Рядки (string)

Рядки та дії з ними. Дане типу рядок — це послідовність довільних символів (тобто елементів типу char). Сталі типу рядок записують за допомогою двох штрих-символів,які охоплюють текст. Рядок може містити від 0 до 255 символів. Наприклад, 'Україна', 'Львівська політехніка', '' - порожній рядок нульової довжини, ' ' -рядок, що містить один символ-пропуск.

Змінну типу рядок оголошують за допомогою слова string так:

var <змінна>: string[n];

де *n* -довжина рядка,n<256. Довжину рядка можна не зазначати.

Приклад.

const slovo='University';

var frazal: string[45]; fraza2: string;

Над змінними типу рядок визначені операції з'єднання (+) та порівняння (<,<=,>,>=,=,<>)• Порівняння двох рядків здійснюється зліва направо до перших різних символів, причому 'A'<'B','B'<'C тощо. "Більшим" вважається символ, який розташований в алфавіті далі (він має більший номер у таблиці кодів комп'ютера ASCII). Числовий код символу дає функція ord, наприклад, ord('B')=66, ord('A')=65. Зворотну дію виконує функція **chr:** chr(66) дає 'B'.

Приклад. Нехай tl='New', t2=' Year'. Тоді з'єднанням цих рядків буде s:=tl+t2 (s матиме значення 'New Year'). Тут s > tl.

Приклад. Вивести на екран малі літери латинського алфавіту та їхні коди можна так: **for** v:= 'a' **to** V **do** writeln(v,ord(v):5).

Функції та процедури для дій з **рядками.** Над даними типу рядок визначені такі стандартні функції:

length(<рядок>) — визначає кількість символів у рядку; **copy(r, m, n)** — дає n символів рядка r, починаючи з символу з номером т;

concat(r1, r2,...,rn) — з'єднує рядки r1,...,rn в один рядок; **pos**(r1, r2) — визначає номер символу, з якого починається входження рядка r1 у рядок r2.

та процедури:

insert(r1, <змінна>, n) — вставляє рядок **r1** у рядок, заданий змінною, починаючи з позиції **n;**

delete(<змінна>, m, n) — вилучає **n** символів з рядка, заданого змінною,починаючи з позиції **n;**

str(<число>,<змінна>) — переводить числове дане в дане типу рядок;

val(rl, s1, s2) — засилає у числову змінну **s1** числовий образ рядка **r1.** Якщо це можливо, то змінна **s2** отримує значення 0, інакше — числове значення номера першого недопустимого символу заданого рядка,

де зазначена змінна посилає в процедуру вхідне дане типу рядок і отримує назад інший рядок - результат виконання процедури.

Приклад. Нехай змінна Lviv має значення 'Львівська політехніка'. Розглянемо приклади функцій та їхні значення:

Функція	***Значення***
length(Lviv)	21;
copy(Lviv,15,ll)	техніка;
concat(Lviv,' - 2000')	Львівська політехніка - 2000;
pos('т',Lviv)	15.

Наступні процедури нададуть змінній Lviv таких значень:

Процедура	*Значення змінної Lviv*
insert('Ну', Lviv,l)	'НУ Львівська політехніка'
delete(Lviv,6,16)	'Львів'
str(2000,Lviv)	'2000'
val('1256', Lvivl, Ozn)	Lviv 1=1256, Ozn=0

Є два способи опрацювання даних типу string. Перший — можна опрацьовувати весь рядок як єдине ціле, застосовуючи до нього функції та процедури, другий — можна розглядати рядок як масив, складений з елементів-символів, і опрацьовувати його за правилами роботи з елементами масиву.

Приклад. Деякі значення змінним frazal та fraza2 з попереднього прикладу можна надати, а потім вивести тексти на екран так: frazal :='Ви любите канікули?';

for i:=1 **to** 19 **do** read(fraza2[i]);{Вводимо *текст з клавіатури}* writeln(frazal); writeln(fraza2).

Задача 1. Дано список з 6 слів. Визначити, скільки слів списку починається на букву «п».

```
Var
s : string[20];
i, k :integer;
Begin
k:=0;
FOR i:=1 TO 6 DO
Begin
Writeln ('Введіть слово');
Readln (s);
if s[1]='п' then k:=k+1;
End;
Writeln (k );
End.
```

Задача 2. Дано список з 8 слів. Знайти найкоротше слово в списку. Якщо таких слів кілька, то роздрукувати їх в один стовпець.

Рішення поставленої задачі зводиться до декількох етапів: ввести список слів у вигляді масиву рядкових змінних; підрахувати довжину кожного рядка; визначити якнайменшу з довжин; роздрукувати ті рядки масиву, довжина яких співпадає з найменшою.

```
type t=array[1..8] of string[20];
tt=array[1..8] of integer;
Var s: t;
n: tt;
i, min: integer;
Begin
FOR i:=1 TO 8 DO
begin
Writeln(' Уведіть слово');
Readln(s[i]);
n[i]:=length(s[i]);
end;
min:=n[1];
35
FOR i:=2 TO 8 DO
IF min>n[i] THEN min:=n[i];
FOR i:=1 TO 8 DO
IF n[i]=min THEN writeln(s[i]);
End.
```

Задача 3 Дано рядок з двох слів, розділених пропуском. Поміняти в даному рядку слова місцями.

```
var s: string[40];
s1,s2: string[20];
i,n: integer;
Begin
Writeln(' Введіть рядок');
Readln(s);
n:=length(s);
i:=pos(' ', s);
s1:=copy(s, 1, i);
s2:=copy(s, i+1, n-i);
s:=s2 + ' ' + s1;
Writeln(s);
End.
```

***Задача** 4.* Кодування інформації. Вилучити з фрази *а* пропуски, коми і крапки, інші символи продублювати. Вивести результат.

```
var a, b, c : string; i : integer;
begin
write('Введіть текст: ');
readln(a);                                   {Вводимо будь-яку фразу}
b := '';
for i := 1 to Length(a) do
begin                                   {Зверніть увагу на коментар унизу:}
    c := Copy(a, i, 1);                          { або так: c := a[i]; }
     if (c <> ',') and (c <>'.') and (c <>' ')
          then b := b + c + c
end;
writeln(b);
readln
end.
```

***Задача** 5.* Скласти програму, яка скрізь у заданому тексті ту-text замінить деяке слово іншим словом такої ж довжини (wordl на word2).

```
var mytext, wordl,word2 : string; i,k : integer;
begin
write('Введіть текст: '); readln(mytext);
write('Введіть шукане слово: '); readln(wordl);
write('Введіть інше слово: '); readln(word2);
 k := length(wordl);
 for i := 1 to length(mytext) - k do
if copy(mytext, i, k) = wordl then
begin
delete(mytext, i, k);
insert(word2, mytext, i)
 end;
writeln(mytext);
readln
end.
```

***Задача** 6.* Нехай задано рядок 'Я люблю інформатику'. Визначити довжину рядка. Вивести на екран друге слово цього рядка.

```
const rl: string = 'Я люблю  інформатику';
var i,k,m,nl,n2: integer;
begin
m:=0;
k:=length(rl); {Визначаємо довжину рядка)
writeln('Довжина рядка k=',k);
for i: = l to k do       {Переглядаємо всі символи рядка)
 if rl[i]=' ' then   {Шукаємо пропуск)
begin
 m:=m+l;
if m=l then nl:=i;
{Визначаємо номер першого та другого пропусків)
 if m=2 then n2:=i
 end;
{Виводимо слово між двома пропусками)
 for i:=nl+l to n2-l do write(rl[i]);
readln
end.
```

Задача 6.
Умова: У даному тексті всі послідовності крапок замінити на одну крапку.
Для розв'язку цієї задачі пропонується організувати цикл перегляду кожного елементу рядка (краще за допомогою циклу з передумовою), причому, якщо буде знайдена група крапок, вилучити її повністю, а потім вставити на це ж місце тільки одну крапку. Для вилучення групи крапок можна скористатися кількома способами. Наприклад, підрахувати їх кількість, а потім всі вилучити.
Ми пропонуємо вилучати всі крапки, доки не зустрінеться символ, що не являється крапкою.

```
Var i:word; {i - змінна циклу}
St:string; {St - даний текст}
Begin
Write ('Введіть текст: ');
Readln (St);
i:=1;
While i<=length(St) do
Begin
While St[i]='.' do Delete(St,i,1);
Insert('.',St,i);
i:=i+1;
End;
Writeln ('Результуючий рядок: ');
Writeln (St);
End.
```

Задача 7
Умова: Дано деякий текст, у якому є хоча б одна кома. Визначити порядковий номер останньої коми в тексті.
Для пошуку останньої коми в тексті необхідно організувати цикл перегляду рядка з кінця до початку, тобто від останнього символу рядка з номером *length(St)* до першого. В зв'язку з тим, що ми не знаємо, якою за номером буде кома, краще це зробити командою повторення з передумовою. Цикл завершить свою роботу при досягненні позиції, в якій знаходиться кома (за умовою вона обов'язково є в тексті). Значення змінної циклу і буде номером шуканої позиції.
Програма, що реалізує описаний алгоритм, має наступний вигляд:

```
Var i:byte; {i - змінна циклу}
St:string; {St - даний текст}
Begin
Write ('Введіть текст: ');
Readln(St);
i:=length(St);
while St[i]<>',' do i:=i-1;
Writeln ('Номер позиції останньої коми в тексті ',i);
End.
```

Задача 8
Умова: Дано деякий текст. Створити новий текст, який утворено із даного читанням з кінця до початку.
Для реалізації даної задачі знов, як і в попередньому випадку, пройдемо рядок з кінця до початку, виконуючи конкатенацію кожного наступного символу до нового рядка.
Програма, що реалізує описаний алгоритм, має наступний вигляд:

```
Var i:byte; {i - змінна циклу}
St,Rez:string; {St - даний текст, Rez - результуючий (перегорнутий) рядок}
Begin
Write ('Введіть текст: ');
```

```
Readln (St);
Rez:=''; {Очищення рядка}
For i:=length(St) downto 1 do
Rez := Rez+St[i];
Writeln ('Результуючий рядок: ');
Writeln (St);
End.
```

Задача 9.

Умова: Перевірити, чи однаково читається дане слово зліва направо і навпаки.

Для розв'язку цієї задачі можна використовувати попередню задачу, як підзадачу, тобто спочатку отримати новий рядок, який являється оберненим відносно даного, а потім порівняти даний та отриманий рядки. Якщо вони співпадають, слово являється паліндромом (читається в обох напрямках однаково, наприклад, "дід", "потоп", "Пилип" тощо), в протилежному випадку - ні.

Програма, що реалізує описаний алгоритм, має наступний вигляд:

```
Var i:byte; {i - змінна циклу}
St,Rez:string; {St - даний текст, Rez -
результуючий (перегорнутий)
рядок}
Begin
Write ('Введіть текст: ');
Readln (St);
Rez:=''; {Очищення рядка}
For i:=length(St) downto 1 do
Rez := Rez+St[i]; {Перегортання рядка}
If Rez = St
Then Writeln ('Слово являється паліндромом.')
Else Writeln ('Слово не являється паліндромом.');
End.
```

Задача 10

Умова: Визначити, скільки разів у даному тексті зустрічається послідовність символів "абв".

Організовуємо прохід по рядку за допомогою циклу з параметром, причому враховуємо, що необхідно перевірити три послідовно розташованих символи (зверніть увагу на можливість виходу за межі рядка!). Один з методів вибору кількох послідовних символів вже розглядався раніше (*i*-ий, *i+1*-ий та *i+2*-ий елементи), тому розглянемо інший метод, що полягає у використанні функції копіювання *Copy*. Нагадуємо, що ця функція містить у якості параметрів вихідний рядок, номер початку копіювання (виділення) та кількість вибраних символів, тобто для вибору трьох символів з будь-якого місця рядка *St* ця функція буде мати вид:

Copy(St,i,3).

Порівнюючи виділені (скопійовані) символи з еталоном, нарощуємо лічильник при виконанні поставлених умов.

Програма, що реалізує описаний алгоритм, має наступний вигляд:

```
Var i:byte; {i - змінна циклу}
St:string; {St - даний текст}
Count:byte; {Count - лічильник  послідовностей}
Begin
Write ('Введіть текст: ');
Readln (St);
Count:=0; {Початкове значення лічильника}
For i:=1 to length(St)-3 do
If Copy (St,i,3) = 'абв'
Then count:=count+1;
```

```
Writeln ('Кількість шуканих послідовностей: ',count);
Readkey; {Затримка зображення на екрані}
End.
```

Задача 11

Умова: Нехай дано формулу. Визначити коректність формули щодо кількості відкритих та закритих дужок. Вважається, що закриті дужки не стоять перед відкритими. Якщо дужки у формулі відсутні - повідомити про це.

Для визначення коректності формули необхідно підрахувати кількість відкритих та кількість закритих дужок. Якщо ці значення дорівнюють одне одному, баланс дужок виконується, якщо ні - не виконується (дійсно, у правильній формулі кількість відкритих та закритих дужок повинні співпадати). Щоб визначити, чи є в формулі дужки взагалі, достатньо перевірити на нуль кількість одних чи других дужок. Визначимо змінні count_left та count_right, як кількість відповідно лівих (відкритих) та правих (закритих) дужок, тоді програма, що реалізує описаний алгоритм, має наступний вигляд:

```
 Var i:byte; {i - змінна циклу}
St:string; {St - даний текст}
count_left, count_right:byte;  {count_left - лічильник кількості  лівих дужок, count_right - лічильник кількості правих дужок}
Begin
Write ('Введіть формулу: ');
Readln(St);
Count_left:=0; {Початкове значення лічильника}
Count_right:=0;
For i:=1 to length(St) do
Begin
If St[i] = '('
Then count_left:=count_left+1;
If St[i] = ')'
Then count_right:=count_right+1;
End;
Writeln ('Кількість лівих дужок: ',count_left);
Writeln ('Кількість правих дужок: ',count_right);
If (count_left=0) or (count_right=0)
Then writeln ('Формула не має дужок.')
Else
Begin
If count_left = count_right
then Writeln ('Формула коректна.')
else writeln ('Формула не коректна.');
end;
End.
```

Задача 12

Умова: Нехай дано текст *S* та значення символьних змінних *x* та *y*. Із тексту вилучити всі символи, що збігаються з *x* і повторити двічі всі символи, що збігаються з *y*.

Для розв'язання запропонованої задачі необхідно переглянути кожен символ рядка і, якщо він співпадає з символом *x*, вилучити його процедурою ***delete***, а якщо збігається із символом *y* - вставити перед ним такий самий процедурою *insert*. Програма для реалізації цього алгоритму має наступний вигляд:

```
Var i:byte; {i - змінна циклу}
S:string; {S - даний текст}
x,y:char; {x,y - шукані символи}
Begin
Write ('Введіть текст: ');
```

```
Readln (S);
Write ('Введіть шукані символи: ');
Readln (x,y);
For i:=1 to length(S) do
Begin
If S[i] = x
Then delete(S,i,1);
If S[i] = ')'
Then insert(y,S,i);
End;
Writeln ('Результуючий рядок після зміни: ',S);
End.
```

Задача 13.

Умова: Розробити програму-шифрувальник тексту, що замінює кожну його літеру наступною по порядку в абетці. Останню літеру абетки необхідно замінити першою.

В даній задачі необхідно, по-перше, перевірити, чи є черговий символ рядка буквою. Відомо, що всі букви латиниці розташовані в таблиці ASCII-кодів двома групами великих та малих літер, а літери кирилиці - двома нерівномірними групами від 'А' до 'п' та від 'р' до 'я':

'A'.. 'Z' - коди *65 .. 90;*

'a'..'z' - коди *97 .. 122;*

'А'..'п' - коди *128 .. 175;*

'р'..'ї' - коди *224 .. 245.*

Якщо черговий символ рядка - буква, необхідно збільшити її код на 1, а потім перетворити отриманий код в букву. Для визначення коду за символом використовується процедура ***ord***, а для визначення символу за ASCII-кодом - процедура ***char***. Для останніх літер необхідно написати незалежні команди розгалуження для заміни їх на перші літери абетки, причому для символів латиниці можна скористатися однією формулою *chr(ord(S[i])-25)*, тому що, кількість великих та малих літер в абетці однакові і відстань від першої літери до останньої дорівнює 25.

Програма для реалізації описаного алгоритму має наступний вигляд:

```
Var i:byte; {i - змінна циклу}
S:string; {S - даний текст}
Begin
Write ('Введіть текст: ');
Readln (S);
For i:=1 to length(S) do
Begin
If (S[i]>=65) and (S[i]<90) or
(S[i]>=97) and (S[i]<122) or
(S[i]>=128) and (S[i]<159) or
(S[i]>=160) and (S[i]<=175) or
(S[i]>=224) and (S[i]<245) or
Then S[i]:=chr(ord(S[i]+1);
If (S[i] = 'Z') or (S[i] = 'z')
Then S[i]:=chr(ord(S[i])-25);
If (S[i] = 'Я')
Then S[i]:='A';
If (S[i] = 'я')
Then S[i]:='a';
End;
Writeln ('Результуючий рядок після зміни: ',S);
End.
```

Задача 14

Умова: Дано деякий текст. Групи символів, які розділені пробілами (одним або кількома) та не містять всередині себе пробілів, називатимемо словами. Вважатимемо, що текст завжди починається зі слова. Визначити кількість слів, у яких перша та остання літера однакові.

При виконанні розв'язку цієї задачі, очевидно, що початок першого слова буде співпадати з першою літерою рядка (дивись умову), а кінець слова можна визначити, знайшовши перший проміжок. Скопіювавши це слово в додатковий рядок, ми отримаємо нескладну задачу порівняти його перший та останній символи. Якщо ці символи співпадають, слово підраховується, і процес продовжується для наступного слова.

Щоб задачу пошуку чергового слова зробити ідентичною для всіх слів, можна вважати, що початком слова є послідовність з проміжку та непроміжку, а відповідно кінцем слова є послідовність з непроміжку та проміжку (цей метод зручний ще тим, що враховує те, що між словами може бути не один проміжок - всі зайві проміжки пропускаються). Виділятися з цього буде перше та останні слова, тому що на початку рядка та в його кінці користувач може не поставити проміжок і ці слова не буде знайденим. Щоб узагальнити задачу пропонуємо один зі штучних методів: дописати на початку та в кінці рядка проміжки . Тоді, програма для реалізації описаного алгоритму буде мати наступний вигляд:

```
Var i:byte; {i - змінна циклу}
S,Slovo:string; {S - даний текст, Slovo - вирізане з тексту слово}
count:byte; {count - лічильник шуканих слів}
Begin
Write ('Введіть текст: ');
Readln (S);
S:=' '+S+' '; {Дописування проміжку перед першим словом та після останнього}
Count:=0; {Початкове значення лічильника}
i:=1; {Початок перегляду рядка}
while i<=length(S)-1 do
Begin
If (S[i]=' ') and (S[i+1]<>' ')
Then
Begin
Slovo:=''; {Очищення рядка для зберігання чергового слова}
While (S[i+1]<>' ') do
Begin
Slovo:=Slovo+S[i+1];
i:=i+1;
End;
If Slovo[1] = Slovo[length(Slovo)]
Then count:=count+1;
End;
i:=i+1;
End;
Writeln ('Кількість шуканих слів: ',count);
End.
```