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»**

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

Выполнил: студент учебной группы

ИСПк-205-52-00

Бирюков Павел Сергеевич

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2023

1. **Цель работы:**

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

1. **Формулировка задания:**
2. Написать программу, выполняющую поиск такого минимального k, что сумма первых k элементов массива превосходит заданное число s.
3. Написать программу, вычисляющую количество вхождений одной строки в другую в качестве подстроки.
4. Все данные вводятся с клавиатуры. При выполнении второго пункта запрещается использовать стандартные функции для работы со строки (за исключением функции определения длины строки).
5. **Описание алгоритма:**
6. Задается тип переменной k, потом программа создает a массив из k рандомных элементов, после этого программа начиная проверять все элементы, начиная с первого элемента массива a, проверяя если сумма элементов массива a превышает или равна числу s. И в конечном итоге программа выводит значение k и завершает программу
7. Запрашиваем у пользователя основную строку и подстроку.

2. Инициализируем переменные count, k, c, i, g.

- count будет хранить количество вхождений подстроки в основную строку.

- k будет хранить длину подстроки.

- c будет использоваться для отслеживания текущей позиции в подстроке.

- i будет использоваться в цикле для перебора символов основной строки.

- g будет хранить длину основной строки.

3. Инициализируем count = 0.

4. В цикле от 1 до длины основной строки:

- Проверяем, если текущий символ основной строки равен первому символу подстроки.

- Если да, то проверяем следующий символ основной строки и следующий символ подстроки.

- Если они совпадают, то продолжаем проверку до конца подстроки.

- Если все символы подстроки совпадают, увеличиваем count на 1 и сбрасываем c в 1 (начало подстроки).

- Если не все символы подстроки совпадают, сбрасываем c в 1 (начало подстроки).

- Если нет, сбрасываем c в 1 (начало подстроки).

5. Выводим количество вхождений подстроки в основную строку.

**4. Схема алгоритма:**

|  |  |
| --- | --- |
| 1) | 2) |

1. **Код программы:**

1)

**var**

a: **array**[1..1000] **of** Integer;

s, sum, k, i: integer;

**begin** // Вводим массив и число s

write('Введите число k: ');

readln(k);

**for** i := 1 **to** k **do**

**begin**

a[i]:=random(1,k);

**if** i **mod** 10 = 0 **then** writeln(a[i]:3)

**else** write(a[i]:3);

**end**;

writeln();

writeln('Введите число s:');

readln(s);

sum := 0;// Ищем минимальное k

k := 1;

**while** (sum <= s) **and** (k <= 10) **do**

**begin**

sum := sum + a[k];

k := k + 1;

**end**;

// Выводим результат

**if** sum > s **then**

writeln('Минимальное k: ', k-1)

**else**

writeln('Невозможно найти такое k');

**end**.

2)

**var**

l, s: string;

count, k, c, i, g: integer;

**begin**

writeln('Введите основную строку:'); // Вводим строки

readln(l);

writeln('Введите подстроку:');

readln(s);

count := 0; // Ищем количество вхождений подстроки

k := length(s);

g := length(l);

c := 1;

**for** i:=1 **to** g **do**

**begin**

**if** l[i] = s[c] **then**

**begin**

**if** s[c] = s[k] **then**

**begin**

count := count + 1;

c := 1;

**end**

**else** c := c + 1;

**end**

**else** c := 1;

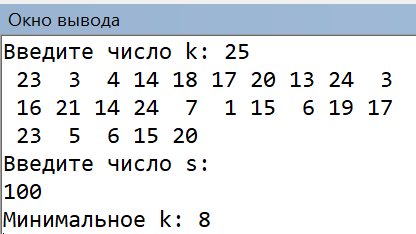
**end**;

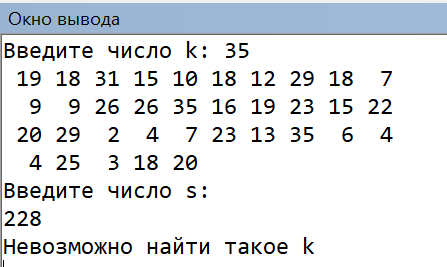
writeln('Количество вхождений подстроки: ', count); // Выводим результат

**end**.

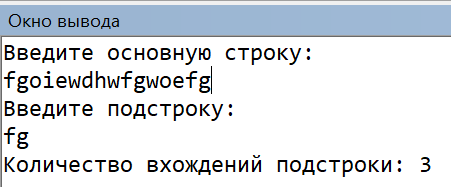
1. **Результат выполнения программы:**

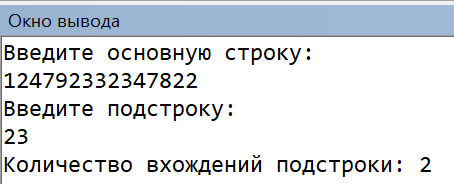
1)



****

2)





1. **Вывод:**

В завершение скажу что заданную задачу я выполнил. В написание кода были трудности, так как при запуске цикла на проверку каждого числа выдавалась ошибка что цикл вышел за пределы массива. Поэтому мне пришлось сделать так чтобы всего этого не было и в этом мне помогла голова. В начале я думал что проблема была в числе, но потом я понял что при каждом выполненном условии в цикле, число оставалось прежним и увеличивалось постоянно. Еще была проблема с удалением старого числа и сохранением количества выполненных условий. По итогу завершения данной задачи, изучил базовую структуру организации программы и основные конструкции языка программирования Pascal.