

Tartalomjegyzék

- 1 Feladatok
 - ◆ 1.1
 - Rekurzív
 - ◇ 1.1.1
 - deep_sum
 - ◇ 1.1.2
 - palindrom
 - ◇ 1.1.3
 - pattern
 - ◇ 1.1.4
 - Determináns
 - ◆ 1.2 Fák
 - ◇ 1.2.1
 - Bináris
 - fák

Feladatok

Rekurzív

deep_sum

Írjunk egy rekurzív függvényt, aminek bemenete egy olyan lista, mely listákat tartalmaz tetszőleges mélységig, melyekben pozitív egész számok vannak.

A függvény adja vissza a listában lévő számok összegét.

Pl.:

[1, 2, 3, [4, 5], [[[6], 7]]] -> 28

palindrom

Írjunk egy rekurzív függvényt, ami eldönti egy stringről, hogy palindrom-e vagy sem.

Pl.:

aba -> True

abb -> False

pattern

Írjunk egy rekurzív függvényt, melynek bemenete egy 5-tel osztható n egész szám, és kiíratja a következő számsorozatot:

n, n-5, n-10, ..., 0, 5, 10, ..., n-5, n,

A megoldáshoz ne használjunk se listát, se for ciklust, se while-t.

pl.:

15, 10, 5, 0, 5, 10, 15,

Determináns

Írjunk egy rekurzív függvényt, aminek bemenete egy négyzetes mátrix és a kimenete a mátrix determinánsa.

Elegendő mindig az első sora szerint kifejtetni aldeterminánsokra és ezekre hívjuk majd meg rekurzív módon a függvényt ismét.

Pl.:

[[1,2,1],[2,3,1],[0,2,1]] -> 1

Fák

Bináris fák

Másoljuk be a bináris fa definícióját. Ezen az osztályon fogunk dolgozni.

- Írjunk egy `count(self)` metódust, ami megszámolja a fa elemeinek a számát!
- Írjunk egy `sum(self)` metódust, ami a fa összes csúcsában lévő értékeket összegezi!
- Írjunk egy `height(self)` metódust, ami megmondja, hogy milyen magas a fa!
- Írjunk egy `is_list(self)` metódust, ami megmondja, hogy a bináris fa listává fajult-e. Egy bináris fa akkor fajult listává, ha minden csúcsának legfeljebb egy gyereke van.