Tartalomjegyzék

Feladatok

Átlaghoz legközelebbi

Írjunk függvényt, mely a kapott valós számokat tartalmazó listában megkeresi a lista elemeinek átlagához legközelebbi számot és ezzel tér vissza.

Csúnya:

```
def kozel(L):
    atlag = 0
    for e in L:
        atlag += e
    atlag = atlag / len(L)
    kozeli = L[0]
    for e in L:
        if abs(atlag - e) < abs(atlag - kozeli):
            kozeli = e
    return kozeli

print kozel([1, 7, 8, 4, 6, 3, 6])
```

Szebb:

```
def atlag_szamol(L):
    szum = 0
    for e in L:
        szum += e
    return szum / len(L)

def kozel(L):
    atlag = atlag_szamol(L)
    kozeli = L[0]
    for e in L:
        if abs(atlag - e) < abs(atlag - kozeli):
            kozeli = e
    return kozeli
```

```
print kozel([1, 7, 8, 4, 6, 3, 6])
```

n hosszú növekvõ részek

Írjunk függvényt, mely kap egy listát és egy egész számot (n). Megkeresi az összes olyan n hosszú részlistát, amire igaz, hogy az elemei növekvõ sorrendben vannak. Ezeket a listákat beteszi egy fõ listába és ezt adja vissza. Segítség: bontsuk részfeladatokra!

```
def novekvo(L):
    for i in range(len(L) - 1):
        if L[i] > L[i + 1]:
            return False
    return True

def novekvo_reszek(L, n):
    uj = []
    for i in range(len(L) - n + 1):
        if novekvo(L[i:i + n]):
            uj.append(L[i:i + n])
    return uj
```

CloudCoder

nevkonfliktus

```
def nevkonfliktus(nevek):
    for i in range(len(nevek)):
        for j in range(i + 1, len(nevek)):
            if nevek[i] == nevek[j]:
                return True
    return False
```

vagy

```
def nevkonfliktus(nevek):
    for i in range(len(nevek)):
        for j in range(len(nevek)):
            if nevek[i] == nevek[j] and i != j:
                return True
    return False
```

ismetles

```
def ismetles(rovid_lista):
    uj = []
    for l in rovid_lista:
        for i in range(l[0]):
            uj.append(l[1])
    return uj
```

kiejtes

```
def kiejtes(szo):
    magan = 0
    maganhangzok = "aeiou"
    for i in range(len(szo)):
        if szo[i] in maganhangzok:
            magan += 1
```

```
    return magan * 2 <= len(szo) - magan
```

szorzotabla

```
def szorzotabla(n, m):
    uj = []
    for i in range(1, n + 1):
        uj.append([])
        for j in range(1, m + 1):
            uj[i - 1].append(i * j)
    return uj
```