Tartalomjegyzék

Maradék mátrixozás

Determináns

Egy mátrix determinánsát kiszámolhatjuk a **det** metódussal:

```
m.det()
```

Számoljuk ki a determinánsát a következõ blokkmátrixnak:

```
X I
O X
```

ahol I a 3x3-as egységmátrix és O a 3x3-as csupa 0 mátrix, X pedig a következõ:

```
 0 -1 -1
-1  0 -1
-1 -1  0
```

Összefüggõ

Határozzuk meg, hogy az alábbi mátrix sorai (vagy oszlopai) milyen **x** értékekre lesznek összefüggõk. (Használjuk a **solve** parancsot a fentiekkel együtt.)

```
x  0  1
0  2  x
1  x -1
```

Sajátérték

Sajátértéket és sajátvektorokat számolhatunk a következõ módon:

```
m.eigenvalues()
m.eigenvectors_right()
```

- Számoljuk ki a korábbi blokkmátrix sajátértékeit!

Listák

```
[kifejezés for elem in bejárható_objektum]
```

Egy olyan listát hoz létre melyben a **kifejezés** szerepel a **bejárható_objektum** minden elemére. Bejárható objektum például egy lista, az is amit a **range** függvény hoz létre.

```
[kifejezés if feltétel else kifejezés_alt for elem in bejárható_objektum]
```

Mint az elõzõ, de csak azok az elemek lesznek benne melyekre teljesül a **feltétel**.

```
[kifejezés if feltétel1 else kifejezés_alt for elem1 in bejárható_objektum1
           for elem2 in bejárható_objektum2
           for elemN in bejárható_objektumN]
```

Több feltétel és ciklus is írható akár.

Pl:

```
[n^2 for n in range(1, 5)]  # [1, 4, 9, 16]
[n for n in [-1, 2, -3, 4] if n > 0]  # [2, 4]
```

Feladatok

Mit csinál?

Futtassuk le az alábbi példákat és értelmezzük õket mi is történik bennük és hogyan érjük ezt el.

```
[n for n in range(1, 10)]

[(n, m) for n in range(1, 10) for m in range(1, 5)]

[n for n in range(1, 10) if is_prime(n)]

[n for n in range(1, 100) if n % 5 == 0 and n % 7 == 1]

[(n, m) for n in range(1, 5) for m in range(n, 5)]

[(m, n) for n in range(1, 10) for m in range(n, 10) if m % n == 0]

sorted([(m, n) for n in range(1, 10) for m in range(n, 10) if m % n == 0])

sum([n for n in range(1, 10) if is_prime(n)])
```

Az utolsóhoz egy kis spoiler, ha nem menne: spoiler

```
[n for n in range(1, 100) if n == sum([m for m in range(1, n) if n % m == 0])]
```

Oldjuk meg

1. Keressük meg az összes olyan 1000 alatti négyzetszámot, melynél eggyel nagyobb szám prím. Pl a 4 ilyen.
2. Keressük meg az összes olyan 100 alatti számpárt, melyekre igaz, hogy mindkett? prím és az egészosztással vett eredményük is prím. Pl (11, 2) ilyen.
3. Keressük meg az összes egy jegy? számhármast, mely egymás után írva megegyezik a köbeik összegével. Ilyen például az 1, 5, 3, mert 1^3 + 5^3 + 3^3 == 153
4. Keressük meg az összes olyan 1000 alatti számot, melynek négyzete megegyezik az nálánál kisebb osztói köbeinek az összegével. (Egy kis csavar a tökéletes számokon)
5. Keressük meg az összes olyan 10000 alatti számot, mely legalább kétféleképpen írható fel 2 darab szám köbének összegeként.

Bonusz

Ha kifutnánk a feladatokból: sagelab.pdf