Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Кубанский государственный университет»

Факультет компьютерных технологий и прикладной математики

**Кафедра информационных технологий**

**ОТЧЕТ**

**ПО УЧЕБНОЙ ПРАКТИКЕ**

**(практика по получению первичных профессиональных умений и навыков)**

Работу выполнила\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.В. Телятник

студентка ФКТиПМ, 2 курса, 27 гр.,

направление 09.03.03 «Прикладная информатика»

Руководитель учебной практики

доцент кафедры информационных технологий

факультета компьютерных технологий

и прикладной математики

к.ф.-м.н., доцент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.В. Гаркуша

Краснодар 2019

1. ***Условие задачи***

 Даны два предложения. Найти самое короткое из слов первого предложения, которого нет во втором предложении.

1. ***Описание алгоритма решения и структур хранения данных***

В блоке описания констант обозначаем константу D, представляющую собой множество разделителей слов в предложении и константу M, отвечающую за наибольшее количество слов. Опишем также две необходимые подпрограммы. Функция Translate – функция перевода символов в верхний регистр, обрабатывает кириллицу. Формальным параметром является переменная строкового типа aStr. Функция FoundW – функция проверки вхождения в предложение aStr слова aWord. Формальные параметры – переменные строкового типа aStr, aWord. В блоке описания переменных обозначим необходимые переменные (S1,S2,Sw,a,I,Len,LenW,LenWMin,Cnt).

В начале программы на экран выводится сообщение c просьбой ввести первое предложение. Далее вызывается функция Translate и применяется для первого предложения. Аналогично поступаем со вторым предложением. Циклом проходим по длине первого предложения и уточняем длину очередного входящего слова, пропуская разделители. Если длина слова не больше текущей минимальной длины и такого же слова нет во втором предложении, то принимаем его длину, как текущую минимальную и начинаем заполнение массива заново. В противном случае добавляем текущее слово в массив. После выполнения данного алгоритма на экран выводится результат работы программы.

1. ***Описание входных и выходных данных***

На вход два предложения – строкового типа.

Выходные данные представляют одно или несколько слов – самые короткие из слов первого предложения, которых нет во втором предложении. Также для удобства на экран выводится число слов, удовлетворяющих условию поставленной задачи.

1. ***Текст программы***

const

 D = [' ', '.', ',', ':', ';', '!', '?', '-']; //Множество разделителей

 M = 100; //Наибольшее количество искомых слов

function Translate (const aStr : String) : String; //функция перевода символов в

var //верхний регистр, обрабатывает кириллицу

 S : String; //переменная типа String

 i : Integer; //переменная типа Integer

begin

 S := aStr;

 for i := 1 to Length(S) do begin //циклом проходим по длине строки

 case S[i] of

 #$A0..#$AF: S[i] := Char(Ord(S[i]) - $20); //буквы а..п

 #$E0..#$EF: S[i] := Char(Ord(S[i]) - $50); //буквы р..я

 #$F1: S[i] := #$F0; //буква ё

 else S[i] := UpCase(S[i]); //остальные символы

 end;

 end;

 Translate := S;

end;

function FoundW(const aStr, aWord : String) : Boolean; //проверяет - есть ли в

var // тексте aStr слово aWord

 i, Len, LenW : Integer; //переменные типа Integer

begin

 FoundW := False;

 Len := Length(aStr);

 LenW := 0;

 for i := 1 to Len do begin

 if aStr[i] in D then Continue; //пропускаем разделители

 Inc(LenW); //уточнение длины слова

 if (i = Len) or (aStr[i + 1] in D) then begin //отслеживание конца слова

 if aWord = Copy(aStr, i - LenW + 1, LenW) then begin

 FoundW := True;

 Break;

 end;

 LenW := 0; //сброс длины очередного слова

 end;

 end;

end;

//основная программа

var

 S1, S2, Sw : String; //переменные типа String

 a : array[1..M] of String; //массив типа String

 i, Len, LenW, LenWMin, Cnt : Integer; //переменные типа Integer

begin

 repeat

 Writeln('Задайте первую строку:'); //вывод сообщения

 Readln(S1); //ввод первого предложения

 S1 := Translate(S1);//вызов функции перевода

 Writeln('Задайте вторую строку:');//вывод сообщения

 Readln(S2); //ввод второго предложения

 S2 := Translate(S2); //вызов функции перевода

 Len := Length(S1);

 LenW := 0;

 LenWMin := Len + 1;

 Cnt := 0; //количество значимых элементов массива

 for i := 1 to Len do begin

 if S1[i] in D then Continue; //пропускаем разделители

 Inc(LenW); //уточнение длины очередного слова

 if (i = Len) or (S1[i + 1] in D) then begin//отслеживаем конец слова

 Sw := Copy(S1, i - LenW + 1, LenW);

 if (LenW <= LenWMin) and not FoundW(S2, Sw) then begin//если длина //слова не больше текущей мин длины и того же слова

 if LenW < LenWMin then begin //если слово короче ранее найденных

 LenWMin := LenW;//принимаем длину как текущую наименьшую

 Cnt := 1;

 a[Cnt] := Sw;//начинаем заполнение массива

 end else begin//иначе добавляем слово в массив

 Inc(Cnt);

 a[Cnt] := Sw;

 end;

 end;

 LenW := 0; //сброс длины очередного слова

 end;

 end;

 Writeln('Самые короткие слова из первой строки, которых нет во второй строке:'); //вывод сообщения

 Writeln('Количество: ', Cnt); //вывод сообщения

 if Cnt > 0 then begin

 Writeln('Перечень:'); //вывод сообщения

 for i := 1 to Cnt do begin

 if i > 1 then Write(', '); //если найденных слов несколько

 Write(a[i]); //вывод результата

 end;

 Writeln;

 end;

 Writeln('Повторить - Enter. Выход - любой символ + Enter.'); //вывод сообщ

 Readln(S1);

 until S1 <> '';

end.

1. ***Тестовый запуск программы и вывод результатов работы***
2. 
3. 
4. 