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

Исупов Максим Олегович

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2023

**Цель работы**

Цель работы: получить базовые навыки работы с одномерными массивами, освоить принципы работы со строками как с частным случаем одномерных массивов.

**Формулировка задания**

Вариант: 8

Лабораторная работа №2

Изучение одномерных массивов и строк

Цель работы: получить базовые навыки работы с одномерными массивами, освоить принципы работы со строками как с частным случаем одномерных массивов.

1. Написать программу, выполняющую поиск максимального элемента в массиве и определяющую количество вхождений этого элемента в массив.
2. Написать программу, определяющую является ли заданная строка подпоследовательностью исходной строки.
3. Все данные вводятся с клавиатуры. При выполнении второго пункта запрещается использовать стандартные функции для работы со строками ( за исключением функции определения длины строки ).

**Описание алгоритма первой программы**

Необходимо найти максимальный элемент в массиве и посчитать сколько раз он входит в массив

1. объявляем переменные i, k, max и массив из 10 элементов. Присваиваем переменной max значение 0;
2. для начала нужно ввести 10 элементов массива с клавиатуры;
3. выводим на экран введённый массив;
4. проходим по массиву A, находим наибольший элемент и сохраняем его в переменной max;
5. проходим по массиву A, сравнивая каждый элемент массива со значением переменной max, если значения равны, то увеличиваем значение переменной k на единицу;
6. выводим на экран максимальный элемент и количество его вхождений в массиве.

**Схема алгоритма первой программы**

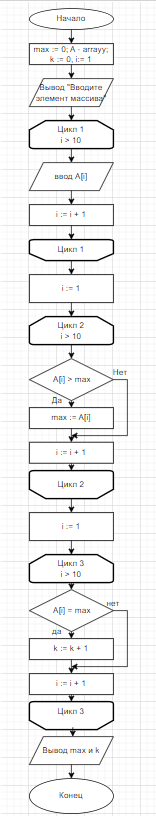


Рисунок 1 – Схема алгоритм задачи

**Код программы**

**Program** ZadDrk21;

**var** i, k,max: integer; A:**array**[1..10] **of** integer;

**begin**

max := 0;

Writeln ('Вводите элемент массив (по одному)');

**for** i := 1 **to** 10 **do begin**

readln (A[i]);

**end**;

writeln (A);

**for** i:= 1 **to** 10 **do begin**

**if** A[i] > max **then**

max := A[i];

**end**;

**for** i:= 1 **to** 10 **do begin**

**if** A[i] = max **then**

k := k + 1;

**end**;

writeln ('Максимальный элемент: ', max);

writeln ('Количество вхождений в массив: ', k);

**end**.

**Результат выполнения программы**

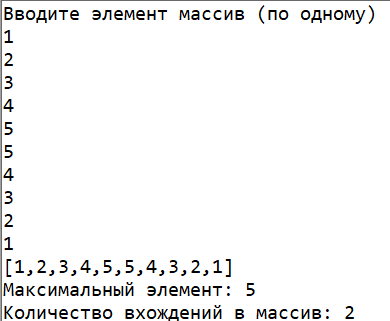
****

Рисунок 2 – Результат выполнения программы 1

**Описание алгоритма второй программы**

Необходимо определить, является ли заданная строка подпоследовательностью введённой исходной строки.

1. объявляем переменные i, k, print, str и str1. Присваиваем переменным i и k значение равное 1. Переменной print присваиваем значение «Строки не совпадают». А переменные str и str1 вводятся с клавиатуры;
2. если длина строки str совпадает с длинной строки str1, то выполняем алгоритм;
3. если предыдущее условие истинно, то запускаем цикл, который будет выполняться до тех пор, пока i не превысит длину строки str и str1;
4. в цикле проверяется условие, если значение i-ого индекса в строках str и str1 совпадает, то увеличиваем значение i на 1;
5. если значение не совпадает, устанавливаем значение переменной print «Строки не совпадают» и завершаем цикл с помощью оператора break;
6. после завершения циклов выводим на экран значение переменной print, которые показывает результат сравнения строк.

**Схема алгоритм второй программы**

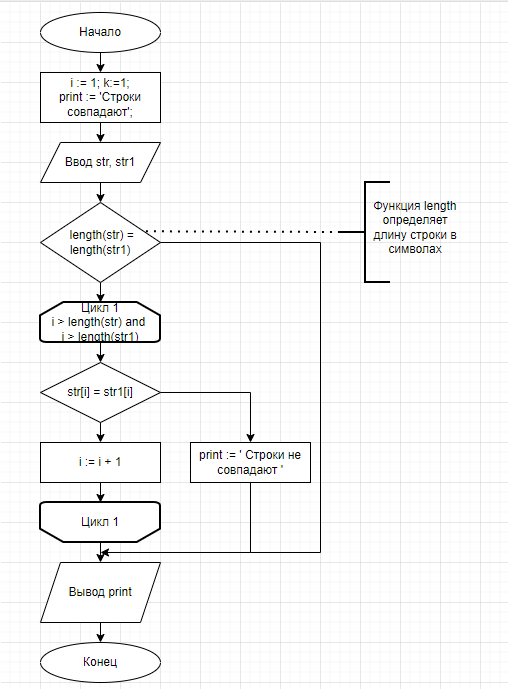


Рисунок 3 - Схема алгоритм задачи

**Код программы**

**Program** Zaddkr22;

**var** str, str1, print: string; I, k: integer;

**begin**

i:= 1;

k:= 1;

print := 'Строки совпадают';

writeln ('Введите исходную строку');

readln (str);

writeln ('Введите строку');

readln (str1);

**if** length(str) = length(str1) **then begin**

**while** (i <= length(str)) **and** (i <= length(str1)) **do begin**

**if** str[i] = str1[i] **then begin**

i := i + 1;

**end**

**else begin**

print := 'Строки не совпадают';

**break**

**end;**

**end**;

**end**;

writeln (print);

**end**.

**Результат выполнения программы**

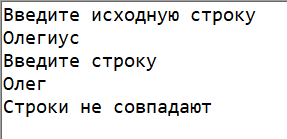
****

Рисунок 4 – Результат выполнения программы

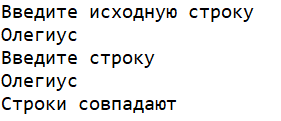
****

Рисунок 5 – Результат выполнения программы

**Вывод**

В ходе выполнения домашней лабораторной работы мы закрепили знания полученные при выполнение лабораторных работа №5-6 и №7-8. При написании кода задачи были использованы конструкции условия if изученные на лабораторных работах №1-2 и конструкции с предусловием While. Так же были использованы знания, полученные на лекциях и при выполнении лабораторных работ №5-6 и №7-8. Никаких сложностей при выполнения домашней контрольной работы не встретилось.

Так же для написания схемы алгоритма был использован интернет -ресурс «draw.io». Его использование упрощает написание схем алгоритмов, уменьшает вероятность ошибок, а также в случае пропуска какого-либо цикла его можно без каких-либо проблем вставить в любое место схемы.