TUTORIAT 3

Linkuri utile:

- https://www.python.org/
- https://docs.python.org/3/
- https://www.codecademy.com/learn/learn-python-3
- https://www.learnpython.org/
- https://stanfordpython.com/#/

Comentariu multiplu PyCharm: CTRL + /

Ce conține tutoriatul ?

I. Liste (**clasa list**)
   1. Creare
   2. Accesare elemente și subsecvențe
   3. Metode uzuale
   4. Matrice

II. Exerciții

I.Clasa **LIST**

➔ **MUTABILE** (li se poate modifica valoarea după creare)
➔ conțin elemente de tipuri diferite (recomandabil, NU)
➔ elementele sunt indexate de la **0**
➔ sunt permise valorile duplicate

II.1. **Creare**

Metoda de creare	Exemplu
cu ajutorul constructorului **list()**	nume_lista = list()
enumerarea **elementelor între []**	nume_lista = [1, 14, -5, 7, 100]
specificarea unui **interabil în constructor**	nume_lista = list("tutoriat")
comprehensiune **[expresie for nume_varianila in iterabil]** **[expresie for nume_varianila in iterabil if conditie]** **[expresie1 if conditie else expresie2 for nume_variabila in iterabil]**	nume_lista = [a**2 for a in range(2,5)] nume_lista = [a for a in range(1, 20) if a % 2 == 0] nume_lista = [a if a > 0 else –a for a in range(-5, 4)]

!ATENȚIE! my_list = [] -> lista vidă

Listele pot avea elemente de tip listă. Acestea poartă denumirea de **liste imbricate**.

Exemplu:

```
my_list = [4, 12, [2, 6], -4, [1, 8, 16]]
# accesare
print(my_list[1])  => 12
print(my_list[4]) => [1, 8, 16]
print(my_list[2][1]) => 6
print(my_list[4][0]) => 1
print(my_list[4][2]) => 16
```

II.2.Accesare elemente și subsecvențe

Pentru a accesa elementele unei liste se utilizează [ ]. **!ATENȚIE!** Indexarea începe de la 0.

Exemple:

my_list[i] => elementul de pe poziția i din lista my_list, (începând cu poziția 0)
!ATENȚIE! i poate fi și negativ.

Tip accesare	Rezultat
nume_lista[i]	elementul de pe poziția i
nume_ lista[i:j]	elementele de pe pozițiile i, i+1,..,j-1
nume_ lista[:j]	elementele de pe pozițiile 0,1,..,j-1
nume_ lista[i:]	elementele de pe pozițiile 0,1,..,len(nume_sir)-1
nume_ lista[:]	toată secvența
nume_lista[i:j:k]	elementele de pe pozițiile i,i+k,i+2k,….

II.3. Metode uzuale

Concatenări: +, *

Exemplu: lista1 = [1, 5, 9, 13]

lista2 = [2, 7, 11]

lista1 = lista1 + lista2 => lista1 = [1, 5, 9, 13, 2, 7, 11]

!ATENȚIE! se creează un obiect nou, nu adaugă la lista lista1

lista = [0]; n = 4

lista * n sau n *lista => [0, 0, 0, 0]

Funcții comune tuturor secvențelor: **len**(nume_lista) => returnează numărul de elemente al listei, **min**(nume_lista) => returnează valoarea celui mai mic element din listă, **max**(nume_lista) => returnează valoarea celui mai mare element din listă

Apartenența: in, not in

Exemplu: my_list = [1, 5, 9, 13]

```
if 10 not in my_list:
    print("10 nu se gaseste in lista")
```

Comparare: <, >, <=, >=, ==, !=

Exemplu:

Lista1 = [1, 5, 9]

Lista2 = [1, 5, 9]

print(lista1 == lista2) => True

lista1 = [1, 5, 9, 13]

lista2 = [2, 3, 11]

print(lista2 > lista1) => True

Metode (funcție a unei clasei)

Metoda	Apel	Efect
count	lista.count(element)	returnează numărul de apariții a lui element în lista
index	lista.index(element)	returnează indexul primei apariții a lui element în lista
append	lista.append(element)	adaugă element la finalul listei
extend	lista.extend(iterabil)	adaugă toate elementele din interabil la finalul listei
insert(index, element)	lista.insert(index, element)	inserează element pe poziția index în lista
reverse	lista.reverse()	returnează lista cu elementele în ordine inversă(de la index len(lista) la index 0)
remove(element)	lista.remove(element)	elimină prima apariție a lui element în lista (EROARE DACĂ NU GĂSEȚTE ELEMENTUL)
pop	lista.pop(i) (implicit i=-1, adică elimină ultimul element)	returnează și elimină valoarea elementului de la indexul i(EROARE DACĂ i>len(lista))
clear	lista.clear()	elimină toate elementele din listă => lista vida
copy	lista.copy()	copiere superficială a listei

sort(key=None,reverse=False)	lista.sort(key=None,reverse=False)	sortează lista în ordine crescătoare(dacă reverse = True, sortează lista în ordine descrescătoare)

Instrucțiuni care modifică o listă

nume_lista[index] = valoare => modifică valoarea elementului de pe poziția index

nume_lista[i:j] = iterabil => înlocuiește elementele de pe pozițiile i,..,j-1 cu elementele din iterabil

nume_lista[i:j:k] = iterabil (trebuie să aibă aceeași lungime)

del lista[i:j] => elimină elementele de po pozițiile i, i+1,..j-1

Exemplu:

```
my_list = [1, 5, -2, 0, 4]
my_list[2] = 10
print(my_list) => [1, 5, 10, 0, 4]
my_list[1:3] = [2, 5]
print(my_list) => [1, 2, 5, 0, 4]
my_list[1:5:2] = [1, 6]
print(my_list) => [1, 1, 5, 6, 4]
del my_list[2:5]
print(my_list) => [1, 1]
```

II.4.Matrice

Exemplu creare si initializare elemente matrice (cu numar linii si coloane citit de la tastatura)

```
numar_linii = int(input("Numar linii:") )
numar_coloane = int(input("Numar coloane: ") )
matrice1 = [[valoare] * numar_coloane] * numar_linii
matrice2 = [[valoare for j in range(numar_coloane)] for i in range (numar_linii)]
matrice3 = [[ valoare for j in numar_coloane]] * numar_linii
matrice4 = [[valoare] * numar_coloane for i in range(numar_linii)]
```

III.Exerciții

1.Se citesc de la tastatură un număr natural n, n elemente ale unei liste și un număr x. Să se elimine din listă toate aparițiile lui x. Dacă x nu se găsește în listă se va afișa un mesaj corespunzător.

Exemplu: n = 8, my_list = [5, 9, 12, 2, 9, 7, -5, 9], x = 9 => [5, 12, 2, 7, -5]

2.Se dă următoarea listă: [1, 2, [3, [4, 5, [6, 7], 11], 12], 13, 14]. Să se adauge la această listă sub lista [8, 9, 10], astfel încât rezultatul să fie următorul: [1, 2, [3, [4, 5, [6, 7, 8, 9, 10], 11], 12], 13, 14].

3.Utilizând crearea listelor prin comprehensiune, să se afișeze următoarele liste:

a) lista numerelor pare din intervalul [2, 25];

b) lista cu elementele de forma -1, 2, -3, 4, -5,.., +/- n, n citit de la tastatură;

c) lista cu elementele de pe poziții pare ale unei liste list1 citită de la tastatură;

d) lista elementelor, dintr-o listă dată, care au paritate diferită de poziția lor;

e) listă ce conține n liste (n citit de la tastatură) astfel: lista de la indexul 0 este lista vidă, lista de la indexul 1 are un singur element egal cu 1, lista de la indexul 2 are două elemente egale cu 2, etc.(pentru n=5 => [[], [1], [2,2],[3,3,3],[4,4,4,4]].

4. Se citește o listă cu n elemente, numere naturale(n citit de la tastatură).Să se ordoneze decrescător elementele care au suma cifrelor număr par și în ordine crescătoare cele care au suma cifrelor număr impar. Să se afișeze lista ca în exemplul următor:

n = 6, lista = [33, 56, 77, 734, 45, 34] => [734, 77, 33, 34, 45, 56]

5.Se citesc de la tastatură numerele naturale m, n, x.

a) Construiți și afișați o matrice cu m linii și n coloane, care să conțină primii m*n multiplii ai lui x, ca în exemplul:

m = 3, n = 4, x = 5 => 0 5 10 15

35 30 25 20

40 45 50 55

b) Afișați suma maximă obținută de pe o coloană și indicele coloanei. În cazul în care o mai multe coloane au suma maximă, atunci va fi afișată oricare dintre ele;

c) Pentru n=m citite de la tastatură, afișați: suma elementelor de pe diagonala principală, produsul elementelor de pe diagonala secundară, suma elementelor impare aflate sub diagonala principală și deasupra celei secundare.

6. https://pynative.com/python-list-quiz/