Laborator 13 – C# 3.0 și 4.0

Obiective

După completarea acestui laborator veți dobândi următoarele cunoștințe:

- Definirea parametrilor opționali și utilizarea parametrilor cu nume
- Inițializarea obiectelor cu câmpuri publice
- Folosirea variabilelor cu tip anonim
- Scrierea de cod care execută dinamic metode
- Utilizarea de lambda expresii
- Prelucrarea folosind LINQ a colecțiilor

Condiţii prealabile

Înainte de a realiza acest laborator trebuie să fiţi familiarizaţi cu următoarele concepte:

- Crearea și apelarea metodelor în C#
- Utilizarea ciclurilor împreună cu vectori
- Clase și obiecte

➢ Exerciţiul 1- Parametrii opționali si parametrii cu nume

În acest exercițiu veți crea o funcție care returnează câte numere dintr-un vector *int[]* se găsesc în intervalul descris de primii 2 parametrii și care sunt divizibile cu al treilea parametru. Ultimii 2 parametrii vor fi opționali. Funcția se va apela utilizând parametrii cu nume.

Crearea metodei

- Creați o metodă statică numită **FilterCount** de tipul *int* care are ca parametrii un vector de tipul *int[]* și 3 întregi în această ordine.
  - **int[] vector**
  - **int min**
  - **int max**
  - **int divisor**
- Creați o variabilă de tipul *int* numită **count**.
  - Inițializați variabila cu 0
- Creați un ciclu **foreach** sau **for** care pentru fiecare element al variabilei **vector** execută urmatoarele:
  - Verifică dacă elementul este mai mare decât **min** și mai mic decât **max.**
  - Dacă da, atunci verifică dacă este divizibil cu **divisor.**
  - Dacă da, atunci incrementează count.

- La sfârșitul metodei returnați valoarea variabilei **count.**

Crearea parametrilor impliciți

- Având în vedere că este vorba de o funcție cu rol de filtru, valorile pentru parametrii opționali trebuie alese astfel încât să nu filtreze elemente.
- Pentru parametrul **max** asignați valoarea implicită **int.MaxValue**.
  - Reprezintă cel mai mare întreg posibil.
- Pentru parametrul **divisor** asignați valoarea 1.
  - Toate numerele sunt divizibile cu 1.

Metoda Main

- Creați un vector și atribuiți câteva valori la întamplare în intervalul [0, 50] .
  - Atribuiți valori la inițializare.
- Apelați funcția **FilterCount** utilizând parametrii cu nume doar atunci când este strict necesar pentru a realiza următoarele:
  - Filtrați elementele în intervalul [0, 20] inversând ordinea parametrilor
  - Filtrați elementele mai mari ca 5 divizibile cu 2
  - Filtrați după toți cei 3 parametrii inversând ordinea ultimilor 2 parametrii

Exercițiul 2 – Inițializarea obiectelor și variabilelor cu tip anonim

În acest exercițiu veți crea o structură numită **Point3D** care va conține 3 câmpuri publice reprezentând coordonatele unui punct tridiminsional. Veți asigna valori câmpurilor unei astfel de variabile la inițializare. Veți crea o variabilă cu tip anonim cu 2 câmpuri pe care o veți folosi pentru a extrage valorile a 2 câmpuri dintr-un obiect **Point3D**. La final veți afișa câmpurile variabilei cu tip anonim.

Crearea structurii **Point3D**

- Într-un fișier nou numit “Point3D.cs” creați structura cu același nume având 3 câmpuri publice de tipul *int* numite **x**, **y** și **z**.

- Metoda Main

- Creați o variabilă de tipul **Point3D** numită **point** și atribuiți valori la inițializare.
  - Folosiți aceeași construcție ca în cazul inițializarii unui obiect
    - **new Point3D()**
    - **new Point3D** - valabil doar dacă urmează să fie asignate valori
  - Adăugați { } și folosiți numele câmpurilor pentru a asigna valori.
    - **{ x = 1, y = 2, z = 3 };**
- Creați o variabilă cu tip anonim numită **variabile** care stochează valorile a 2 câmpuri din variabila **point.**
  - **var variable = new { oy = point.y, oz = point.z };**
- Afișați cele 2 valori stocate în variabila cu tip anonim **variable.**

➢ Exercițiul 3 – Legarea dinamică

În acest exercițiu veți utiliza legarea dinamică pentru a afișa, utilizând aceeași metodă, lungimea unui vector și a unui șir de caractere.

- Metoda PrintLength

- Creați o metodă statica care nu intoarce nici o valoare numită **PrintLength** care are ca parametru un obiect dinamic numit **obj**.
  - public static void PrintLength(dynamic obj)
- În corpul metodei afișați lungimea obiectului folosing metoda **Length.**

Observați ce se sugerează Visual Studio în momentul în care tastați "**obj.**" .

- Metoda Main

- Creați un vector de orice lungime și un **string.**
- Apelați metoda **PrintLength** pentru cele 2 variabile create mai sus.

➢ Exercițiul 4 – Metode extensie

În acest exercițiu va trebui să adăugați funcționalitate clasei string astfel încât să puteți afișa text indentat și cu diferite simboluri la începul, în funcție de nivelul de indentare.

Metoda CreateIndentation

- Creați o clasă statică numită **StringExtension** în fișierul Program.cs.
- În interiorul acestei clase creați o metodă publică statică numită **CreateIndentation** care primește ca argument un întreg numit **level** și întoarce o valoare de tipul *string*.
  - **public static string CreateIndentation(int level)**
- Înainte de primul parametru inserați un parameru de tipul *string* precedat de cuvântul cheie *this*.
  - **this string str**
- În corpul metodei folosiți instrucțiunea *switch* pentru a acoperii mai multe posibilități de indentare.
  - pentru valoarea "1" adăugați înaintea lui **str** "\tx "
    - **str = "\tx " + str;**
  - pentru valoarea "2" adăugați înaintea lui str "\t\t+ "
    - **str = "\t\t+ " + str;**
  - pentru valoarea "3" adăugați înaintea lui str "\t\t\to "
    - **str = "\t\t\to " + str;**

Metoda Main

- Folosiți *Console.WriteLine* pentru a testa metoda **CreateIndentation** a unui obiect de tipul *string*.
  - Console.WriteLine("heading 1".CreateIndentation(1));
  - Console.WriteLine("heading 2".CreateIndentation(2));
  - Console.WriteLine("heading 3".CreateIndentation(3));

Exercițiul 5 – Expresii lambda

În acest exercițiu veți utiliza o lambda expresie pentru a putea sorta descrescător un vector folosind metoda *Sort*.

Metoda Main

- Declarați un vector de întregi numit **vector**
  - **int[] vector = {1, 5, 2, 9, 4, 0};**
- Utilizând *Array.Sort* sortați vectorul **vector** utilizând ca expresie lambda care primește 2 parametrii și returnează valoarea primului scăzut cu al doilea
  - **Array.Sort(vector, (x,y) => y-x);**
- Afișați valoarea fiecărui element utilizând instrucțiunea *foreach*.