**Спецификация к Лабораторной работе №1**

**Сортировка. Символы. Множества**

**Вариант 22**

**Задание**

Упорядочить символы строки, расположив символы в указанном в нижеследующей таблице порядке (по вариантам) двумя способами:

- с созданием своей таблицы символов и

- без ее создания (со сложным условием сравнения и множествами).

Провести анализ аномалий: строка не пустая и других символов (кроме указанных) нет.

При выводе результата в начало строки и в конец добавить символ " (двойная кавычка) для обозначения начала и конца строки, что особенно актуально для строк из пробелов и других непечатных символов (s:= +<нужный символ>+s+<нужный символ или строка>).

***Указание 1****.* Для проверки и сортировки использовать процедуры, расположенные в отдельном модуле.

***Указание 2****.* Для проверки принадлежности символа к одной из трёх групп использовать множества.

***Указание 3****.* Изменить (упорядочить) заданную строку по указанному в каждом варианте правилу «на месте» без создания дополнительной строки (или массива, файла и т.д.).

**1. Условие задачи**

Упорядочить строку, расположив символы в указанном в таблице порядке, убедившись, что строка не пустая и других символов (кроме указанных) нет.

|  |  |  |
| --- | --- | --- |
| **Сначала идут символы** | **Затем символы** | **А в конце – символы** |
| Кириллицы строчные в исходно заданном порядке, включая ё | Кириллицы ЗАГЛАВНЫЕ в алфавитном порядке (А..ЕЁ..Я) | Цифры в исходно заданном порядке |

**2. Уточненное условие задачи**

Задана строка S.

Проверить, что строка S не пустая и состоит только из цифр, заглавных и строчных букв кириллицы:

- если строка пустая - вывести сообщение «Пустая строка»,

- если присутствуют иные символы - вывести сообщение «Некорректные символы»,

- в противном случае, расположить символы строки в указанном в таблице порядке и вывести результат на экран.

**3. Пример**

Пусть задана строка Тв37ОрЛД990йЦу.

Строка непустая и в ней нет запрещенных символов.

После сортировки получаем строку: врйуДЛОТЦ37990.

*# python*

import re

test\_strings = [u'',u'iJщюкQ4',u'01234',u'Ау',u'ЁЖИКВТУМАНЕ',u'StartEnd',u'adressADRESS',

u'358морозИсолнцеДЕНЬчудестный123',u'123FirstSecondThirdОЮИ']

symbol\_table = u'абвгдеёжзийклмнопрстуфхцчшщъыьэюяАБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ0123456789'

scyrillic\_letters,bcyrillic\_letters,figures = ur'[а-я,ё]',ur'[А-Я,Ё]',ur'[0-9]'

bcyrillic\_letters1,bcyrillic\_letters2 = ur'[А-Е]',ur'[Ж-Я]'

# combined functions

def prov\_sort1(s):

s1 = ''.join(re.findall(scyrillic\_letters,s,re.U))+\

''.join(sorted(re.findall(bcyrillic\_letters1,s,re.U)))+\

''.join(re.findall(ur'[Ё]',s,re.U))+\

''.join(sorted(re.findall(bcyrillic\_letters2,s,re.U)))+\

''.join(re.findall(figures,s,re.U))

if len(s)==0: return 'Пустая строка'

elif len(s1)!=len(s): return 'Некорректные символы'

else: return '"'+s1+'"'

def prov\_sort2(s):

s1 = ''.join([s[i] for i in range(len(s)) if s[i] in symbol\_table[:33]])

s2 = ''.join([s[i] for i in range(len(s)) if s[i] in symbol\_table[33:66]])

s3 = ''.join([s[i] for i in range(len(s)) if s[i] in symbol\_table[66:]])

if len(s)==0: return 'Пустая строка'

elif len(s1+s2+s3)!=len(s): return 'Некорректные символы'

else:

m=1; n=len(s2); s2list=list(s2); flag=False

while not (flag or m==n):

flag=True

for i in range(n-m):

if (symbol\_table.find(s2[i])>symbol\_table.find(s2[i+1])):

ch=s2list[i]; s2list[i]=s2list[i+1]; s2list[i+1]=ch;

s2 = ''.join(s2list); flag=False

m+=1

return '"'+s1+s2+s3+'"'

*# variant 2*

def prov\_sort2\_2(s):

s1 = ''.join([s[i] for i in range(len(s)) if s[i] in symbol\_table[:33]])

s2 = ''.join(sorted([s[i] for i in range(len(s)) if s[i] in symbol\_table[33:39]]))

s3 = ''.join([s[i] for i in range(len(s)) if s[i] == symbol\_table[39]])

s4 = ''.join(sorted([s[i] for i in range(len(s)) if s[i] in symbol\_table[40:66]]))

s5 = ''.join([s[i] for i in range(len(s)) if s[i] in symbol\_table[66:]])

if len(s)==0: return 'Пустая строка'

elif len(s1+s2+s3+s4+s5)!=len(s): return 'Некорректные символы'

else: return '"'+s1+s2+s3+s4+s5+'"'

**4. Таблица данных**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Класс** | **Имя** | **Описание (смысл)** | **Тип** | **Структура** | **Формат** |
| исходные | S | исходная строка | символьный | строка |  |
| выходные | S1 | отсортированная строка без своей таблицы символов | символьный | строка |  |
| выходные | S2 | отсортированная строка со своей таблицей символов | символьный | строка |  |
| промежуточные | N | номер аномальной ситуации (1 или 2), или 0, если нет аномалий | цел | простая переменная |  |

**5,6. Форма ввода/вывода**

|  |  |
| --- | --- |
| **Форма ввода** | **Форма вывода** |
| Введите строку <S> | Строка до сортировки <S1> |
|  | Отсортированная 1-м способом строка <S1> |
|  | Строка до сортировки <S2> |
|  | Отсортированная 2-м способом строка <S2> |

**7. Аномалии**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Описание** | **Условие** | **Ожидаемая реакция** |
| 1 | Пустая строка S | S = '' | Пустая строка |
| 2 | Есть запрещенные символы | Есть символ S[i],  для которого НЕ выполняется условие  (S[i] in ['а'..'я','ё']) or  (S[i] in ['А'..'ЕЁ'..'Я']) or  (S[i] in ['0'..'9']) | Некорректные символы |

**8. Функциональные тесты**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Исходные данные** | **Ожидаемый результат** | **Смысл теста** |
| 1 | S = ‘’ | ‘Пустая строка’ | Аномалия 1  Пустая строка |
| 2 | S = ‘iJщюкQ4’ | ‘Некорректные символы’ | Аномалия 2  Есть запрещенные символы |
| 3 | S = ‘01234’ | S = ‘01234’ | Есть только символы *третьей части*, и они выводятся в исходном порядке |
| 4 | S = 'Ау' | S = 'уА' | Есть только символы *первой и второй части*, из *первой* части выводятся раньше, чем из *второй* |
| 5 | S = 'ЁЖИКВТУМАНЕ' | S = 'АВЕЁЖИКМНТУ' | Есть только символы *второй* части, и они выводятся в алфавитном порядке |
| 6 | S = 'StartEnd' | ‘Некорректные символы’ | Аномалия 2 Есть запрещенные символы |
| 7 | S = 'adressADRESS' | ‘Некорректные символы’ | Аномалия 2 Есть запрещенные символы |
| 8 | S = '358морозИсолнцеДЕНЬ  чудестный123' | S = 'морозсолнцечудестный  ДЕИНЬ358123' | Есть символы всех видов, выводятся в заданном порядке |
| 9 | S = '123FirstSecondThirdОЮИ' | ‘Некорректные символы’ | Аномалия 2 Есть запрещенные символы |

**9. Метод**

1. Введем строку (ввод с клавиатуры в режиме диалога), создадим две копии исходной строки – S1 и S2.

Для каждого из двух методов выполним пункты 2-4:

2. Проверим аномалии. Для проверки аномалий напишем функции Prov1 и Prov2 (без и со своей таблицей символов), возвращающие номер аномальной ситуации N.

3. В случае корректной строки – отсортируем ее по указанному правилу с помощью процедур Sort1 и Sort2. Сортировать будем «пузырьком», указав ВСЕ случаи, когда два СОСЕДНИХ элемента стоят в НЕправильном порядке.

4. Выведем сообщение или отсортированную строку на экран.

**10. Основной алгоритм**

начало

Ввод S

А0.1 Проверка строки

Nom:=Prov1(S1)

Вх: S1

Вых: N

N=?

Пустая строка

Некорр. символы

Вывод S1

А0.2 Сортировка

Sort1(S1)

Вх: S1

Вых: S1

конец

N = 1

N = 2

+

–

А0.1 Проверка строки

Nom:=Prov2(S2)

Вх: S2

Вых: N

N=?

Пустая строка

Некорр. символы

Вывод S2

А0.2 Сортировка

Sort2(S2)

Вх: S2

Вых: S2

N = 1

N = 2

+

–

Создание копии строки

S1:=Copy(S)

Вх: S

Вых: S1

Создание копии строки

S2:=Copy(S)

Вх: S

Вых: S2

**11. Программный код**

program Lab1;

{$AppType CONSOLE}

Uses Windows, UnLAB1;

var

S,S1,S2: ANSIString;

N: Byte;

begin

// сменим кодовые страницы для консольного ввода и вывода

setConsoleCP(1251); // для ввода

setConsoleOutputCP(1251); // для вывода

writeln('Введите строку (смените шрифт на Lucida Console)');

readln(S); // ввод

S1:=copy(S,1,length(S)); S2:=copy(S,1,length(S)); // создание копий дин.строки

// без своей таблицы символов (со сложным условием обмена)

writeln(#13#10, 'Строка до сортировки'#13#10,S1);

N:= Prov1(s1); // проверка

case N of

1: writeln('Пустая строка');

2: writeln('Некорректные символы');

else

begin // сортировка

Sort1(s1);

writeln('Отсортированная строка 1'#13#10,S1); // вывод строки S1

end; {else}

end; {case}

// со своей таблицей символов

writeln(#13#10'Строка до сортировки'#13#10,S2);

N:= Prov2(s2); // проверка

case N of

1: writeln('Пустая строка');

2: writeln('Некорректные символы');

else

begin // сортировка

Sort2(s2);

writeln('Отсортированная строка 2'#13#10, S2); // вывод строки S2

end; {else}

end; {case}

writeln(#13#10'Press ENTER to exit');

readln

end.

Исходный код модуля

Unit UnLAB1; // имя модуля – меняется при сохранении File → Save As…

Interface // раздел описания межмодульного интерфейса

Uses

SysUtils;

Const

SymbTable = 'абвгдеёжзийклмнопрстуфхцчшщъыьэюяАБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ0123456789';

Const

BukvKirS: set of char = ['а'..'я','ё'];

BukvKirB: set of char = ['А'..'Я','Ё'];

Cifr : set of '0'..'9' = ['0'..'9'];

BukvKirB1: set of char = ['А'..'Е'];

BukvKirB2: set of char = ['Ж'..'Я'];

Function Prov1(const s: ANSIString): Byte; // проверка без SymbTable

Function Prov2(const s: ANSIString): Byte; // проверка с SymbTable

Procedure Sort1(var s: ANSIString); // сортировка без SymbTable

Procedure Sort2(var s: ANSIString); // сортировка с SymbTable

Implementation // раздел реализации и описания закрытых процедур

Function Prov1; // проверка без SymbTable

Var

N: byte; // номер аномалии

i,Len: word; // текущий символ и длина строки

Begin

N:=0; Len:=Length(S);

if Len=0 then N:=1

else

begin

i:=1;

while (i<=Len) and (N=0) do

begin

if not ((S[i] in Cifr) or (S[i] in BukvKirS) or (S[i] in BukvKirB)) then N:=2;

Inc(i);

end;

end;

Prov1:=N;

End;

Function Prov2; // проверка с SymbTable

Var

N: byte; // номер аномалии

i,Len: word; // текущий символ и длина строки

Begin

N:=0; Len:=Length(S);

if Len=0 then N:=1

else

begin

i:=1;

while (i<=Len) and (N=0) do

begin

if not (Pos(S[i],SymbTable)>0) then N:=2;

Inc(i);

end;

end;

Prov2:=N;

End;

Procedure Sort1; // сортировка без SymbTable

Var

i,z,Len: word; // номер текущего символа, номер итерации (шага), длина строки

flag: boolean; // упорядочено? (нет обменов?)

ch: char; // для обмена

Begin

z:=1; Len:=length(s);

repeat // пузырёк

flag:=true;

for i:=1 to Len-z do

if ( (s[i] in Cifr) and ((s[i+1] in BukvKirS) or (s[i+1] in BukvKirB)) ) or // цифры «легче» всех

( (s[i] in BukvKirB) and (s[i+1] in BukvKirS) ) or

// кириллица заглавные «легче» строчных – всплывает

( (s[i] in BukvKirB2) and (s[i+1] in BukvKirB1) ) or

// кириллица заглавные 2 «легче» кириллицы заглавных 1 – всплывает

( (s[i]='Ё') and (s[i+1] in BukvKirB1) ) or

// кириллица Ё «легче» кириллицы заглавных 1 – всплывает

( (s[i] in BukvKirB1) and (s[i+1] in BukvKirB1) and (s[i]>s[i+1]) ) or

// кириллица заглавные 1 – по алфавиту

( (s[i] in BukvKirB2) and (s[i+1] in BukvKirB2) and (s[i]>s[i+1]) ) or

// кириллица заглавные 2 – по алфавиту

then

begin // обмен

ch:=s[i]; s[i]:=s[i+1]; s[i+1]:=ch; flag:= false;

end;

inc(z);

until flag or (z=Len);

End;

Procedure Sort2; // сортировка с SymbTable

Var

i,z,Len,P1,P2: word; // номер текущего символа, номер итерации (шага), длина строки, позиции текущих символов

flag: boolean; // упорядочено? (нет обменов?)

ch: char; // для обмена

Begin

z:=1; Len:=length(s);

repeat // пузырёк

flag:=true;

for i:=1 to Len-z do

P1:=Pos(s[i],SymbTable); P2:=Pos(s[i+1],SymbTable);

// неправильный порядок по своей «таблице»

if (P1>66 and P2<67) or (P1>33 and P2<34)

or (P1>P2 and P1>33 and P2>33 and P1<67 and P2<67)

then

begin // обмен

ch:=s[i]; s[i]:=s[i+1]; s[i+1]:=ch; flag:=false;

end;

inc(z);

until flag or (z=Len);

End;

End.

**12. Алгоритмы процедур**

**A0.1. Проверка аномалий**

Начало А0.1, Вх: S1

N:=0; Len:= Длина(S1)

Len=0

N:=1

i:=1

I ≤ Len ∧ N=0

(\*)

N:=2

i:=i+1

Конец А0.1, Вых: N

(\*) ≡ ¬((S1[i] ∈ [‘0’..‘9’]) ∧

(S1[i] ∈ [‘А’..‘Я’, ‘Ё’]) ∧ (S1[i] ∈ ['а'..'я','ё']

+

–

+

–

+

–

**А0.2. Сортировка «пузырьком»**

Начало А0.2, Вх: S1

z:=1; Len:= Длина(S1)

flag:=true

i:=1; +1; Len-z

\*\*

ch:=S1[i]; S1[i]:=S1[i+1]; S1[i+1]=ch

flag:=false

z:=z+1

flag ∨(z=Len)

Конец А0.2, Вых: S1

+

+

–

(\*\*) ≡ (S1[i] ∈ [‘0’..‘9’]) ∧ ((S1[i+1] ∈ ['а'..'я','ё']) ∨ (S1[i+1] ∈ [‘А’..‘Я’, ‘Ё’])) ∨

(S1[i] ∈ [‘А’..‘Я’, ‘Ё’]) ∧ (S1[i+1] ∈ ['а'..'я','ё']) ∨

(S1[i] ∈ ['Ж'..'Я']) ∧ (S1[i+1] ∈ ['А'..'Е']) ∨

(S1[i] = ‘Ё’) ∧ (S1[i+1] ∈ ['А'..'Е']) ∨

(S1[i] ∈ ['А'..'Е']) ∧ (S1[i+1] ∈ ['А'..'Е']) ∧ (S1[i]>S1[i+1]) ∨

(S1[i] ∈ ['Ж'..'Я']) ∧ (S1[i+1] ∈ ['Ж'..'Я']) ∧ (S1[i]>S1[i+1])

Менять пару символов S1[i], S1[i+1] местами, если

-- i - цифра и (i+1) – буква кириллицы (большая или маленькая)

-- i - кириллица заглавные и (i+1) – кириллица прописные

-- i - кириллица заглавные после Ё по алфавиту и (i+1) – кириллица заглавные до Ё по алфавиту

-- i - кириллица заглавная Ё по алфавиту и (i+1) – кириллица заглавные до Ё по алфавиту

-- обе буквы - кириллица заглавные до Ё по алфавиту и не по порядку

-- обе буквы - кириллица заглавные после Ё по алфавиту и не по порядку