1. ITERAREA PE TABLOURI

Iterarea pe tablouri înseamnă **accesarea elementelor** tablorilor. Acest lucru se poate realiza folosind instrucțiunile ciclice ale limbajului C#.

De exemplu, pentru a afişa valorile elementelor unui tablou putem folosi o instrucțiune **for**:

```
tip[] tab = {...};
```

unde **tip** este un tip predefinit sau tipul **string**.

```
for(int i = 0;i<tab.Length;i++)
   Console.WriteLine(tab[i]);
```

Acelaşi lucru se poate realiza utilizând o instrucțiune **foreach** în locul instrucțiunii **for**:

```
tip[] tab = {...};
foreach(int i in tab)
   Console.WriteLine(i);
```

Instrucțiunea **foreach** poate fi folosită şi în situații în care utilizăm tablouri de **tipuri implicite**.

Fie de exemplu clasa **angajat** definită astfel:

```
class angajat{
   string nume,adresa;
   ushort salar;
   public angajat(string n,string a,ushort s)
   {
      nume = n;
      adresa a;
      salar = s;
   }
   public void afis()
   {
      Console.WriteLine(nume+",adresa:"+
                        adresa+",salar="+salar);
   }
}
```

Fie instanțierile:

```
angajat a1 = new angajat(...);
angajat a2 = new angajat(...);
...
angajat an = new angajat(...);
```

Definim variabila **angajati** ca o referință la un tablou ale cărui elemente sunt inițializate cu elementele **a1, a2, ..., an**:

```
var angajati = new []{a1, a2, ..., an};
```

Fololsind o instrucțiune **foreach** putem să afișăm componentele obiectelor **a1, a2, ..., an**:

```
foreach(var v in angajati)
        v.afis();
```

Despre corpul instrucțiunilor **for**, **while** sau **foreach** se spune că formează un **bloc iterator**.

Într-un bloc iterator se pot folosi **instrucțiuni yield** (**produce** sau **oferă**, tradus în limba română).

O instrucțiune **yield** are formatele:

```
yield return expresie;
yield break;
```

Exerciții

1.1. Să se scrie o aplicație care definește clasa **angajat** amintită mai sus, iar în metoda **Main** se definește tabloul **angajati** (vezi mai sus) și apoi se afișează valorile elementelor realizând o **iterare** printr-o instrucțiune **foreach**.

Proiectul aplicației va avea numele **IterareForeach1**.

Mai jos este listat textul sursă al aplicației.

În figura 1.1. este afișată o fereastră MS-DOS ce conține rezultatele execuției aplicației.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace IterareForeach1
{
   class Program
   {
      static void Main(string[] args)
      {
         angajat a1 = new angajat("Popescu Ion",
            "Eminescu 32 Cluj-Napoca", 12345);
         angajat a2 = new angajat("Pop Ana",
            "Paris 100 Cluj-Napoca", 10000);
         angajat a3 = new angajat("Muresan Mihai",
            "Rebreanu 25 Cluj-Napoca", 9999);
         var angajati = new [ ]{ a1, a2, a3 };
         foreach (var v in angajati)
                  v.afis();
      }
   }
   class angajat
   {
      string nume;
      string adresa;
      ushort salar;
      public angajat()
```

```
        {
        }
        public angajat(string n, string a, ushort s)
        {
           nume = n;
           adresa = a;
           salar = s;
        }
        public void afis()
        {
           Console.WriteLine(nume + ", adresa: " + adresa +
                             ", salar = " + salar);
        }
    }
}
```

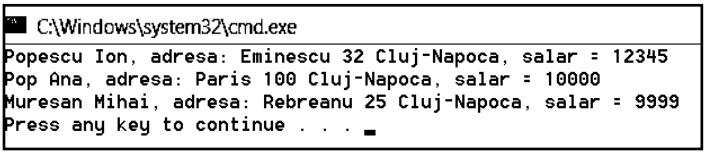

Figura 1.1.

REZUMAT

Iterarea pe tablori înseamnă accesarea elementelor tablorilor. Se poate realiza folosind instrucțiunile **while**, **for** sau **foreach**. Corpul acestor instrucțiuni formează un **bloc iterator**.

Instrucțiunea **foreach** se poate folosi şi în sitații în care se utilizează tablouri de **tipuri implicite**.

Într-un **bloc iterator** se pot folosi **instrucțiuni yield** pentru a ieşi din ele.

Formatul **instrucțiunii yield**:

```
yield return expresie;
yield break;
```

V-241

Limbajul C# pentru începători
delegare, evenimente, fire de execuţie, data calendaristică şi ora, clasa ArrayList

Liviu Negrescu, Lavinia Negrescu