



# PODSTAWY PROGRAMOWANIA W PYTHON

## PO PIĄTYCH I SZÓSTYCH ZAJĘCIACH:

### 1. Omawiane zagadnienia wtorek:

#### a. listy, listy zagnieżdżone

##### i. listy jako stosy (Stacks -

<https://docs.python.org/3/tutorial/datastructures.html#using-lists-as-stacks> )

##### ii. listy jako kolejki (queues:

<https://docs.python.org/3/tutorial/datastructures.html#using-lists-as-queues> )

##### iii. kopiowanie list

##### iv. list comprehension (magia Pythona) -

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

#### b. tuple - <https://docs.python.org/3/tutorial/datastructures.html#tuples-and-sequences>

#### c. słowniki - <https://docs.python.org/3/tutorial/datastructures.html#dictionaries>

### 2. Omawiane zagadnienia - czwartek:

#### a. Funkcje - <https://docs.python.org/3/tutorial/controlflow.html#defining-functions>

##### i. tworzenie – **def**

##### ii. argumenty funkcji

###### 1. wymagane (pozycyjne)

###### 2. domyślne

a. uwaga na typy referencyjne (listy, obiekty) przy definiowaniu argumentu domyślnego – słowo kluczowe

**None**

##### iii. funkcje:

###### 1. zwracające wartość / obiekt - **return**

###### 2. modyfikujące, bez zwracania wartości

- 3. dokumentacja funkcji - docstring
  - iv. zakres zmiennych:
    - 1. zmienne lokalne funkcji
    - 2. zmienne globalne używane w funkcji – użycie **global**
  - v. import własnych plików i re-użycie kodu
- pamiętać o tym aby **nie używać spacji w nazwach plików** - nazwy plików są traktowane przez Python'a jak nazwy modułów – a te nie mogą mieć spacji, zaczynać się od cyfr (zmienne, nazwy funkcji też!)

3. Popatrzeć sobie jak wygląda przebieg programu gdy mamy funkcje na <http://pythontutor.com>

4. Dokumentacja i przydatne linki:

[https://www.tutorialspoint.com/python/python\\_functions.htm](https://www.tutorialspoint.com/python/python_functions.htm)

<https://www.python.org/dev/peps/pep-0257/>

Najczęstsze pomyłki początkujących Python'istów:

[http://www.onlamp.com/pub/a/python/2004/02/05/learn\\_python.html](http://www.onlamp.com/pub/a/python/2004/02/05/learn_python.html)

## 5. Zadania domowe:

- a. Napisz program, który będzie bazą kontaktów (albo jakąkolwiek inną bazą - zobacz plik *praca\_domowa.py* w day6)), program ma pytać użytkownika, co chce zrobić, dając mu minimum te opcje: dodanie imienia, usunięcie imienia, sprawdzenie czy imię jest w bazie, usunięcie imienia, sprawdzenie ilości imion w bazie oraz zakończenie programu.

Jeśli czujesz się pewniej to postaraj się aby użytkownik mógł podać więcej szczegółów np. nazwisko, nr telefonu, adres itp.

Program ten będziemy pomału rozbudowywać, w kolejnych tygodniach

Oczywiście piszemy już „Czysty kod” stosując się do konwencji

Python'owych: piszemy docstringi, właściwe i znaczące nazwy zmiennych oraz funkcji. I jeśli damy radę to możemy postarać się stworzyć moduły z oddzielnymi funkcjonalnościami.

### b. dodatkowe zadanie dla czujących się pewniej:

„Fair and square”

Little John likes palindromes, and thinks them to be fair (which is a fancy word for nice). A *palindrome* is just an integer that reads the same backwards and forwards - so 6, 11 and 121 are all palindromes, while 10, 12, 223 and 2244 are not (even though 010=10, we don't consider leading zeroes when determining whether a number is a palindrome).

He recently became interested in squares as well, and formed the definition of a *fair and square* number - it is a number that is a palindrome **and** the square of a *palindrome* at the same time. For instance, 1, 9 and 121 are fair and square (being palindromes and squares, respectively, of 1, 3 and 11), while 16, 22 and 676 are **not** fair and square: 16 is not a palindrome, 22 is not a square, and while 676 is a palindrome and a square number, it is the square of 26, which is not a palindrome.

Now he wants to search for bigger fair and square numbers. Your task is, given an interval Little John is searching through, to tell him how many fair and square numbers are there in the interval, so he knows when he has found them all.

Example data:

range(from, to(including) )	output(number of fair and square numbers in range)
1 4	2
10 120	0
100 1000	2