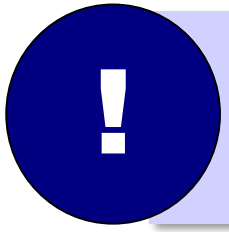
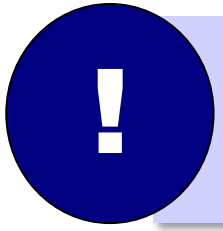
Bazele programării I
Prelucrarea șirurilor de caractere

Șirul de caractere. Definiție



Șirul de caractere reprezintă o succesiune de caractere.



Un caracter se reprezintă prin tipul de date standard **Char**.

Modalități de reprezentare a șirurilor de caractere (Metoda 1)

Șirul de caractere poate fi definit printr-un grup de variabile de tip *Char*.

De exemplu, cuvântul ***algoritm*** poate fi reprezentat în felul următor:

```
Var
   A1: Char
   A2: Char
   A3: Char
   A4: Char
   A5: Char
   A6: Char
   A7: Char
   A8: Char
```

```
A1:= 'a'
A2:= 'l'
A3:= 'g'
A4:= 'o'
A5:= 'r'
A6:= 'i'
A7:= 't'
A8:= 'm'
```

Modalități de reprezentare a șirurilor de caractere (Metoda 2)

Șirul de caractere poate fi definit ca vector, elementele căruia sunt caractere.

De exemplu, cuvântul ***algoritm*** poate fi reprezentat în felul următor:

```
Type
   Sir=Array[1..8] Of Char
Var
   A: Sir
```

```
For K:=1 To 8 Step 1
    ReadChar(A[K])
End
```

```
A[1]:='a'    A[6]:='i'
A[2]:='l'    A[7]:='t'
A[3]:='g'    A[8]:='m'
A[4]:='o'
A[5]:='r'
```

```
For K:=1 To 8 Step 1
    WriteChar(A[K])
End
```

Dezavantajele vectorului din caractere?

Avem următoarea declarație:

```
var A: array[1..N] of char
```

- fiecare simbol – obiect distinct;
- vectorul are lungimea N, definită la declararea acestuia.

Este necesar:

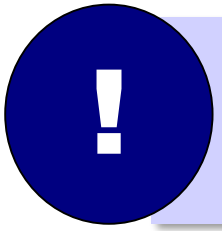
- prelucrarea secvenței de caractere ca un element integru;
- șirul de caractere trebuie să posede o lungime variabilă.

Tipul de date String

Datorită faptului că în programe operațiile cu șirurile de caractere sunt foarte frecvente, limbajele de programare conțin mecanisme care facilitează lucrul cu astfel de date.

Există tipuri speciale de reprezentare a șirurilor de caractere, fapt ce permite:

- introducerea;
- afișarea;
- atribuirea;
- compararea șirurilor ca un tot întreg.



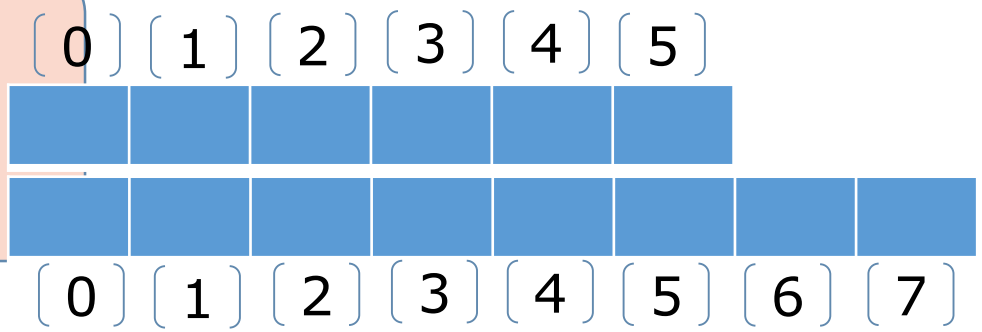
Tipul de date șir de caractere este definit ca un tip special de tablou **String** cu elemente de tip Char.

Declararea tipului de date String

```
Type
  <Nume_tip> = String[Lungimea maximă a șirului]
```

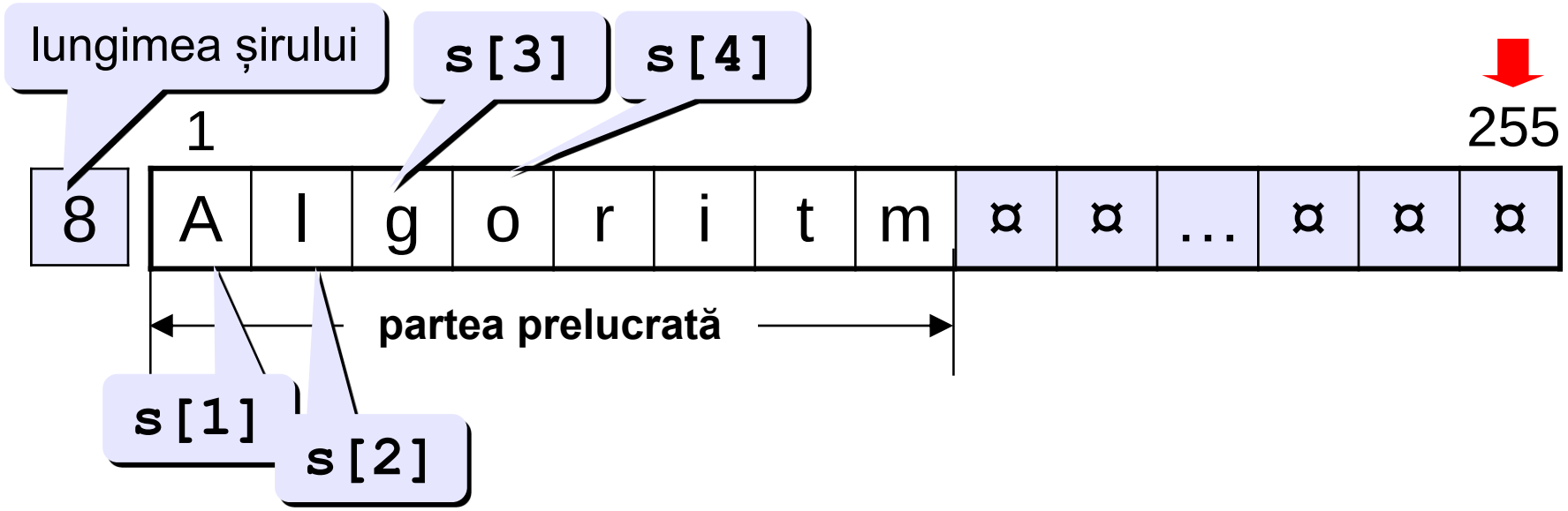
```
Type
    Sir1 = String[5]
    Sir2 = String[7]
```

```
Var
    A: Sir1
    B: Sir2
```

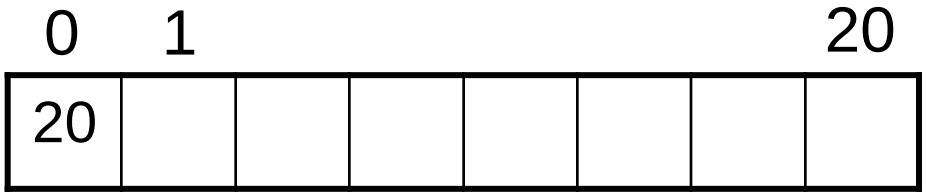


Șirul de caractere

```
var s: string
```




```
var s: string[20]
```



Operații cu șirurile: Citirea de la tastatură

```
ReadString(X)
```

```
Type
    Sir1=String[5]
    Sir2=String[7]
```

```
Var
  A: Sir1
  B: Sir2
```

4	P	a	c	e	

7	P	r	o	g	r	a	m

```
ReadString(A)
```

Tastatura

P	a	c	e	↵

```
ReadString(B)
```

Tastatura

P	r	o	g	r	a	m	↵

Operații cu șirurile: Afișarea șirului

```
WriteString(X)
```

```
Type
    Sir1=String[5]
    Sir2=String[7]
```

```
Var
  A: Sir1
  B: Sir2
```

4	P	a	c	e	

7	P	r	o	g	r	a	m

```
WriteString(A)
```

Ecran			
P	a	c	e

```
WriteString(B)
```

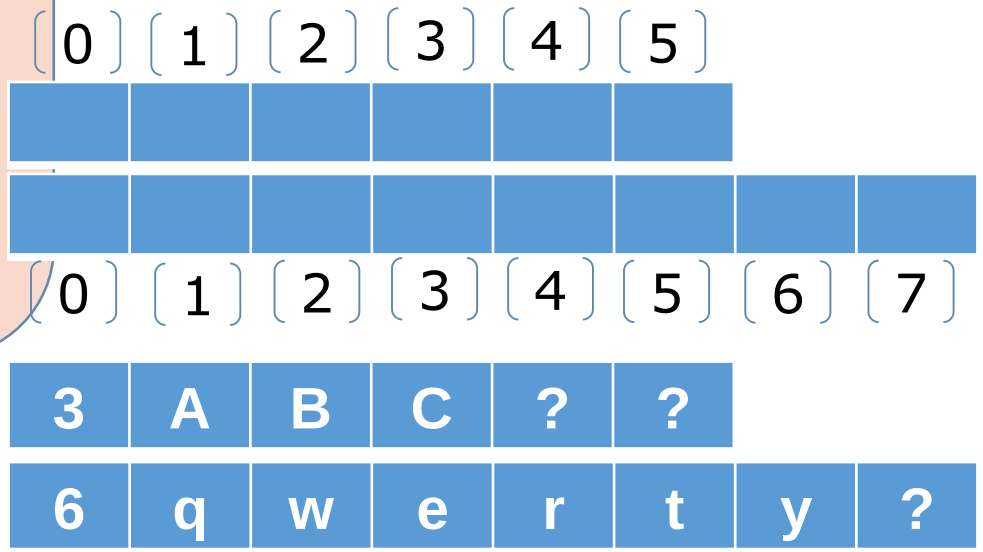
Ecran						
P	r	o	g	r	a	m

Operații cu șirurile: Atribuirea

Spre deosebire de tablouri, unei variabile de tip *String* i se poate atribui valoarea unei expresii de tip *String*.

```
Type
    Sir1 = String[5]
    Sir2 = String[7]
Var
    A: Sir1
    B: Sir2
Begin
```

```
A:= 'ABC'
B:= 'qwerty'
```

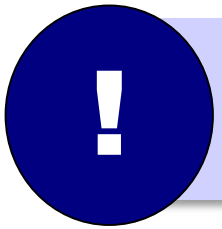


Operații cu șirurile: Compararea șirurilor

Spre deosebire de tablouri variabilele de tip *String* pot să apară în **expresii relaționale**. Operatorii relaționali:

=, #, <,>, <=, >=, aplicați unor date de tip *String*, furnizează o valoare logică.

Comparația a două date de tip *String* constă în **compararea caracter cu caracter**, de la stânga la dreapta, a codurilor ASCII ale caracterelor, până când se ajunge **la două caractere diferite** sau se epuizează caracterele unuia dintre șiruri.



Șirurile sunt **egale** numai dacă atât lungimile cât și conținutul lor sunt identice.

```
'abc' < 'abcd'    →
'axy' < 'abcdef'  →
```

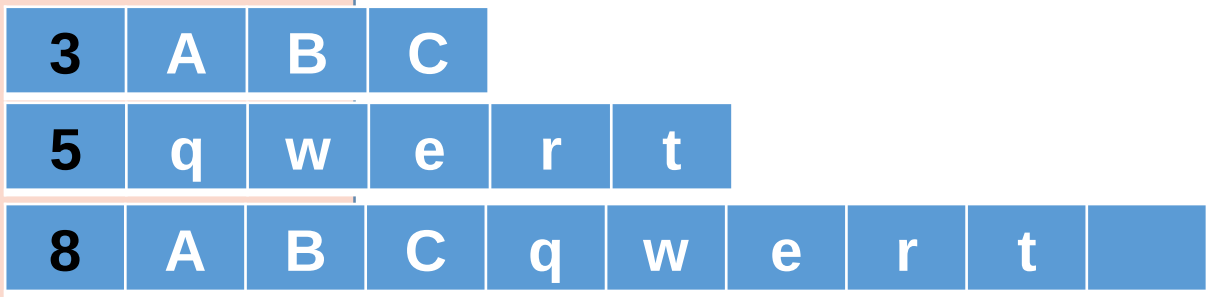
True

False

Operații cu șirurile: Concatenarea

Se realizează folosind operatorul **+**.

```
Type
    Sir1 = String[3]
    Sir2 = String[5]
    Sir3 = String[9]
Var
    A: Sir1
    B: Sir2
    C: Sir3
Begin
```

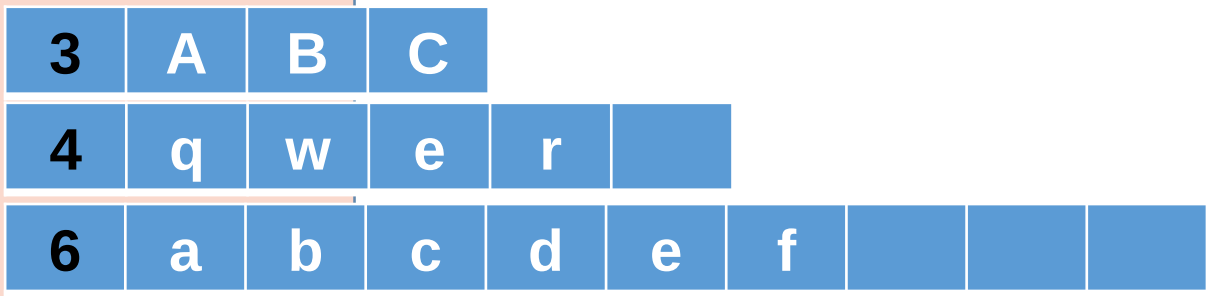



```
A:= 'ABC'
B:= 'qwert'
C:= A + B
```

Operații cu șirurile: Determinarea lungimii

Se utilizează funcția **Length(X)**

```
Type
    Sir1 = String[3]
    Sir2 = String[5]
    Sir3 = String[9]
Var
    A: Sir1
    B: Sir2
    C: Sir3
Begin
```




```
Var
  n: integer
```

```
  ...        Length(A)     ->
WriteInt(Length(B))       ->
n:= Length(C)             ->
```

3
4
6

Prelucrarea șirului de caractere caracter cu caracter

```
Type
    Sir = String[30]
Var
    S: Sir
```

```
For I:=1 to Length(S) step 1
        <prelucrare S[I]>
End
```

Numărarea elementelor şirului care posedă o proprietate dată

```
Rez := 0
For I:=1 to Length(S) step 1
   If S[I] = <proprietate> Then
        <prelucrare S[I]>
   End
End
```

Exemplu 1.

De la tastatură se introduce un şir de caractere. Să se determine numărul de cifre din acest şir.

```
ReadString(S)
Rez:=0
For I:=1 to Length(S) step 1
    If (S[I]>='0') and
                (S[I]<='9') Then
       Rez:= Rez + 1
    End
End
WriteNat(Rez)
```

Determinarea existenței în şir a caracterelor care posedă o proprietate

```
I:= 1
L:= Length(S)
While (I<=L) and (Not(S[I]=<proprietate>)) do
    I:= I+1
End
If I<=L Then
    WriteString('E
Else
    WriteString('N
End
```

Exemplu 2.

Să se determine dacă şirul A conține cifre.

```
I := 1
L:= Length(S)
While (I<=L) and (S[I]<'0' Or
                  S[I]>'9') do
    I := I+1
End
If I<=L Then
    WriteString('Există')
Else
    WriteString('Nu există')
End
```

Determinarea poziției în şir a caracterelor care posedă o proprietate

În multe probleme este necesar de determinat următoarele:

- poziția **primului caracter** care posedă o careva proprietate;
- poziția **ultimului caracter** care posedă o careva proprietate;
- pozițiile **tuturor caracterelor** care posedă proprietatea.

Determinarea poziției primului caracter care posedă o proprietate

```
I:= 1
L:= Length(S)
While (I<=L) and (Not(S[I]=<proprietate>)) do
    I:= I+1
End
If I<=L Then
    WriteNat(I)
Else
    WriteString('Nu există')
End
```

Determinarea poziției ultimului caracter care posedă o proprietate

```
L:= Length(S)
I:= L
While (I>=1) and (Not(S[I]=<proprietate>)) do
    I:= I-1
End
If I>=1 Then
    WriteNat(I)
Else
    WriteString('Nu există')
End
```

Extragerea unui subşir din şirul de caractere

```
Function SubStr (S1:String[255], N:Natural,
                 Poz: Natural): String
Var
 S2: String[255]
 I: Natural
Begin
 S2:= ''
 For I:=Poz to Poz+N-1 step 1
     S2:= S2+S1[I]
 End
 Return S2
End
```

S1 =	8	A	B	C	X	Y	Z	D	E

N = 3

Poz = 4

I	4	5	6
S2	X	Y	Z

Ștergerea unui subșir din șirul de caractere

```
Procedure DeleteStr (Var S1: String[255],
                     N:Natural, Poz:Natural)
Var
    S2: String[255]
    I: Natural
Begin
 S2:= ''
 For I:=1 to Poz-1 step 1
    S2:= S2+S1[I]
 End
 For I:=Poz+N to Length(S1)
    S2:= S2+S1[I]
 End
 S1:= S2
End
```

S1 =	7	A	B	C	X	Y	Z	D

N = 3

Poz = 4

I	1	2	3
S2	A	B	C

I	7	8
S2	A	B	C	D
---	---	---	---	---

Includerea unui subşir într-un şirul de caractere

```
Procedure InsertStr(S1: String[255],
                    Var S2: String[255],Poz:Natural)
Var
   S3: String[255];  I: Natural
Begin
  S3:=''
  For I:=1 to Poz-1 step 1
      S3:= S3+S2[I]
  End
  For I:=1 to Length(S1) step 1
      S3:= S3+S1[I]
  End
  For I:=Poz to Length(S2) step 1
      S3:= S3+S2[I]
  End
  S2:= S3
End
```

Includerea unui subşir într-un şirul de caractere

S1 =	3	X	Y	Z

S2 =	5	A	B	C	D	E

Poz = 4

I	1	2	3
S3	A	B	C

I	1	2	3
S3	A	B	C	X	Y	Z
---	---	---	---	---	---	---

I	4	5
S3	A	B	C	X	Y	Z	D	E
---	---	---	---	---	---	---	---	---

Modelul de prelucrare secvențială înainte

Elementele unui vector pot fi șiruri de caractere.

```
Type
 Cuvint = String[8]
 Tb = Array[1..6] of Cuvi
Var
 Lista: Tb; I, J: Natural
Begin
 For I:=1 to 6 step 1
  ReadString(Lista[I])
 End
 For I:=1 to 6 step 1
   For J:=1 to length(Lista[i])-1 step 1
     If Lista[i][j] > Lista[i][j+1] then
       ...
End
```

	1	2	3	4	5	6	7	8
1	M	A	R	I	A			
2	V	A	S	I	L	E		
3	A	N	A					
4	P	E	T	R	U			
5	M	I	H	A	I			
6	I	O	N					

Sarcini pentru lucrul independent

1. Se citește un text de la tastatură. Să se stabilească dacă textul conține doar litere și caracterul '.'
2. Să se elaboreze o procedură care are 3 parametri - șiruri de caractere. Cel de-al treilea șir fiind obținut prin concatenarea primelor două șiruri.
3. Să se elaboreze un algoritm care numără aparițiile unui anumit caracter într-o secvență de caractere citită de la tastatură.