(15g.py)

Lista metódusok

A „pont” operátor is használható a lista objektumok beépített metódusainak elérésére. Próbáljuk ki a leghasznosabb metódusokat, amelyekkel módosíthatunk a listáinkon!

- Elöször létrehozunk egy üres listát, majd feltöltjük elemekkel! Ehhez az **új_lista.append(x)** utasítást használjuk a minta alapján
- Beszúrunk 5 db számot a listánkba egymás után sorban.
- Majd mindig kiíratjuk a lista tartalmát a képernyőre, hogy lássuk a változást!
- A második helyre (a 9-es helyére) beszúrunk egy 15-ös értéket, az **uj_lista.insert(2,15)** utasítással!
- Az **uj_lista.count(15)** utasítással megszámolhatjuk, hogy hányszor fordul elő a listánkban a 15-ös szám!
- Az eddigi listánk végére hozzáadhatunk egy másik listát a **uj_lista.extend([5, 9, 5, 11])** utasítással!
- Keressük megy az index értékét, sorszámát a listában az **print(uj_lista.index(9))** utasítással!
- Fordítsuk meg a listánkat az **uj_lista.reverse()** utasítással!
- Rendezzük növekvő sorrendbe a listánk értékeit az **uj_lista.sort()** utasítással!
- Töröljük, távolítsuk el az első előfordulását a 15-ös számnak az **uj_lista.remove(15)** utasítással!

Gépeljük be a kódot, kövessük végig a parancsokat, és a megjelenített eredményt!

```
uj_lista = []
uj_lista.append(15)
uj_lista.append(27)
uj_lista.append(9)
uj_lista.append(12)
uj_lista.append(23)
print(uj_lista)
print("------------------")
uj_lista.insert(2, 15)
print(uj_lista)
print("------------------")
print(uj_lista.count(15))
print("------------------")
uj_lista.extend([5, 9, 5, 11])
print(uj_lista)
print("------------------")
print(uj_lista.index(9))
print("------------------")
uj_lista.reverse()
print(uj_lista)
print("------------------")
uj_lista.sort()
print(uj_lista)
print("------------------")
uj_lista.remove(15)
print(uj_lista)
```

```
C:\Users\kolmank\AppData\Local\Program
[15, 27, 9, 12, 23]
------------------
[15, 27, 15, 9, 12, 23]
------------------
2
------------------
[15, 27, 15, 9, 12, 23, 5, 9, 5, 11]
------------------
3
------------------
[11, 5, 9, 5, 23, 12, 9, 15, 27, 15]
------------------
[5, 5, 9, 9, 11, 12, 15, 15, 23, 27]
------------------
[5, 5, 9, 9, 11, 12, 15, 23, 27]
```

(15h.py)

Ebben a példában egy fix listáról készítünk **másolat**ot, majd egy **hivatkozás**t! Aztán az eredeti listából törlünk két elemet. Látszik, hogy **a hivatkozásból szintén eltűntek az elemek, a másolatból nem**.

```
szamok=[12,6,8,11,5,7,9,2,1]
print("A számok lista tartalma: ",szamok)
masolat_1=szamok.copy()
print("A másolat lista tartalma: ",masolat_1)
szamok_2 = szamok
print("A számok 2 lista tartalma: ",szamok_2)
szamok.remove(12)
szamok.remove(11)
print("A számok lista tartalma: ",szamok)
print("A másolat lista tartalma: ",masolat_1)
print("A számok 2 lista tartalma: ",szamok_2)
```

```
E:\00_MM\12_Python_programozas\programok\venv\Scripts\pyt
A számok lista tartalma:  [12, 6, 8, 11, 5, 7, 9, 2, 1]
A másolat lista tartalma:  [12, 6, 8, 11, 5, 7, 9, 2, 1]
A számok 2 lista tartalma:  [12, 6, 8, 11, 5, 7, 9, 2, 1]
A számok lista tartalma:  [6, 8, 5, 7, 9, 2, 1]
A másolat lista tartalma:  [12, 6, 8, 11, 5, 7, 9, 2, 1]
A számok 2 lista tartalma:  [6, 8, 5, 7, 9, 2, 1]
```

(15i.py)

Lista bejárása

for ciklussal

Értelmezd a példákat!

```
napok =["Hétfő","Kedd","Szerda","Csütörtök"
        "Péntek","Szombat","Vasárnap"]
for i in napok:
    print(i, end=" ")
print()
for j in range(len(napok)):
    print("%d. nap: %s" % (j+1, napok[j]))
```

```
E:\00_MM\12_Python_programozas\programok\venv\Script
Hétfő Kedd Szerda Csütörtök Péntek Szombat Vasárnap
1. nap: Hétfő
2. nap: Kedd
3. nap: Szerda
4. nap: Csütörtök
5. nap: Péntek
6. nap: Szombat
7. nap: Vasárnap
```

(15j.py)

Karakterláncok darabolása listává

Gyakran az összetartozó adatokat egymás mellett szóközzel, vesszővel, pontosvesszővel választjuk el egymástól. Adatbázisokban ezeket hívjuk rekordoknak. Például ha egy autó adatait nézzünk:
AAA-111, szűrke, Citroen, C4, 2012, 3500000
Mindenki érti, hogy melyik adat mit jelenthet. Ezekre az adatokra általában külön lehet szükségünk, ezért nézzünk pár parancsot, melyekkel változtatunk a struktúrán! (Tulajdonképpen feldolgozhatóvá tesszük az adatokat!)
Tömböt készítünk, melynek elelmeivel műveleteket végezhetünk!

```
16c.py
1  auto="AAA-111, szűrke, Citroen, C4, 2012, 3500000"
2  auto=auto.replace(",","")
3  print(auto)
4  print("_____________________")
5  auto_adatok=auto.split()
6  print(auto_adatok)
7  print("_____________________")
8  for i in auto_adatok:
9      print(i)
```

```
Run:  16c
C:\Users\eloado\PycharmProjects\pythonProject\venv\Scripts
AAA-111 szűrke Citroen C4 2012 3500000
_____________________
['AAA-111', 'szűrke', 'Citroen', 'C4', '2012', '3500000']
_____________________
AAA-111
szűrke
Citroen
C4
2012
3500000
```

- Az első sorban megadunk egy sztring adatsort vesszőkkel elválasztva.
- A második sorba, az „auto” változóba visszatesszük a vesszők nélküli szöveget. Erre a **replace()** utasítást használjuk a minta szerint. A vesszők helyére a („”) semmit tesszük. Erre azért van szükség, mert a darabolásnál, ha bent marad a vessző, akkor az a szó végén maradna egy felesleges vessző karakter. (pl.: fehér,).
- Aztán ellenőrzésül kiíratjuk az aktuális auto változó tartalmát. Vonallal elválasztva a következő soroktól. (3-4. sor)
- Majd behozunk egy új tömböt (auto_adatok néven), mely tömbbe a szóközök mentén feldarabolt szöveget beleteszem egyesével. Erre a **split()** parancsot használjuk a minta alapján. Így lesznek a különböző adat elemek egy tömb elemei. (5. sor)
- Aztán kiíratom egy sorba a tömböt, hogy lássuk az eredményt. (6. sor)
- Végül a tömb elemeit kiíratom egymás alá for utasítás segítségével, hogy lássuk hogy az elemekkel külön -külön dolgozhatunk. (8-9. sor)

(15k.py)

Képzeljünk el egy autókölcsönzőt! Az előző programból kiindulva most több autó (4db) adatait tároljuk egy tömbben.

Készítsünk programot, melynek az eredménye a jobb oldali képernyőkép!

A kérdés az, hogy átlagosan hány évesek az adatbázisban lévő autók?
Mennyi a legdrágább autó ára?

```
Run:  16d
C:\Users\kolmank\AppData\Local\Programs\Python\Python38\pyt
['AAA-111 szűrke Citroen C4 2012 3500000', 'BBB-222 fehér A
_____________________
['AAA-111', 'szűrke', 'Citroen', 'C4', '2012', '3500000']
['BBB-222', 'fehér', 'Audi', 'A4', '2015', '5500000']
['CCC-333', 'kék', 'BNW', 'Q6', '2020', '25000000']
['DDD-444', 'zöld', 'Skoda', 'Octávia', '2008', '1300000']
_____________________
Az autók átlagéletkora:  8.25  év
A legdrágább autó ára:  25000000  Ft.
```