1	2	3	4	5	$\sum$

Informatika 2, 2. ZH (2019-03-08)

A feladatok megoldása elfér a feladat mellett, ha külön lapra írjuk, tegyünk egy jól látható nyilat a helyére!

1. Írjunk két python osztályt, az egyik a sík pontjait, a másik háromszögeket reprezentáljon. A szükséges metódusok: *(6 pont)*

- `Pont`

  konstruktor két paraméterrel: a pont x és y koordinátája.

  str szöveggé alakító metódus, adja vissza a pont koordinátáit szövegesen, valahogy így: `P(3,4)`

- `Haromszog`

  konstruktor három paraméterrel: a három csúcs, mint `Pont` típusú objektum. Ez a metódus emeljen `ValueError` kivételt, ha a három pont egybeesne.

  sulypont Adja vissza a háromszög súlypontját `Pont` típusú objektumként, ez a három csúcs átlaga.

```
class Pont(object):
    def __init__(self,         ):




    def __str__(self           ):


class Haromszog(object):
    def __init__(self,         ):




    def sulypont(self           ):
```

Ha jól írtuk meg, így kell működnie:

```
>>> A, B, C = Pont(0,0), Pont(1,0), Pont(0,1)
>>> print B
P(1,0)
>>> h = Haromszog(A, B, C)
>>> print h.sulypont()
P(0.33333, 0.33333)
```

2. Elméleti kérdések *(4 pont)*

a) Hogyan kell saját kivétel oszályt írni?

b) Mire való a `__mul__` speciális metódus? Hogyan használjuk?

c) Hogyan lehet egy szöveget szavakra vágni? (string-ből string-ek listája kell)

d) Mi az a *variadikus függvény*?

3. Írjunk egy iterálható osztályt, amin egy adott számig adott lépésközzel lehet iterálni! *(3 pont)*

A konstruktorának legyen két paramétere, a szám ameddig el kell menni és egy szám, hogy milyen lépésközzel. Az iterálás 0-tól kezdődjön és az első paraméterben kapot számot már nem kell beleérteni. Írjuk meg az `__iter__` és a `next` metódusokat is. Az `__iter__` visszatérési értéke `self` legyen!

```
class Iterable(object):
    def __init__(self, n, k):


    def __iter__(self):



    def next(self):
```

Példa

```
for i in Iterable(10,2):
    print i,
```

```
0 2 4 6 8
```

4. Mit csinálnak a következő függvények? *(3 pont)*
Magyarázzuk meg és mutassunk példát!

```
def f(s, width=10):
    return s.rjust(width)

def g(L):
    R = ""
    for l in L:
        R += f(l)
    return R

def h():
    x = raw_input()
    L = []
    while len(x) > 0:
        L.append(x)
        x = raw_input()
    print g(L)
```

5. Az alábbi kódban 4 hiba található, találjuk meg ezeket! *(4 pont)*

```
class A(object):
    def__init__(self, x):
        self.x = x

class B(object):
    def __init__(self, *x)
        self.x = x

    def add(self, other):
        return B(self.x + other.x)

    def __str__(self):
        return str(x)
```

Példa:

```
a = A(0)
b = B(0)
print b + b
```