

Алгоритмы обработки текстовых данных



Строковый тип данных (string)

Строковый тип данных (string) — это сложный тип данных (в отличие от integer, real, boolean и char).

Значением строковой величины является последовательность СИМВОЛОВ (т.е. элементов типа char).

Объявление текстовых данных

- `Var Text: String;`

Ввод данных с клавиатуры

- `Write('Введите строку : '); ReadLn(Text);`

Ввод данных в программе

- `Text : string[5]='ПКС-7';`

Вывод данных

- `WriteLn('Текст: ', Text);`

Процедуры

- Delete (Var S : String; N, M : Integer)
{Удаляет M символов из строки S, начиная с позиции N}
- Insert (SubS : String; Var S : String; N : Integer)
{Вставляет подстроку SubS в строку S, начиная с позиции N}
- Str (X : Integer; Var S : String)
{Возвращает представление числа X в его символьной форме S}
- Val (S : String; Var X, Code : Integer)
{Возвращает представление символов строки S в ее числовой форме X. Параметр Code содержит признак ошибки преобразования (если Code = 0, ошибки нет)}

Пример:

```
var stroka, stroka1, stroka2: string;  
x, y, z: integer; begin  
x:=1234; y:=5678; z:=x+y;  
writeln(z);  
str(x, stroka1); str(y, stroka2);  
stroka:=stroka1+stroka2;  
writeln(stroka);  
end.
```

Вывод:

6912

12345678

Функции

- `Ord (X : Char) : LongInt` {Порядковый номер символа X в таблице кодов символов (из таблицы ASCII)}
- `Chr (X : Byte) : Char` {Возвращает символ с заданным порядковым номером X (из таблицы ASCII)}

	000	016	032	048	064	080	096	112		128	144	160	176	192	208	224	240	
00		◆		0	@	P	`	p	00	00	A	P	a	░	▬	p	≡	00
01		◆	!	1	A	Q	a	q	01	01	Б	С	б	▒	▴	с	±	01
02		↕	"	2	B	R	b	r	02	02	В	Т	в	▓	▵	т	≥	02
03	♥		#	3	C	S	c	s	03	03	Г	У	г		▿	у	≤	03
04	◆	¶	\$	4	D	T	d	t	04	04	Д	Ф	д	└	▯	ф	┌	04
05	♣	§	%	5	E	U	e	u	05	05	Е	Х	е	├	┴	х	┘	05
06	♠		&	6	F	V	f	v	06	06	Ж	Ц	ж	▯	┐	ц	÷	06
07	.		'	7	G	W	g	w	07	07	З	Ч	з	▯	▯	ч	≈	07
08		↑	(8	H	X	h	x	08	08	И	Ш	и	▯	▯	ш	°	08
09	°	↓)	9	I	Y	i	y	09	09	Й	Щ	й	▯	▯	щ	•	09
10		→	*	:	J	Z	j	z	10	10	К	Ъ	к	▯	▯	ъ	.	10
11		←	+	;	K	[k	{	11	11	Л	Ы	л	▯	▯	ы	√	11
12		└	,	<	L	\	l		12	12	М	Ь	м	▯	▯	ь	π	12
13		•	-	=	M]	m	}	13	13	Н	Э	н	▯	▯	э	z	13
14		•	.	>	N	^	n	~	14	14	О	Ю	о	▯	▯	ю	■	14
15	Ц	◆	/	?	O	_	o	\$	15	15	П	Я	п	▯	▯	я		15

Функции

- Concat (S1 , S2 , ... , SN): String
{Выполняет сцепку (конкатенацию) последовательности строк}
- Copy (S : String; N , M : Integer) : String
{Выделяет подстроку из строки S, начиная с позиции N и длиной M символов}
- Length (S : String) : Byte
{Количество символов в строке S - длина}
- Pos (SubS , S : String) : Byte
{Номер позиции, начиная с которой в строке S располагается подстрока SubS (если значение функции равно нулю, то S не содержит SubS)}

Пример:

```
var i: integer;  
s, t, u: string; begin  
s := 'индустриализация';  
t := Copy(s, 3, 2);  
u := Copy(s, 9, 8);  
WriteLn(Concat(t, u));  
i := Pos('ус', s);  
Delete(s, i, 11);  
WriteLn(s);  
Insert('онез', s, 4);  
WriteLn(s); end.
```

Вывод:

дуализация

индия

индонезия

Посчитать количество слов в
предложении

Program KolSlov;

Var Text : String; i, k : Integer; Flag: Boolean;

BEGIN

WriteLn('Введите текст :'); ReadLn(Text);

k := 0; Flag := TRUE;

For i := 1 to Length(Text) do begin

 If (Text[i] <> ' ') and Flag then

 k := k+1;

 Flag := (Text[i]=' ')

end;

WriteLn('О т в е т : количество слов в тексте равно ', k);

END.

Заменить слово в предложении

Программа, использующая
стандартную функцию Pos , не
требует, чтобы длины заменяемого
и вставляемого слов были
одинаковыми

```
Program Replace;  
Var Text, Slovo1, Slovo2 : String;  
i, DlinaSlova, P : Integer;  
BEGIN;  
Write('Введите строку : '); ReadLn(Text);  
Write('Какое слово заменить ? '); ReadLn(Slovo1);  
Write('На какое слово заменить ? '); ReadLn(Slovo2); WriteLn;  
DlinaSlova:=Length(Slovo1);  
P:=Pos(Slovo1,Text); {номер позиции, с которой в строке Text в первый раз  
встречается подстрока Slovo1}  
While P>0 do {цикл продолжается до тех пор пока подстрока Slovo1  
встречается в строке Text}  
begin Delete(Text, P, DlinaSlova); {удаление подстроки Slovo1,  
начинающейся с позиции P, из строки Text }  
Insert(Slovo2, Text, P); {вставка подстроки Slovo2 в строку Text с позиции  
P}  
P:=Pos(Slovo1, Text); {номер позиции, с которой подстрока Slovo1  
встречается в строке Text в очередной раз} end;  
WriteLn('Новый текст: ', Text); END.
```

Определить, является ли заданное
слово "перевёртышем"
(слово называется "перевёртышем",
если совпадает с собой после
реверса)

```
Program;  
Var Slovo  : String;  
Dlina, i : Integer;  
Flag : Boolean;  
BEGIN  Write('Введите слово : '); ReadLn(Slovo);  
Dlina:= Length(Slovo);  
i:=1; Flag := TRUE;  
While (i <= Dlina/2) and Flag do begin  
    Flag := (Slovo[i]=Slovo[Dlina-i+1]);  
    {Сравниваются пары букв: первая буква с последней,  
    вторая буква с предпоследней и т.д. }  
    i := i+1  
end;  
WriteLn; Write( 'О т в е т : слово ', Slovo);  
If Flag then WriteLn(' — перевертыш. ')  
    else WriteLn(' — не перевертыш');  
END.
```

Заданную последовательность слов переупорядочить в алфавитном порядке

Данные	Результат
Words=("стул", "гора", "яма", "стол")	Words=("гора", "стол", "стул", "яма")

```
Program Poryadok;  
Var Words  : Array[1..10] of String; Tmp : String; i, j, k : Integer;  
BEGIN  
Write('Количество слов в тексте — '); ReadLn(k);  
For i := 1 to k do begin  
Write(i, '-ое слово : '); ReadLn(Words[i]) end;  
For i := 1 to k-1 do  
    For j := i+1 to k do  
        If Words[i]>Words[j] then begin  
            Tmp := Words[i]; Words[i]:=Words[j]; Words[j]:=Tmp;  
        end;  
    end;  
WriteLn('О т в е т:');  
For i := 1 to k do  
Write(Words[i], ' ');  
END.
```


Проверить, имеется ли в формуле
баланс открывающих и
закрывающих скобок

Program Balance;

Var S : String;

k, i : Integer;

BEGIN

Write('Введите формулу :'); ReadLn(S);

i:=1; k:=0;

While i<=Length(S)) do begin

 If S[i] = '(' then k:=k + 1;

 If S[i] = ')' then k:=k - 1;

 i:=i+1;

end;

WriteLn('О т в е т');

If k=0 then Write('Есть баланс ') else Write('Нет баланса ');

WriteLn('открывающих и закрывающих скобок');

END.