Programozási tételek Pythonban

IKT projektmunka I. · 10. Íbisz · segédlet

Mi az a programozási tétel?

Sok programozási feladat **ugyanarra a néhány mintára** épül: végigmegyünk egy lista elemein, és közben számolunk, keresünk vagy válogatunk. Ezeket a mintákat hívjuk **programozási tételeknek**. Ha felismered, melyik kell, a program fele már kész.

Minden tétel ugyanígy épül fel: egy változó a kezdőértékkel **a ciklus előtt** → egy `for` vagy `while` ciklus a listán → benne egy `if` → a végén az eredmény kiírása.

#	Tétel	A kérdés	Eredmény
1	Összegzés	Mennyi az összeg?	egy szám
2	Megszámlálás	Hány darab felel meg?	egy szám
3	Eldöntés	Van-e olyan?	igen / nem
4	Kiválasztás	Hol van? (biztosan benne van)	egy hely
5	Lineáris keresés	Van-e, és ha igen, hol?	igen / nem + hely
6	Maximum / minimum	Melyik a legnagyobb / legkisebb?	egy elem (és a helye)
7	Másolás	Minden elemmel csinálj valamit!	egy új lista
8	Kiválogatás	Melyek felelnek meg?	egy új lista
9	Szétválogatás	Bontsd kettőbe!	két új lista
10	Metszet	Mi van mindkettőben?	egy új lista
11	Unió	Mi van legalább az egyikben?	egy új lista

Az 1–6. az alap tételek (egy listából egy eredmény), a 7–11. az összetettek (listából lista). A végén kitekintésként: bináris keresés és rendezés.

A példák adatai (ugyanazokon a listákon dolgozunk végig):

```
pontok = [12, 18, 7, 20, 15, 9, 18]                  # egy dolgozat pontjai
nevek = ["Attila", "Viki", "Vera", "Levi", "Dimi"]
termekek = ["kenyér", "tej", "alma", "sajt"]
```

Emlékeztető: a lista elemeit 0-tól számozzuk: `pontok[0]` → `12`, `pontok[3]` → `20`. A lista hossza: `len(pontok)` → `7`.

1. Összegzés

A kérdés: Mennyi az elemek **összege**? (És belőle az átlag.)

A minta:

```
osszeg = 0
for elem in lista:
    osszeg = osszeg + elem
```

Példa:

```
pontok = [12, 18, 7, 20, 15, 9, 18]

osszeg = 0
for p in pontok:
    osszeg = osszeg + p
print("Összesen:", osszeg, "pont")
print("Átlag:", round(osszeg / len(pontok), 2))
```

Kimenet:

```
Összesen: 99 pont
Átlag: 14.14
```

Pythonban röviden (beépített függvénnyel): `sum(pontok)` · átlag: `sum(pontok) / len(pontok)`

Figyelj: Az `osszeg = 0` a ciklus **előtt** legyen! Ha a cikluson belül van, minden körben lenullázódik.

2. Megszámlálás

A kérdés: Hány elem felel meg egy feltételnek?

A minta:

```
db = 0
for elem in lista:
    if feltétel:
        db = db + 1
```

Példa:

```
pontok = [12, 18, 7, 20, 15, 9, 18]

db = 0
for p in pontok:
    if p >= 15:
        db = db + 1
print("Legalább 15 pontos dolgozat:", db, "db")
```

Kimenet:

```
Legalább 15 pontos dolgozat: 4 db
```

Pythonban röviden (beépített függvénnyel): egy konkrét értékre: `pontok.count(18)` · feltételre: `len([p for p in pontok if p >= 15])`

Figyelj: A megszámlálás nem az elemeket adja össze, hanem **egyet** ad hozzá minden találatnál: `db + 1`, nem `db + elem`.

3. Eldöntés

A kérdés: Van-e olyan elem, amely megfelel egy feltételnek? (Igen vagy nem.)

A minta:

```
van = False
for elem in lista:
    if feltétel:
        van = True
        break       # ha megtaláltuk, nem kell tovább nézni
```

Példa:

```
pontok = [12, 18, 7, 20, 15, 9, 18]

van = False
for p in pontok:
    if p < 10:
        van = True
        break
if van:
    print("Van 10 pont alatti dolgozat.")
else:
    print("Nincs 10 pont alatti dolgozat.")
```

Kimenet:

```
Van 10 pont alatti dolgozat.
```

Pythonban röviden (beépített függvénnyel): egy konkrét értékre: `20 in pontok` · feltételre: `any(p < 10 for p in pontok)`

Figyelj: Az `else: van = False` **ne** kerüljön a ciklusba! Akkor az utolsó elem döntene, nem az, hogy volt-e bármelyik.

4. Kiválasztás

A kérdés: Hányadik helyen van az az elem, amelyről **biztosan tudjuk**, hogy benne van?

A minta:

```
i = 0
while lista[i] != keresett:
    i = i + 1
# az i a keresett elem helye (0-tól számolva!)
```

Példa:

```
nevek = ["Attila", "Viki", "Vera", "Levi", "Dimi"]

i = 0
while nevek[i] != "Levi":
    i = i + 1
print(f"Levi a névsorban a(z) {i + 1}. helyen áll.")
```

Kimenet:

```
Levi a névsorban a(z) 4. helyen áll.
```

Pythonban röviden (beépített függvénnyel): `nevek.index("Levi")`

Figyelj: Csak akkor használd, ha **biztos**, hogy benne van! Ha nincs, a ciklus túlfut a lista végén: `IndexError`. Ilyenkor a keresés kell (5.).

5. Lineáris keresés

A kérdés: Van-e ilyen elem, és ha igen, **hol**?

A minta:

```
i = 0
while i < len(lista) and lista[i] != keresett:
    i = i + 1
if i < len(lista):
    # megvan, az i. helyen
else:
    # nincs benne
```

Példa:

```
termekek = ["kenyér", "tej", "alma", "sajt"]
keresett = "alma"

i = 0
while i < len(termekek) and termekek[i] != keresett:
    i = i + 1
if i < len(termekek):
    print(f"{keresett}: megvan, a(z) {i + 1}. helyen.")
else:
    print(f"{keresett}: nincs a listában.")
```

Kimenet:

```
alma: megvan, a(z) 3. helyen.
```

Pythonban röviden (beépített függvénnyel): `if keresett in termekek: termekek.index(keresett)`

Figyelj: A feltételben a sorrend számít: **előbb** az `i < len(lista)`, aztán a `lista[i]`. Fordítva a lista végén `IndexError` lesz.

6. Maximum- és minimumkiválasztás

A kérdés: Melyik a **legnagyobb** vagy a **legkisebb** elem, és hol van?

A minta:

```
max_hely = 0                        # feltételezzük: az első a legnagyobb
for i in range(1, len(lista)):
    if lista[i] > lista[max_hely]:
        max_hely = i
```

Példa:

```
pontok = [12, 18, 7, 20, 15, 9, 18]

max_hely = 0
for i in range(1, len(pontok)):
    if pontok[i] > pontok[max_hely]:
        max_hely = i
print(f"A legtöbb pont: {pontok[max_hely]}, a(z) {max_hely + 1}. dolgozat")

legkisebb = pontok[0]
for p in pontok:
    if p < legkisebb:
        legkisebb = p
print("A legkevesebb pont:", legkisebb)
```

Kimenet:

```
A legtöbb pont: 20, a(z) 4. dolgozat
A legkevesebb pont: 7
```

Pythonban röviden (beépített függvénnyel): `max(pontok)`, `min(pontok)` · a helye: `pontok.index(max(pontok))`

Figyelj: Ne `0`-val kezdd a maximumot (`legnagyobb = 0`)! Ha minden elem negatív, rossz lesz az eredmény. **Az első elemmel** kezdj.

7. Másolás

A kérdés: Minden elemmel csinálj valamit, és az eredményeket tedd egy **új listába**.

A minta:

```
uj = []
for elem in lista:
    uj.append(valami(elem))
```

Példa:

```
arak_ft = [450, 1200, 380, 2990]

arak_eur = []
for ar in arak_ft:
    arak_eur.append(round(ar / 400, 2))
print(arak_eur)
```

Kimenet:

```
[1.12, 3.0, 0.95, 7.47]
```

Pythonban röviden (beépített függvénnyel): `[round(ar / 400, 2) for ar in arak_ft]` (listaértelmezés)

Figyelj: Az új lista ugyanannyi elemű, mint a régi. Ha kevesebb kell, az már a kiválogatás (8.).

8. Kiválogatás

A kérdés: Gyűjtsd ki egy **új listába** azokat, amelyek megfelelnek a feltételnek.

A minta:

```
uj = []
for elem in lista:
    if feltétel:
        uj.append(elem)
```

Példa:

```
pontok = [12, 18, 7, 20, 15, 9, 18]

jok = []
for p in pontok:
    if p >= 15:
        jok.append(p)
print("A legalább 15 pontosak:", jok)
```

Kimenet:

```
A legalább 15 pontosak: [18, 20, 15, 18]
```

Pythonban röviden (beépített függvénnyel): `[p for p in pontok if p >= 15]`

Figyelj: A megszámlálás azt mondja meg, **hány** ilyen van. A kiválogatás azt, hogy **melyek** azok.

9. Szétválogatás

A kérdés: Bontsd **két listára** az elemeket: ami megfelel, és ami nem.

A minta:

```
igen = []
nem = []
for elem in lista:
    if feltétel:
        igen.append(elem)
    else:
        nem.append(elem)
```

Példa:

```
pontok = [12, 18, 7, 20, 15, 9, 18]

atment = []
nem_ment_at = []
for p in pontok:
    if p >= 10:
        atment.append(p)
    else:
        nem_ment_at.append(p)
print("Átment:", atment)
print("Nem ment át:", nem_ment_at)
```

Kimenet:

```
Átment: [12, 18, 20, 15, 18]
Nem ment át: [7, 9]
```

Pythonban röviden (beépített függvénnyel): két kiválogatás: `[p for p in pontok if p >= 10]` és `[p for p in pontok if p < 10]`

Figyelj: A két új lista elemszámának összege mindig ugyanannyi, mint az eredeti listáé. Ezzel ellenőrizheted magad.

10. Metszet

A kérdés: Mely elemek vannak **mindkét** listában?

A minta:

```
kozos = []
for elem in lista1:
    if elem in lista2:
        kozos.append(elem)
```

Példa:

```
foci = ["Attila", "Levi", "Bence", "Ádám"]
kosar = ["Levi", "Vera", "Ádám", "Hero"]

mindketto = []
for nev in foci:
    if nev in kosar:
        mindketto.append(nev)
print("Mindkét sportot űzi:", mindketto)
```

Kimenet:

```
Mindkét sportot űzi: ['Levi', 'Ádám']
```

Pythonban röviden (beépített függvénnyel): halmazokkal: `set(foci) & set(kosar)`

Figyelj: A metszet egy kiválogatás: azokat válogatjuk ki az elsőből, amelyek a másodikban is benne vannak.

11. Unió

A kérdés: Mely elemek vannak **legalább az egyik** listában? (Mindegyik csak egyszer.)

A minta:

```
unio = []
for elem in lista1:
    unio.append(elem)
for elem in lista2:
    if elem not in unio:
        unio.append(elem)
```

Példa:

```
foci = ["Attila", "Levi", "Bence", "Ádám"]
kosar = ["Levi", "Vera", "Ádám", "Hero"]

valamelyik = []
for nev in foci:
    valamelyik.append(nev)
for nev in kosar:
    if nev not in valamelyik:
        valamelyik.append(nev)
print("Legalább az egyiket űzi:", valamelyik)
```

Kimenet:

```
Legalább az egyiket űzi: ['Attila', 'Levi', 'Bence', 'Ádám', 'Vera', 'Hero']
```

Pythonban röviden (beépített függvénnyel): halmazokkal: `set(foci) | set(kosar)`

Figyelj: Az ismétlődést figyeld: Levi és Ádám mindkét listában benne van, de az unióban csak egyszer szerepelhet.

Kitekintés: két haladóbb tétel

12. Bináris (logaritmikus) keresés

A kérdés: Keresés **rendezett** listában, felezéssel: sokkal gyorsabb, mint a lineáris.

```
szamok = [3, 8, 12, 17, 25, 31, 40, 52]   # NÖVEKVŐ sorrendben!
keresett = 31

eleje = 0
vege = len(szamok) - 1
megvan = -1
while eleje <= vege and megvan == -1:
    kozep = (eleje + vege) // 2
    if szamok[kozep] == keresett:
        megvan = kozep
    elif szamok[kozep] < keresett:
        eleje = kozep + 1
    else:
        vege = kozep - 1
if megvan >= 0:
    print(f"{keresett}: megvan, a(z) {megvan + 1}. helyen.")
else:
    print(f"{keresett}: nincs a listában.")
```

Kimenet:

```
31: megvan, a(z) 6. helyen.
```

Mint a számkitalálós játékban: mindig a közepét nézed, és a felét eldobod. 1000 elemből legfeljebb 10 lépés (a lineáris keresésnek akár 1000).

13. Rendezés (egyszerű cserés rendezés)

A kérdés: Rakd **sorba** a lista elemeit.

```
szamok = [12, 18, 7, 20, 15]

for i in range(len(szamok) - 1):
    for j in range(i + 1, len(szamok)):
        if szamok[j] < szamok[i]:
            szamok[i], szamok[j] = szamok[j], szamok[i]
print(szamok)
```

Kimenet:

```
[7, 12, 15, 18, 20]
```

Minden elemet összehasonlít az utána lévőkkel, és ha kell, megcseréli őket. Pythonban röviden: `sorted(lista)` vagy `lista.sort()`.

Összefoglaló: melyiket mikor?

Ha a feladatban ezt olvasod…	…ez a tétel kell
„mennyi összesen”, „mennyi az átlag”	összegzés
„hány olyan van”, „hány darab”	megszámlálás
„van-e olyan”, „igaz-e, hogy valamelyik”	eldöntés
„hányadik”, „hol van” (és biztosan benne van)	kiválasztás
„benne van-e, és ha igen, hol”	lineáris keresés
„a legnagyobb”, „a legkisebb”, „a legjobb”, „a legolcsóbb”	maximum / minimum
„mindegyiket alakítsd át”, „számold át”	másolás
„írd ki / gyűjtsd ki azokat, amelyek…”	kiválogatás
„bontsd két csoportra”	szétválogatás
„mindkettőben”, „közös”	metszet
„legalább az egyikben”, „összesen kik”	unió

A beépített függvények egy helyen (az eredmények a `pontok` listára):

```
pontok = [12, 18, 7, 20, 15, 9, 18]
print(sum(pontok), max(pontok), min(pontok), len(pontok))
print(20 in pontok, pontok.index(20), pontok.count(18))
print(sorted(pontok))
```

```
99 20 7 7
True 3 2
[7, 9, 12, 15, 18, 18, 20]
```

Fontos: a dolgozatban, ha a feladat azt kéri, hogy **„tétellel, ciklussal”** oldd meg, a beépített függvény (`sum`, `max`, `min`, `index`) nem elég. Azt kell megmutatnod, hogy érted, mi történik a háttérben.

Gyakorló feladatok

Adat: `homerseklet = [3, −2, 5, 8, −4, 1, 6]` (egy hét napi hőmérséklete, °C).

1. Mennyi a heti átlaghőmérséklet? *(összegzés)*
2. Hány fagyos nap volt (0 °C alatt)? *(megszámlálás)*
3. Volt-e 7 °C-nál melegebb nap? *(eldöntés)*
4. Melyik volt a leghidegebb nap, és hányadik napon? *(minimum, a hellyel)*
5. Gyűjtsd ki a fagyos napok hőmérsékletét egy listába! *(kiválogatás)*
6. Számold át mindegyiket Fahrenheitbe (`F = C * 9 / 5 + 32`)! *(másolás)*
7. Bontsd a napokat „fagyos” és „nem fagyos” listára! *(szétválogatás)*