Tartalomjegyzék

El?adás

8. elõadás

Feladatok

Nyissuk meg a Sypdert.

Alakzatok a vásznon

Alakzatok

Írjunk egy **Shape** osztályt.

- Legyen **x** és **y** változója, ezek tárolják az alakzat pozícióját a síkon.
- Legyen egy **move** metódusa, aminek egyetlen **v** paramétere van, egy kételem? lista, a vektor, amivel el kell mozgatni az alakzatot.

Definiáljuk a **Shape** osztály leszármazottaiként az

- **Ellipse** ellipszis, legyen meg a kis- és nagytengelye (**a,b**)
- **Rectangle** téglalap, legyen meg az oldalak hossza (**a,b**)

osztályokat. Mindkét esetben a pozíciójuk a súlypontjukat jelentse. Írjunk mindkét osztályhoz egy **area** függvényt, ami kiszámítja az alakzat területét!

Definiáljuk az **Ellipse** osztály **equation** metódusát, ami kiírja az adott ellipszis egyenletét!

Vászon

Definiáljuk a **Canvas** (vászon) osztályt.

- Egyetlen tagváltozója legyen a **shapes**, ami alakzatok listáját tárolja.
- Definiáljuk egy **add** metódust, amivel újabb **Shape**-et adunk a vászonhoz!
- Oldjuk meg, hogy az osztályunk iterálható legyen! Ehhez definiáljuk az **__iter__(self)** metódust, valamit a **next(self)** metódust, ahogy az el?adáson láttuk.

- Definiáljuk a **crop** metódust a következ?képp: a bemeneti paraméter két pont koordinátája, ezek egy téglalap bal fels? és jobb alsó pontjai. A függvény térjen vissza azon alakzatok listájával, amelyek a vásznon vannak és teljesen belefférnek az így definiált téglalapba. Ehhez a feladathoz az **Ellipse** és a **Rectangle** osztálynak is szüksége lesz egy **box()** metódusra, ami a legkisebb tartalmazó doboz bal fels? és jobb alsó sarkait adja vissza.

Overload

1. Írjunk egy függvényt, aminek az els? argumentuma **n**, egy **int** típusú változó. A függvény térjen vissza **True**-val, ha annyi extra paraméterrel hívták meg, mint az els? bemeneti paraméter értéke, egyébként térjen vissza **False**-szal.
2. Definiáljunk egy **szumma** függvényt, ami tetsz?legesen sok bemeneti paraméterének összegével tér vissza!
    1. Kezeljük le a kivételt, ha a paraméterek típusa nem azonos!
3. Definiáljunk egy **print_words** függvényt, úgy, hogy a megadott (akármennyi) szavakat annyiszor írja ki, amennyit megadunk bemenetnként (szavanként)!
    1. Kezejük le kivételként, ha a bemeneten nem egész számot adtak meg a szó gyakoriságára!

Sakk

Ha még maradt id?. Definiáljuk a **Piece** osztályt. Ez reprezentál egy sakkbábut, tároljuk a pozícióját a táblán két koodinátával, a színét (black/white), illetve a **__repr__** írja ki, hogy hol áll (A2, G3 etc.)!

- Definiáljuk a bábu leszármazottjaként a **King** és a **Pawn** osztályokat!
- Minden leszármazottnak legyen egy **move(pos)** metódusa, ahol a **pos** egy sztring (A3, G2 etc.)! Mozgassuk el a bábut, ha szabályos a lépés!
- Definiáljuk a **PieceMoveError** osztályt. Ha szabálytalan a lépés, dobjunk egy ilyen exceptiont és kezeljük le!

Legyen most egy **Board** osztályunk! Két listát tároljunk: **white** és **black**, a játékosok bábuival!

- Legyen a **Board** osztálynak egy **move(player, pos1, pos2)** metódust, ami a **pos1** pozícióban álló bábut a **pos2** helyre mozgatja, ha a lépés szabályos!
    - A normál szabályok mellett vegyük figyelembe, hogy áll-e ott más bábu! Ha saját bábu áll a mez?n, a lépés szabálytalan, ha ellenfél bábuja, akkor távolítsuk el a pályáról!
    - Ha maradt id?nk, kezdjük el írni a **Knight, Rook, Bishop** és **Queen** osztályokat.
    - Oldjuk meg, hogy a **Pawn** ne állhasson az ellenfél alapvonalán!