Fájlkezelés (szövegfájl olvasás/írás), hibakezelés (try/except)

Mi az a fájlkezelés?

Eddig az adat:

- program futása közben létezett
- a program leállásával elveszett
- Fájl = tartós adattárolás (pl. .txt).

Alap minta:

```
#r = read (olvasás)
#w = write (felülír)
#a = append (hozzáfűz)

f = open("adat.txt", "r")
tartalom = f.read()
f.close()
```

Fájl olvasása

Tegyük fel, hogy az adat.txt tartalma: 10 20 30

```
#Beolvasás:

with open("adat.txt", "r") as f:
    tartalom = f.read()
print(tartalom)

#Sorokra bontás:

with open("adat.txt", "r") as f:
    sorok = f.readlines()
print(sorok)
```

Feladat: Olvass be egy fájlt. Tárold el a sorokat listában. Írd ki hány sorból áll.

```
with open("adat.txt", "r") as f:
    sorok = f.readlines()
```

```
print("Sorok száma:", len(sorok))
```

Feladat: Az adat.txt fájl számokat tartalmaz, soronként egyet. Írj programot, ami:

- Beolvassa a számokat
- Listába teszi őket (float-ként)
- Kiszámolja az összegüket

```
szamok = []

with open("adat.txt", "r") as f:
    for sor in f:
        szamok.append(float(sor.strip()))

osszeg = 0
for szam in szamok:
    osszeg += szam

print("Összeg:", osszeg)

#vagy:
print(f"Összeg: {sum(szamok)}")
```

Fájl írása

```
with open("eredmeny.txt", "w") as f:
    f.write("Ez egy teszt.\n")

#w mód felülírja a fájlt
```

Feladat: Írj programot, ami:

- Bekér 3 számot
- Fájlba menti őket soronként

```
with open("szamok.txt", "w") as f:
    for i in range(3):
        szam = input("Adj meg egy számot: ")
        f.write(szam + "\n")
```

Hibakezelés (try/except)

Hiba példa:

- szam = int(“abc”)

```
try:
    szam = int(input("Adj meg egy számot: "))
    print("Sikeres konvertálás")
except ValueError:
    print("Hibás adat!")
```

Feladat: Írj programot, ami:

- addig kér be számot,
- amíg a felhasználó helyes számot nem ad meg

```
while True:
    try:
        szam = float(input("Adj meg egy számot: "))
        break
    except ValueError:
        print("Nem számot adtál meg, próbáld újra!")

print("Megadott szám:", szam)
```

Feladat: Írj programot, ami:

- Beolvassa az adat.txt fájlt
- Kiszámolja az átlagot
- Kiírja egy új fájlba az eredményt

```
szamok = []

with open("adat.txt", "r") as f:
    for sor in f:
        szamok.append(float(sor.strip()))

osszeg = sum(szamok)
atlag = osszeg / len(szamok)

with open("atlag.txt", "w") as f:
```

```
    f.write("Átlag: " + str(atlag))

print("Átlag kiszámolva és fájlba írva.")
```

Alapvető kivételek

Exception

- Az összes beépített kivétel ősosztálya.
- Általános hibakezeléshez használható, de nem ajánlott túl általánosan elkapni.

```
try:
    ...
except Exception as e:
    print(e)
```

ValueError

- Akkor dobódik, ha a típus megfelelő, de az érték hibás.
- Példa: szám konvertálás sikertelen

```
int("abc")  # ValueError
```

TypeError

- Hibás adattípus használata.
- Példa: nem kompatibilis típusok művelete

```
"2" + 2  # TypeError
```

NameError

- Nem létező változó használata.

```
print(x)  # NameError, ha x nincs definiálva
```

IndexError

- Lista vagy szekvencia nem létező indexének elérése.

```
lst = [1, 2]
lst[5]  # IndexError
```

KeyError

- Szótárban nem létező kulcs elérése.

```
d = {"a": 1}
d["b"]  # KeyError
```

FileNotFoundError

- Fájl megnyitásakor nem található az adott fájl.

```
open("nemletezo.txt")  # FileNotFoundError
```

IOError / OSError

- Általános fájl- vagy rendszerhibák.
- Pl. jogosultság, eszközhiba.

```
open("/root/file.txt")  # OSError (pl. permission denied)
```

ZeroDivisionError

- Nullával való osztás.

```
10 / 0  # ZeroDivisionError
```

ImportError

- Modul importálási hiba.

```
import nemletezo_modul  # ImportError
```

AttributeError

- Nem létező attribútum/metódus elérése.

```
"abc".nemletezo()  # AttributeError
```

StopIteration

- Iterátor véget ér (pl. next() hívásnál).

```
it = iter([1])
next(it)
next(it)  # StopIteration
```

AssertionError

- assert utasítás sikertelensége.

```
assert 2 > 3  # AssertionError
```

Mikor melyiket használjuk?

- ValueError → ha az érték tartalma hibás
- TypeError → ha az adattípus nem megfelelő
- FileNotFoundError → fájl nem létezik
- KeyError / IndexError → nem létező kulcs/index
- ZeroDivisionError → matematikai hiba
- AttributeError → rossz objektumhasználat
- ImportError → modul probléma

Gyakorló feladatok

Feladat: Jegyek beolvasása és átlag számítása

Adott egy jegyek.txt fájl, amely soronként a következő formátumú:

```
Anna 5 4 3
Bence 2 3 4
Csaba 1 2
```

- Olvasd be a fájlt.

- Számold ki minden diák átlagát.
- Írd ki az eredményt egy atlagok.txt fájlba ilyen formában:
    - Anna átlaga: 4.0
    - Bence átlaga: 3.0
    - Csaba átlaga: 1.5
- Ha a fájl nem létezik, kezeld hibával.

```
def atlag(lista):
    if len(lista) == 0:
        return 0
    return sum(lista) / len(lista)

try:
    with open("jegyek.txt", "r", encoding="utf-8") as f:
        sorok = f.readlines()
except FileNotFoundError:
    print("A jegyek.txt fájl nem található!")
else:
    eredmeny = []

    for sor in sorok:
        adatok = sor.strip().split()
        nev = adatok[0]
        jegyek = []

        for j in adatok[1:]:
            try:
                jegyek.append(int(j))
            except ValueError:
                print(f"Hibás jegy: {j}")

        eredmeny.append(f"{nev} átlaga: {atlag(jegyek):.2f}")

    with open("atlagok.txt", "w", encoding="utf-8") as f:
        for sor in eredmeny:
            f.write(sor + "\n")
```

Feladat: Szógyakoriság fájlból

Olvass be egy szoveg.txt fájlt.

- Számold meg a szavak előfordulását dictionary segítségével.
- Írd ki az eredményt gyakorisag.txt fájlba (szó – darabszám).
- Hibakezelés: ha a fájl nem létezik.

```
def szo_gyakorisag(szoveg):
    szamlalo = {}
    szavak = szoveg.split()
```

```
    for szo in szavak:
        szamlalo[szo] = szamlalo.get(szo, 0) + 1

    return szamlalo

try:
    with open("szoveg.txt", "r", encoding="utf-8") as f:
        tartalom = f.read()
except FileNotFoundError:
    print("A szoveg.txt nem található!")
else:
    eredmeny = szo_gyakorisag(tartalom)

    with open("gyakorisag.txt", "w", encoding="utf-8") as f:
        for szo, db in eredmeny.items():
            f.write(f"{szo} - {db}\n")
```

Feladat: Bevásárlólista és árösszesítés

Adott egy bevasarlas.txt fájl:

```
alma 300
korte 250
banan 400
```

- Olvasd be a fájlt.
- Számold ki az összértéket.
- Írd ki egy osszeg.txt fájlba.
- Kezeld, ha az ár nem szám.

```
def osszeg_szamitas(lista):
    total = 0
    for ar in lista:
        total += ar
    return total

arak = []

try:
    with open("bevasarlas.txt", "r", encoding="utf-8") as f:
        for sor in f:
            adatok = sor.strip().split()
            try:
                ar = int(adatok[1])
                arak.append(ar)
            except (ValueError, IndexError):
                print("Hibás sor:", sor.strip())
```

```
except FileNotFoundError:
    print("A bevasarlas.txt nem található!")
else:
    vegosszeg = osszeg_szamitas(arak)

    with open("osszeg.txt", "w", encoding="utf-8") as f:
        f.write(f"Összesen fizetendő: {vegosszeg} Ft\n")
```

Feladat: Naplófájl hibaszűrés

- Adott egy naplo.txt fájl.
- Írd ki külön fájlba (hibak.txt) azokat a sorokat, amelyek tartalmazzák az “ERROR” szót.
- Hibakezelés kötelező.

```
try:
    with open("naplo.txt", "r", encoding="utf-8") as f:
        sorok = f.readlines()
except FileNotFoundError:
    print("A naplo.txt nem található!")
else:
    hibak = []

    for sor in sorok:
        if "ERROR" in sor:
            hibak.append(sor.strip())

    with open("hibak.txt", "w", encoding="utf-8") as f:
        for sor in hibak:
            f.write(sor + "\n")
```

Feladat: Felhasználói adatok mentése és betöltése

Írj programot, amely:

- Bekér neveket és életkorokat.
- Elmenti egy felhasznalok.txt fájlba.
- Újra beolvassa a fájlt.
- Dictionary-be rendezi (név → életkor).
- Kiírja a 18 év alattiakat.

```
def ment_fajlba(adatok):
    with open("felhasznalok.txt", "w", encoding="utf-8") as f:
        for nev, kor in adatok:
            f.write(f"{nev} {kor}\n")

def betolt_fajlbol():
    felhasznalok = {}
```

```
    try:
        with open("felhasznalok.txt", "r", encoding="utf-8") as f:
            for sor in f:
                adatok = sor.strip().split()
                try:
                    nev = adatok[0]
                    kor = int(adatok[1])
                    felhasznalok[nev] = kor
                except (ValueError, IndexError):
                    print("Hibás adat:", sor.strip())
    except FileNotFoundError:
        print("A felhasznalok.txt nem található!")

    return felhasznalok

# Példa adat
adatok = [("Anna", 17), ("Bence", 20), ("Csaba", 15)]

ment_fajlba(adatok)
felhasznalok = betolt_fajlbol()

print("18 év alattiak:")
for nev, kor in felhasznalok.items():
    if kor < 18:
        print(nev, kor)
```

Feladat: Adott egy diakok.txt fájl, amely a következő formátumú:

```
Anna;matek=5,tori=4,info=5
Bence;matek=2,tori=3
Csaba;matek=4,info=öt
Dora;matek=5,tori=5,info=5
```

A program feladata:

1. Olvassa be a fájlt.
2. Alakítsa át az adatokat dictionary formára:
    - ```
      {
      "Anna": {"matek": 5, "tori": 4, "info": 5},
      ...
      }
      ```
3. Hibakezelés:
    - Ha a fájl nem létezik → hibaüzenet
    - Ha egy jegy nem szám → jelezze, de a program menjen tovább
4. Számolja ki minden diák átlagát.
5. Írja ki az eredményt eredmeny.txt fájlba.
6. Külön listázza a bukott diákokat (átlag < 2).
      ```

```
def fajl_beolvasas(fajlnev):
    diakok = {}

    try:
        with open(fajlnev, "r", encoding="utf-8") as f:
            sorok = f.readlines()
    except FileNotFoundError:
        print("A fájl nem található!")
        return {}

    for sor in sorok:
        sor = sor.strip()

        if not sor:
            continue

        try:
            nev_resz, jegy_resz = sor.split(";")
        except ValueError:
            print("Hibás sorformátum:", sor)
            continue

        tantargyak = {}
        jegyek = jegy_resz.split(",")

        for adat in jegyek:
            try:
                tantargy, jegy = adat.split("=")
                jegy = int(jegy)
                tantargyak[tantargy] = jegy
            except ValueError:
                print(f"Hibás jegy adat: {adat} ({nev_resz})")

        diakok[nev_resz] = tantargyak

    return diakok

def atlag_szamitas(jegyek_dict):
    if len(jegyek_dict) == 0:
        return 0
    return sum(jegyek_dict.values()) / len(jegyek_dict)

def eredmeny_mentes(fajlnev, diakok):
    with open(fajlnev, "w", encoding="utf-8") as f:
        for nev, jegyek in diakok.items():
            atlag = atlag_szamitas(jegyek)
            f.write(f"{nev} átlaga: {atlag:.2f}\n")

def bukottak_listaja(diakok):
    bukott = []
```

```
    for nev, jegyek in diakok.items():
        atlag = atlag_szamitas(jegyek)
        if atlag < 2:
            bukott.append(nev)

    return bukott

# -------- FŐPROGRAM --------

diakok = fajl_beolvasas("diakok.txt")

if diakok:
    eredmeny_mentes("eredmeny.txt", diakok)

    bukott = bukottak_listaja(diakok)

    print("Bukott diákok:")
    for nev in bukott:
        print("-", nev)
```

From:
https://edu.iit.uni-miskolc.hu/ - **Institute of Information Science - University of Miskolc**

Permanent link:
https://edu.iit.uni-miskolc.hu/tanszek:oktatas:muszaki_informatika:fajlkezeles_hibakezeles

Last update: **2026/03/31 06:25**