Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №2**

**«Изучение одномерных массивов и строк»**

**ПО ДИСЦИПЛИНЕ «Основы алгоритмизации и программирования»**

Выполнил: студент учебной группы

ИСПк-204-52-00

Зараменских Илья Алексеевич

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2024

1. **Цель работы:** получить базовые навыки работы с одномерными массивами, освоить принципы работы со строками как с частным случаем одномерных массивов.
2. **Формулировка задания:** Вариант 6
   1. Написать программу, выполняющую поиск среднеарифметического в заданном массиве.
   2. Написать программу, вычисляющую количество вхождений одной строки в другую в качестве подстроки.
   3. Все данные вводятся с клавиатуры. При выполнении второго пункта запрещается использовать стандартные функции для работы со строками (за исключением функции определения длины строки).
3. **Описание алгоритма:**
   1. Программа запрашивает у пользователя количество элементов массива, которое сохраняется в переменной n. Это значение используется для определения размера массива. Затем программа создает массив и предлагает пользователю поочередно ввести значения каждого элемента массива. Далее программа инициализирует переменную *sum* со значением 0 и использует цикл для прохождения по всем элементам массива, суммируя их в *sum*. После завершения цикла программа проверяет, больше ли n нуля. Если *n* > 0, то среднее арифметическое рассчитывается как *sum* / *n* и сохраняется в переменной *average*. Если же n равно нулю, среднее арифметическое устанавливается равным 0, чтобы избежать деления на ноль. В конце программа выводит результат с округлением до двух знаков после запятой
   2. Программа начинает с запроса основной строки, а затем подстроки, которую нужно найти. Далее программа определяет длины основной строки и подстроки, сохраняя их в *mainLen* и *subLen* соответственно. Если длина подстроки *subLen* больше нуля, программа переходит к поиску: внешний цикл последовательно проходит по каждому символу основной строки от первой позиции до позиции *mainLen* - *subLen* + 1, чтобы избежать выхода за границы. На каждой позиции запускается вложенный цикл, который сравнивает текущие символы основной строки и подстроки, начиная с текущей позиции. Переменная *matchCount* отслеживает количество совпавших символов. Если все символы подстроки совпали с символами основной строки (то есть *matchCount* равно *subLen*), это значит, что найдено вхождение подстроки, и *count* увеличивается на единицу. В завершение программа выводит количество вхождений подстроки в основную строку

1. **Схема алгоритма:**

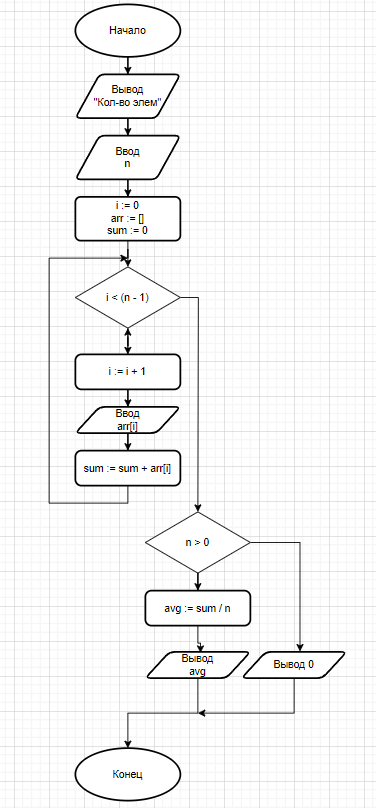


Рисунок 1 – Схема алгоритма 1

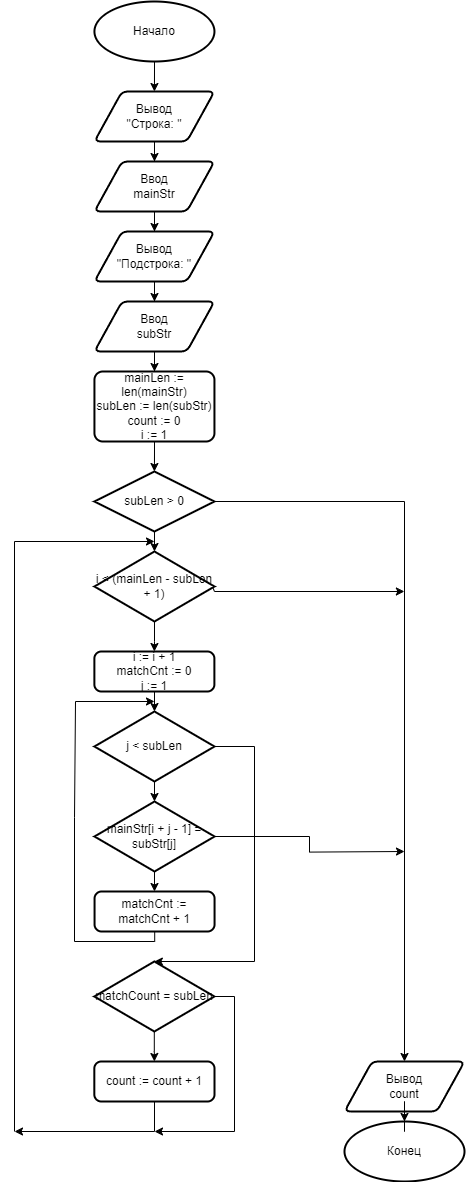


Рисунок 2 – Схема алгоритма 2

1. **Код программы:**
   1. Код 1

var

arr: array of Integer;

n, i, sum: Integer;

average: Real;

begin

Write('Кол-во элем: ');

ReadLn(n);

SetLength(arr, n);

sum := 0;

for i := 0 to n - 1 do

begin

Write('Элемент ', i + 1, ': ');

ReadLn(arr[i]);

sum := sum + arr[i];

end;

if n > 0 then

average := sum / n

else

average := 0;

WriteLn('Ср. арифм. = ', average:0:2);

end.

* 1. Код 2

var

mainStr, subStr: string;

mainLen, subLen, i, j, count, matchCount: Integer;

begin

Write('Строка: ');

ReadLn(mainStr);

Write('Ищем: ');

ReadLn(subStr);

mainLen := Length(mainStr);

subLen := Length(subStr);

count := 0;

if subLen > 0 then

for i := 1 to mainLen - subLen + 1 do

begin

matchCount := 0;

for j := 1 to subLen do

if mainStr[i + j - 1] = subStr[j] then

Inc(matchCount)

else

break;

if matchCount = subLen then

Inc(count);

end;

WriteLn('Кол-во вхождений = ', count);

1. end.
2. **Результат выполнения программы**

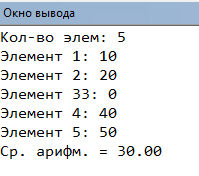


Рисунок 3 - результат выполнения кода 1

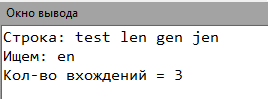


Рисунок 4 - результат выполнения кода 2

1. **Вывод:** в ходе работы были рассмотрены основные конструкции языка Pascal, такие как массивы, циклы, условные операторы и работа со строками, которые позволяют эффективно организовывать хранение и обработку данных.

Массивы в Pascal позволяют хранить набор значений одного типа. Размер массива может быть задан динамически с помощью функции `SetLength`, что дает гибкость при работе с массивами. Циклы используются для обработки элементов массива, таких как ввод значений и вычисление суммы, что делает код компактным и удобочитаемым.

Во втором алгоритме для обработки строк использовались вложенные циклы для ручного поиска подстрок, что позволяет лучше понять принципы работы с текстовыми данными на низком уровне.

Для форматирования чисел с плавающей запятой используется конструкция `:0:2`, позволяющая задать количество знаков после запятой, что полезно при выводе результатов, таких как среднее арифметическое.