Laborator 2

1. Liste, dictionare, multimi

1. Dandu-se o lista, sa se inverseze ordinea elementelor. Exista cel putin Jdoua variante de rezolvare: slicing si folosind o metoda a tipului lista.
2. Pentru o lista de cuvinte (stringuri), sa se obtina un dictionar care are drept cheie cuvintele distincte din lista, iar valorile corespunzatoare sa fie numarul de aparitii ale fiecarui cuvant
3. Dandu-se o lista de tuple binare, sa se obtina o lista care contine valoarea maxima a fiecarui tuplu.
4. Sa se calculeze suma numerelor de indice par si produsul numerelor de indice impar dintr-o lista.
5. Documentati-va asupra tipului de date Set (multime) si folositi-l pentru a determina elementele unice dintr-o lista de numere.
6. Pentru o lista de stringuri si un numar n > 1, sa se produca o lista care contine toate stringurile din lista initiala cu toate numere 1..n. Exemplu: pentru lista = ['a', 'b', 'c'], n=2 se va produce lista_2 = ['a1', 'a2', 'b1', 'b2', 'c1', 'c2']; ordinea poate sa difere. Incercati o implementare bazata pe list comprehension.
7. Dandu-se o lista de cuvinte, sa se produca o alta lista cu aceleasi cuvinte, dar: prima litera a fiecarui cuvant sa fie mare, celelalte din cuvant - mici.
8. Documentati-va si explicati comportamentul pentru urmatoarea secventa de cod:

```
this = ['I', 'am', 'not', 'a', 'crook']
that = ['I', 'am', 'not', 'a', 'crook']
print("Test 1:", this is that)
that = this
print("Test 2:", this is that)
```

2. Functii

1. Pentru o lista de numere, sa se scrie o functie care returneaza suma elementelor si diferenta maxima dintre ele.
2. Scrieti o functie care preia doua liste si returneaza `True` daca cele doua liste contin cel putin k elemente comune (k dat ca parametru, intreg >= 1, valoare implicita 1), `False` altfel.
3. Sa se scrie o functie care determina daca o lista contine doar cuvinte. Functia trebuie sa testeze daca elementele din lista sunt stringuri; un string este cuvant daca nu contine: spatiu, virgula si alti separatori.
4. Sa se scrie o functie care returneaza numarul de litere mici si mari - 2 rezultate - dintr-un parametru dat.
5. Sa se scrie o functie care primind o lista de dictionare, returneaza `True` daca toate dictionarele sunt goale si `False` altfel.
6. Sa se scrie o functie recursiva care sa faca ridicarea la putere a unui numar, astfel:

$$a^n = \begin{cases} a & \text{daca } n = 1 \\ \left(a^{n//2}\right)^2 & \text{daca } n \text{ e par} \\ \left(a^{n//2}\right)^2 \cdot a & \text{daca } n \text{ e impar}, n > 1 \end{cases}$$

unde prin $n//2$ se reprezinta catul impartirii intregi intre n si 2.

3. Exercitii simple folosind API de NumPy

1. Folosind functia `np.where`, determinati care sunt liniile si coloanele care contin un element dat intr-o matrice.
2. Pentru o matrice de numere, sa se determine care este produsul elementelor pe linii.
3. Folosind functia `np.all`, determinati daca toate elementele unei matrice `a` sunt mai mari sau egale decat toate de pe pozitiile corespondente din `b`.
4. Determinati care e maximul pe coloanele unei matrice (functia `max` cu argumentul `axis` setat corespunzator) si maximul intregii matrice.
5. Plecand de la un vector de numere de tip float pentru care un element este pus la valoarea `np.nan`, calculati suma elementelor. Explicati rezultatul.
6. Pentru un vector de numere, care sunt *toate pozitiile* pe care apar valoarea maxima? Folositi functie NumPy.
7. Sa se calculeze suma elementelor non-NaN intr-un 2d numpy array, in 3 feluri. Folosind comanda magica `%%timeit` Built-in magic commands (http://ipython.readthedocs.io/en/stable/interactive/magics.html#cell-magics), IPython Magic Commands (https://jakevdp.github.io/PythonDataScienceHandbook/01.03-magic-commands.html), sa se determine care e mai rapida.
   A. Peste o copie a tabelului initial (functia membru `copy()` a unui tablou) se suprascriu elementele nan cu 0, apoi se face suma;
   B. Folosind functia `np.nansum`;
   C. Folosind indiciere booleana si functi `np.isnan`, se determina colectia elementelor non-nan si se calculeaza suma lor in mod obisnuit.
8. Se considera vectorul: `a = np.array([230, 10, 284, 39, 76])`. Folosind indexarea logica, sa se inmulteasca elementele din vector care sunt mai mici ca 100 cu 2 (restul raman neschimbate). Folosind ciclare, sa se aplice aceeasi operatie pana cand toate elementele devin mai mari ca 100. Sa se afiseze din vectorul final elementele care sunt mai mari de 150 si mai mici de 200.
9. La curs s-a prezentat problema gasirii celei mai apropiate peerchi de puncte in spatiul 2d. Rescrieti rezolvarea pentru a determina cele mai apropiate k puncte (k dat) intr-un spatiu 5-dimensional.
10. Se dau doi vectori **x** si **y** de *m*, respectiv *n* componente. Sa se calculeze, folosind cod vectorizat si cod Python clasic (ciclu `for`):

$$\sum_{i,j} x_i \cdot y_j$$

Efectuati masuratori asupra vitezei de calcul folosind `%%timeit`.