CAPITOLUL 1. **PRELIMINARII**

1.1 Prima aplicație C#

Limbajul C# este un limbaj relativ tânăr, a apărut în jurul anului 2000, a fost dezvoltat de catre compania Microsoft ca un limbaj simplu, ce permite programarea structurată, modulară si orientată obiect. Este foarte asemanator cu limbajul Java și moștenește sintaxa și principiile de programare ale limbajului C/C++, însă într-o maniera mult simplificată.

1) Crearea aplicatiilor de tip Consola:

File/New Project/Visual C#/Windows/Clasic Desktop/Console Application

2) Structura unui program C#:

Un program C# este un ansamblu de clase, grupate în spații de nume. Fiecare clasă conține una sau mai multe funcții, dar una singură conține funcția principală Main() („punctul de intrare” = entry point). Așadar, instrucțiunile sunt grupate în metode (funcții), metodele în clase, iar clasele în spații de nume.

Cel mai simplu program C#, care dealtfel nu face nimic, este:

```
using System;
namespace ConsoleApplication1 //spatiu de nume
{
   class Program //clasa
   {
      static void Main() //functia Main
      {

      }
   }
}
```

Trebuie precizat încă de pe acum că limbajul C# face distincţie între literele mari şi cele mici: este un limbaj case-sensitive.

Exemplul 1:

```
using System; //spatiu de nume pentru clasa Console
namespace ConsoleApplication
{
   class Program
   {
      static void Main()
      {
        Console.WriteLine("Prima mea aplicatie C#!");
        Console.ReadKey();

      }
   }
}
```

Programul va afişa:

Prima mea aplicatie C#!

Orice instrucţiune se încheie cu punct şi virgulă (;).Un şir de caractere, cum este mesajul afişat, se încadrează între ghilimele.

Mai multe instrucţiuni pot fi scrise pe o singură linie. Reciproc, o instrucţiune poate fi scrisă pe mai multe linii. Aplicaţia de mai sus poate fi rescrisă astfel:

```
using System;
namespace ConsoleApplication1
{
   class Program
   {
      static void Main(string[] args)
      {
         Console.WriteLine("Prima "); Console.WriteLine(
            "mea aplicatie C#!");
         Console.ReadKey();
```

```
    }
  }
}
```

Textul va fi afişat pe un singur rând:
Prima mea aplicatie C#!

În aplicaţia care urmează ne propunem să adunăm două numere întregi preluate de la terminal. Înainte de a lucra cu cele două numere, ele trebuiesc declarate. Rolul unei declaraţii este de a informa compilatorul asupra numelui şi tipului variabilei. O declaraţie are sintaxa:
<tip> <nume1>, <nume2>, ...,<nume_n>;

Pentru numere întregi, limbajul C# dispune de tipul primitiv de date *int*. Aşadar, vom putea declara cele două numere întregi, fie ele *a* şi *b* astfel:
int a,b;

Citirea de la terminal se face cu funcția *Console.ReadLine()*, însă aceasta returnează un text care trebuie convertit într-un număr întreg pentru prelucrarea numerică pe care ne-am propus-o. Conversia unui text (string) către un întreg se face cu *Convert.ToInt32*.

Astfel, citirea de la terminal a unui număr întreg a va fi:
a = Convert.ToInt32(Console.ReadLine());

Afișarea se va face cu *Console.WriteLine*. De exemplu, apelul:
Console.WriteLine("Suma lor este: " + suma);
afisează conținutul variabilei suma, precedată de un text.

ReadLine și *WriteLine* sunt două funcții din clasa *Console*, declarată în spațiul de nume (namespace) *System*, inclus la începutul aplicației:
using System;

Fără includerea spațiului de nume, cele două funcții puteau fi utilizate astfel: *System.Console.ReadLine* si *System.Console.WriteLine()*.

Exemplul 2:

```
using System;
namespace ConsoleApplication1
{
   class Program
   {
      static void Main(string[] args)
      {

         Console.WriteLine("Introduceti doua numere intregi:");
         int a, b, suma;
         a = Convert.ToInt32(Console.ReadLine());
         b = Convert.ToInt32(Console.ReadLine());
         suma = a + b;
         Console.WriteLine("Suma lor este: " + suma);

         Console.ReadKey();
         Console.ReadKey();
}
   }
}
```

Output:

```
Introduceti doua numere intregi:
3
5
Suma lor este: 8
```

Dacă dorim să afişăm, ca rezultat, şi valorile variabilelor *a* şi *b*, vom rescrie ultimul apel *Console.WriteLine*:

```
         Console.WriteLine(a+"+"+b+"="+suma);
```

Output:

```
Introduceti doua numere intregi:
3
5
3 + 5 =8
```

Exemplul 3:
O altă aplicaţie are rolul de a citi un număr întreg de la terminal şi de a-i afişa radicalul. Pentru extragerea radicalului putem utiliza funcţia *sqrt* (square root = rădăcină pătrată) a clasei Math, declarată tot în spațiul de nume System. Pentru stocarea rădăcinii pătrate se va folosi o variabilă reală de tip *double*.

```
using System;
namespace ConsoleApplication1
{
   class Program
   {
      static void Main(string[] args)
      {

         Console.WriteLine("Introduceti un numar intreg:");
         int a = Convert.ToInt32(Console.ReadLine());

         double rad = Math.Sqrt(a);
         Console.WriteLine("Radicalul lui "+a+" este : " + rad);

         Console.ReadKey();
          }
   }
}
```

Output:
Introduceti un numar intreg:
8
Radicalul lui 8 este : 2,82842712474619

Observăm că numerele reale sunt afişate cu zecimale. Putem rotunji radicalul la un anumit numar de zecimale cu ajutorul funcției Round a clasei Math. De exemplu, putem calcula rezultatul rotunjit la numai 2 zecimale, rescriind apelul WriteLine:

```
Console.WriteLine("Radacina patrata este : " + Math.Round(rad,2));
```

Output:
Introduceti un numar intreg:
8
Radacina patrata este : 2,83

Exemplul 4:

```
using System;
namespace ConsoleApplication1
{
   class Program
   {
      static void Main(string[] args)
      {

         Console.WriteLine("Introduceti numele dvs:");
         string nume = Console.ReadLine();
         Console.WriteLine("Buna, " + nume + "!");

         Console.ReadKey();
      }
   }
}
```

Output:
Introduceti numele dvs:
Daniela
Buna, Daniela!

Exemplul 5: Să se determine partea întreagă şi partea fracţionară a unui număr real preluat de la terminal.

```
using System;
  class Program
   {
   static void Main()
      {
```

```
    Console.Write("Introduceti un numar real:");
    double d = double.Parse(Console.ReadLine());

    Console.WriteLine("Nr. initial: "+d);
    Console.WriteLine("Parte intreaga: " + (int)d);
    Console.WriteLine("Parte zecimala: " + (d-(int)d));
    Console.ReadKey();
    }
  }
```

Output:
Introduceti un numar real:7,555
Nr. initial: 7,555
Parte intreaga: 7
Parte zecimala: 0,555

Cele trei instrucțiuni:
Console.WriteLine("Nr. initial: "+d);
Console.WriteLine("Parte intreaga: " + (int)d);
Console.WriteLine("Parte zecimala: " + (d-(int)d));
pot fi compactate într-una singură astfel:
Console.WriteLine("Nr. initial: "+d+"\nParte intreaga: " + (int)d+"\nParte zecimala: " + (d-(int)d));
unde secvența \n determină salt la rândul următor.

Ultima afișare poate fi încă o dată rescrisă, folosind afișarea formatată, astfel:
Console.WriteLine("Nr. initial: {0}\nParte intreaga: {1}\nParte zecimala: {2}", d, (int)d, d-(int)d);
unde cei trei descriptori {0}, {1} și {2} vor fi înlocuiți la afișare cu valorile celor trei expresii: d, (int)d și d-(int)d.

Sau, și mai simplu:
Console.WriteLine("{0} = {1} + {2}", d, (int)d, d-(int)d);
care va afișa:
Output:
Introduceti un numar real:7,555
7,555 = 7 + 0,555