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 ДИСЦИПЛИНЕ «МДК 05.02 Разработка кода информационных систем»**

Выполнил: студент учебной группы

ИСПк-202-52-00

Абдулатипова Амина Заирбеговна

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2024

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

**Задание:**

1. Написать программу для работы со структурой данных "Стек"
2. ﻿﻿﻿Структура данных должна быть реализована на основе статической памяти.
3. ﻿﻿﻿Работа со структурой должна осуществляться с помощью case-меню. Предусмотреть наглядную визуализацию содержимого структуры.

**Описание алгоритма:**

Стек — это динамическая структура данных, состоящая из переменного числа элементов одинакового типа.

Свойства стека:

* Размещение новых элементов и удаление существующих производится только с одного конца, который называют вершиной стека.
* Стек — структура типа LIFO (Last In, First Out) — последним вошел, первым вышел: последний добавленный элемент, который первым выйдет.
* Максимальное число элементов стека не должно ограничиваться.
* Состояние стека рассматривается только по отношению к его вершине, а не по отношению к количеству его элементов.
* Операции, выполняемые над стеком, имеют специальные названия: push — добавление элемента в стек (вталкивание), pop — выталкивание элемента из стека, верхний элемент стека удаляется.

Стек можно хранить с помощью массива, где каждый элемент массива представляет собой узел стека. Узлы не связаны явным образом, а вершина стека отслеживается указателем на последний элемент массива.

Стек представлен в виде записи с массивом данных типа Integer и переменной Top, которая указывает на вершину стека.

Процедура InitializeStack инициализирует стек, устанавливая Top в 0.

Функции IsEmpty и IsFull проверяют, пуст ли стек или полон соответственно.

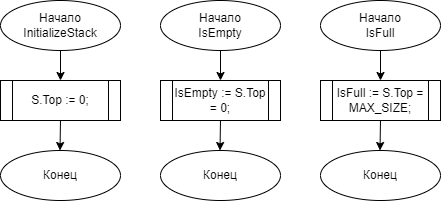
Процедура Push добавляет элемент в стек, увеличивая Top и записывая значение в массив данных.

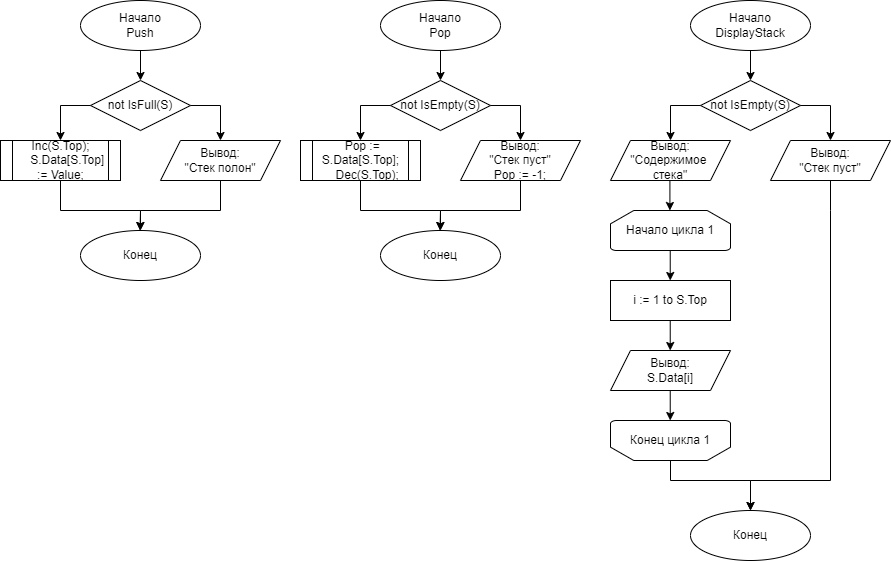
Функция Pop извлекает элемент из стека, уменьшая Top и возвращая извлеченное значение. Если стек пуст, возвращает -1.

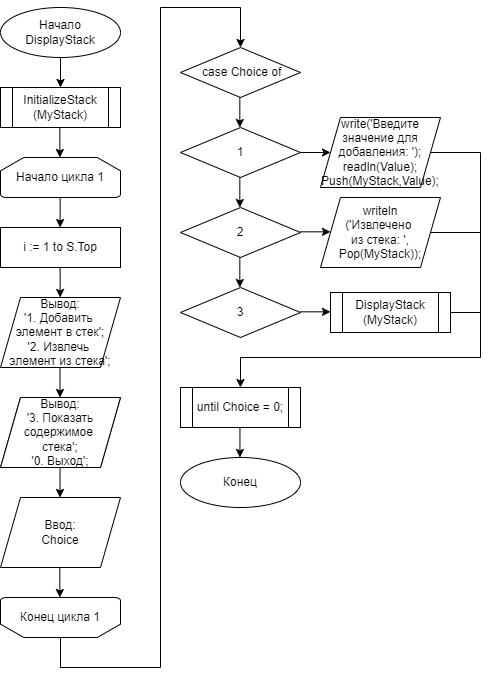
Процедура DisplayStack показывает содержимое стека, выводя все элементы начиная с вершины.

В основной части программы создается стек MyStack, и пользователю предлагается выбирать операции: добавление элемента, извлечение элемента или показ содержимого стека. Пользователь может повторять операции до выбора выхода из программы (ввод 0).

**Схема алгоритма:**

****

****

****

**Код программы:**

**program** StackDemo;

**const**

MAX\_SIZE = 10;

**type**

TStack = **record**

Data: **array**[1..MAX\_SIZE] **of** Integer;

Top: Integer;

**end**;

**procedure** InitializeStack(**var** S: TStack);

**begin**

S.Top := 0;

**end**;

**function** IsEmpty(**const** S: TStack): Boolean;

**begin**

IsEmpty := S.Top = 0;

**end**;

**function** IsFull(**const** S: TStack): Boolean;

**begin**

IsFull := S.Top = MAX\_SIZE;

**end**;

**procedure** Push(**var** S: TStack; Value: Integer);

**begin**

**if not** IsFull(S) **then**

**begin**

Inc(S.Top);

S.Data[S.Top] := Value;

**end**

**else**

writeln('Стек полон');

**end**;

**function** Pop(**var** S: TStack): Integer;

**begin**

**if not** IsEmpty(S) **then**

**begin**

Pop := S.Data[S.Top];

Dec(S.Top);

**end**

**else**

**begin**

writeln('Стек пуст');

Pop := -1;

**end**;

**end**;

**procedure** DisplayStack(**const** S: TStack);

**var**

i: Integer;

**begin**

**if not** IsEmpty(S) **then**

**begin**

writeln('Содержимое стека:');

**for** i := 1 **to** S.Top **do**

writeln(S.Data[i]);

**end**

**else**

writeln('Стек пуст');

**end**;

**var**

MyStack: TStack;

Choice, Value: Integer;

**begin**

InitializeStack(MyStack);

**repeat**

writeln('1. Добавить элемент в стек');

writeln('2. Извлечь элемент из стека');

writeln('3. Показать содержимое стека');

writeln('0. Выход');

write('Введите ваш выбор: ');

readln(Choice);

**case** Choice **of**

1:

**begin**

write('Введите значение для добавления: ');

readln(Value);

Push(MyStack, Value);

**end**;

2: writeln('Извлечено из стека: ', Pop(MyStack));

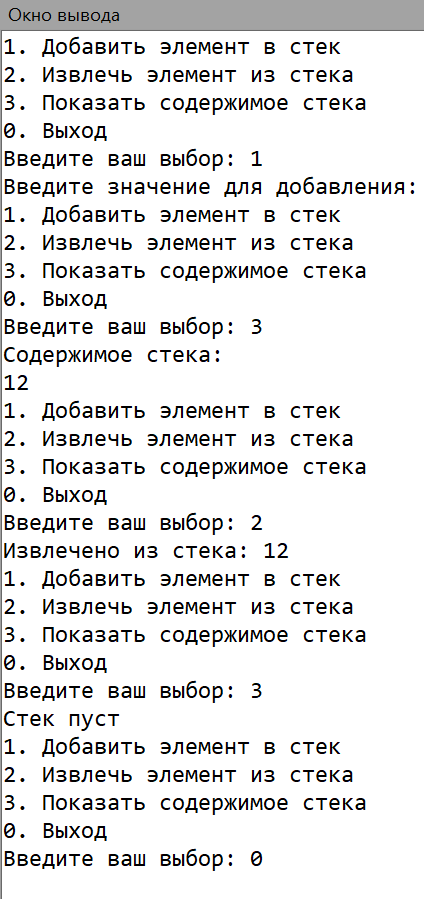
3: DisplayStack(MyStack);

**end**;

**until** Choice = 0;

**end**.

**Результаты выполнения программы:**



**Вывод:** во время выполнения домашней контрольной работы мы получили принципов работы с базовыми структурами данных, получение навыков организации case-меню, изученный материал применили при написании программы для работы со структурой данных "Стек", полученные навыки будут применяться в будущей практике.