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»**

Выполнила: студент учебной группы

ИСПк-202-52-00

Долгополов Ярослав Максимович

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2024

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

1. **Формулировка задания 1. Вариант 8**

**Постановка задачи**

Написать программу, выполняющую поиск двух элементов заданного массива, таких, что их разность по модулю минимальна.

1. **Описание алгоритма** 
   1. Задаём переменной **min** значение максимально возможное в данном языке программирования у данного целочисленного типа данных (у int в PascalABC – это 32767).
   2. Задать длину динамического массива введённым значением переменной **n**. Задать массив **s** вводом с клавиатуры.
   3. Устанавливаем цикл 1 с параметром **i**: изначальное значение **i** = 0. Установить ограничение для выполнения цикла <= **n**-1.
   4. В теле цикла 1:
      1. Устанавливаем цикл 2 с параметром **j**: изначальное значение **j** = **i**+1. Установить ограничение для выполнения цикла <= **n**-1.
      2. Задаём условие: если модуль разности элемента массива **s** с индексом **i** и элемента массива **s** с индексом **j** меньше переменной **min**, то даём переменной **min** значениемодуля разности элемента массива **s** с индексом **i** и элемента массива **s** с индексом **j**.
      3. После выполнения тела цикла 2 увеличиваем значение **j** на 1.
   5. После выполнения тела цикла 1 увеличиваем значение **i** на 1
   6. Выводим **min**.
   7. Устанавливаем цикл 3 с параметром **i**: изначальное значение **i** = 0. Установить ограничение для выполнения цикла <= **n**-1.
   8. В теле цикла 3:
      1. Устанавливаем цикл 4 с параметром **j**: изначальное значение **j** = **i**+1. Установить ограничение для выполнения цикла <= **n**-1.
      2. В теле цикла 4 задаём условие: если модуль разности элемента массива **s** с индексом **i** и элемента массива **s** с индексом **j** равен переменной **min**, то выводим эти элементы.
      3. После тела цикла 4 увеличиваем **j** на 1.
   9. После тела цикла 3 увеличиваем **i** на 1.
2. **Схема алгоритма с комментариями**

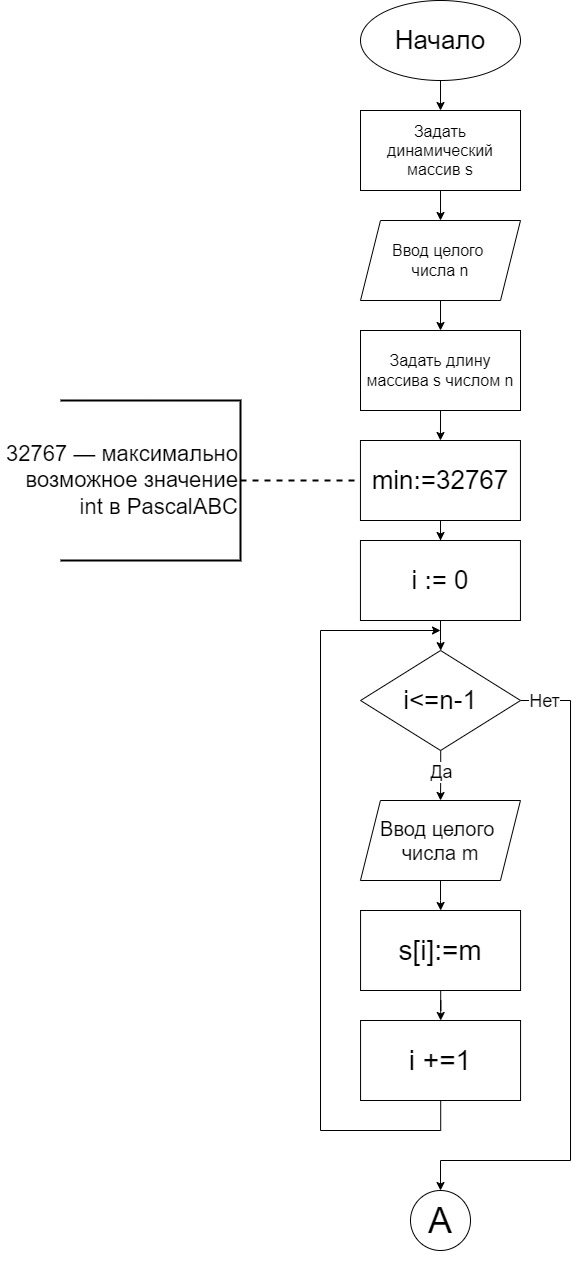
****

Рисунок 1.1 - Алгоритм решения

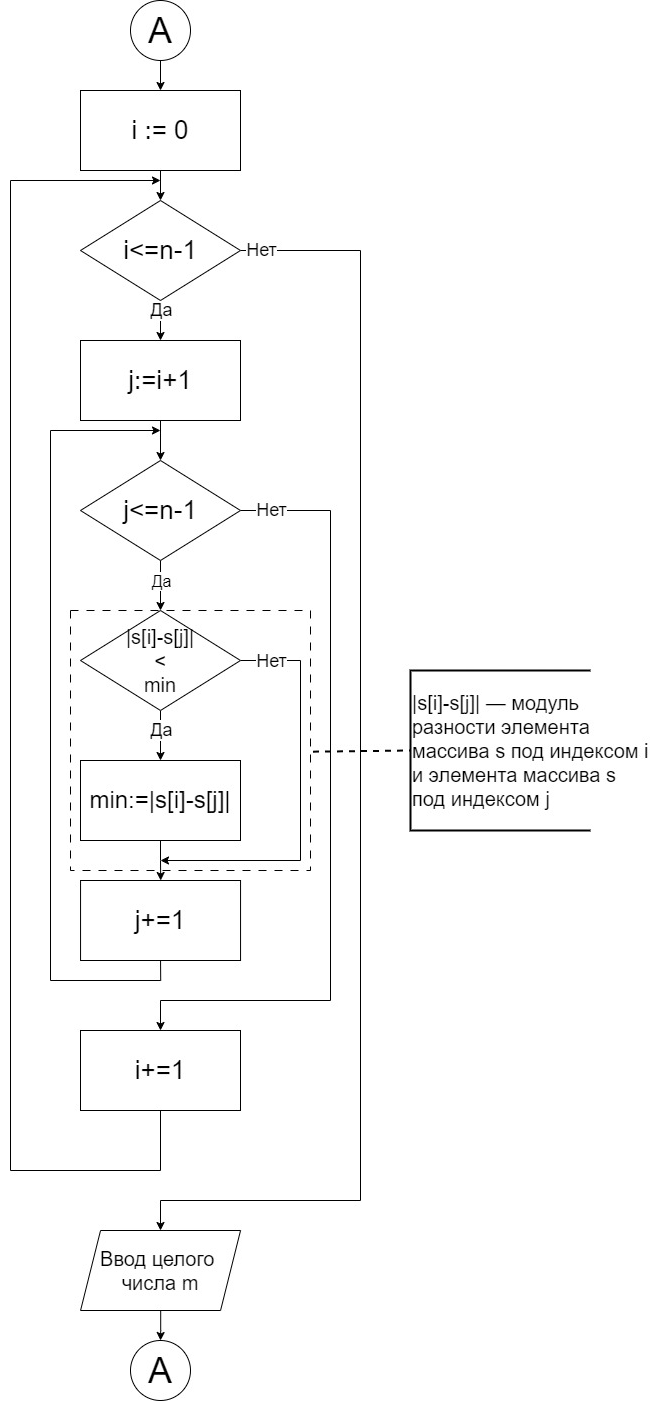
****

Рисунок 1.2 - Алгоритм решения

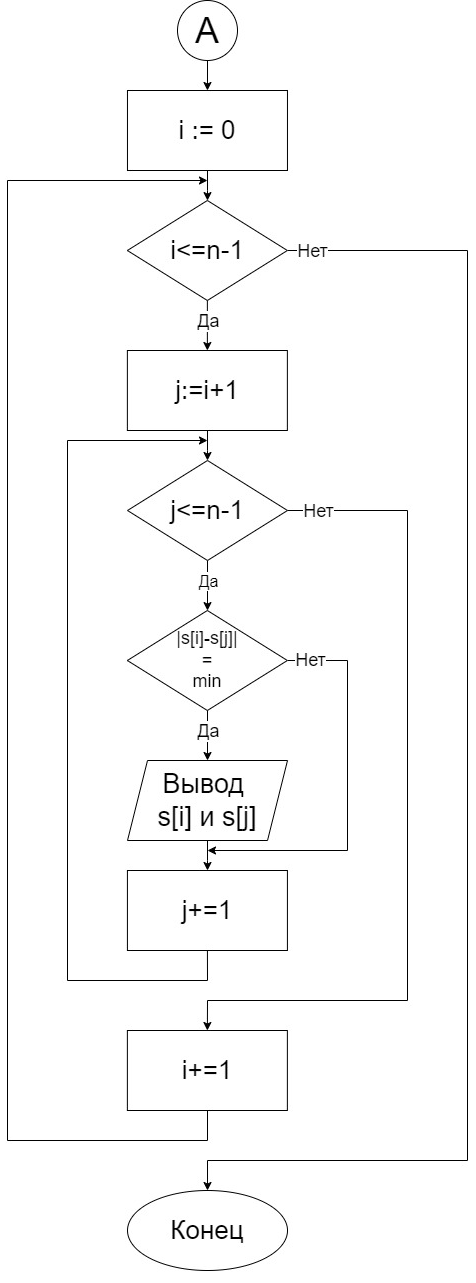


Рисунок 1.3 - Алгоритм решения

1. **Код программы**

**var**

s: **array of** integer;

min, i, j, n, m: integer;

**begin**

writeln('Введите количество чисел в массиве');

readln(n);

SetLength(s, n);

min := 32767;

writeln('Введите числа массива');

**for** i := 0 **to** (n-1) **do**

**begin**

readln(m);

s[i] := m

**end**;

**for** i := 0 **to** (n-1) **do**

**begin**

**for** j := (i + 1) **to** (n-1) **do**

**begin**

**if** abs(s[i] - s[j]) < min **then**

**begin**

min := abs(s[i] - s[j])

**end**

**end**

**end**;

writeln('Минимальный модуль разности: ', min);

writeln('Пары:');

**for** i := 0 **to** (n-1) **do**

**begin**

**for** j := (i + 1) **to** (n-1) **do**

**begin**

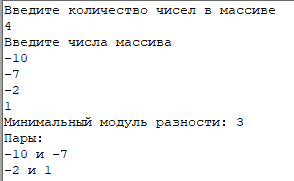
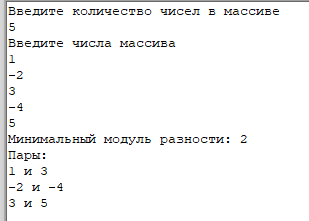
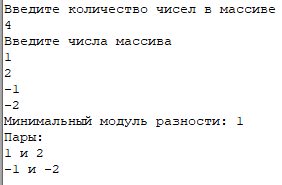
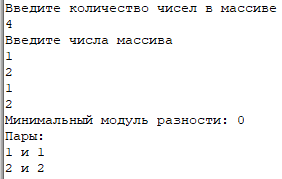
**if** abs(s[i] - s[j]) = min **then** writeln(s[i], ' и ', s[j])

**end**

**end**

**end**.

1. **Результат выполнения программы**

****

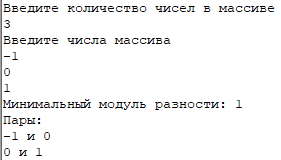
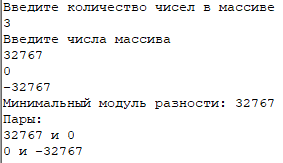
** **

Рисунок 2- Результаты выполнения программы

1. **Формулировка задания 2. Вариант 8**

**Постановка задачи**

Написать программу, определяющую какое из слов встречается в заданной строке чаще всего. Запрещается использовать стандартные функции для работы со строками (за исключением функции определения длины строки)

1. **Описание алгоритм**
2. Задаём массив **arr** для строк.
3. Задаём пустую строку **s** и строку **cyrillic**, которая состоит из строчных букв кириллицы (кроме ё).
4. Устанавливаем цикл с параметром **i**: изначальное значение **i** = 0. Установить ограничение для выполнения цикла <= length**(input)** (длина введённой строки).
5. В теле цикла 1 устанавливаем оператор выбора case:
   1. Если номер символа **s[i]** в кодировке ASCII соответствует номеру строчной латинской или кириллической букв или пробелу, то эти символы присоединяем к строке **s**.
   2. Если номер символа **s[i]** в кодировке ASCII соответствует номеру заглавной латинской букве, то к строке **s** присоединяется символ с номером в кодировке ASCII:

номер нового символа = номер изначального символа + 32.

* 1. Если номер символа **s[i]** в кодировке ASCII соответствует номеру заглавной кириллической букве (кроме Ё), то к строке **s** присоединяется символ из строки **cyrillic** с индексом:

индекс нужного символа = номер изначального символа по кодировке ASCII – 191.

* 1. Если номер символа **s[i]** в кодировке ASCII соответствует номеру заглавной букве «Ё», то к строке **s** присоединяется строчная буква «ё».

1. После тела цикла 1 увеличиваем **i** на 1.
2. Если на конце строки **s** нет пробела, то присоединяем к концу строки пробел.
3. Устанавливаем переменную **num** и присваиваем ему значение 1.
4. Устанавливаем пустую строку **wor**.
5. Устанавливаем цикл 2 с параметром **i**: изначальное значение **i** = 1. Установить ограничение для выполнения цикла <= **length(s)** (длине новой строки).
6. В теле цикла 2 устанавливаем условие: если **s[i]** не равно пробелу, то к строке **wor** присоединяем **s[i]**. В ином случае:
   1. Присваиваем элементу **arr[num]** значение **wor**;
   2. Присваиваем **wor** значение пустой строки;
   3. Увеличиваем **num** на 1.
7. После тела цикла увеличиваем **i** на 1.
8. Устанавливаем переменную **max** и присваиваем ему значение 0.
9. Устанавливаем цикл 3 с параметром **a**: изначальное значение **a** = 1. Установить ограничение для выполнения цикла <= **num**-1.
10. В теле цикла 3:
    1. Устанавливаем переменную **count** и присваиваем ему значение 0;
    2. Устанавливаем цикл 4 с параметром **b**: изначальное значение **b**= **a**. Установить ограничение для выполнения цикла <= **num**-1.
    3. В теле цикла 4 устанавливаем условие: если **arr[b]** = **arr[a]**, то увеличиваем **count** на 1. После тела цикла 4 увеличиваем **b** на 1.
    4. Устанавливаем условие: если **max** меньше **count**, то присваиваем **max** значение **count**.
11. После тела цикла 3 увеличиваем **a** на 1.
12. Устанавливаем цикл 5 с параметром **a**: изначальное значение **a** = 1. Установить ограничение для выполнения цикла <= **num**-1.
13. В теле цикла 5:
    1. Устанавливаем переменную **count** и присваиваем ему значение 0;
    2. Устанавливаем цикл 6 с параметром **b**: изначальное значение **b**= **a**. Установить ограничение для выполнения цикла <= **num**-1.
    3. В теле цикла 6 устанавливаем условие: если **arr[b]** = **arr[a]**, то увеличиваем **count** на 1. После тела цикла 6 увеличиваем **b** на 1.
    4. Устанавливаем условие: если **max** равно **count**, то выводим **arr[a]**.
14. После тела цикла 5 увеличиваем **a** на 1.
15. **Схема алгоритма с комментариями**

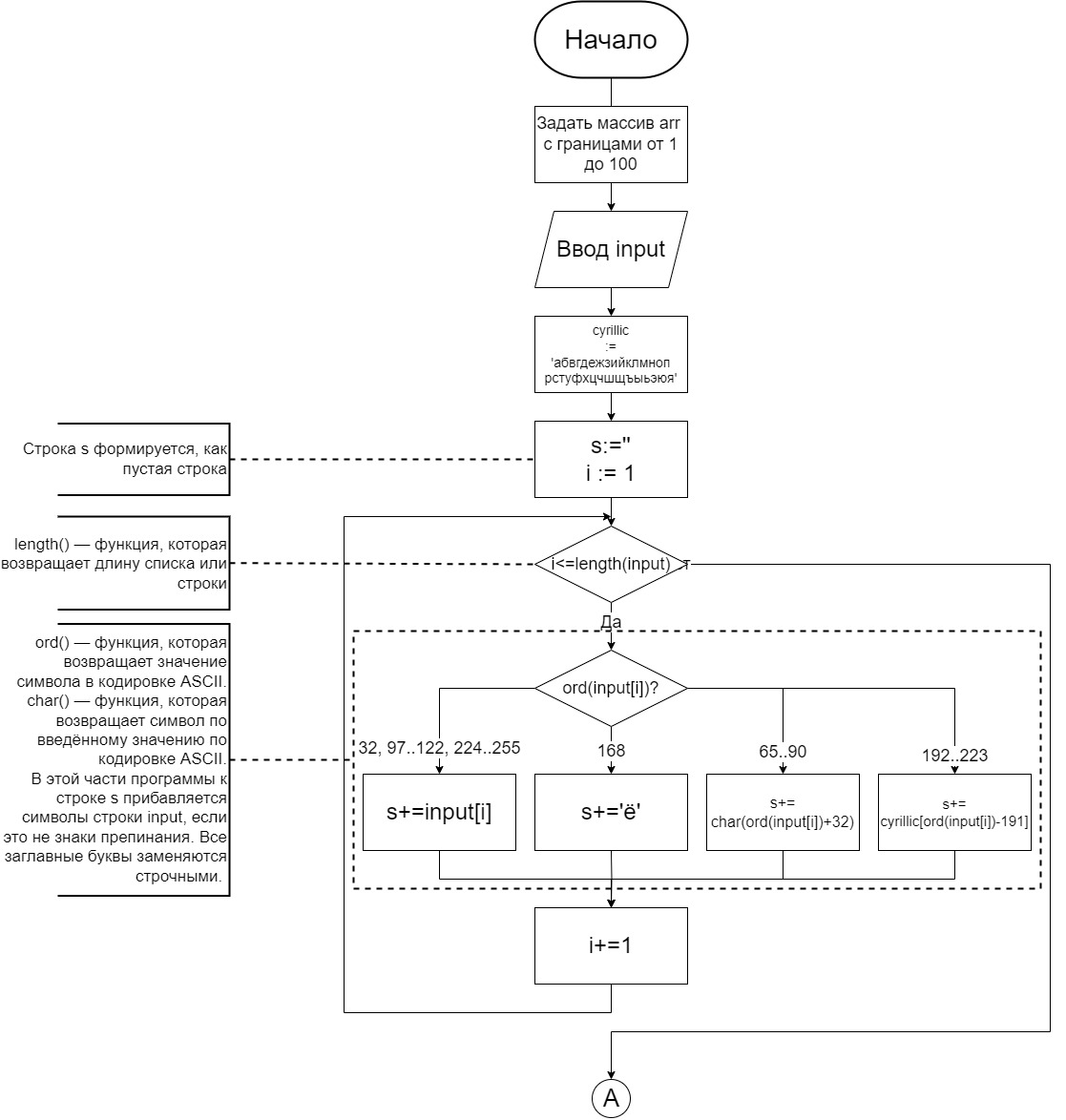


Рисунок 3.1 - Алгоритм решения

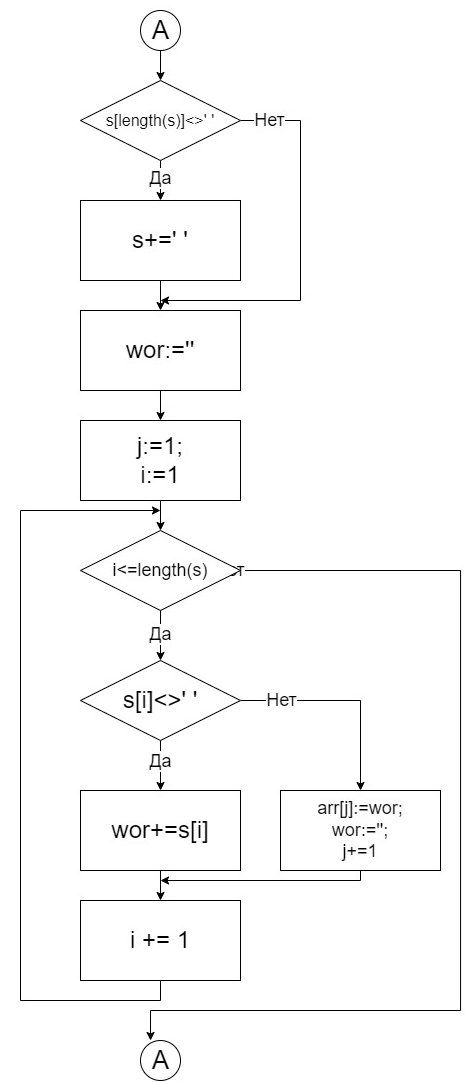


Рисунок 3.2 - Алгоритм решения

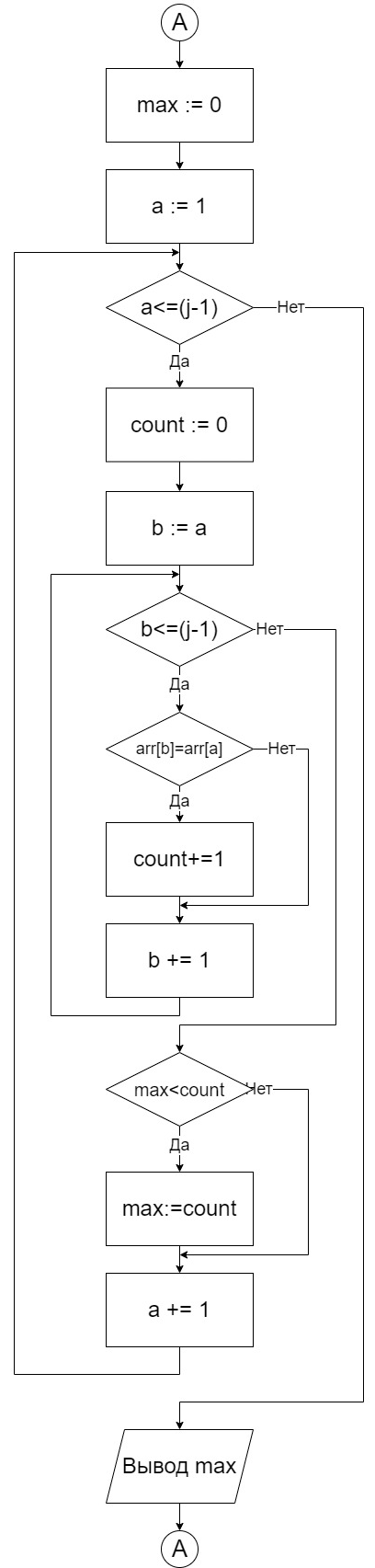


Рисунок 3.3 - Алгоритм решения

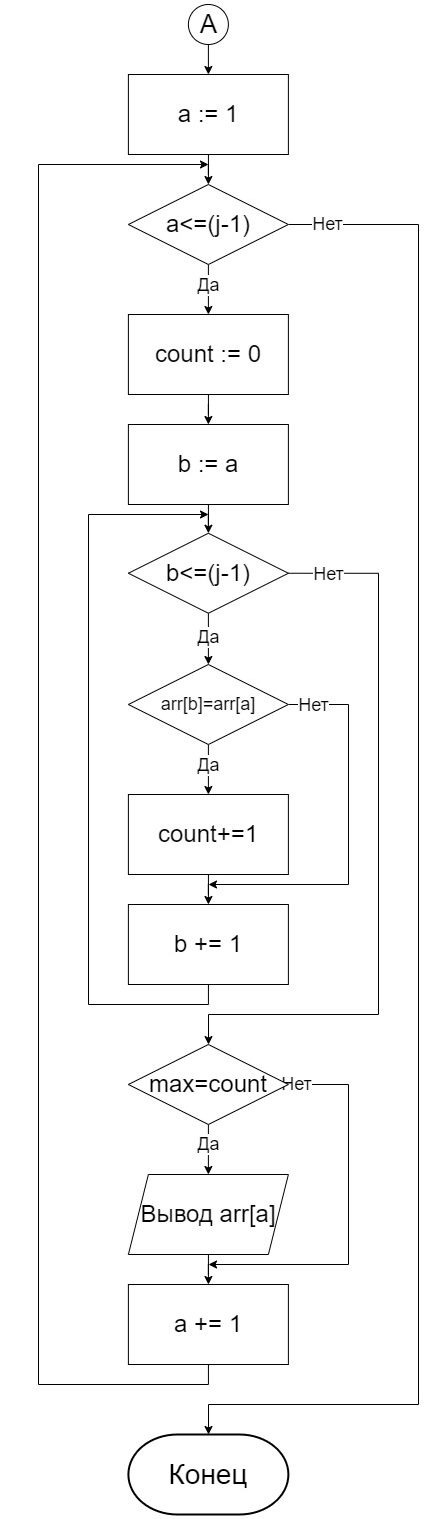


Рисунок 3.4 - Алгоритм решения

1. **Код программы**

**var**

input, s, wor, cyrillic: string;

count, max, i, num, a, b: integer;

arr: **array**[1..100] **of** string;

**begin**

readln(input);

s:='';

cyrillic:='абвгдежзийклмнопрстуфхцчшщъыьэюя';

**for** i:=1 **to** length(input) **do**

**begin**

**case** ord(input[i]) **of**

32, 97..122, 224..255: s+=input[i];

65..90: s+=char(ord(input[i])+32);

192..223: s+=cyrillic[ord(input[i])-191];

168: s+='ё';

**end**

**end**;

**if** s[length(s)]<>' ' **then** s+=' ';

wor:='';

num:=1;

**for** i:=1 **to** length(s) **do**

**begin**

**if** s[i]<>' ' **then** wor+=s[i]

**else**

**begin**

arr[num]:=wor;

wor:='';

num+=1

**end**

**end**;

max:=0;

**for** a:=1 **to** (num-1) **do**

**begin**

count:=0;

**for** b:=a **to** (num-1) **do if** arr[b]=arr[a] **then** count+=1;

**if** max<count **then**

**begin**

max:=count;

**end**

**end**;

writeln('Кол-во повторов чаще встречающихся слов: ', max);

writeln('Слова:');

**for** a:=1 **to** (num-1) **do**

**begin**

count:=0;

**for** b:=a **to** (num-1) **do if** arr[b]=arr[a] **then** count+=1;

**if** count=max **then**

**begin**

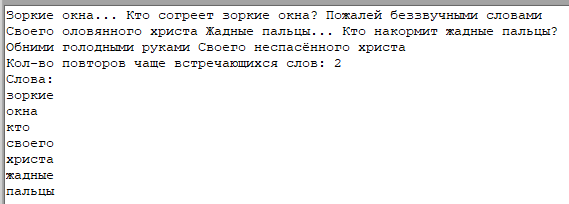
writeln(arr[a])

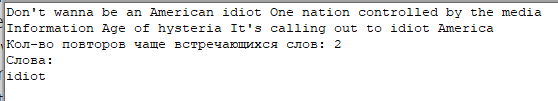
**end**

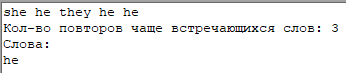
**end**;

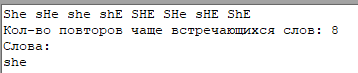
**end**.

1. **Результат выполнения программы**

****

****

****

****

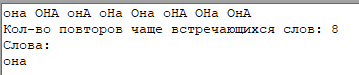
****

Рисунок 4 - Результаты выполнения программы

1. **Вывод**

В процессе работы над домашней контрольной мы изучили базовые навыки работы с одномерными массивами и освоили принципы работы со строками как с частным случаем одномерных массивов.

Для создания алгоритмических схем мы освоили правильное построение письменных алгоритмов, а также научились представлять их визуально с помощью сервиса draw.io — бесплатного онлайн-инструмента для создания и совместного редактирования диаграмм и схем.

В ходе работы мы столкнулись с некоторыми трудностями. Например, в Pascal для кодировки символов используется ASCII, который не поддерживает кириллицу, из-за чего, хоть при вводе кириллической буквы в функцию ord (возвращает номер ASCII для заданного символа) будет возвращаться определённый номер, эти буквы не получится вывести через функцию char (возвращает символ по заданному номеру ASCII). Решением этой проблемы стало вынесение кириллицы в отдельную переменную и обращением в коде по индексу к этой переменной для получения определённых букв.

В итоге, благодаря полученным знаниям и устранённым ошибкам, мы смогли успешно решить задачу, применив код с условными и циклическими конструкциями, а также расширили наши знания о языке программирования Pascal.