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-203-52-00

Калинин Максим Константинович

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2023

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

**Вариант 11**

**Задача 1**

Написать программу, определяющую для каждого различного элемента сколько раз он встречается в заданном массиве.

**Алгоритм решения**



Рис.1 - Алгоритм задания №1

**Описание алгоритма**

1. Ввод константы z - максимального допустимого размера массива;
2. Ввод значения n - размера массива;
3. Если n меньше 1 или больше z, программа выводит сообщение о неверном размере массива и завершает выполнение;
4. Ввод элементов массива a;
5. Выполнение операций над каждым значением массива a:
   1. Присваивание переменной cnt значения 0;
   2. Если значение с таким же индексом в массиве b равно false, то выполняется цикл подсчета количества вхождений данного элемента в массиве:
      * Если значение элемента не было выведено (не отмечено как true в b), проверяется каждый элемент массива на равенство с текущим;
      * Если обнаруживается совпадение с другим элементом (не с текущим индексом), текущий элемент помечается true, как уже выведенный, и увеличивается счетчик cnt;
   3. Текущий элемент помечается true, как уже выведенный;
   4. Увеличивается счетчик cnt для текущего элемента;
   5. Выводится информация о количестве вхождений данного элемента в массиве.

**Код решения программы**

**const** z=1000; //макс размер

**var** a: **array**[1..z] **of** integer; b: **array**[1..z] **of** boolean;

n, i, j, cnt: integer;

**begin**

writeln('Введите размер массива:');

readln(n);

**if** (n<1) **or** (n>z) **then**

**begin**

writeln('Неверный/слишком большой размер массива');

**exit**;

**end**;

writeln('Введите элементы массива:');

**for** i:=1 **to** n **do** read(a[i]);

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

**begin**

cnt:=0;

**if not** b[i] **then** //проверка на выведен ли

**begin**

**for** j:=1 **to** n **do**

**begin**

**if** (i<>j) **and** (a[i]=a[j]) **then**

**begin**

b[j]:=true; //уже выведены

cnt:=cnt+1;

**end**;

**end**;

b[i]:=true;

cnt:=cnt+1;

writeln('Элемент ', a[i], ' встречается ', cnt, ' раз(а)');

**end**;

**end**;

**end**.

**Результат выполнения программы**

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 3  1  2  1 | Элемент 1 встречается 2 раз(а)  Элемент 2 встречается 1 раз(а) |
| 4  -1  -20  0  0 | Элемент -1 встречается 1 раз(а)  Элемент -20 встречается 1 раз(а)  Элемент 0 встречается 2 раз(а) |
| 3  1  1  1 | Элемент 1 встречается 3 раз(а) |
| -2 | Неверный/слишком большой размер массива |
| 5  1  2  3  4  5 | Элемент 1 встречается 1 раз(а)  Элемент 2 встречается 1 раз(а)  Элемент 3 встречается 1 раз(а)  Элемент 4 встречается 1 раз(а)  Элемент 5 встречается 1 раз(а) |

**Задача 2**

Написать программу, определяющую количество уникальных подстрок из трех символов, входящих в заданную строку

**Алгоритм решения**



Рис.1 – Алгоритм задания №2

**Описание алгоритма**

1. Ввод константы z - максимального допустимого размера массива для символов;
2. Ввод строки orig с клавиатуры;
3. Если длина введенной строки меньше 3 или больше константы n, программа выводит сообщение о некорректной длине строки и завершает выполнение;
4. Заполнение массива sus символами из строки orig;
5. Присваивание переменной cnt значения 0 для подсчета уникальных подстрок из трех символов;
6. Поиск уникальных подстрок:
   1. Проход по массиву sus от 1 до (длины строки)-2 элемента;
   2. Если элемент b[i] равен false – цикл проверки уникальности текущей подстроки:
      * Проверка всех последующих элементов массива sus на совпадение с текущим элементом sus[i], sus[i+1], sus[i+2];
      * Если найдено совпадение (кроме текущего индекса), пометка соответствующих элементов массива b значением true, чтобы отметить, что эти подстроки уже были выведены;
      * Увеличение счетчика cnt для текущей уникальной подстроки;
   3. Если текущая подстрока не была помечена как уже выведенная, вывод информации о ней;
7. Вывод общего количества уникальных подстрок из трех символов.

**Код решения программы**

**const** n=1000;

**var** orig:string;

sus:**array**[1..n] **of** char;

b:**array**[1..n] **of** boolean;

i,j,cnt:integer;

**begin**

writeln('Введите строку:');

readln(orig);

**if** (length(orig)<3) **or** (length(orig)>n) **then begin**

writeln('Слишком большая/маленькая длина строки');

**exit**;

**end**;

**for** i:=1 **to** length(orig) **do** sus[i]:=orig[i];

cnt:=0;

**for** i:=1 **to** length(orig)-2 **do begin**

**if not** b[i] **then begin**

**for** j:=i+1 **to** length(orig)-2 **do begin**

**if** (sus[i]=sus[j]) **and** (sus[i+1]=sus[j+1]) **and** (sus[i+2]=sus[j+2]) **then begin**

b[j]:=true;

b[i]:=true;

**end**;

**end**;

**if not** b[i] **then begin**

cnt:=cnt+1;

writeln('Уникальная подстрока: ', sus[i], sus[i+1], sus[i+2]);

**end**;

**end**;

**end**;

writeln('Количество уникальных подстрок из трех символов: ', cnt);

**end**.

**Результат выполнения программы**

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| лохлох | Уникальная подстрока: охл  Уникальная подстрока: хло  Количество уникальных подстрок из трех символов: 2 |
| и | Слишком большая/маленькая длина строки |
| пен | Уникальная подстрока: пен  Количество уникальных подстрок из трех символов: 1 |
| Все все | Уникальная подстрока: Все  Уникальная подстрока: се  Уникальная подстрока: е в  Уникальная подстрока: вс  Уникальная подстрока: все  Количество уникальных подстрок из трех символов: 5 |
| все все | Уникальная подстрока: се  Уникальная подстрока: е в  Уникальная подстрока: вс  Количество уникальных подстрок из трех символов: 3 |

Вывод:

1. Научился более корректно строить электронные схемы алгоритмов.
2. Изучил новые функции языка Pascal.
3. Перевод элементов строки в отдельный массив.
4. При написании программы использовать противоположные условия.
5. Быстрый проход по элементам массива с использованием цикла for.
6. Альтернативы использования стандартных функции для работы со строками, их написание.
7. Использовать различные индексы для одного массива для хранения отдельный значений.
8. Использование второго массива с одинаковыми индексами в качестве хранилища дополнительный значений для основного.
9. Улучшение различных навыков по совмещению нескольких условий в одно.