Języki Symboliczne, laboratorium 03

Zadanie 1. Napisz funkcję przyjmującą listę jako argument, wypisującą wartość najmniejszego elementu listy, największego oraz średnią arytmetyczną wszystkich elementów.

https://docs.python.org/3/tutorial/datastructures.html

Zadanie 2. Napisz funkcję kopiującą listę przesłaną jako argument. Wykorzystaj mechanizm *list comprehensions*.

https://docs.python.org/3/tutorial/datastructures.html#list-comprehensions

Zadanie 3. Napisz funkcję przyjmującą jako argument listę, kopiującą ją i zmieniającą każdy element nowej listy tak, aby był on średnią arytmetyczną siebie oraz swoich bezpośrednich sąsiadów. Lista przesłana jako argument nie powinna być modyfikowana. Pierwszy i ostatni element listy mogą pozostać niezmienione.
Stwórz listę 20 losowych liczb i zaprezentuj działanie powyższej funkcji. Przykładowo:

```
>>> lista
[3, 1, 2, 0, 4]
>>> blur(lista)
[3, 2.0, 1.0, 2.0, 4]
```

Zadanie 4. Napisz funkcję przyjmującą jako argumenty dwie listy, zwracającą listę, której elementami jest suma odpowiednich elementów przesłanych list (c[0]=a[0]+b[0], c[1]=a[1]+b[1] itd.). Jeśli listy nie są równej długości, wyjściowa lista powinna mieć tyle elementów, ile ma krótsza z list. Wykorzystaj mechanizm *list comprehensions*.

```
>>> a
[1, 2, 3, 5, 0]
>>> b
[4, 3, 0, 1, 2, 5]
>>> add(a, b)
[5, 5, 3, 6, 2]
```

Zadanie 5. Podobnie jak w zadaniu 4., napisz funkcję przyjmującą jako argumenty dwie listy, zwracającą listę, której elementami jest iloraz odpowiednich elementów przesłanych list. Jeśli listy nie są równej długości, wyjściowa lista powinna mieć tyle elementów, ile ma krótsza z list. Wykorzystaj mechanizm *list comprehensions*. Pamiętaj, że nie można dzielić przez zero – jeśli dzielnik miałby być zerem, to pomiń element (wykorzystaj *if* w *list comprehensions*).

```
>>> a
[1, 2, 3, 5, 0]
>>> b
[2, 1, 0, 5, 3, 6]
>>> div(a, b)
[0.5, 2.0, 1.0, 0.0]
```

Zadanie 6. Napisz funkcję przyjmującą jako argument listę, wybierającą z niej trzy najmniejsze

elementy w kolejności malejącej, trzy największe elementy również w kolejności malejącej, zwracającą listę złożoną z tych elementów. Wykorzystaj metody/funkcje *sort*/*sorted* oraz *reverse*/*reversed*.

```
>>> a
[0, 1, -5, 8, 14, 9, 6]
>>> select(a)
[1, 0, -5, 14, 9, 8]
```

Zadanie 7. Napisz funkcję przyjmującą jako argument listę, kasującą z niej co trzeci element oraz wszystkie elementy o wartości mniejszej od 0. Wykorzystaj *del*.

https://docs.python.org/3/tutorial/datastructures.html#the-del-statement

Zadanie 8. Napisz funkcję przyjmującą jako argument listę oraz pewną liczbę całkowitą *odstęp*, wstawiającą 0 do listy co *odstęp* elementów.

```
>>> lista
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
>>> insert(lista, 2)
[1, 2, 0, 3, 4, 0, 5, 6, 0, 7, 8, 0, 9, 10, 0]
>>> insert(lista, 3)
[1, 2, 3, 0, 4, 5, 6, 0, 7, 8, 9, 0, 10]
```

Zadanie 9. Napisz funkcję przyjmującą jako argument wielkość planszy, zwracającą tablicę dwuwymiarową wypełnioną na zmianę 1 i 0, na kształt szachownicy. Wykorzystaj mechanizm *list comprehensions*.

```
>>> checkboard(4)
[[1, 0, 1, 0], [0, 1, 0, 1], [1, 0, 1, 0], [0, 1, 0, 1]]
>>> checkboard(3)
[[1, 0, 1], [0, 1, 0], [1, 0, 1]]
```

https://docs.python.org/3/tutorial/datastructures.html#nested-list-comprehensions

Zadanie 9000. Napisz funkcję zwracającą liczbę samogłosek w tekście przesłanym jako argument. Napisz funkcję przyjmującą jako argument listę tekstów, zwracającą listę utworzoną z tych elementów przesłanej listy, które zawierają parzystą liczbę samogłosek. Wykorzystaj mechanizm *list comprehensions* oraz *in* do sprawdzania czy litera jest samogłoską [spoiler: strona 3].

```
>>> samogloski(["Robot", "Mech", "Metal Gear", "Android"])
['Robot', 'Metal Gear']
```

ad. 9000.

```
if znak in "aiueoy":
    print("samogloska")
else:
    print("nie samogloska")
```