Tartalomjegyzék

- 1 Feladatok
  - 1.1 Komplex
- 2 Reservation
  - 2.1 Reservation_1
    - 2.1.1 Segítség
  - 2.2 Reservation_2
    - 2.2.1 Segítség
  - 2.3 Reservation_3
    - 2.3.1 Segítség
  - 2.4 Reservation_4
  - 2.5 Reservation_5
    - 2.5.1 Segítség

Feladatok

Komplex

A feladat az el?adáson elkezdett Komplex osztályt befejezni:

```
class Komplex(object):
    def __init__(self, real, imaginary):
        self.re = real
        self.im = imaginary

    def __add__(self, k2):
        uj_re = self.re + k2.re
        uj_im = self.im + k2.im
        return Komplex(uj_re, uj_im)

    def __str__(self):
        s = ""
        s += str(self.re)
        s += " + "
        s += str(self.im)
        s += "i"
        return s

k1 = Komplex(4, 3)
k2 = Komplex(-2, 1)
k3 = k1 + k2

print(k3)
```

- Valósítsuk meg a kivonás, szorzás és osztás m?veleteket. (__sub__, __mul__, __truediv__) Az osztás el?tt érdemes lehet a következ? részt megoldani el?ször.
- Valósítsuk meg a `norm` metódust, mely a komplex szám hosszát adja meg.
- Javítsuk ki a __str__ metódust, hogy szépen írja ki a számokat, pl:

```
2 - 4i
5i
2
```

Teszteléshez használhatjuk pl ezt a kódot, de írjunk saját teszteket is!

```
k1 = Komplex(4, 3)
k2 = Komplex(-2, 1)
k3 = Komplex(4, 1)

print k1 + k2
print k1 - k3
print k2 * k1
print k3 / k1
print k1.norm()
```

Reservation

A következ? feladatokban egy olyan rendszeren dolgozunk, ami egy moziban fogja nyomon követni a különböző emberek helyfoglalásait. Ehhez létre fogunk hozni egy `Reservation` nevű osztályt, ami egy ember foglalását tárolja, és ezt az osztály ellátjuk minden szükséges metódussal a kényelmes használathoz.

Reservation_1

Ebben a feladatban hozzuk létre a **konstruktort**. A foglalás osztály két tagváltozót tartalmazzon majd, és ezek legyenek a konstruktor paraméterei is, ebben a sorrendben:

- **name**, a mozi foglalás lefoglalójának neve
- **seats**, egy lista, ami az üléseket tartalmazza amiket *name* lefoglalt. Minden ülés egy karakterlánccal van leírva, pl. "http://wiki.math.bme.huE12"http://wiki.math.bme.hu

Egyelőre csak elég a konstruktort megírni, semmi más nem kell az osztályba még.

Segítség

- Ne felejtsük el, hogy (mint minden metódusnál) a konstruktor definiálásakor az első paraméter a *self* legyen.

Reservation_2

Ebben a feladatban adjunk egy metódust a *Reservation* osztályhoz. A metódus neve legyen **not_taken**, és egy paramétere legyen (a *self*-en kívül):

- **lehetosegek**, ami egy lista, ami lehetséges üléseket tartalmaz.

A *not_taken* metódus térjen vissza a *lehetosegek* lista azon elemeivel, amik ebben a *Reservation* objektumban nincsenek lefoglalva. Az elemek maradjanak ugyanabban a sorrendben mint a *lehetosegek* listában voltak.

Segítség

- Ne felejtsük el, hogy az aktuális objektum foglalt helyeit a *self* referencia *seats* tagváltozóján keresztül érhetjük el.

Reservation_3

Ebben a feladatban kipróbáljuk, milyen használni az eddig megalkotott *Reservation* osztályt. Írjuk meg a **reservation_3** nevű függvényt, aminek paraméterei:

- **foglalasok**, egy lista, ami az éppen aktuális foglalásokat tartalmazza, *Reservation* típusú objektumok formájában
- **osszes_ules**, egy lista, az adott mozi terem összes ülését tartalmazza

A függvény azoknak az üléseknek a listájával térjen vissza, amik még szabadak, tehát amik még egyik foglalásnak sem részei.

Segítség

- Használjuk a *Reservation* osztály *not_taken* nevű metódusát.

Reservation_4

Ebben a feladatban olyan metódust írunk, ami módosítja az objektum tartalmát. Az eddigi *Reservation* osztályunkban hozzunk létre egy új metódust, aminek a neve **change**, és paramétere (a *self*-en kívül):

- **uj_helyek**, egy lista, ami azt tartalmazza hogy mik a helyek amiket foglalni akar az illető a változtatás után.

A metódus változtassa meg az objektumot úgy, hogy átírja a **self.seats** változót, de ne térjen vissza semmivel.

Reservation_5

Most használjuk a *Reservation* objektumainkat kicsit bonyolultabb feladatra. Írjuk meg a **reservation_5** nevű függvényt, aminek paraméterei:

- **foglalasok**, egy lista, ami az éppen aktuális foglalásokat tartalmazza, *Reservation* típusú objektumok formájában
- **osszes_ules**, egy lista, az adott mozi terem összes ülését tartalmazza
- **nev**, egy karakterlánc, az éppen foglalni próbáló vendég neve
- **darab**, egy egész szám, hogy hány helyet szeretne lefoglalni

A függvény keressen *darab* olyan helyet, amik az *osszes_ules* listában egymás után szerepelnek, és még nincsenek lefoglalva. Ha talál ilyeneket, akkor azt a lehetőséget vegye, amelyiknek az ülései a legkorábban szerepelnek az *osszes_ules* listában, és adja vissza a foglalást egy *Reservation* típusú objektumban. Ha nem talál ilyen üléseket, akkor *None*-t adjon vissza.

Segítség

- Használjuk ez el?z? feladatok megoldását