Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Основы алгоритмизации и программирования (ОАиП)

ОТЧЕТ

по лабораторной работе №7(отчёт 2)

Тема работы: Работа со строками

Выполнил

студент: гр. 151003 Матошко И.В

Проверил: Фадеева Е.П.

Минск 2021

содержание

[1 Постановка задачи 3](#_Toc91271420)

[2 Текстовый алгоритм решения задачи 4](#_Toc91271421)

[3 Структура данных 7](#_Toc91271422)

[4 Схема алгоритма решения задачи по ГОСТ 19.701-90 8](#_Toc91271423)

[5 Результаты расчетов 13](#_Toc91271424)

[ПРИЛОЖЕНИЕ А 14](#_Toc91271425)

[ПРИЛОЖЕНИЕ Б 20](#_Toc91271426)

# Постановка задачи

**1.1 Первоначальная постановка**

Дана последовательность, содержащая от 2 до 50 слов, в каждом из которых от 1 до 8строчных латинских букв; между соседними словами – запятая или не менее одного пробела, за последним словом точка.

Напечатать те слова последовательности, которые отличны от последнего слова и удовлетворяют следующему свойству: каждая буква входит в слово не менее 3 раз.

**1.2 Окончательная постановка**

Дана последовательность, содержащая от 2 до 50 слов, в каждом из которых от 1 до 8строчных латинских букв; между соседними словами – запятая или не менее одного пробела, за последним словом точка. Удалить из введенной строки лишние запятые и пробелы.

Напечатать те слова последовательности, которые отличны от последнего слова и удовлетворяют следующему свойству: каждая буква входит в слово не менее 3 раз.

# Текстовый алгоритм решения задачи

Таблица 1 ­­– Алгоритм решения

|  |  |
| --- | --- |
| Номер  Шага | Назначение шага |
|  | Ввод Str |
|  | i := 1 |
|  | Начало цикла А1. Проверка выполнения условия (i<= length(Str)). Если условие истинно, идти к шагу 4, иначе – к шагу 10 |
|  | Проверка выполнения условия ((Str[i] = Str[i + 1]) and (Str[i] = ',')). Если условие истинно, идти к шагу 5, иначе – к шагу8 |
|  | Начало цикла Б1. Идти к шагу 6 |
|  | delete(Str, i + 1, 1) |
|  | Конец цикла Б1. Проверка выполнения условия (Str[i]<>Str[i+1]). Если условие истинно, идти к шагу 8, иначе – к шагу 5 |
|  | i :=i+1 |
|  | Конец цикла А1. Идти к шагу 3 |
|  | Начало цикла Б2. Идти к шагу 11 |
|  | Проверка выполнения условия (Str[length(Str) - 1] = ' '). Если условие истинно, идти к шагу 12, иначе – к шагу 13 |
|  | delete(Str, length(Str) - 1, 1) |
|  | Конец цикла Б2. Проверка выполнения условия (Str[length(Str) - 1] <> ' '). Если условие истинно, идти к шагу 14, иначе – к шагу 10 |
|  | Temp := '' |
|  | i := length(Str) - 1 |
|  | Начало цикла А2. Проверка выполнения условия ((Str[i] <> ',') and (Str[i] <> ' ')). Если условие истинно, идти к шагу 17, иначе – к шагу 20 |
|  | Temp := Temp + copy(Str, i, 1) |
|  | dec(i) |
|  | Конец цикла А2. Идти к шагу 16 |
|  | LastWord := '' |
|  | i := 1 |
|  | Начало цикла А3. Проверка выполнения условия (i<= length(Temp)). Если условие истинно, идти к шагу 23, иначе – к шагу 26 |
|  | LastWord := Temp[i] + LastWord |
|  | i :=i+1 |
|  | Конец цикла А3. Идти к шагу 22 |
|  | Вывод Str |
|  | Вывод LastWord |
|  | i := 1 |

Таблица 2 – Продолжение таблицы 1

|  |  |
| --- | --- |
|  | counterArr := 1 |
|  | Начало цикла А4. Проверка выполнения условия (Str[i] <> '.'). Если условие истинно, идти к шагу 31, иначе – к шагу 66 |
|  | Проверка выполнения условия ((Str[i] <> ' ') and (Str[i] <> ',')). Если условие истинно, идти к шагу 32, иначе – к шагу 64 |
|  | Checker := true |
|  | Word := '' |
|  | Начало цикла А5. Проверка выполнения условия ((Str[i] <> ' ') and (Str[i] <> ',') and (i < length(Str))). Если условие истинно, идти к шагу 35, иначе – к шагу 38 |
|  | Word := Word + Str[i] |
|  | inc(i) |
|  | Конец цикла А5. Идти к шагу 34 |
|  | l := 1 |
|  | Начало цикла А6. Проверка выполнения условия (l<=8). Если условие истинно, идти к шагу 40, иначе – к шагу 43 |
|  | BoolMas[l] := true |
|  | l:=l+1 |
|  | Конец цикла А6. Идти к шагу 39 |
|  | j := 1 |
|  | checkForK := true |
|  | Начало цикла А7. Проверка выполнения условия ((j <= length(Word)) and (checkForK = true)). Если условие истинно, идти к шагу 46, иначе – к шагу 60 |
|  | Проверка выполнения условия (BoolMas[j] = true). Если условие истинно, идти к шагу 47, иначе – к шагу 58 |
|  | Letter := copy(Word, j, 1) |
|  | k := 1 |
|  | l := j + 1 |
|  | Начало цикла А8. Проверка выполнения условия (l<= length(Word)). Если условие истинно, идти к шагу 51, иначе – к шагу 56 |
|  | Проверка выполнения условия (Letter = Word[l]). Если условие истинно, идти к шагу 52, иначе – к шагу 54 |
|  | k := k + 1 |
|  | BoolMas[l] := false |
|  | l:=l+1 |
|  | Конец цикла А8. Идти к шагу 50 |
|  | Проверка выполнения условия (k<3). Если условие истинно, идти к шагу 57, иначе – к шагу 58 |
|  | checkForK := false |
|  | j := j + 1 |

Таблица 3 – Продолжение таблицы 2

|  |  |
| --- | --- |
|  | Конец цикла А7. Идти к шагу 45 |
|  | Проверка выполнения условия ((Word <> LastWord) and (checkForK = true)). Если условие истинно, идти к шагу 61, иначе – к шагу 63 |
|  | WordArr[counterArr] := Word |
|  | counterArr := counterArr + 1 |
|  | Идти к шагу 65 |
|  | inc(i) |
|  | Конец цикла А4. Идти к шагу 30 |
|  | Вывод WordArr[1..counterArr-1] |
|  | Останов. |

# Структура данных

Таблица 4 – Данные

|  |  |  |
| --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение |
| BoolMas | Array[1..8] of boolean | Логический массив для "удаления" отмеченных букв в слове |
| Str | String | Исходная строка |
| LastWord | String | Последнее слово в строке |
| Word | String | Каждое последующее проверяемое слово в строке |
| Letter | String | Буква проверяемого слова |
| Temp | String | Хранит «палиндром» последнего слова |
| i | Integer | Счетчик цикла |
| j | Integer | Счетчик цикла |
| l | Integer | Счетчик цикла |
| k | Integer | Количество повторений каждой буквы в слове |
| Checker | Boolean | Логическая переменная для проверки допустимых символов в строке |
| checkForK | Boolean | Логическая переменная для остановки проверки слова при кол-ве повторений буквы меньше 3 |
| counterArr | Integer | Счетчик цикла |
| WordArr | Array[1..50] of string | Массив выводимых слов |

# Схема алгоритма решения задачи по ГОСТ 19.701-90



Рисунок 1– Схема алгоритма решения задачи по ГОСТ 19.701-90



Рисунок 2 – Схема алгоритма решения задачи по ГОСТ 19.701-90



Рисунок 3 – Схема алгоритма решения задачи по ГОСТ 19.701-90



Рисунок 4 – Схема алгоритма решения задачи по ГОСТ 19.701-90



Рисунок 5 – Схема алгоритма решения задачи по ГОСТ 19.701-90

# Результаты расчетов

Вследствие выполнения программы на экран выводятся следующие результаты:

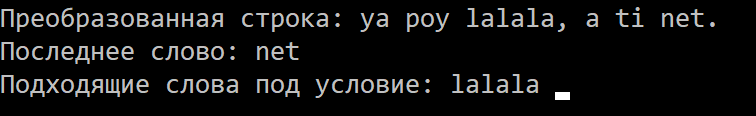


Рисунок 6 – Результаты расчетов

ПРИЛОЖЕНИЕ А

(обязательное)

Исходный код программы

Program lab7R2;

{ A sequence is given containing from 2 to 50 words,

each of which has from 1 to 8 lowercase Latin letters;

there is a comma or at least one space between adjacent

words, followed by a period after the last word. Remove

extra commas or spaces from the entered line.

Print those words of the sequence that are different from

the last word and satisfy the following property: each letter

enters the word at least 3 times. }

{$APPTYPE CONSOLE} // Console app

// Modules declaration

Uses

System.SysUtils;

// Variables declaration

Var

Str, LastWord: string;

Word, Letter, Temp: string;

i, j, k, l, counterArr: integer;

Checker: boolean;

BoolMas: array [1 .. 8] of boolean;

checkForK: boolean;

WordArr: array [1 .. 50] of string;

{ Str - original string;

LastWord - the last word;

Word - current checking word;

Letter - 1-st, 2-nd, etc. letter of Word;

Temp - variable, which contains palindrom of LastWord;

i - cycle counter;

j - cycle counter;

l - cycle counter;

counterArr - cycle counter for array;

k - the number of repetitions of the letter;

Checker - variable, which is used to find mistakes and

to delete words withrepeating letters;

BoolMas - Logical array for "deleting" checked letters

in a word;

checkForK - variable to check the number of

repetitions of letters in a word;

WordArr - an array that stores all matching words. }

// Start the program

Begin

// Mistakes checker

repeat

// Set checker to the original value

Checker := true;

// Input String

write('Введите вашу строку: ');

readln(Str);

// Check condition of only english letters, allowing

// input spaces and сommas

if length(Str) <> 0 then

begin

for i := 1 to length(Str) - 1 do

begin

if ((ord(Str[i]) < 96) or (ord(Str[i]) > 125)) and

(Str[i] <> ' ') and (Str[i] <> ',') then

Checker := false;

end;

end

// If nothing was inputted

else

Str := ' '; (\* In order to make insufficient input \*)

// If mistake was found

if (Str[1] = ',') or (Str = '') or

((pos(' ', Str) = 0) and (pos(',', Str) = 0)) or

(Checker = false) or (Str[length(Str)] <> '.') then

writeln('Ошибка ввода');

// If input is sufficient

until (Str[1] <> ',') and (Checker = true) and

(Str[length(Str)] = '.') and (Str <> '') and

((pos(' ', Str) <> 0) or (pos(',', Str) <> 0));

// Removing extra commas

for i := 1 to length(Str) do

begin

if (Str[i] = Str[i + 1]) and (Str[i] = ',') then

begin

repeat

delete(Str, i + 1, 1);

until Str[i] <> Str[i + 1];

end;

end;

// Removing extra spaces before the dot

repeat

if Str[length(Str) - 1] = ' ' then

delete(Str, length(Str) - 1, 1);

until (Str[length(Str) - 1] <> ' ');

// Set Temp to the original value

Temp := '';

// Set i to the original value

i := length(Str) - 1;

// Find palindrom of the last word

while (Str[i] <> ',') and (Str[i] <> ' ') do

begin

Temp := Temp + copy(Str, i, 1);

dec(i);

end;

// Set LastWord to the original value

LastWord := '';

// Write palindrom of the last word in reverse order

for i := 1 to length(Temp) do

LastWord := Temp[i] + LastWord;

// Output original string and last word

writeln('Преобразованная строка: ', Str);

writeln('Последнее слово: ', LastWord);

write('Подходящие слова под условие: ');

// Set i and counterArr to the original value

i := 1;

counterArr := 1;

// While Str[i] not equals '.'

while Str[i] <> '.' do

begin

// If new word was found

if (Str[i] <> ' ') and (Str[i] <> ',') then

begin

// Set checker to the original value

Checker := true;

// Set Word to the original value

Word := '';

// Find Word

while (Str[i] <> ' ') and (Str[i] <> ',') and (i <

length(Str)) do

begin

Word := Word + Str[i];

inc(i);

end;

// Boolean massive for removing checking letters from

// Word

for l := 1 to 8 do

BoolMas[l] := true;

// Check if there are repeated letters in Word

j := 1;

// Set checkForK to the original value

checkForK := true;

while (j <= length(Word)) and (checkForK = true) do

begin

// If this letter hasn't been there yet

if BoolMas[j] = true then

begin

// Copy 1-st, 2-nd, ... letter of the word

Letter := copy(Word, j, 1);

// Number of repetitions of a new letter

k := 1;

// Search for repetitions of letters and their

//"removal"

for l := j + 1 to length(Word) do

begin

if Letter = Word[l] then

begin

// Increasing the number of repetitions

k := k + 1;

// "Delete" the letter

BoolMas[l] := false;

end;

end;

// Check for at least 3 repetitions of the

// letter in the word, if this is incorrect,

// do not check this word further

if k < 3 then

checkForK := false

end;

j := j + 1;

end;

// If found word is not equals to last word and

// there is no repeated lettters

if (Word <> LastWord) and (checkForK = true) then

begin

// Filling the array for output

WordArr[counterArr] := Word;

counterArr := counterArr + 1;

end;

end

// If we met space or comma

else

inc(i);

end;

// Printing all words from array

for i := 1 to counterArr - 1 do

begin

Write(WordArr[i], ' ');

end;

// Final operations

readln;

End.

ПРИЛОЖЕНИЕ Б

(обязательное)

Тестовые наборы

Тестовый набор 1

«Проверка ввода»

Тест 1

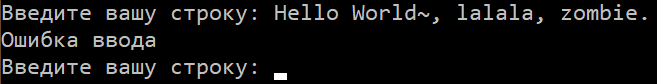
Тестовая ситуация: проверка ввода неподходящего символа в строку

Исходные данные: Str = «Hello World~, lalala, zombie.»

Ожидаемый результат:

Ошибка ввода

Полученный результат:



Тест 2

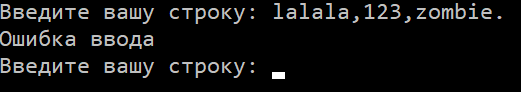
Тестовая ситуация: проверка ввода чисел в поле для строки

Исходные данные: Str = «lalala,123,zombie.»

Ожидаемый результат:

Ошибка ввода

Полученный результат:



Тест 3

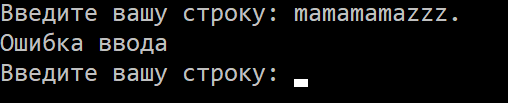
Тестовая ситуация: проверка ввода сплошной строки

Исходные данные: Str = «mamamamazzz.»

Ожидаемый результат:

Ошибка ввода

Полученный результат:



Тест 4

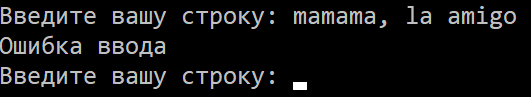
Тестовая ситуация: проверка ввода без точки

Исходные данные: Str = «mamama, la amigo»

Ожидаемый результат:

Ошибка ввода

Полученный результат:



Тест 5

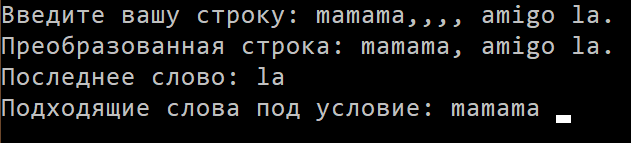
Тестовая ситуация: проверка ввода лишних запятых

Исходные данные: Str = «mamama,,,, amigo la.»

Ожидаемый результат:

Преобразованная строка: mamama, amigo la.

Полученный результат:



Тест 6

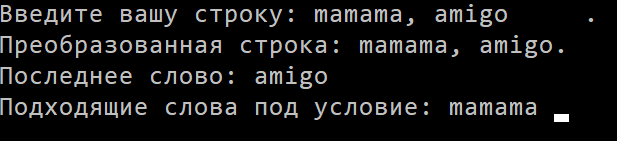
Тестовая ситуация: проверка ввода без точки

Исходные данные: Str = «mamama, amigo .»

Ожидаемый результат:

Преобразованная строка: mamama, amigo.

Полученный результат:



Тестовый набор 2

«Проверка решений»

Тест 1

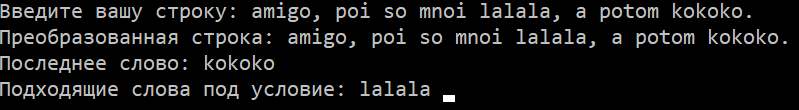
Тестовая ситуация: проверка решений

Исходные данные: Str = «amigo, poi so mnoi lalala, a potom kokoko.»

Ожидаемый результат:

Подходящие слова под условие: lalala

Полученный результат:



Тест 2

Тестовая ситуация: проверка решений

Исходные данные: Str = «amigo, hohoho, noviy god blizko, bobobo hlopushki.»

Ожидаемый результат:

Подходящие слова под условие: hohoho bobobo

Полученный результат:

