F Á J L K E Z E L É S

Készítette: Monok Judit

Fájlkezelés

Adatainkat sokszor fájlokban tároljuk, ezért fájlokat kell írnunk és olvasnunk. Ehhez szükségünk van az IO névtérre, melyet a using részben vehetünk fel:

using System.IO;

Fájlok kezelése Stream-ekkel:

A fájlok használatakor létre kell hoznunk egy példányt (fs) a FileStream osztályból:

FileStream példánynév = new FileStream(”filenév elérési útvonallal”, megnyitás módja);

FileStream fs = new FileStream(”adatok.txt”, FileMode.OpenOrCreate);

Ha nem adunk meg elérési útvonalat, akkor a fájl a Debug mappába kerül íráskor, illetve onnan olvassuk be. Ezért beolvasáskor nekünk kell biztosítani, hogy a fájl ott legyen!!!

A használat módja többféle lehet, ha az OpenOrCreate-t használjuk, akkor írni és olvasni is tudjuk a megadott fájlt.

Ha befejeztük a fájlokkal való műveleteket, akkor le kell zárni:

példánynév.Colse();
fs.Close();

Fájlok írása Stream-ekkel:

Fájlok írásakor egy írót (sw) kell példányosítanunk a StreamWriter osztályból:

StreamWriter sw = new StreamWriter(fs);

Ezt követően írhatunk a fájlba:

sw.Write(”alma”);

vagy

sw.WriteLine(”alma”);

Itt is érvényes a Write és a WriteLine közötti különbség: ha a Write-t használjuk, akkor a következő kiírás ugyanabba a sorba fog kerülni, ha pedig a WriteLine-t akkor a következő kiírás új sorba kerül.

Ha befejeztük az írást, akkor ezt is le kell zárni:

sw.Close();

Pl.: Írjuk ki a *hello.txt* fájlba: HELLO!

```
FileStream fs = new FileStream(”hello.txt”, FileMode.OpenOrCreat);
StreamWriter sw = new StreamWriter(fs);
sw.Write(”HELLO”);
sw.Close();
fs.Close();
```

A FileStream osztály példányosítását el is hagyhatjuk, ekkor a StreamWriter példányosításakor adjuk meg a fájl elérési útvonalát és nevét:

```
StreamWriter sw = new StreamWriter(”filenév elérési útvonallal”);
```

Itt is érvényes: ha nem adunk meg elérési útvonalat, akkor a Debug mappába kerül a fájl.

Pl.:

```
StreamWriter sw = new StreamWriter(”hello.txt”);
sw.Write(”HELLO”);
sw.Close();
```

A) Egyetlen sornyi adat kiírása:

Pl.:Írjuk ki az első 5 pozitív számot, szóközzel elválasztva egymás mellé!

```
StreamWriter sw = new StreamWriter(”szamok.txt”);
for (int i = 1; i < 6; i++)
        {
                sw.Write(”{0}  ”, i);
        }
sw.Close();
```

B) Több sornyi adat kiírása:

Pl.:Írjuk ki az első 5 pozitív számot, egymás alá!

```
StreamWriter sw = new StreamWriter(”szamok.txt”);
for (int i = 1; i < 6; i++)
        {
                sw.WriteLine(i);
        }
sw.Close();
```

C) Több sor, soronként több adat

Pl.:Írjunk ki 5 sorban soronként 3 számot, szóközzel elválasztva!

```
StreamWriter sw = new StreamWriter(”szamok.txt”);
for (int i = 1; i < 6; i++)
        {
                for (int j = i; j < i+3; j++)                // egy sornyi szám kiírása
                        {
                                sw.Write(”{0}  ”, j);
                        }
                sw.WriteLine();         //ezzel biztosítjuk, hogy új sorba kerüljön a következő három szám
        }
sw.Close();
```

Fájlok kiírása Stream-ek nélkül:

A) Egyetlen sornyi adat kiírása:

File.WriteAllText(”filenév elérési útvonallal”, sztring);

Pl.:

File.WriteAllText(”hello.txt”, ”HELLO”);

vagy:

```
string szoveg = ”HELLO”;
File.WriteAllText(”hello.txt”, szoveg);
```

Szöveg típusú tömb elemeinek elválasztó karakterrel egyetlen sornyi szöveggé való átalakítását a következő módon végezhetjük el:

string.Join(”elválasztó karakter”, szöveg típusú tömb);

Pl:

```
string[] adatok = {”1”, ”2”, ”3”, ”4”};
string kiiras = string.Join(”:”, adatok);
File.WriteAllText(”szamok.txt”, kiiras);
```

Ez a módszer megoldja a Random számok kiírását is, ha először egy szöveg típusú tömbbe átkonvertáljuk szöveggé.

Pl:

```
Random rnd = new Random();
string[] adatok = new string[10];
for (int i = 0; i < adatok.Length; i++)
  {
      adatok[i] = Convert.ToString(rnd.Next(1, 100));
  }
string kiiras = string.Join(":", adatok);
File.WriteAllText("proba.txt", kiiras);
```

B) Több sornyi adat kiírása:

Több sornyi adat kiírásakor szükségünk lesz egy szöveg típusú tömbre (kiir), amelybe tároljuk a kiírandó sorokat.

File.WriteAllLines(”filenév elérési útvonallal”, szöveg típusú tömb);

Pl.:

```
string[] kiir=new string{”alma”, ”körte”, ”szilva”};
File.WriteAllLines(”gyumolcsok.txt”, kiir);
```

C) Több sor, soronként több adat

Ebben az esetben az előző két rész megoldásait kell ötvöznünk. Először biztosítani kell, hogy az elemek a megfelelő elválasztó karakterrel együtt egy sorba kerüljenek. Majd ezeket a sorokat egy sztring típusú tömbben tároljuk. Itt is hasznos lehet a string.Join(" ", sztring típusú tömb) megoldás.

Pl.: Írjunk ki 5 sorba, soronként 3 darab 1 és 100 közé eső véletlenszámot * karakterrel el elválasztva a *veletlenek.txt* fájlba!

```
Random rnd = new Random();
string[] egy_sornyi = new string[3];
string[] kiir = new string[5];
for (int i = 0; i < kiir.Length; i++)
{
   for (int j = 0; j < egy_sornyi.Length; j++)
   {
      egy_sornyi[j] = Convert.ToString(rnd.Next(1, 101));       //véletlenek sztringgé alakítása
   }
   kiir[i] = string.Join("*", egy_sornyi);       // egy sornyi adat sztringgé alakítása
}
File.WriteAllLines("veletlenek.txt", kiir);    // a sztring típusú tömb kiírása
```

Feladatok:

Több sor, soronként 1 adat

A következő feladatoknál minden adatot egymás alá, új sorba írass ki!

1. Írj programot, amely 1-től 10-ig kiírja a számokat a *novekvo.txt* állományba!
2. Írj programot, amely kiírja az első 15 négyzetszámot a *negyzetek.txt* állományba!
3. Írj programot, amely kiír 10 darab 100 és 200 közé eső véletlenszámot a *veletlenek.txt* állományba!
4. Írj programot, amely Peti 15 matekjegyét írja ki a *matek.txt* fájlba!
5. Írj programot, amely egy 28 fős osztály minden tanulójának a lábméretét (35-45) írja ki a *cipomeret.txt* állományba!
6. Írj programot, amely kiírja ez első 10 páros számot a *parosak.txt* állományba!
7. Írj programot, amely kiír 100 bitet (csupa 1-et és 0-t) a *bitek.txt* állományba!
8. Írj programot, amely kiír 100 darab N és F betűt! Generálj 1-et és 2-t, ha 1-t generál a program, akkor F betű, ha 2-t, akkor pedig N betű kerüljön a *nemek.txt* fájlba!
9. Írj programot, amely az előző feladathoz hasonló módszerrel 500 darab fej-et vagy írás-t generál a *penzdobas.txt* fájlba!
10. Egy szellemi vetélkedőn mérték a 120 résztvevő IQ-ját. Mindegyikük átlag feletti intelligenciát mutatott. Készíts programot, amely kiírja az *intelligencia.txt* állományba az IQ értéküket (110 és 145 közötti értékek).
11. 2015. novemberében minden nap kiszámolták az átlagos légnyomást. A légnyomás ebben a hónapban 993 hPa és 1041,1 hPa között ingadozott. Írj programot, amely ennek megfelelően kiírja a *legnyomas.txt* fájlba a napi átlagokat!
12. Írj programot, amely150 darab 65-90 és 97-122 közötti ASCII kódokat generál, majd a generált kódnak megfelelő karaktereket egymás alá kiírja a *karakterek.txt* állományba!
13. Írj programot, amely a felhasználótól bekéri 20 gyümölcsnek a nevét, majd egymás alá kiírja a *gyumolcsok.txt* fájlba!
14. Véradáson 150 ember jelent meg. Vércsoportjaikat (A, B, AB, 0) feljegyezték. Készíts programot, amely a vércsoportokat írja ki a *veradas.txt* fájlba! Használd a generálást (0-1-2-3) a vércsoportok megadásához, valamint többirányú elágazást a kiíráshoz!
15. A 2015-ös F1 Magyar Nagydíjat Sebastian Vettel nyerte, míg a 14. helyen körhátrány nélkül PastorMaldonado érkezett be. Vettel ideje 1:46:09.985, Maldonadoé 1:47:35.127. Készíts programot, amely 12 különböző számot generál 6369985 és 6455127 között, majd a kapott számok ezredrészét a
    TimeSpan.FromSeconds(valóssá konvertált szám/1000)
    átalakítással írasd ki Vettel és Maldonado idejével együtt a *forma1.txt* fájlba.

Egy sornyi adat

A következő feladatoknál minden adatot egyetlen sorba, egymás mellé írass ki!

16. Egy számsorozat első tagja 22, a következő tagját úgy kapjuk, hogy az előzőhöz hozzáadunk 4-et. Írj programot, amely kiírja ennek a számsorozatnak az első 10 tagját szóközzel elválasztva a *sorozat.txt* állományba!
17. Egy dobókockával 50-szer dobunk. Írj programot, amely kiírja a *dobokocka.txt* fájlba a dobásokat egymás mellé szóközzel elválasztva!
18. A felhasználótól egy mondat szavait kérjük be, egészen addig, amíg a beírt szöveg utolsó karaktere nem pont. Írj programot, amely kiírja egymás mellé szóközzel elválasztva a szavakat a *mondat.txt* állományba!
19. Írj programot, amely 1 és 50 között 10 különböző véletlenszámot generál! A számokat szóközzel elválasztva *kicsik.txt* fájlba írd ki!
20. Írj programot, amely a felhasználótól országokat kér be. Az utolsó ország Magyarország legyen! Az országokat az *orszagok.txt* fájlba írasd ki egymás mellé * karakterrel elválasztva!
21. Csabi a totózóba megy. Feladja a heti tippjeit (13+1) a mérkőzésekre (1 – az otthoni csapat nyer, 2 – az idegenben játszó csapat nyer, x – döntetlen az eredmény). A 13+1 es szelvény tippjeit a *toto.txt* fájlba egymás mellé vesszővel elválasztva írasd ki!
22. Budapest és Vác távolsága légvonalban 32 km. Budapesten a pesti oldal legalacsonyabb tengerszint feletti magassága 100 m, Vác legnagyobb tengerszint feletti magassága 146 m. 4 km-enként megmérték a tengerszint feletti magasságot. Írj programot, amely a mért értékeket a *tengerszint.txt* fájlba szóközzel elválasztva egymás mellé írja!
23. A skandináv lottón 35 számból 7-t kell megjelölni. Írj programot, amely a *skanditipp.txt* fájlba szóközzel elválasztva kiírja a heti tippjeinket!
24. Egy atlétika versenyen Kiss Pista 6 alkalommal ugrik távol. Az eredményeit cm-ben mérve szóközzel elválasztva írasd ki a *tavol.txt* fájlba. Ha az eredménye 600 cm alatti, akkor az ugrást tekintsük belépettnek, eredménye 0 legyen! A magyar rekordot Szalma László tartja 830 cm-rel 1985 óta.
25. Az edzőtábor legnehezebb napján komoly fizikai megterhelésnek van kitéve minden résztvevő. Kati 3 x 800 m-t és 10 x 60 m-t fut. A 800 m időeredményeit 2:45 min és 3:30 min között futja, míg a 60 m-t 9 s és 12 s között. Írj programot, amely kiírja Kati eredményeit szóközzel elválasztva egymás mellé a *futas.txt* fájlba!

Több sor, soronként több adat

26. Egy gépjármű tulajdonos havonta átlagosan 1 alkalommal tankol. Ekkor feljegyzi az előző tankolás óta megtett kilométerek számát (480 és 540 között), valamint az elhasznált üzemanyag mennyiséget (36,7 és 39,8 között). Írj programot, amely a *fogyasztas.txt* fájlba kiírja egy teljes év adatait úgy, hogy egy sorba egy hónap két adata kerül szóközzel elválasztva (km liter)!
27. A vércsoportokat, AB0 (A, B, AB, 0) és Rh (+, -) rendszer alapján 8 csoportba sorolhatjuk: A+, A-, B+, B-, AB+, AB-, 0+, 0-. Írj programot, amely 100 ember vércsoportját írja ki a *vercsoportok.txt* állományba. Egy sorba szóközzel elválasztva kerüljön az ABO és az Rh rendszer szerinti érték!
28. Tomi és Zolikő-papír-olló játékot játszik. Írj programot, amely a *kopapirollo.txt* állományba kiírja a 10 játék alkalmával 10 sorban és 2 oszlopban a gyerekek által mutatott jeleket (k=kő, p=papír, o=olló)!
29. Klári, Miki és Peti társasoznak. Mindig annyit léphetnek, amennyit a dobókockával dobnak. Az nyer, aki előbb eléri a pálya végét, amely 100 egység távolságra van. Írj programot, amely, a *tarsas.txt* állományba kiírja a dobásokat. Egy sorban Klári, Miki és Peti 1-1 dobása legyen, szóközzel elválasztva!
30. A skandináv lottón 35 számból 7-t kell megjelölni. Húzáskor egy gépi és egy kézi sorsolás történik. Írj programot, amely a *skandinyerő.txt* állományba két sorba kiírja a gépi és a kézi húzás számait!