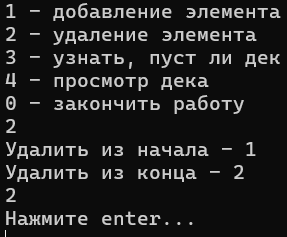


Рисунок 16

7. Вывод.

Выполняя домашнюю контрольную работу, мы смогли достичь поставленной цели: мы изучили принципы работы с базовыми структурами данных, а также получили навыки организации case-меню.

В ходе работы над задание мы вспомнили и поработали с процедурами и функциями внутри Pascal, вспомнили как работает case-меню.