Tartalomjegyzék

- 1 3. gyakorlat - rekurzió és ciklus
  - 1.1 Feladatok
  - 1.2 Feladatok megoldása
    - 1.2.1 zarojel
    - 1.2.2 cserebere
    - 1.2.3 osztosor2
    - 1.2.4 tornyok
  - 1.3 Házi feladatok megoldása
    - 1.3.1 pascal
    - 1.3.2 nalattak

3. gyakorlat - rekurzió és ciklus

Ezen a gyakorlaton az el?z? heti el?adás anyagát dolgozzuk fel.

Feladatok

Feladatok a CloudCoder-en megtalálhatóak. Ajánlott sorrend:

1. zarojel
2. cserebere
3. osztosor2
4. tornyok

Feladatok megoldása

zarojel

```
def zarojel(l):
    szamlalo = 1
    for elem in l:
        szamlalo += zarojel(elem)
    return szamlalo
```

cserebere

```
def cserebere(szo, cserek):
    for hely, betu in cserek:
        szo_eleje = szo[:hely]
        szo_vege = szo[hely+1:]
        szo = szo_eleje + betu + szo_vege
    return szo
```

osztosor2

```
megoldott = {}

def osztosor2(szam, osztok):
    if (szam, osztok) in megoldott:
        return megoldott[(szam, osztok)]
    else:
        megoldas = 1
        for oszto in osztok:
            if szam % oszto == 0:
                megoldas += osztosor2(szam / oszto, osztok)
        megoldott[(szam, osztok)] = megoldas
        return megoldas
```

tornyok

```
def tornyok(magassagok, k):
    n = len(magassagok)
    # Létrehozzuk az k*n-es táblázatot csupa 0 elemmel.
    tablazat = [[0 for j in range(n)] for i in range(k)]

    # tablazat[i][j] értékének reszfeladat(i+1, j+1) értékét
    # szeretnénk kiszámolni (a 0-tól indexelés miatt).

    tablazat[0][0] = magassagok[0]
    for j in range(1, n):
        tablazat[0][j] = max(magassagok[j], tablazat[0][j-1])

    for i in range(1, k):
        for j in range(n):
            if magassagok[j] >= j:
                epit_osszeg = magassagok[j]
            else:
                epit_osszeg = magassagok[j] + tablazat[i-1][j - magassagok[j] - 1]
            nem_epit_osszeg = tablazat[i][j-1]
            tablazat[i][j] = max(epit_osszeg, nem_epit_osszeg)
    return tablazat[k-1][n-1]
```

Házi feladatok megoldása

pascal

```
def pascal(n):
    haromszog = [[0 for j in range(i+1)] for i in range(n)]
    haromszog[0][0] = 1
    for i in range(1, n):
        for j in range(i+1):
            if j == 0 or j == i:
                haromszog[i][j] = 1
            else:
                haromszog[i][j] = (
                    haromszog[i-1][j] +
                    haromszog[i-1][j-1])
    return haromszog
```

nalattak

Ezt a feladatot többféleképpen meg lehetett oldani, pl. a faktoriálisok kiszámításával. Számomra a legegyszer?bb megoldás a *pascal* megoldásának felhasználása volt:

```
def nalattak(n, k):
    haromszog = [[0 for j in range(i+1)] for i in range(n+1)]
    haromszog[0][0] = 1
    for i in range(1, n+1):
        for j in range(i+1):
            if j == 0 or j == i:
                haromszog[i][j] = 1
            else:
                haromszog[i][j] = haromszog[i-1][j] + haromszog[i-1][j-1]
    return haromszog[n][k]
```