\times

8. gyakorlat

A nyolcadik gyakorlaton a C# nyelvvel ismerkedünk meg konzolos illetve *Windows Forms* alapú grafikus alkalmazásokon keresztül.

Feladat 1.

Valósítsuk meg a Racionális számok típusát a 4 alapművelettel. A Visual Studio 2019 elindítása után válasszuk a *Create a new projekt* menüpontot.

| pen recent | | Get sta | rted | |
|---|---|------------|--|--|
| ہ ب | 2019. 10. 26. 15:49 | * | Clone or check out code Get code from an online repository like GitHub or Azure DevOps | |
| C:\Users\pepe\source\repos\gyakorlat_8_console ConsoleApp1.sln C:\Users\pepe\source\repos\ConsoleApp1 | 2019. 10. 23. 11:50 | Ъ | Open a project or solution Open a local Visual Studio project or .sln file | |
| GameOfLife_CSharp.sin C\\C# and C++ and F# and VB\C#,C++,F#,VB\GameOfL App1.sin C\UserSynene\source\report\App1 | 2019. 09. 10. 22:59 ife\GameOfLife_CSharp 2019. 05. 27. 18:46 | | Open a local folder Navigate and edit code within any folder | |
| WpfApp1.sln C\Users\pepe\source\repos\WpfApp1 | 2019. 05. 18. 9:06 | * D | Create a new project Choose a project template with code scal | |
| WpfApp4.sln C:\Users\pepe\Documents\Visual Studio 2017\Projects\W | 2019. 01. 01. 12:13 pfApp4 | | Continue without code → | |
| ■ WpfApp3.sln C\Users\pepe\Documents\Visual Studio 2017\Projects\W | 2018. 12. 19. 11:14 pfApp3 | | | |

A következő ablakban pedig válasszuk ki a **Console App (.Net Framework)** projekt templatet:

| reate a new p | roject | Search for templates (Alt+S) | <u>C</u> lear |
|------------------------------|--------|--|---------------|
| Recent project templates | | C# • All Platforms • All Project Type | es * |
| Console App (.NET Framework) | C# | WPF User Control Library (NET Core) Windows Presentation Foundation user control library | * |
| WPF App (.NET Framework) | C# | C# Windows Desktop Library | |
| | | Blank App (Universal Windows) A project for a single-page Universal Windows Platform (UWP) app that has no predefined controls or layout. C Windows Xbox UWP Desktop Class Library (Universal Windows) | 1 |
| | | TQ1B A project for creating a managed class library (.dll) for Universal Windows Platfi (UWP) apps. C# Windows Library UWP | m |
| | | Azure Cloud Service (classic) A project for creating a scalable service that runs on Microsoft Azure. | |
| | | Console App (NET Framework) A project for creating a command-line application Ce Windows Console | |
| | | - CO ARGUNATION FORMA AND FAITH FORMATION | - |
| | | Back | Next |
| | | | |

A *next* gombra való kattintás után megadhatjuk a projekt és a solution nevét. A .NET megoldásokban (Solution) gondolkodik, egy Solution több projektet is tartalmazhat. Alapértelmezetten a solution neve megegyezik az első projekt nevével, de ezt lehet módosítani. Az induló projekt a *Solution Explorer*ben a projekt nevén jobb klikk, majd a *Set as StartUp Project* menüpontot választva adható meg.

| Annova Canada Canada Caldarni yaraha Canada Canada Samenoo K Mil Temenoo K A 22 | CONCOLO DOD / BILL LEDINGOLOGICA CA Management | | |
|---|---|---|--|
| Ingent gener Felded) Joalion Clamitgenit sound repail genetic (Lannale) generation AET framework 422 • | LONSOLE App (INET Framework) C* Windows Console | | |
| Feddal conton CCUbertgepelsowstrepolgskolds (Lonsole •) jemenosk METframenosk 472 • | voject Dame | | |
| Jordion CUSheri Japer Saunch repairing also nie di Lancole (Jammonok MET Frenenosk 4/2 - | Feladat3 | | |
| CUbergregelaworkreperigyskolid, Lonnole , , , , , , , , , , , , , , , , , , , | position | | |
| venerok AtT famerok 422 - | C/Users\pepe\source\repos\gyakorlat_8_console | · | |
| APT Francesch 472 - | ramasovir | | |
| | NET Example 4.7.2 | | |
| | and Fridinghout And a | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

A Create gomb megnyomása után elkezdhetjük szerkeszteni az alkalmazásunk kódját.



A racionális típus reprezentálására adjuk a projektünkhöz egy új osztályt, a név legyen **Rational**.



A Racionális számok két egész számmal reprezentálhatók. Készítsünk számukra egy Nominator és egy Denominator tulajdonságot az egyszerűbb használhatóság érdekében, valamint egy üres és egy paraméteres konstruktort!

Az alapműveletek megvalósítására az adott műveletnek megfelelő statikus publikus operator függvényt használjuk C# ban.

```
public static Rational operator *(Rational a, Rational b)
{
    return new Rational
    {
        p = a.p * b.p,
        q = a.q * b.q
    };
}
```

Egy-egy racionális szám megjelenítéséhez definiáljuk felül az object osztálytól örökölt toString() függvényt override kulcsszóval.

```
public override string ToString()
{
    return p + "/" + q;
}
```

A szövegként megadott racionális szám Rational típusra konvertálásához készítsünk szintén statikus Parse függvényt. A Parse függvény egy lehetséges megvalósítása:

```
1 reference
public static Rational Parse(string number)
ſ
   int poz = number.IndexOf('/');
   if (poz == -1)
   {
        return new Rational(Int32.Parse(number), 1);
   }
   else if (poz == 0)
   {
        return new Rational(1, Int32.Parse(number.Substring(1)));
   }
   else if (number.EndsWith("/"))
    {
        return new Rational(Int32.Parse(number.TrimEnd('/')), 1);
    }
   else
    {
        return new Rational(Int32.Parse(number.Split('/')[0]), Int32.Parse(number.Split('/')[1]));
    }
```

A main függvényben példányosítsunk két racionális számot és írjuk ki az összegüket.

```
namespace Feladat1
{
    Oreferences
    class Program
    {
        Oreferences
        static void Main(string[] args)
        {
            Rational a = new Rational(1, 2);
            Rational b = Rational.Parse("3/2");
            Console.WriteLine("{0} plusz {1} egyenlő {2}", a, b, a + b);
            Console.ReadKey();
        }
    }
}
```

Feladat 2.

Töltsünk fel egy 1 dimenziós tömböt és egy 2 dimenziós tömböt szöveg fájlból és írjuk ki a tartalmukat a képernyőre. A feladat megvalósításához oldjuk fel a System.IO névteret a típusnevek egyszerűsítése érdekében: *using System.IO*;

A szövegfájl elérésére egy TextReader objektumot használunk, melyet egy StreamReader osztály segítségével példányosítunk:

TextReader reader = new StreamReader(filename);

A szövegfájlból egy sor beolvasása a TextReader osztály ReadLine() függvényével történik. A függvény visszatérési értéke *string* típusú. Ha a függvény már a szövegfájlból nem tud olvasni akkor null értékkel tér vissza.

Szövegből számba konvertálást például az Int32.Parse(str) utasítás tud végezni. Egy szöveget lehetőség van tömbbé konvertálni egy megadott elválasztójel (pl. vessző) mentén az str.Split(',') függvény segítségével.

Ne felejtsük el bezárni az olvasót a használat végén!

Feladat 3.

Készítsünk grafikus alkalmazást mely egy szövegdobozba beírt számsorozatból kiválasztja a legnagyobb értéket, megadja a számok átlagát, és a számokat hozzáadja egy ListBox vezérlőhöz. Hozzunk létre új *Windows Forms App (.Net Framework)* projektet vagy adjunk hozzá egy újat a meglévő solutionhöz:

| Recent project templates | | C# | All <u>P</u>latforms | - | All Project <u>T</u> ypes | |
|---|----|-------------------------------|---|--|--|---|
| Console App (.NET Framework) WPF App (.NET Framework) | C# | Cit Console A projec C# | e App (.NET Framework) ct for creating a command-line ap Windows Console | oplication | | |
| | | Window A projec | vs Forms App (.NET Framework) ct for creating an application with Windows Desktop | a Windows Forms (| WinForms) user interface | l |
| | | ASP.NE Project Forms, I | T Web Application (.NET Framewor templates for creating ASP.NET a MVC, or Web API applications and Windows Cloud Web | ork) pplications. You can d add many other fe: | create ASP.NET Web atures in ASP.NET. | |
| | | A project | brary (.NET Framework) ct for creating a C# class library (.4 Windows Library | dll) | | |
| | | A project | brary (.NET Core) ct for creating a class library that t Windows Linux macOS | targets .NET Core. Library | | |
| | | Azure V | VebJob (.NET Framework) | | | |

A megjelenő tervező felületen helyezünk el az űrlapon a következő vezérlőket:

- 3 db TextBox -szöveg beviteli és megjelenítő vezérlő
- 3 db Button elsődleges parancs vezérlő
- 1 db ListBox lista megjelenítő, az elemeket a vezérlő *Items* listájához kell hozzáadni
- 4 db Label elsődleges szöveg megjelenítő vezérlő
- •

A grafikus elemek tulajdonságait a *Properties* ablakban tudjuk szerkeszteni tervező nézetben: Például az ablak (Form1) felirata a Text tulajdonság megadásával változtatható:



Az egyes vezérlőket a baloldali eszköztárból (*Toolbox*) tudjuk kiválasztani. A legnagyobb szám és az átlag megjelenítésére szolgáló TextBox-ok *ReadOnly* tulajdonságát állítsuk *True* -ra. A feladatot a 3 gombhoz tartozó *Click* eseményvezérlők fogják megoldani. Az egyes eseményvezérlők szerkesztéséhez kattintsunk duplán az egyes gombokra.

A TextBox vezérlő Text tulajdonsága tartalmazza a beleírt szöveget, jelen esetben számsorozatot. A string objektum tömbbé darabolásához használjuk a String osztály *Split()* függvényét.

```
Egyféle elválasztó karakter (pl. szóköz):
textBox1.Text.Split(' ');
Többféle elválasztó karakter:
textBox1.Text.Split(new char[] {' ',',','\t',';' });
Egy szövegről a String osztály IsNullOrEmpty(str) statikus metódusa dönti el, hogy
üres-e.
```

Érdemes listát (fontosabb műveletei: Add, Count) használni a számok tárolására, és kezelni, amikor a felhasználó nem egész számot ad meg. Ebben az esetben a MessageBox.Show("Nem egész szám"); utasítással elég jelezni, hogy hibás a bemenet.

Feladat 4.

Legördülő menü (*ComboBox*) vezérlő és több Ablak használatának demonstrációja. Egy műszaki boltban alkatrészeket lehet a kiválasztani egy listából, majd a kosár tartalmának az összértékét meghatározni. A listát lehet bővíteni új alkatrészekkel. Az elkészítendő alkalmazás 2 űrlapot használ, a fő ablak (Form1):

| 🖳 Feladat4 | | | _ | | \times |
|---|--------------|-----------|---|---------|----------|
| Válassz egy alkatrészt | Új alkatrész | | | | |
| | Kosárba | | | | |
| SelectedIndex értéke SelectedItem értéke | Kosár | Osszérték | | Kilépés | |
| | | | | | |

Az új alkatrész bevitelére szolgáló ablak (Form2):

| Kód | | - | | \times |
|-----|--|---|---------|----------|
| Név | | [| Hozzáad | |
| Kód | | [| Bezár | |
| Ar | | | | |

Az alkatrészeket reprezentáló Product osztály:

| Product Class | * |
|---|---|
| Properties | |
| Code Name Price | |
| Methods | |
| ToString | |

A főablak eseménykezelői:

- A ComboBox (comboBox1) vezérlő SelectedIndexChanged eseménykezelője
- A 4 Button (newBtn, exitBtn, button1, button2) vezérlő Click eseménykezelője
- A *SelectedIndexChanged* esemény hatására a két textbox-ba jelenjen meg a kiválasztott elem index-e illetve objektuma.
- A newBtn Click eseménykezelője példányosítsa a Form2 osztályt és nyissa is meg (if (newForm.ShowDialog() == DialogResult.OK){...}), majd az ablak bezárása után aktualizálja a ComboBox vezérlő Items listáját.
- A button1 *Click* eseménykezelője adja hozzá a kosárhoz (listbox1) a kiválasztott alkatrészt.
- A button2 *Click* eseménykezelője összegezze a kosárba lévő alkatrészek árait.

A Form2 ablakban a Hozzáad gombra kattintva adja hozzá a Főablak leíró osztályának egy listájához a Product osztály egy példányát. Ehhez a Form2 rendelkezzen egy Product adattaggal, ami a gomblenyomásra kap értéket, amennyiben jó a bemenet. Jó bemenetnél záródjon is be a form (DialogResult = DialogResult.OK; Close();). Amennyiben a felhasználó mégsem ad meg új alkatrészt, a DialogResult = DialogResult.Cancel; Close(); utasításokkal jelezhető.)

Hasznos link: A Windows Forms vezérlők összefoglalója: <u>http://www.dotnetperls.com/controls</u>