Igen 😊 És **pontosan arra gondolsz**, ami a régebbi VB-s programozásból is ismerős lehet: előre deklaráljuk a változókat, majd a TextBox tartalmát beletesszük, közben számmá alakítjuk.

Érdemes ezt most rendbe tenni, mert kezdő C#-nál ez egy nagyon fontos alap.

1. Először: miért kell átalakítani?

A TextBox tartalma mindig:

```
C#
txtSugar.Text
```

és ez **string**, vagyis szöveg.

Ha például ezt írja be a felhasználó:

```
5
```

attól ez még a program számára:

```
C#
"5"
```

Tehát szöveg.

Ezért ez nem működik:

```
C#
double eredmeny = 2 * txtSugar.Text * Math.PI;
```

Mert nem lehet egy számot (`2`) közvetlenül egy szöveggel megszorozni.

---

2. A leggyakrabban használt átalakítások

`int.Parse()`

Egész számhoz:

```
C#
int szam = int.Parse(txtSzam.Text);
```

Például:

```
25
```

`double.Parse()`

Tizedes számhoz:

```
C#
double szam = double.Parse(txtSzam.Text);
```

Például:

```
25,5
```

`decimal.Parse()`

Pénzügyi számításoknál használható:

```
C#
decimal ar = decimal.Parse(txtAr.Text);
```

---

3. Van `Convert` is

C#-ban találkozhatsz ezekkel is:

```
C#
int szam = Convert.ToInt32(txtSzam.Text);
```

vagy:

```
C#
double szam = Convert.ToDouble(txtSzam.Text);
```

Tehát például:

```
C#
double sugar = Convert.ToDouble(txtSugar.Text);
```

Ez is működik.

Kezdőként viszont én ezt tanítanám:

```C#
double sugar = double.Parse(txtSugar.Text);
```

Egyszerűbb és nagyon jól megmutatja, hogy **szöveget számmá alakítunk**.

---

4. És van a `TryParse`

Ez egy biztonságosabb megoldás.

```C#
double.TryParse(txtSugar.Text, out double sugar);
```

Ez viszont már egy kicsit másképpen működik.

A `TryParse` megpróbálja átalakítani a szöveget.

Ha sikerül:

```
"5,5" → 5,5
```

ha nem sikerül, nem dob hibát, hanem `false` értéket ad.

Például:

```C#
double sugar;

if (double.TryParse(txtSugar.Text, out sugar))
{
    lblEredmeny.Text = (2 * sugar * Math.PI).ToString();
}
else
{
    MessageBox.Show("Hibás adat!");
}
```

Ezt én csak később tanítanám, amikor már az `if` is megvan.

---

5. Amit te mondasz: előre deklaráljuk a változókat

Igen! 👍

Lehet így:

```
C#
double sugar;
double kerulet;

sugar = double.Parse(txtSugar.Text);

kerulet = 2 * sugar * Math.PI;

lblKerulete.Text = kerulet.ToString();
```

Ez szerintem **kezdő tanulóknál kifejezetten jó megoldás**, mert szépen látszik a folyamat.

1. változó létrehozása

```
C#
double sugar;
```

2. TextBoxból adat beolvasása és átalakítása

```
C#
sugar = double.Parse(txtSugar.Text);
```

3. számítás

```
C#
kerulet = 2 * sugar * Math.PI;
```

4. eredmény kiírása

```
C#
lblKerulete.Text = kerulet.ToString();
```

---

6. Ugyanez rövidebben

A két első lépést össze is lehet vonni:

```
C#
double sugar = double.Parse(txtSugar.Text);
```

Tehát:

```
C#
double sugar;
sugar = double.Parse(txtSugar.Text);
```

ugyanazt jelenti, mint:

```
C#
double sugar = double.Parse(txtSugar.Text);
```

Kezdőknél én először akár a hosszabb változatot is használnám, mert jobban megértik, hogy:

> létrehozom a változót → értéket adok neki.

---

7. És akkor a `ToString()` – miért kell a végére?

Na, ez nagyon fontos! 🙂

Pont fordítva történik, mint az elején.

Az elején:

```
C#
double sugar = double.Parse(txtSugar.Text);
```

szövegből számot csinálunk.

A végén:

```
C#
lblKerulete.Text = kerulet.ToString();
```

számból szöveget csinálunk.

Miért?

Mert a Label:

```
C#
lblKerulete.Text
```

szintén **stringet vár**.

Tehát:

```
TextBox
   ↓
string
   ↓ Parse
double
   ↓
számolás
   ↓
double
   ↓ ToString()
string
   ↓
Label
```

Ez az egész folyamat. 🔄

---

8. Egy teljes példa

Kör kerületének kiszámítása:

```
C#
private void btnSzamol_Click(object sender, EventArgs e)
{
    double sugar;
    double kerulet;

    sugar = double.Parse(txtSugar.Text);

    kerulet = 2 * sugar * Math.PI;

    lblKerulete.Text = kerulet.ToString();
}
```

Ezt szerintem **nagyon jó első mintakódként** megtanítani.

---

9. És mi van, ha TextBoxba akarjuk kiírni?

Pont ugyanaz:

```
C#
txtEredmeny.Text = kerulet.ToString();
```

A Label és a TextBox ebből a szempontból ugyanúgy működik:

```
C#
lblEredmeny.Text = kerulet.ToString();
```

vagy

```
C#
txtEredmeny.Text = kerulet.ToString();
```

Mindkettőnek **string kell**.

---

10. Ezt a kis táblázatot érdemes megtanulni

Mit akarunk?	Kód
Szöveg → egész szám	`int.Parse()`
Szöveg → tizedes szám	`double.Parse()`
Szöveg → pénzügyi szám	`decimal.Parse()`
Szöveg → egész szám biztonságosan	`int.TryParse()`
Szöveg → tizedes szám biztonságosan	`double.TryParse()`
Szöveg → egész szám Convert-tel	`Convert.ToInt32()`
Szöveg → double Convert-tel	`Convert.ToDouble()`

Mit akarunk?	Kód
Szám → szöveg	`ToString()`

Én a ti tananyagotokban ezt a sorrendet követném:

1. `int.Parse()`
2. `double.Parse()`
3. `ToString()`
4. változók külön deklarálása
5. `Convert.To...`
6. `TryParse()` + `if`

Így nem zúdítjuk rájuk egyszerre az összes lehetőséget, hanem először megértik az alapelvet:

> **A TextBoxból szöveg jön → számmá alakítjuk → számolunk → az eredményt szöveggé alakítjuk → kiírjuk.**

Ez a WinForms-os kezdő feladatoknál az egyik legfontosabb alap.