11. KIÍRÁS FÁJLBA

Az írás művelete is a fájl megnyitásával kezdődik (1. sor). Ilyenkor az **open** utasítás második paramétere **"w"**. Ha a fájl nem létezik, a program létrehozza. Alapértelmezés szerint, ha a fájl már létezik, az újbóli létrehozása teljesen felülírja a régi tartalmat. Tehát nem bővül, hanem törlődik. A fájl elérési útjának megadására ugyanazok a szabályok vonatkoznak, mint az olvasásnál.

A fájl megnyitása után kezdhetünk írni a fájlba. A folyamatot ugyanúgy képzelhetjük el, mintha a képernyőre írnánk. A fájlba írás utasítása a **write** (2. sor). A write utasítással is lehet formázott adatmegjelenítést végezni, mivel ugyanúgy működnek benne a jelölők, mint a **print** esetén. A bekezdés vége jelről (\n) nekünk kell gondoskodni, az utasítás nem tartalmazza. A megnyitott fájl a beleírások hatására folyamatosan bővül. A legutóbb kiírt adat mindig a fájl végére kerül.

```
1 wr = open("test.txt","w")
2 wr.write("András\n")
3 wr.close()
```

Az írás végeztével a fájlt be kell zárni a beolvasáskor megszokott módon a **close** utasítással (3. sor).

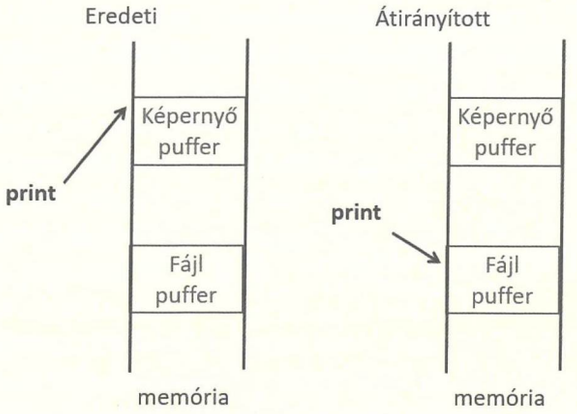
Egy fájl utólagosan is *bővíthető*. Ekkor a fájlt az előzőtől eltérő módon, **"a"** paraméterrel kell megnyitni. Bővítés esetén az adatok a fájl végéhez fognak csatlakozni. A fájlok bővítése érettségin nem követelmény.

63. mintafeladat – ismerkedés a fájlba írással

1. Álljunk a parancsértelmező ablakba (**Python 3.8.0 Shell**).
2. A prompt jel >>> mögött villog a kurzor. Oda írjuk majd be a parancsokat, mindegyik után megnyomva az **Enter** billentyűt.
3. Nyissunk meg egy *test.txt* nevű fájlt írásra. Írjuk be, majd üssünk **Entert**:
   `wr = open("test.txt",'w')`
4. Írassuk ki a fájlba a nevünket. Írjuk be, majd üssünk **Entert**:
   `wr.write("Magyary Gyula")`
5. A parancsértelmező ablakában megjelenik: 13. Ez a kiírt karakterek száma. Keressük meg a *test.txt* fájlt a **Fájlkezelőben** (alapértelmezésben **C:\Python** könyvtárban találjuk), majd nyissuk meg a **Jegyzettömb** segítségével. Nem lesz benne semmi, mert még nem íródott ki ténylegesen a fájlba a név. Zárjuk be a Jegyzettömböt.
6. Zárjuk be a parancsértelmező segítségével a *test.txt* fájlt. Írjuk be, majd üssünk **Entert**:
   `wr.close()`
7. Nézzük meg ismét **Jegyzettömbbel** a *test.txt* fájlt. Ott lesz benne a nevünk. Zárjuk be a **Jegyzettömböt**. Tanulság: a fájlt be kell zárni a fájlművelet végén, mert ellenkező esetben adatvesztés léphet fel.
8. Nyissunk meg újra a *test.txt* nevű fájlt írásra. Írjuk be, majd üssünk **Entert**:
   `wr = open("test.txt",'w')`
9. Írassuk ki a fájlba a Python szót. Írjuk be, majd üssünk **Entert**:
   `wr.write("Python")`
10. Zárjuk be a parancsértelmező segítségével a *test.txt* fájlt. Írjuk be, majd üssünk **Entert**:
    `wr.close()`
11. Nézzük meg ismét **Jegyzettömbbel** a *test.txt* fájlt. Benne lesz a Python, de a nevünk nem. Zárjuk be a **Jegyzettömböt**. Tanulság: ha egy fájlt újra megnyitunk írásra, a régi fájl törlődik, adatai elvesznek, helyette ugyanolyan néven új fájl készül az új tartalommal.
12. Nyissunk meg újra a *test.txt* nevű fájlt, de most bővítésre. Írjuk be, majd üssünk **Entert**:
    `wr = open("test.txt",'a')`
13. Írassuk ki a fájlba egy új bekezdésben a dátumot. Az új bekezdés jele a *\n*. Írjuk be, majd üssünk **Entert**:
    `wr.write("\n2020.04.01.")`
14. Zárjuk be a parancsértelmező segítségével a *test.txt* fájlt. Írjuk be, majd üssünk **Entert**:
    `wr.close()`
15. Nézzük meg ismét **Jegyzettömbbel** a *test.txt* fájlt. Benne lesz a Python szó és a dátum. Zárjuk be a **Jegyzettömböt**. Tanulság: ha egy már bezárt fájlt szeretnénk bővíteni, akkor bővítésre *(a – append)* kell megnyitni, és nem írásra.

Átirányítás

A parancsértelmező ablakba és a fájlba hasonlóan írnak a programok. Mindkettőhöz puffereket használnak, ahogy a fájlkezelés elején láttuk. Ha a parancsértelmezőt kezelő utasítások számára a képernyőhöz használt puffer címét (ahol a parancsablak adatai tárolódnak a memóriában) átírjuk az adott fájl pufferének címére, a parancsablakba író utasítással (print) írhatunk a fájlba.



Tegyük fel, hogy az adatok képernyőre történő kiíratására van egy kész programrészletünk, és a feladat ezeknek az adatoknak egy szövegfájlba írása. Ekkor a legegyszerűbb megoldás a parancsablak fájlba irányítása, majd a programrészlet megismétlése. Ha ezt nem tesszük, két példányban kell létrehozni az adatok megjelenítését, egyszer képernyőre, egyszer a fájlba, s ez pluszmunka és helyfoglalás a kódismétlés miatt.

A kódoláshoz először ki kell bővíteni az utasításkészletet (1. sor).

Aztán eltároljuk a képernyő puffer címét egy *oldout* nevű változóban (2. sor). Erre azért van szükség, mert a fájlba írás után szeretnénk ismét a képernyőre írni, s ahhoz vissza kell állítani a parancsablakba író utasítások számára az eredeti helyzetet.

```
1 import sys
2 oldout = sys.stdout
3 print("Képernyőre ír.")
4 wr = open("test2.txt",'w')
5 sys.stdout = wr
6 print("Fájlba ír.")
7 wr.close()
8 sys.stdout = oldout
9 print("Képernyőre ír ismét.")
```

Elvégezzük a kiírást a képernyőre a szokásos módon (3. sor).

Megnyitjuk a fájlt írásra a szokásos módon (4. sor).

Átállítjuk a képernyő puffert a fájl pufferre (5. sor).

Kiírjuk a fájlba az adatokat, mintha a konzolablakba írnánk (6. sor).

Mivel a fájlt többet nem használjuk, bezárjuk a fájlt (7. sor).

Visszaállítjuk a parancsablakba író parancs kimenetét a parancsablak pufferre (8. sor), hogy ismét tudjunk írni a képernyőre. Az *oldout* változó tartalmazza a képernyő puffer címét.

Ismét írunk a képernyőre, hogy lássuk, sikerült-e a visszairányítás (9. sor).

64. mintafeladat – ismerkedés az átirányítással

1. Álljunk a parancsértelmező ablakba (**Python 3.8.0 Shell**).
2. A prompt jel >>> mögött villog a kurzor. Oda írjuk majd be a parancsokat, mindegyik után megnyomva az **Enter** billentyűt.
3. Az átirányítás használatához bővítsük a felhasználható utasításkészletet a rendszerutasításokkal (sys). Írjuk be, majd üssünk **Entert**:
   `import sys`
4 Jegyezzük fel egy változóban *(oldout)* a parancsértelmező pufferének címét, hogy majd vissza tudjuk állítani az eredeti értékre. Írjuk be, majd üssünk **Entert**:
   `oldout = sys.stdout`
5. Ellenőrzésként írjunk valamit a képernyőre. Írjuk be, majd üssünk **Entert**:
   `print("Képernyőre ír.")`
6. A parancsértelmező ki is írja a fenti szöveget a szokásos kék betűkkel.
7. Nyissunk meg egy *test2.txt* nevű fájlt írásra. Írjuk be, majd üssünk **Entert**:
   `wr = open("test2.txt",'w')`
8. Irányítsuk át a kiírást a fájl pufferébe. Írjuk be, majd üssünk **Entert**:
   `sys.stdout = wr`
9. Ellenőrzésként írjunk valamit a *test2.txt* fájlba a **print** utasítással, ami alapvetően a parancsértelmező ablakba írna. Írjuk be, majd üssünk **Entert**:
   `print("Fájlba ír.")`
10. Azt fogjuk látni, hogy a parancsablakban nem íródik ki kékkel a "Fájlba ír" szöveg. Valószínűleg sikerült az átirányítás. Azonban a fájlban még hiába keresnénk a szöveget. Ahhoz előbb be kell zárni a fájlt.
11. Zárjuk be a *test2.txt* nevű fájlt. Írjuk be, majd üssünk **Entert**:
    `wr.close()`
12. Keressük meg a *test2.txt* fájlt a **Fájlkezelőben** (alapértelmezésben C:\Python könyvtárban találjuk), majd nyissuk meg a **Jegyzettömb** segítségével. Benne lesz a "Fájlba ír." szöveg. Zárjuk be a **Jegyzettömböt**.
13. Próbáljunk meg írni valamit a *test2.txt* fájlba a **print** utasítással. Írjuk be, majd üssünk **Entert**:
    `print("Hová ír?"))`
14. Hibaüzenetet kapunk, hiszen a fájlt bezártuk, oda nem lehet írni. A parancsértelmezőbe azért nem tud írni, mert a kiíratások még mindig a fájl pufferre mutatnak.
15. Állítsuk vissza az átirányítást a képernyő pufferre. Írjuk be, majd üssünk **Entert**:
    `sys.stdout = oldout`

16. Ellenőrzésként írjunk valamit a képernyőre. Írjuk be, majd üssünk **Entert**:
    `print("Képernyőre ír ismét.")`
17. Megjelenik a parancsértelmezőben kék színnel a "Képernyőre ír ismét." szöveg, tehát a visszairányítás sikeres volt.

Feladatok

1. Írjunk programot, ami beolvassa az adat.txt fájl tartalmát soronként, majd minden beolvasott sort rögtön ki is ír az *oper.txt* fájlba az alábbi formátumban: *24/32 + 8/3 =*.
2. Írjunk programot, ami egy *fracts.txt* nevű szöveges fájlba kiír 100 sor véletlenszerű adatot. Egy sorban négy darab, 100 alatti pozitív egész szám és egy műveleti jel legyen, egymástól szóközzel elválasztva. Például: 8 20 8 30 *. A lehetséges műveleti jelek: +, -, *, :.
3. Készítsük el a 15 x 15-ös szorzótáblát a képernyőre. A kódrészlet megismétlése nélkül írassuk ki egy *multtable.txt* nevű fájlba is. Végül a program végén a képernyőre írjuk ki: *Kész a fájl.*