Programozás C# nyelven

10. előadás

http://e-learning.ujs.sk/

Struktúra („struct”): hasonló, mint az osztály („class”).

Különbségek:

Osztály	**Struktúra**
referencia (reference) típus	érték (value) típus
továbbszármaztatható	nem származtatható

Példák:

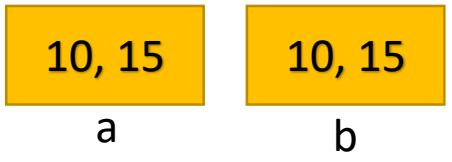
Osztály	**Struktúra**
string, String	int, Int32
object, Object	double, Double
Random	Point
Pen	Rectangle
SolidBrush	

```
public class PontOsztaly
{
    public int X { set; get; }
    public int Y { set; get; }

    public PontOsztaly(int x, int y)
    {
        X = x;
        Y = y;
    }

    public override string ToString()
    {
        return "(" + X + "," + Y + ")";
    }
}
```

```
public struct PontStruktura
{
    public int X { set; get; }
    public int Y { set; get; }

    public PontStruktura(int x, int y)
    {
        X = x;
        Y = y;
    }

    public override string ToString()
    {
        return "(" + X + "," + Y + ")";
    }
}
```

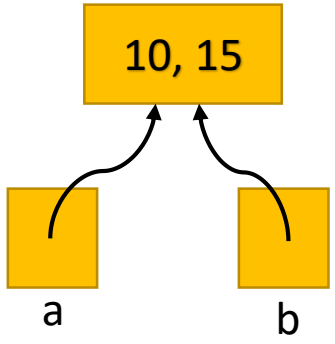
```
private void button1_Click(object sender, EventArgs e)
{
    textBox1.Clear();
    PontStruktura a = new PontStruktura(10, 15);
    PontStruktura b = a;
    textBox1.AppendText("A pont: " + a.ToString() + "\r\n");
    textBox1.AppendText("B pont: " + b.ToString() + "\r\n");
    b.X = 8;
    b.Y = 23;
    textBox1.AppendText("A pont: " + a.ToString() + "\r\n");
    textBox1.AppendText("B pont: " + b.ToString() + "\r\n");
}
```

Érték típus:




```
private void button2_Click(object sender, EventArgs e)
{
    textBox1.Clear();
    PontOsztaly a = new PontOsztaly(10, 15);
    PontOsztaly b = a;
    textBox1.AppendText("A pont: " + a.ToString() + "\r\n");
    textBox1.AppendText("B pont: " + b.ToString() + "\r\n");
    b.X = 8;
    b.Y = 23;
    textBox1.AppendText("A pont: " + a.ToString() + "\r\n");
    textBox1.AppendText("B pont: " + b.ToString() + "\r\n");
}
```

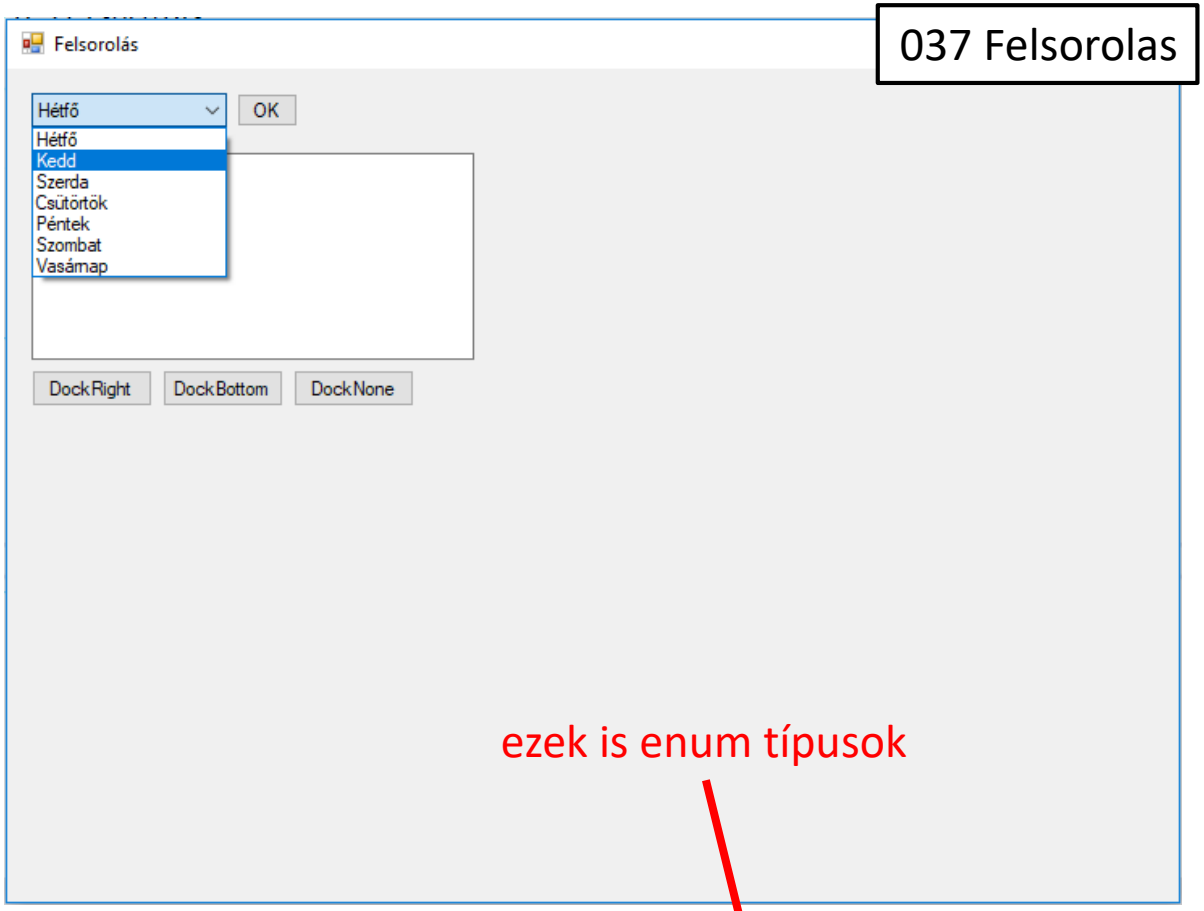
Referencia típus:



Felsorolás („enum”): úgy képzelhetjük el, mint konstansok halmazát.

```
public enum Nap
{
    Hétfő = 1,
    Kedd = 2,
    Szerda = 3,
    Csütörtök = 4,
    Péntek = 5,
    Szombat = 6,
    Vasárnap = 7,
    Ismeretlen = -1
}
```

037 Felsorolas



ezek is enum típusok

```
private void Form1_Load(object sender, EventArgs e)
{
    this.FormBorderStyle = FormBorderStyle.FixedSingle;
    comboBox1.DropDownStyle = ComboBoxStyle.DropDownList;
    comboBox1.SelectedIndex = 0;
}
```

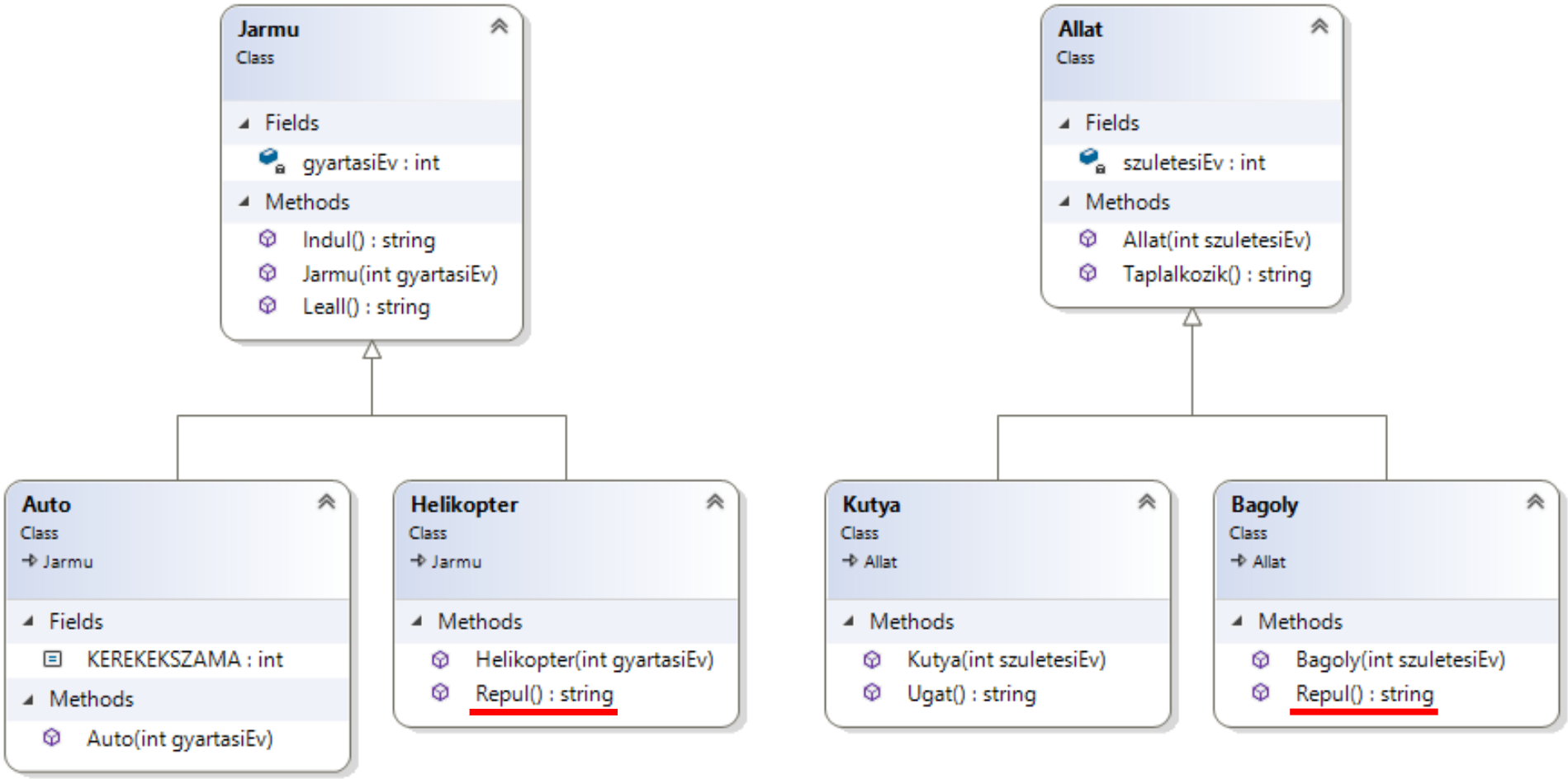
```
private void button1_Click(object sender, EventArgs e)
{
    Nap n = Nap.Hétfő;
    switch (comboBox1.Text)
    {
        case "Hétfő":
            n = Nap.Hétfő;
            break;
        case "Kedd":
            n = Nap.Kedd;
            break;
        case "Szerda":
            n = Nap.Szerda;
            break;
        case "Csütörtök":
            n = Nap.Csütörtök;
            break;
        case "Péntek":
            n = Nap.Péntek;
            break;
        case "Szombat":
            n = Nap.Szombat;
            break;
        case "Vasárnap":
            n = Nap.Vasárnap;
            break;
    }
    label1.Text = "A kiválasztott nap: " + n + ", a hét " + (int)n + ". napja.";
}
```

```
private void button2_Click(object sender, EventArgs e)
{
    textBox1.Dock = DockStyle.Right;
}

private void button3_Click(object sender, EventArgs e)
{
    textBox1.Dock = DockStyle.Bottom;
}

private void button4_Click(object sender, EventArgs e)
{
    textBox1.Dock = DockStyle.None;
}
```

Interfész („interface”): implementáció nélküli metódusok neveit tartalmazza. Pl.:

```
namespace System
{
    public interface IComparable
    {
        int CompareTo(object obj);
    }
}
```

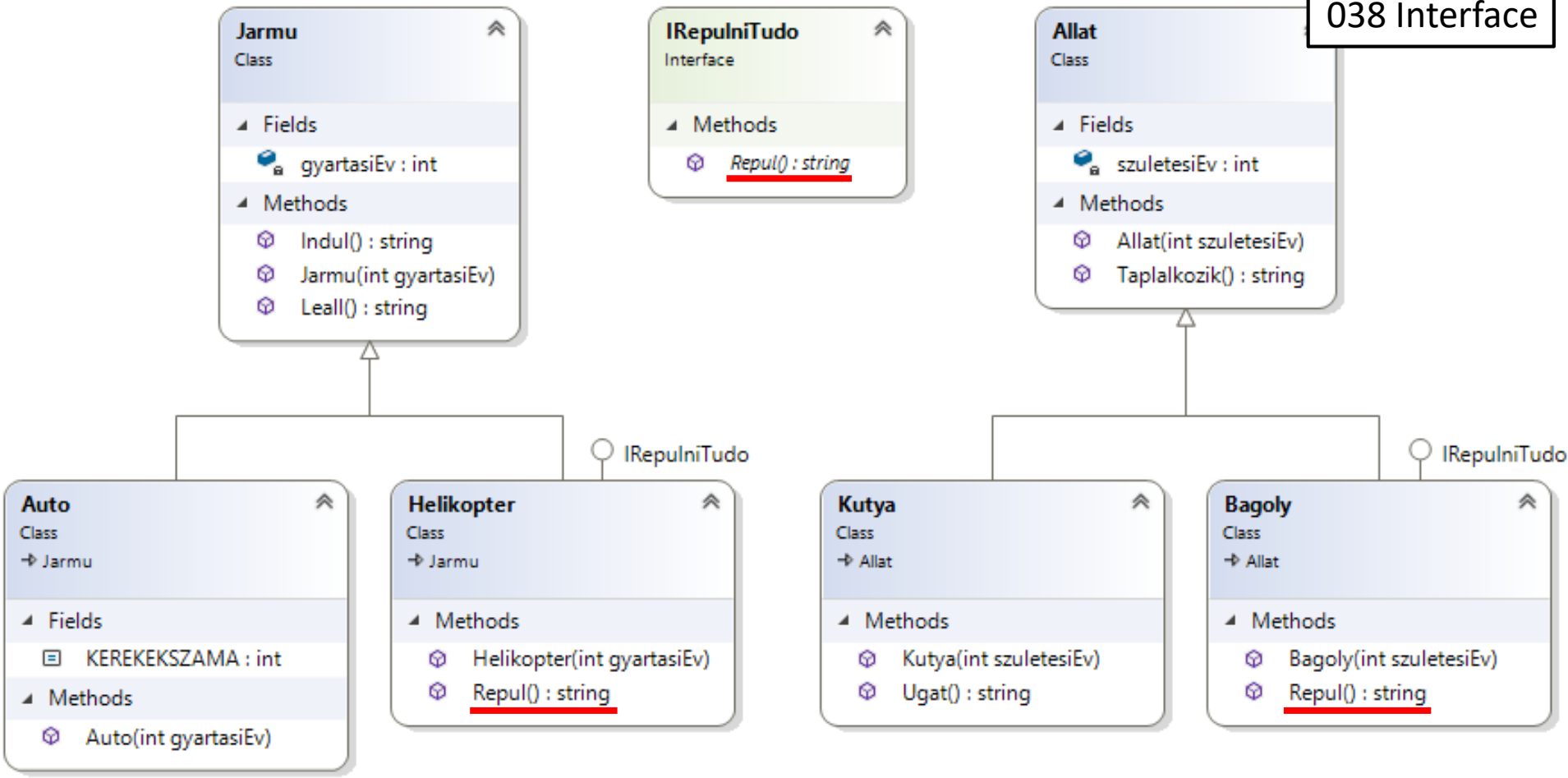
Mire jó egy interfész? Nézzünk egy egyszerűsített példát…




```
private void Repules(TextBox tbox,            valami)
{
    tbox.AppendText(valami.Repul() + "\r\n");
}
```

Milyen típus legyen itt? „Helikopter” és „Bagoly” osztályokból létrehozott objektumokat szeretnénk ide átadni paraméterként.

038 Interface




```
private void Repules(TextBox tbox, IRepulniTudo valami)
{
    tbox.AppendText(valami.Repul() + "\r\n");
}
```

```
public interface IRepulniTudo
{
    string Repul();
}
```

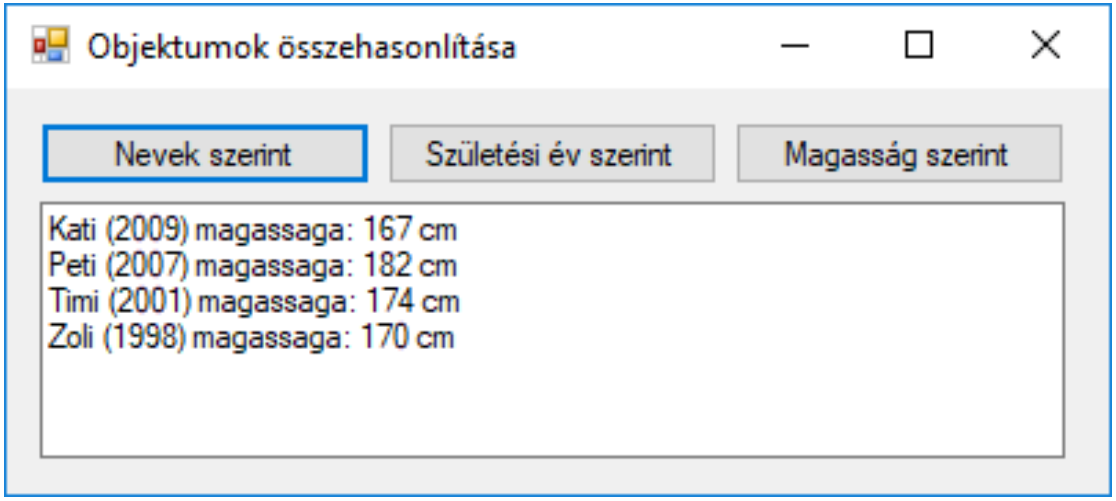
```
public class Helikopter : Jarmu, IRepulniTudo
{
    public Helikopter(int gyartasiEv) : base(gyartasiEv)
    {
    }

    public string Repul()
    {
        return "A helikopter repül a város felett.";
    }
}
```

A C#-ban is léteznek interfészek, melyeket mi is használhatunk a programjainkban. Pl.:

- Az „**IComparable**” interfész segítségével tudjuk megoldani saját osztályunkból létrehozott objektumok rendezését valamilyen adatmezőjük szerint (természetes sorrend). Pl. személyek név szerinti rendezése.

- Az „**IComparer**” interfész segítségével további „rendező” objektumokat hozhatunk létre, amelyek segítségével a saját osztályainkból létrehozott objektumok rendezhetők más-más adatmező szerint. Pl. személyek születési év szerinti vagy magasság szerinti rendezése.

- Az „**IEnumerable**” interfészt megvalósító objektumok bejárhatók foreach ciklus segítségével. Pl. egy verem adatszerkezetet implementáló osztály elemei.



039 Objektumok osszehasonlítása

```
private Szemely[] szemelyek;

private void Form1_Load(object sender, EventArgs e)
{
    szemelyek = new Szemely[4];
    szemelyek[0] = new Szemely("Peti", 2007, 182);
    szemelyek[1] = new Szemely("Kati", 2009, 167);
    szemelyek[2] = new Szemely("Zoli", 1998, 170);
    szemelyek[3] = new Szemely("Timi", 2001, 174);
}
```

```
private void button1_Click(object sender, EventArgs e)
{
    Array.Sort(szemelyek);
    textBox1.Clear();
    foreach (Szemely x in szemelyek)
    {
        textBox1.AppendText(x + "\r\n");
    }
}

private void button2_Click(object sender, EventArgs e)
{
    Array.Sort(szemelyek, Szemely.SzulEvSzerintiRendezes());
    textBox1.Clear();
    foreach (Szemely x in szemelyek)
    {
        textBox1.AppendText(x + "\r\n");
    }
}

private void button3_Click(object sender, EventArgs e)
{
    Array.Sort(szemelyek, Szemely.MagassagSzerintiRendezes());
    textBox1.Clear();
    foreach (Szemely x in szemelyek)
    {
        textBox1.AppendText(x + "\r\n");
    }
}
```

Ehhez a Szemely osztálynak meg kell valósítania az **IComparable** interfészt!

Ez a metódus egy olyan osztályból létrehozott objektumot ad vissza, amely megvalósítja az **IComparer** interfészt.

Ez a metódus egy olyan osztályból létrehozott objektumot ad vissza, amely megvalósítja az **IComparer** interfészt.

```
public class Szemely : IComparable
{
    private string nev;
    private int szulEv;
    private int magassag;

    public Szemely(string nev, int szulEv, int magassag)
    {
        this.nev = nev;
        this.szulEv = szulEv;
        this.magassag = magassag;
    }

    public override string ToString()
    {
        return nev + " (" + szulEv + ") magassaga: " + magassag + " cm";
    }

    // az objektumokat alapertelmezetten nev szerint hasonlitjuk ossze
    public int CompareTo(object obj)
    {
        Szemely sz = (Szemely)obj;
        return nev.CompareTo(sz.nev);
    }
```

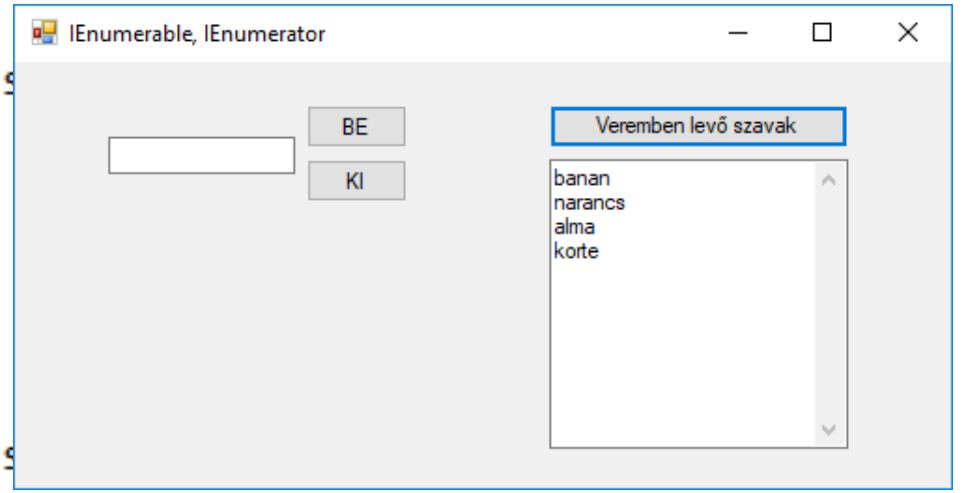
```
// ----- szuletesi ev szerinti osszehasonlitashoz -----
private class SzulEvComparer : IComparer
{
    public int Compare(object x, object y)
    {
        Szemely sz1 = (Szemely)x;
        Szemely sz2 = (Szemely)y;
        return sz1.szulEv - sz2.szulEv;
    }
}

public static IComparer SzulEvSzerintiRendezes()
{
    return new SzulEvComparer();
}
```

```
// ----- magassag szerinti osszehasonlitashoz -----
private class MagassagComparer : IComparer
{
    public int Compare(object x, object y)
    {
        Szemely sz1 = (Szemely)x;
        Szemely sz2 = (Szemely)y;
        return sz1.magassag - sz2.magassag;
    }
}
```

```
    public static IComparer MagassagSzerintiRendezes()
    {
        return new MagassagComparer();
    }
}
```

040 IEnumerable, IEnumerator

```
Verem v = new Verem(10);
```




```
private void button1_Click(object s
{
    v.ElemBe(textBox1.Text);
    textBox1.Clear();
}

private void button2_Click(object s
{
    textBox1.Text = v.ElemKi();
}

private void button3_Click(object sender, EventArgs e)
{
    textBox2.Clear();
    foreach(string s in v)
    {
        textBox2.AppendText(s + "\r\n");
    }
}
```

```
public class Verem : IEnumerable
{
    private string[] elemek;
    private int n = 0;

    public Verem(int meret)
    {
        elemek = new string[meret];
    }

    public void ElemBe(string s)
    {
        if (n < elemek.Length)
        {
            elemek[n] = s;
            n++;
        }
    }
```

```
    public string ElemKi()
    {
        if (n > 0)
        {
            n--;
            return elemek[n];
        }
        return null;
    }

    public IEnumerator GetEnumerator()
    {
        string[] tomb = new string[n];
        Array.Copy(elemek, 0, tomb, 0, n);
        return new VeremEnumerator(tomb);
    }
}
```

```
public class VeremEnumerator : IEnumerator
{
    private string[] tomb;
    private int index;

    public object Current
    {
        get
        {
            return tomb[index];
        }
    }

    public VeremEnumerator(string[] tomb)
    {
        this.tomb = tomb;
        index = tomb.Length;
    }
```

```
    public bool MoveNext()
    {
        index--;
        return index >= 0;
    }

    public void Reset()
    {
        index = tomb.Length;
    }
}
```

Összefoglalás:

- Struktúra (struct)
- Felsorolás (enum)
- Interfész (interface)
  - IComparable, IComparer
  - IEnumerable, IEnumerator