V-191

Limbajul C# pentru începători
Noțiuni de bază

Liviu Negrescu, Lavinia Negrescu

3. ŞIRURI DE CARACTERE

În acest capitol revenim la şirurile de caractere pentru a preciza unele aspecte privind definirea şi utilizarea acestora. În prima parte a capitolului indicăm modul în care se pot defini şi gestiona şirurile de caractere. Astfel, un şir de caractere se poate reprezenta printr-un literal şir (vezi paragraful 1.3.3.), dar mai există şi alte posibilităţi cum ar fi de exemplu definirea unui tablou ale cărui elemente sunt de tip *char* şi care au ca valori caracterele din compunerea şirului respectiv sau o altă posibilitate este de a instanţia obiecte ale clasei *string* în care putem păstra şiruri de caractere.

În partea a doua a cestui capitol se arată că programele C# pot fi lansate în execuţie utilizând parametri care sunt interpretaţi ca obiecte ale clasei *string.*

3.1. Din nou despre şiruri de caractere

Prin *şir de caractere* înţelegem o succesiune de zero sau mai multe caractere oarecare incluse între ghilimele.

În paragraful 1.3.3. s-au introdus şirurile de caractere. Cu acestă ocazie s-a indicat faptul că într-un şir de caractere putem utiliza *secvenţe escape* (secvenţe de forma \ **...**) pentru caractere *neimprimabile* (amintim că sunt *imprimabile* caracterele care au codul ASCII în intervalul [32,126]). O excepţie de la această regulă o reprezintă caracterele *ghilimele* şi caracterul *backslash.* Deşi aceste caractere sunt imprimabile ele pot fi componente ale unui şir de caractere numai dacă le reprezentăm prin *secvenţele lor escape*. Aceasta este normal să fie aşa deoarece ghilimele se utilizează pentru a defini caracterele ce aparţin unui şir de caractere. Deci, pentru a include caracterul ghilimele într-un şir de caractere, vom utiliza *secvenţa escape:*

\”

Caracterul *backslash* se poate reprezenta într-un şir de caractere dublându-l:

\\

Şirurile de caractere se pot păstra în *tablouri* ale căror elemente sunt de tip *char.*

Ştim că un tablou unidimensional se declară astfel:

```
tip [ ] nume;
```

unde *nume* este numele tabloului, iar *tip* este tipul comun elementelor tabloului. În cazul tablourilor destinate pentru a păstra şiruri de caractere, *tip* va fi cuvântul rezervat *char:*

```
char [ ] sir;
```

În paragraful 1.5.1. s-a arătat că la declararea tablourilor, acestea se pot şi iniţializa. Tablourile ale căror elemente sunt de tip *char* se pot iniţializa prin caractere incluse între acolade şi separate prin virgulă

Exemplu

1. `char [] sir = {'+', '-', '(', ')','*','/'}`

Tabloul *sir* declarat în acest fel are 6 elemente, fiecare este de tip *char* şi au următoarele valori iniţiale:

`sir[0]` – codul ASCII al caracterului +
`sir[1]` – codul ASCII al caracterului -
`sir[2]` – codul ASCII al caracterului (
`sir[3]` – codul ASCII al caracterului)
`sir[4]` – codul ASCII al caracterului *
`sir[5]` – codul ASCII al caracterului /

Şirurile de caractere se pot păstra şi în obiecte ale clasei *String.* Clasa *String* se află în spaţiul de nume *System.* Un obiect al clasei *String* se poate declara sub forma:

```
System.String numeobiect;
```

Evident, în cazul în care este prezentă directiva *using System* putem să declarăm mai simplu obiectul respectiv, astfel:

```
String numeobiect;
```

Există şi un sinonim pentru *System.String* şi anume, cuvântul *string*. Astfel, indiferent dacă este prezentă sau nu directiva *using System;* putem întotdeuna să utilizăm cuvântul *string* pentru a declara obiecte ale clasei *String,* astfel:

```
string numeobiect;
```

Aici, *string* nu înseamnă altceva decât clasa *String* din spaţiul de nume *System.*

În cele ce urmează vom accepta faptul că *string* este şi el un *nume de tip* şi anume acelaşi *tip* care este definit de clasa *String* din spaţiul de nume *System*.

Obiectele clasei *string* pot fi iniţializate la declaraea lor. În acest scop putem folosi o declaraţie de forma:

```
string nume="...";
```

Prin această declaraţie, *nume* este un obiect al clasei *string* şi el conţine şirul de caractere situat la dreapta caracterului =.

De exemplu declaraţia:

```
string sir="abc";
```

declară pe *sir* ca obiect al clasei *string* şi acesta este iniţializat cu şirul de caractere "abc".

Un obiect al clasei *string* poate fi parametru al metodei *WriteLine*, deci apelul de mai jos este corect:

```
Console.WriteLine(sir);
```

În urma acestui apel se va afişa pe ecranul monitorului caracterele *abc* din compunerea şirului păstrat în obiectul *şir*.

Ştim că acelaşi efect se obţine dacă am fi apelat metoda *WriteLine* astfel:

```
Console.WriteLine("abc");
```

În fine, metoda *WriteLine* se poate apela având ca parametru şi un tablou ale cărui elemente sunt de tip *char*. Acelaşi efect se obţine dacă considerăm declaraţia:

```
char []tab = {'a','b','c'};
```

şi apoi apelul metodei *WriteLine* având ca parametru pe *tab*:

```
Console.WriteLine(tab);
```

În limbajul C# putem defini tablouri ale căror elemente să fie obiecte ale clasei *string*. Un astfel de tablou îl vom declara printr-o declaraţie de forma:

```
string []nume;
```

unde *nume* este numele tabloului care se declară. De asemenea, la declararea unui tablou ale cărui elemente sunt de tip *string* se pot indica şi valori pentru iniţializarea elementelor tabloului. În acest scop, vom utiliza o declaraţie de forma:

```
string []nume = {sir1, sir2, . . ., sirn};
```

unde *sir1, sir2, . . ., sirn* sunt şiruri de caractere.