Szövegkezelés

`string szoveg = "informatika óra";` ←idézőjel!

A stringeket sok tekintetben úgy kezelhetjük, mint karakterek (char) tömbjét.

A szöveg i-dik karaktere:

`szoveg[i];`

`Pl. szoveg[2]` → `f`

Ha valamelyik karaktert akarjuk vizsgálni: `szoveg[i]=='e';` ← aposztrofot kell használni, nem idézőjelet

A szöveg hossza (hány karakter):

`szoveg.Length` //15 (beleszámítja a szóközt is!)

Helyettesítés:

`szoveg.Replace(" ","_")` → a szóközt lecseréli _-ra; informatika_óra

Tartalmazás:

`szoveg.Contains("forma")` → true értéket ad vissza, ha tartalmazza a megadott szöveget, false értéket ad vissza, ha nem

Nagybetűssé/kisbetűssé alakítás:

`szoveg.ToUpper()`

`szoveg.ToLower()`

Törlés a szövegből:

`szoveg.Remove(11,4)` A 11 indexű karaktertől kezdve 2 darab karaktert töröl: informatika óra→ informatika. Ha nem adjuk meg a második paramétert, akkor a szöveg végéig töröl.

Beszúrás a szövegbe:

`szoveg.Insert(index, beszúrandó_szöveg)` A megadott indextől kezdve beszúrja a megadott szöveget.

Szövegrész:

`szoveg.Substring(index, darab)` A megadott indextől kezdve a megadott darab karaktert adja vissza.

Szövegrész helye:

`szoveg.IndexOf('f')` A megadott karakter vagy szöveg (kezdő) helyének indexét adja vissza. Ha nem található, akkor -1-et.

`szoveg.IndexOf("for");`

Darabolás:

```
string darabok=szöveg.Split(' '); //több karaktert is meg lehet adni vesszővel elválasztva, aposztrofok között
                                  //két „darab” keletkezett: a darabok[0] → informatika
                                  //és a darabok[1] → óra
```

String átalakítása Char típusú tömbbé:

```
Char[] szovegTomb = szoveg.ToCharArray();
```

Így most már a szovegTomb-re alkalmazhatjuk az Array osztály metódusait, például a rendezést:

```
Array.Sort(szovegTomb);  //informatika → aafiikmort
```

Megjegyzés: kiíratásnál nem kell karakterenként végigmenni a tömbön, egyben ki lehet íratni:

Console.WriteLine(szövegTömb);