Министерство образования Республики Беларусь

Учреждение образования

БелорусскиЙ государственный университет

информатики и радиоэлектроники

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

|  |
| --- |
|  |
|  |

# оТЧЕТ

по лабораторной работе

на тему:

**Изучение строкового типа данных**

|  |  |  |
| --- | --- | --- |
| Выполнил  Студент гр. 351001 |  | А.Д. Матюшенко |
| Проверил |  | Асс. Е.Е. Фадеева |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Минск, 2023

1. Теоретические сведения по теме лабораторной работы

**Алгоритм** – система правил, четко описывающая последовательность действий, которые необходимо выполнить для решения задачи.

**Свойства правильного алгоритма:**

1) Дискретность – значения величин в каждый следующий момент времени должны получаться по определенным правилам из значений величин, имевшихся в предшествующий момент времени.

2) Определенность (детерминированность) – каждое правило алгоритма должно быть однозначным. Значения величин, получаемых в какой-то момент времени, однозначно связаны со значениями величин, вычисленных ранее.

3) Результативность (конечность) – алгоритм должен приводить к решению задачи за конечное число шагов.

4) Массовость – алгоритм должен разрабатываться в общем виде так, чтобы его можно было применить для класса задач, различающихся лишь исходными данными.

В последних версиях Delphi тип string, обозначающий строку, является псевдонимом встроенного типа System.UnicodeString. Т.е. когда вы объявляете переменную str: string, то автоматически вы объявляете переменную типа UnicodeString.

1. Функция length возвращает длину указанной строки. Данная функция обладает одним параметром, который представляет собой выражение, имеющее строковый тип данных. В качестве значения функции length выступает целое число, обозначающее количество символов, входящих в состав строки.
2. Функция роs. С помощью функции pos возможно определение положения подстроки в указанной строке.
3. Функция copy выделяет фрагмент указанной строки.
4. Задание на лабораторную работу
   1. Постановка задачи

**Задание № 62**

Даны натуральное число N и текстовый файл, последовательность компонент которого представляет собой текст на русском языке. Абзац (красная строка) обозначен символом $. Вывести этот текст строками длины N. Абзац начинать с трех пробелов. При форматировании слова разбивать для переноса, учитывая правила, описанные в примечании.

*Примечание:* Разбиение русского слова на части для переноса с одной строки на другую с большой вероятностью выполняется правильно, если пользоваться следующими простыми правилами:

1 – две идущие подряд гласные можно разделить, если первой из них предшествует согласная, а за второй идет хотя бы одна буква (буква й при этом рассматривается вместе с предшествующей гласной как одно целое).

2 - две идущие подряд согласные можно разделить, если первой из них предшествует гласная, а в той части слова, которая идет за второй согласной имеется хотя бы одна гласная (буквы ь, ъ вместе с предшествующей согласной рассматриваются как единое целое).

3 – если не удается применить пункты 1, 2, то следует попытаться разбить слово так, чтобы первая часть содержала более чем одну букву и оканчивалась на гласную, а вторая содержала хотя бы одну гласную.

Вероятность правильного разбиения увеличивается, если предварительно воспользоваться хотя бы неполным списком приставок, содержащих гласные, и попытаться прежде всего выделить из слова такую приставку.

сиреневенький длинношеее

* 1. Эскиз ожидаемого результата

Raw string: qwerty ty ty u ty

Word to delete: "ty"

Replaced the last word string (S1): "qwerty u"

qw: 1

we: 1

er: 1

rt: 1

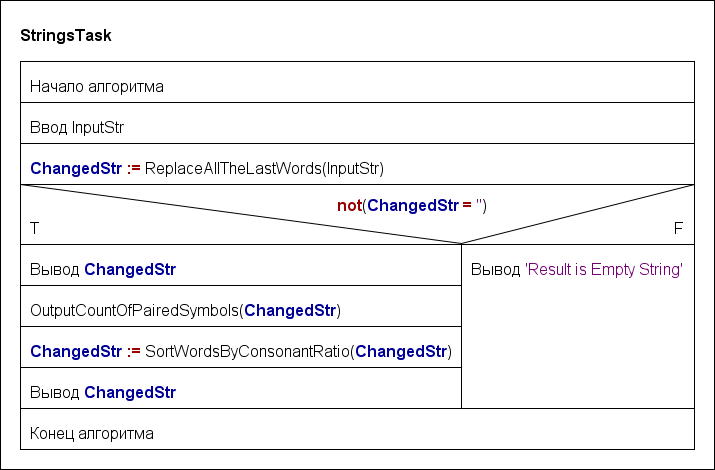
ty: 1

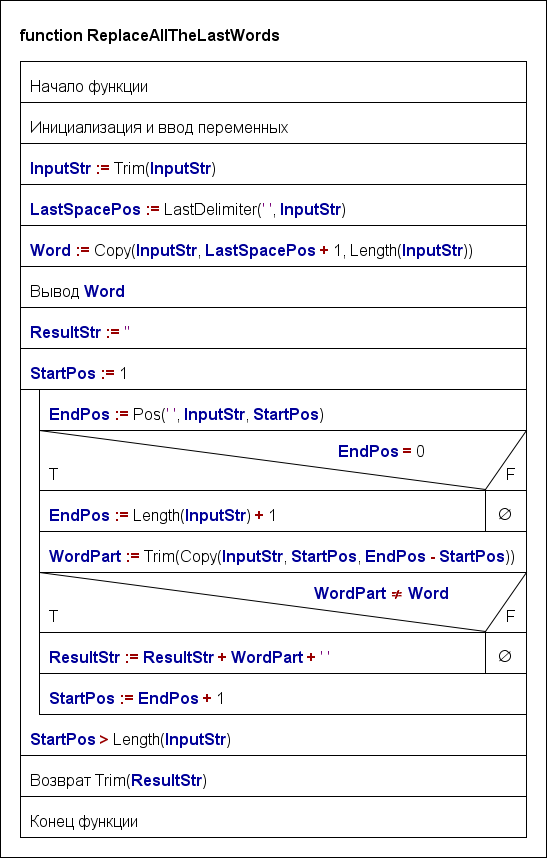
Sorted by consonantr ratio string (S2): "u qwerty"

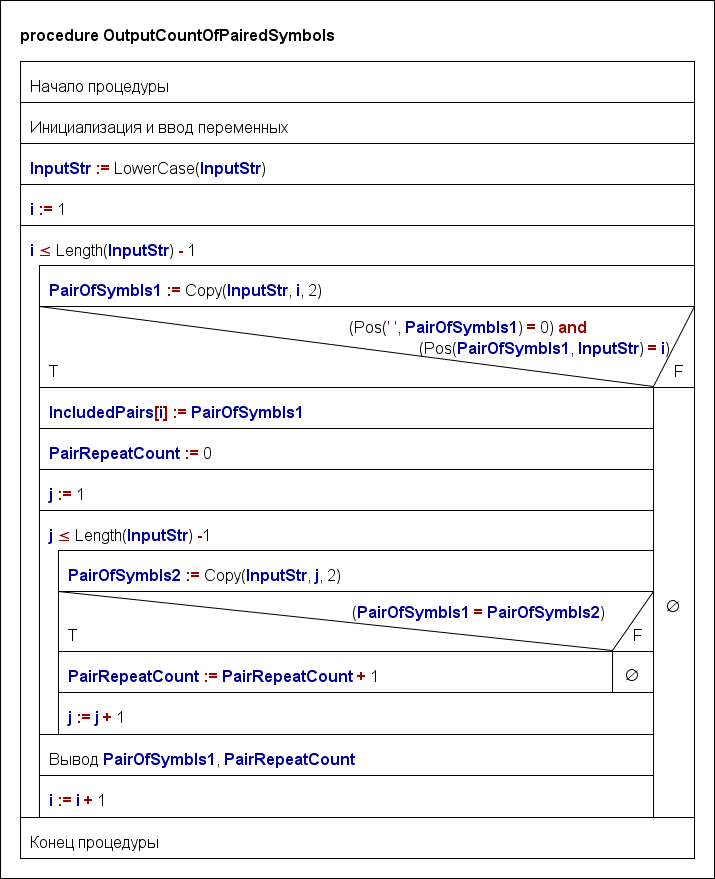
* 1. Разработка алгоритма

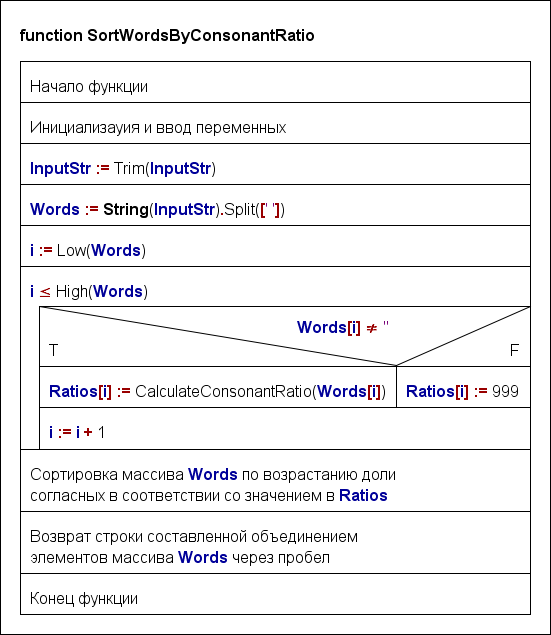
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя идентификатора | назначение | Тип идентификатора | Начальное значение | Закон изменения |
| i | Итерируемая переменная. Индекс массива/строки | integer | – | i := i + 1 |
| j | Итерируемая переменная. Индекс массива/строки | integer | – | j := j + 1 |
| InputStr | Вводимая строка | TString = string[200] | -­ | readln(InputStr) |
| ChangedStr | Измененная строка пункты 1, 2 | TString | -­ | ChangedStr := ReplaceAllTheLastWords(InputStr),  ChangedStr := SortWordsByConsonantRatio(ChangedStr) |
| LastSpacePos | Индекс последнего пробела в строке.  function ReplaceAllTheLastWords | Integer | - | LastSpacePos := LastDelimiter(' ', InputStr) |
| IncludedPairs | Пары символов в строке.  procedure OutputCountOfPairedSymbols | array [1 .. 999] of TString | - | IncludedPairs[i] := PairOfSymbls1 |
| Ratios | Массив с % согласных в слове(строке).  function SortWordsByConsonantRatio | array [0 .. 999] of Real | - | Ratios[i] := CalculateConsonantRatio(Words[i]),  Ratios[i] := 999; // Если пустое слово(пробелы) |

Таблица 1 используемые идентификаторы









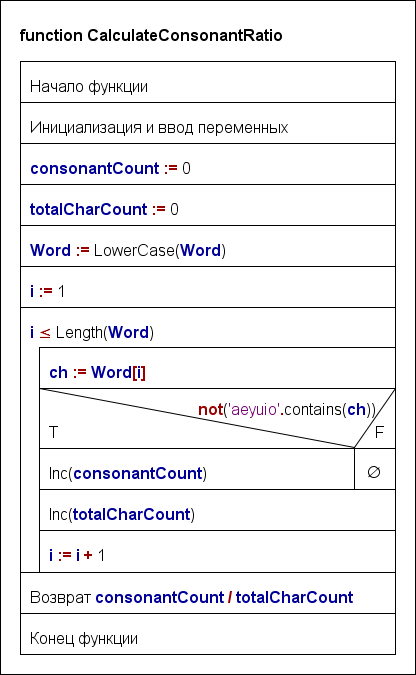


Рисунок 1 – Схема работы программы

Данный код представляет решение задачи, описанной в пунктах 1 и 2.

В пункте 1 алгоритм выполняет следующие шаги:

* Вводится исходная строка InputStr.
* Функция ReplaceAllTheLastWords заменяет все последние слова в строке на пустую строку и возвращает результат в переменную ChangedStr.
* Если ChangedStr не является пустой строкой, выполняются следующие действия:
* Выводится строка после замены последнего слова (S1).
* Вызывается процедура OutputCountOfPairedSymbols, которая подсчитывает количество вхождений каждого двухбуквенного сочетания, встречающегося в строке ChangedStr.

В пункте 2 алгоритм выполняет следующие шаги:

* Вызывается функция SortWordsByConsonantRatio, которая сортирует слова в строке ChangedStr в порядке возрастания доли согласных букв в словах.
* Выводится отсортированная строка (S2).

Используется пользовательский тип TString = String[200] для работы со строками.

Таким образом, алгоритм позволяет выполнить замену последних слов в строке и подсчитать вхождения двухбуквенных сочетаний, а также отсортировать слова в строке по доле согласных букв.

* 1. Текст программы и его описание

program StringsTask;

uses

System.SysUtils;

type

TString = string[200];

var

InputStr, ChangedStr: TString;

function ReplaceAllTheLastWords(InputStr: TString): TString;

var

ResultStr, Word: TString;

LastSpacePos, StartPos, EndPos: integer;

WordPart: TString;

begin

InputStr := Trim(InputStr);

LastSpacePos := LastDelimiter(' ', InputStr);

// Находим слово для удаления по последнему пробелу

Word := Copy(InputStr, LastSpacePos + 1, Length(InputStr));

writeln('Word to delete: "', Word, '"');

ResultStr := '';

StartPos := 1;

repeat

EndPos := Pos(' ', InputStr, StartPos);

if EndPos = 0 then

EndPos := Length(InputStr) + 1;

WordPart := Trim(Copy(InputStr, StartPos, EndPos - StartPos));

if WordPart <> Word then

ResultStr := ResultStr + WordPart + ' ';

StartPos := EndPos + 1;

until StartPos > Length(InputStr);

Result := Trim(ResultStr);

end;

procedure OutputCountOfPairedSymbols(InputStr: TString);

var

IncludedPairs: array [1 .. 999] of TString;

PairOfSymbls1, PairOfSymbls2: TString;

PairRepeatCount: integer;

begin

InputStr := LowerCase(InputStr);

for var i := 1 to Length(InputStr) - 1 do

begin

PairOfSymbls1 := Copy(InputStr, i, 2);

// Если двухбуквенное сочетание не включает пробелы и

// если оно не встречалось ранее (проверка на первое вхождение в строку)

if (Pos(' ', PairOfSymbls1) = 0) and (Pos(PairOfSymbls1, InputStr) = i) then

begin

IncludedPairs[i] := PairOfSymbls1;

PairRepeatCount := 0;

for var j := 1 to Length(InputStr) - 1 do

begin

PairOfSymbls2 := Copy(InputStr, j, 2);

if (PairOfSymbls1 = PairOfSymbls2) then

PairRepeatCount := PairRepeatCount + 1;

end;

writeln(PairOfSymbls1, ': ', PairRepeatCount);

end;

end;

end;

function CalculateConsonantRatio(Word: TString): Real;

var

i, consonantCount, totalCharCount: integer;

ch: AnsiChar;

begin

consonantCount := 0;

totalCharCount := 0;

Word := LowerCase(Word);

for i := 1 to Length(Word) do

begin

ch := Word[i];

if not('aeyuio'.contains(ch)) then

Inc(consonantCount);

Inc(totalCharCount);

end;

Result := consonantCount / totalCharCount;

end;

function SortWordsByConsonantRatio(InputStr: TString): TString;

var

Ratios: array [0 .. 999] of Real;

i, j: integer;

tempRatio: Real;

tempWord: TString;

begin

InputStr := Trim(InputStr);

var

Words := String(InputStr).Split([' ']); // Cтрока преобразуется в массив слов

for i := Low(Words) to High(Words) do

begin

if Words[i] <> '' then

Ratios[i] := CalculateConsonantRatio(Words[i])

else

Ratios[i] := 999; // Если пустое слово(пробелы)

end;

// Сортировка пузырьком.

for i := Low(Words) to High(Words) - 1 do

begin

for j := Low(Words) to High(Words) - i - 1 do

begin

if Ratios[j] > Ratios[j + 1] then

begin

tempRatio := Ratios[j];

Ratios[j] := Ratios[j + 1];

Ratios[j + 1] := tempRatio;

tempWord := Words[j];

Words[j] := Words[j + 1];

Words[j + 1] := tempWord;

end;

end;

end;

Result := string.Join(' ', Words);

end;

begin

write('Raw string: ');

readln(InputStr);

ChangedStr := ReplaceAllTheLastWords(InputStr);

if not(ChangedStr = '') then

begin

writeln('Replaced the last word string (S1): "', ChangedStr, '"');

OutputCountOfPairedSymbols(ChangedStr);

ChangedStr := SortWordsByConsonantRatio(ChangedStr);

writeln('Sorted by consonant ratio string (S2): "', ChangedStr, '"');

end

else

writeln('Result is Empty String');

readln;

end.

* 1. Тестирование и отладка программы

Таблица 3.2 Прохождение тестов программы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Специфика тестирования | Номер теста | Вводимые данные | Ожидаемый результат | Полученный результат |
| Тест нахождения строки s1 | 1 | Raw string: qwert qwert y y rt y | Replaced the last word string (S1): "qwert qwert rt"  qw: 2  we: 2  er: 2  rt: 3  Sorted by consonant ratio string (S2): "qwert qwert rt" | Тест пройден успешно |
| Тест корректности вывода пустой строк | 2 | Raw string: | Result is Empty String | Тест пройден успешно |
|  | 3 | Raw string: qwert | Result is Empty String | Тест пройден успешно |
| Тест корректности отличия слов разного регистра | 4 | Raw string: Qwert rt qwert qwert | Replaced the last word string (S1): "Qwert rt"  qw: 1  we: 1  er: 1  rt: 2  Sorted by consonant ratio string (S2): "Qwert rt" | Тест пройден успешно |
| Тест корректности отличия слов разного регистра (для кол-ва сочетаний двух букв) | 5 | Raw string: Qwert rt qwert qwert q | Replaced the last word string (S1): "Qwert rt qwert qwert"  qw: 3  we: 3  er: 3  rt: 4  Sorted by consonant ratio string (S2): "Qwert qwert qwert rt" | Тест пройден успешно |
| Тест нахождения строки s2 | 6 | Raw string: q w qwe qwwe qwwwe e y | Replaced the last word string (S1): "q w qwe qwwe qwwwe e"  qw: 3  we: 3  ww: 3  Sorted by consonant ratio string (S2): "e qwe qwwe qwwwe q w" | Тест пройден успешно |

Анализ прохождения тестов, причины возникших ошибок, внесение изменений в алгоритм и текст программы

* 1. Итоговый текст программы

program StringsTask;

uses

System.SysUtils;

type

TString = string[200];

var

InputStr, ChangedStr: TString;

function ReplaceAllTheLastWords(InputStr: TString): TString;

var

ResultStr, Word: TString;

LastSpacePos, StartPos, EndPos: integer;

WordPart: TString;

begin

InputStr := Trim(InputStr);

LastSpacePos := LastDelimiter(' ', InputStr);

// Находим слово для удаления по последнему пробелу

Word := Copy(InputStr, LastSpacePos + 1, Length(InputStr));

writeln('Word to delete: "', Word, '"');

ResultStr := '';

StartPos := 1;

repeat

EndPos := Pos(' ', InputStr, StartPos);

if EndPos = 0 then

EndPos := Length(InputStr) + 1;

WordPart := Trim(Copy(InputStr, StartPos, EndPos - StartPos));

if WordPart <> Word then

ResultStr := ResultStr + WordPart + ' ';

StartPos := EndPos + 1;

until StartPos > Length(InputStr);

Result := Trim(ResultStr);

end;

procedure OutputCountOfPairedSymbols(InputStr: TString);

var

IncludedPairs: array [1 .. 999] of TString;

PairOfSymbls1, PairOfSymbls2: TString;

PairRepeatCount: integer;

begin

InputStr := LowerCase(InputStr);

for var i := 1 to Length(InputStr) - 1 do

begin

PairOfSymbls1 := Copy(InputStr, i, 2);

// Если двухбуквенное сочетание не включает пробелы и

// если оно не встречалось ранее (проверка на первое вхождение в строку)

if (Pos(' ', PairOfSymbls1) = 0) and (Pos(PairOfSymbls1, InputStr) = i) then

begin

IncludedPairs[i] := PairOfSymbls1;

PairRepeatCount := 0;

for var j := 1 to Length(InputStr) - 1 do

begin

PairOfSymbls2 := Copy(InputStr, j, 2);

if (PairOfSymbls1 = PairOfSymbls2) then

PairRepeatCount := PairRepeatCount + 1;

end;

writeln(PairOfSymbls1, ': ', PairRepeatCount);

end;

end;

end;

function CalculateConsonantRatio(Word: TString): Real;

var

i, consonantCount, totalCharCount: integer;

ch: AnsiChar;

begin

consonantCount := 0;

totalCharCount := 0;

Word := LowerCase(Word);

for i := 1 to Length(Word) do

begin

ch := Word[i];

if not('aeyuio'.contains(ch)) then

Inc(consonantCount);

Inc(totalCharCount);

end;

Result := consonantCount / totalCharCount;

end;

function SortWordsByConsonantRatio(InputStr: TString): TString;

var

Ratios: array [0 .. 999] of Real;

i, j: integer;

tempRatio: Real;

tempWord: TString;

begin

InputStr := Trim(InputStr);

var

Words := String(InputStr).Split([' ']); // Cтрока преобразуется в массив слов

for i := Low(Words) to High(Words) do

begin

if Words[i] <> '' then

Ratios[i] := CalculateConsonantRatio(Words[i])

else

Ratios[i] := 999; // Если пустое слово(пробелы)

end;

// Сортировка пузырьком.

for i := Low(Words) to High(Words) - 1 do

begin

for j := Low(Words) to High(Words) - i - 1 do

begin

if Ratios[j] > Ratios[j + 1] then

begin

tempRatio := Ratios[j];

Ratios[j] := Ratios[j + 1];

Ratios[j + 1] := tempRatio;

tempWord := Words[j];

Words[j] := Words[j + 1];

Words[j + 1] := tempWord;

end;

end;

end;

Result := string.Join(' ', Words);

end;

begin

write('Raw string: ');

readln(InputStr);

ChangedStr := ReplaceAllTheLastWords(InputStr);

if not(ChangedStr = '') then

begin

writeln('Replaced the last word string (S1): "', ChangedStr, '"');

OutputCountOfPairedSymbols(ChangedStr);

ChangedStr := SortWordsByConsonantRatio(ChangedStr);

writeln('Sorted by consonant ratio string (S2): "', ChangedStr, '"');

end

else

writeln('Result is Empty String');

readln;

end.