



8. hét: osztályok

 Czirkos Zoltán, Frey Balázs ·  2023.10.26.
Osztályok definíciója, példányosítása. Paraméterátadás.

Tartalom

- [Összetett adatszerkezet](#)
- [Az __str__\(self\) függvény](#)
- [Időpontok](#)
- [Kerítés](#)
- [Szállóvendégek](#)
- [Charlie](#)

1. Összetett adatszerkezet

Az itt látható táblázat egy futóverseny eredményeit tartalmazza.

Index	Név	Születés	Helyezés
0	Am Erika	1994. 05. 06.	1
1	Break Elek	2001. 09. 30.	3
2	Dil Emma	1998. 08. 25.	2
3	Kasza Blanka	1989. 06. 10.	5
4	Reset Elek	2001. 04. 05.	4

Alább egy elkezdett programot látsz, amelyben a megfelelő típusok már definiálva vannak, és az adatokat egy lista tartalmazza. Egészítsd ki a programot, hogy kiírja a képernyőre a kommentekben megadott adatokat!

```
class Datum:
    def __init__(self, ev, honap, nap):
        self.ev = ev
        self.honap = honap
        self.nap = nap

class Versenyzo:
    def __init__(self, nev, szuletes, helyezes):
        self.nev = nev
        self.szuletes = szuletes
        self.helyezes = helyezes

def datum_str(d):
    pass # később kiegészítendő

def versenyzo_str(v):
    pass # később kiegészítendő

def main():
    versenyzok = [
        Versenyzo("Am Erika", Datum(1994, 5, 6), 1),
        Versenyzo("Break Elek", Datum(2001, 9, 30), 3),
        Versenyzo("Dil Emma", Datum(1998, 8, 25), 2),
        Versenyzo("Kasza Blanka", Datum(1989, 6, 10), 5),
        Versenyzo("Reset Elek", Datum(2001, 4, 5), 4),
    ];

    # 0-s versenyző neve
    # 2-es versenyző helyezése
    # 4-es versenyző születési dátuma (írd meg a datum_str függvényt!)
    # 1-es versenyző nevének kezdőbetűje
    # az 1-es versenyző dobogós-e? igen/nem
    # az 4-es versenyző gyorsabb-e, mint a 3-as versenyző?
    # az 1-es versenyző ugyanabban az évben született-e, mint a 2-es?
    # egészítsd ki a versenyzo_str() függvényt, és írd ki az 1-es versenyző adatait
    # végül listázd ki az összes versenyzőt sorszámozva, összes adatukkal.

main()
```

► **Megoldás**

2. Az __str__(self) függvény

Az előadáson szerepelt, hogy az osztály belsejében definiált `__str__` függvénnyel „meg lehet tanítani” a Pythonnak, hogy egy adott típus hogyan konvertálható sztringgé.

Módosítsd az előző programot úgy, hogy a `datum_str()` és a `versenyzo_str()` függvények helyett `__str__` függvényeket kapnak az osztályok! Ezek után pedig, írd át a főprogramot, ahol kell.

► **Megoldás**

3. Időpontok

Írj programot, amely egy osztályban időpontot tárol: óra, perc. Írjunk függvényeket ehhez:

ora	perc
15	45

- `ido_kiir(i)`: kiírja az időpontot óra:perc formában.
- `ido_hozzaad(i, p)`: hozzáad `p` percet az `i` időponthoz, és visszatér az új időponttal. Pl. 15:15 + 45 = 16:00.
- `ido_eltelt(i1, i2)`: megmondja, hány perc telt el a két időpont között, pl. 16:30-15:15 = 75 perc. (A paraméterek sorrendje a kivonásnál megszokott: kisebbbitendő, kivonandó.)
- `ido_kivon(i, p)`: kivon `p` percet az `i` időpontból, és visszatér az új időponttal. Pl. 15:45 - 30 = 15:15.

► **Tipp**

► **Megoldás**

4. Kerítés

Az előadásban szerepelt a [pont típus](#). Ebben a feladatban ezzel kell megoldanod egy problémát.

x	y
3.2	-5.4

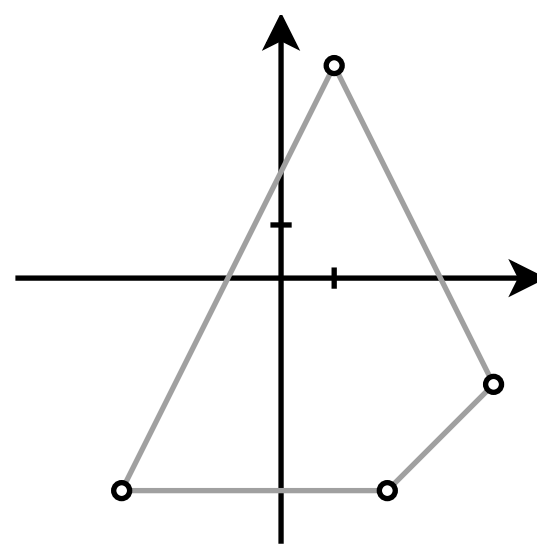
Add meg a `Pont` típust, amely kétdimenziós koordinátát (`x`, `y`) tárol! Írj ehhez függvényeket:

- `tav()`: a paraméterként kapott két pont távolságával tér vissza! (Ehhez Pitagorasz tételét kell használni.)
- `egyenlo()`: megvizsgál két pontot, és megmondja, hogy egybeesnek-e.
- `beolvas()`: beolvassa egy pont koordinátáit a billentyűzetről, és visszatér vele.

Ha ezek megvannak, az eddigiek használatával oldd meg az alábbi feladatot:

Kerítés hossza

Egy gazda szeretné körbekeríteni a telkét drótkerítéssel. Írj programot, amely kiszámítja, hogy mennyi kerítésre lesz szüksége! A program kérje egymás után a kerítésoszlopok koordinátáit (`x`, `y` koordinátpárok), számítsa ki az aktuális és az előző oszlop távolságát, és összegezze a távolságokat! Az összegzést addig folytassa, amíg a megadott koordináták nem egyeznek az elsőként megadott koordinátpárral, vagyis míg vissza nem ér a kezdőoszlophoz!



Ehhez célszerű egy változóba följegyezni a kezdőpontot, azután pedig két további, pont típusú változóval dolgozni: az egyik tárolja az új pont adatait, a másik pedig mindig az eggyel előzőt.

► **Megoldás**

5. Szállóvendégek

Egy héteemeletes szállodában a szobafoglalásokat listában tárolják. A szobák a szokásos módon vannak számozva, a százasok adják meg az emeletet, a többi pedig a szoba sorszámát (pl. 712 = 7. emelet, 12. szoba). A földszint a 0. szint, utána 1-től 7-ig az emeletek. Ennél a feladatnál nem kell teljes programot írni, csak a megadott részeket.

- Definiálj `Vendeg` nevű típust, amelyik egy szállóvendég adatait (név: sztring, szobaszám: egész) tartalmazza! Írj függvényt, amely átvesz egy vendéget, és visszaadja, hogy melyik emeleten lakik!
- Írj függvényt, amely átvesz egy `Vendeg` elemekből álló listát és egy nevet! Keresse ez meg a névhez tartozó foglalást és adja vissza a megtalált listaelemet vagy `None`-t, ha nincs találat!
- Írj függvényt, amely paraméterként kapja a vendégek listáját és visszaad egy másik listát, amelyet a szint sorszámával indexelünk! Írja be az utóbbi listába, hogy az egyes emeleteken hány vendég lakik! Ha van üres emelet, annak is szerepelnie kell ebben.
- Írj függvényt, amely megkapja a vendégek listáját, az előző függvénnyel előállítja a betöltöttiségek listáját, és végül visszatér a legzsúfoltabb emelet sorszámával – tehát azzal, ahol a legtöbb vendég van éppen!
- Írj főprogramot, amelyben létrehozod a listát az alábbi foglalásokkal, és megkeresed a legzsúfoltabb emeletet!

Név	Szoba
Dia Dóra	712
Elektro M Ágnes	713
Érték Elek	506

► **Megoldás**

6. Charlie

Charlie fagyaltot árul: jelenleg pisztácia, vanília, tutti-frutti, karamell, rumos dió és kávé a választék, de lehetne többféle is. A fagyit íz alapján kérhetik a gyerekek, egyszerűen csak egy gombócot. A készlet véges, minden gombóc eladásával értelemszerűen eggyel csökken.

Definiálj osztályt, ami egy fagyí adatait (íz, hány gombóc van) tárolja! Írj függvényt, amely a fagyí objektumok listáját kapja és egy ízt; vissza pedig a megtalált elem referenciáját adja, vagy `None`-t!

Egészítsd ki ezt teljes programmá, amely a vásárlásokat kezeli! Hozz létre egy fagyí listát, és töltsd fél adatokkal. Olvasd be a vásárlásokat (ízeket) fájl vége jeléig. Keresd meg az előbb megírt függvénnyel a kapott ízt, és jelezd a vásárlás eredményét: sikeres, kifogyott (volt, de 0-ra csökkent), nem is volt!

Íz	Mennyiség
pisztácia	0
vanília	3
tutti-frutti	8
karamell	4
rumos dió	5
kávé	9

