Tartalomjegyzék

- 1 Vezérlési szerkezetek (if, for, while)
- 2 Lista és tuple alapok
- 3 Listák kezelése
- 4 Vegyes feladatok
    - 4.1 Tökéletes számok
    - 4.2 A 3n+1 probléma
- 5 Szótárak kezelése

Vezérlési szerkezetek (if, for, while)

1. Definiálj egy Sage függvényt *elojel* néven, amelynek egy bemenete van (*a*), és a "http://wiki.math.bme.hupozitiv"http://wiki.math.bme.hu karakterláncot (vagyis stringet) írja ki ha a kapott paraméter pozitív, "http://wiki.math.bme.hunegativ"http://wiki.math.bme.hu-at ad vissza ha a szám negatív, és "http://wiki.math.bme.hunulla"http://wiki.math.bme.hu-ta ad ha nulla volt a paraméter értéke.
2. Írj egy Sage függvényt *primek_szama* néven, amely bemenetként kap egy számot (*n*) és visszaadja az n-nél kisebb prímek darabszámát!
3. Írj egy függvényt *elso_primek* néven, amely bemenetként kap egy számot (*n*) és kiírja az els? *n* prímszámot!

Lista és tuple alapok

1. Adj meg egy legalább 5 elem?, egész számokat tartalmazó listát, és rendeld az *L* változóhoz!
2. Írasd ki a lista második elemét!
3. Írasd ki a lista második, harmadik, és negyedik eleméb?l álló részlistát (használd a kett?spontot a szögletes zárójelen belül)!
4. Írasd ki a lista els? 3 elemét!
5. Írasd ki a lista utolsó elemét (negatív index)!
6. F?zz a lista végére egy új elemet, értéke legyen ugyanaz, mint az els? elem! (*append()*)
7. Keresd meg hogy egy elem hányadik indexen szerepel a listádban! (*index()*)
8. Számold meg, hányszor szerepel az els? elem a listában! (*count()*)
9. Mennyi a listádban szerepl? számok összege? (*sum()*)
10. Rendezd a listádat növekv? sorrendbe! (*sort()*)
11. F?zd össze az *L* listát az [1,2,3] listával! (használd a + operátort)
12. Készíts listát (*A* néven) az "http://wiki.math.bme.huabrakadabra"http://wiki.math.bme.hu stringb?l! (*list()*)
13. Készíts stringet az *A* listából! (*str()*)
14. Készíts tuple-t *T* néven az *A* listából, majd írd ki az utolsó elemét!
15. Változtasd meg a *T* els? elemét!

Listák kezelése

1. Írj függvényt *sokszorozo* néven, amely bemenetként kap egy számot (*n*) és még egy paramétert, *a*-t (ennek a típusa bármi lehet). A függvény adjon vissza egy listát, amiben *n*-szer szerepel az *a* értéke. Például: *sokszorozo(3, "http://wiki.math.bme.hubla"http://wiki.math.bme.hu)* kimenete ["http://wiki.math.bme.hubla"http://wiki.math.bme.hu, "http://wiki.math.bme.hubla"http://wiki.math.bme.hu, "http://wiki.math.bme.hubla"http://wiki.math.bme.hu] legyen.
2. Írj függvényt *reszlista* néven, amely bemenetként kap egy listát, és a függvény kimenete az a részlistája legyen az eredetinek, amely a két els? *0* közötti elemekb?l áll. Ha nincs a bemeneti

listában legalább 2 nulla, akkor legyen a kimenet az üres lista.
Például *reszlista(1, 4, 0, 3, 5, 6, 3, 0, 23, 5, 0, 1, 0, 4)* eredménye legyen *[3, 5, 6, 3]*

3. Írj Sage függvényt amely megfordít egy bemenetként kapott listát!

Vegyes feladatok

Tökéletes számok

- Írj függvényt, amely egy *a* számról eldönti, hogy az tökéletes szám-e (megegyezik a nála kisebb osztóinak az összegével, pl: 6, 28).
- Írj függvényt, amely 1-t?l *n*-ig összegy?jti és egy listában visszaadja a talált tökéletes számokat!

A 3n+1 probléma

A híres 3x+1 probléma (Collatz-sejtés) : végy egy számot, ha páratlan, szorozd meg 3-mal és adj hozzá 1-et, ha páros, oszd el 2-vel. Az az állítás, hogy így bármilyen pozitív egész számból indulva el?bb-utóbb eljutunk 1-ig.

Írj Sage függvényt, amely *x*-et kap bemenetként, és sorban kiírja a lépéseket 1-ig!

Szótárak kezelése

Legyen egy *gyumolcs_arak* nev? szótárunk, a következ? kulcs-érték párokkal:
'alma': 150
'szilva': 190
'ananász': 450
'banán': 300

És legyen egy másik, *vasarlas* nev? szótár, amely azt tárolja, mib?l mennyit vettünk:
'banán': 0.6
'alma': 1.5
'ananász': 2

Írj Sage függvényt (legyen a neve *ar_szamolo*), amely megkapja a fenti két szótárat (els? paramétere legyen az árakat tartalmazó), és kiszámolja, hogy mennyit kell fizetnünk a gyümölcsökért!