SMARTFACTOR

# BootCamp: zajęcia 5

# Spis treści



# Biblioteki w Pythonie







*Źródła grafik i zestawiania bibliotek: https://www.mygreatlearning.com/blog/open-source-python-libraries/, http://pythoneyes.nl/?p=81, https://keras.io/*

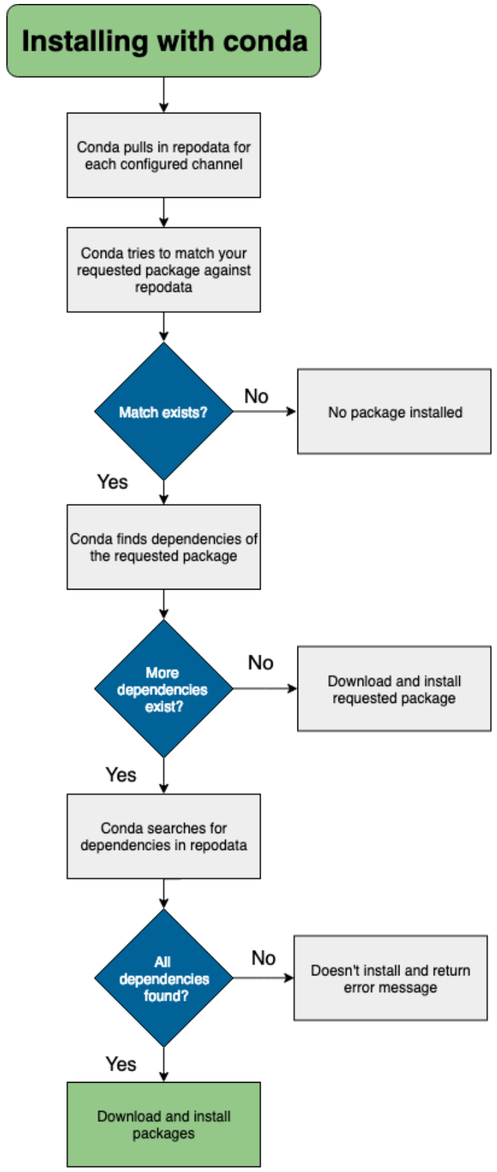
# Skąd brać informacje?

- https://www.mygreatlearning.com/blog/open-source-python-libraries/
- https://pypi.org/
- https://www.edureka.co/blog/python-libraries/

- Artykuły
- Blogi
- Podcasty
- Kursy on-line (MOOC - *massive open online course*)
- Książki
- Rozmowy, wymiana doświadczeń z innymi osobami

*Źródło grafiki: link*

# Instalacja bibliotek z poziomu minicondy




*Źródło: [1]*

## Installing with conda

To install conda packages, in the terminal or an Anaconda Prompt, run:

```
conda install [packagename]
```

During the install process, files are extracted into the specified environment, defaulting to the current environment if none is specified. Installing the files of a conda package into an environment can be thought of as changing the directory to an environment, and then downloading and extracting the artifact and its dependencies---all with the single `conda install [packagename]` command.

Read more about conda environments and directory structure.

- When you `conda install` a package that exists in a channel and has no dependencies, conda:
  - Looks at your configured channels (in priority).
  - Reaches out to the repodata associated with your channels/platform.
  - Parses repodata to search for the package.
  - Once the package is found, conda pulls it down and installs.

SMARTFACTOR

# Importowanie bibliotek, tworzenie własnych funkcji, *wrappery*

1. Słowo kluczowe import
2. Różne sposoby importu
3. Importowanie wybranych funkcji
4. Importowanie z aliasami

5. Definiowanie własnych funkcji
6. Wywoływanie funkcji
7. Zapisywanie funkcji w zmiennej
8. `*args`, `**kwargs`
9. Zwracanie wartości
10. Wrappery – opakowanie dla funkcji
11. Tworzenie *cache* dla funkcji

# Miejsce na notatki

# Zadania do wspólnego rozwiązania

**Zadanie 1**
Napisz funkcję, która jako argument przyjmuję promień kuli i zwraca jej objętość z dokładnością do dwóch miejsc po przecinku

**Zadanie 2**
Napisz funkcję, która jako argument przyjmuje 4 liczby i zwraca:
a) Największą z nich
b) Najmniejszą z nich

**Zadanie 3**
Napisz funkcję, która jako argument przyjmuje listę z liczbami i zwraca:
a) Największy element listy
b) Najmniejszy element listy

**Zadanie 4**
Napisz funkcję, która jako argument przyjmuje liczbę całkowitą i zwraca wartość True jeśli liczba jest pierwsza, lub False jeśli liczba nie jest pierwsza

**Zadanie 5**
Napisz skrypt, który losuje liczbę z przedziału od 0 do 2π i wyświetla wartość funkcji sin(x), cos(x) i tg(x) dla tej liczby w tej kolejności ale wyniki pojawiają się co 3 sekundy

**Zadanie 9**
Napisz funkcję, która generuje losowe hasło o długości k używając symboli z klawiatury.

**Zadanie 10**
Napisz funkcję, która generuje n losowych haseł, każde o długości k. Użyj w niej funkcji z poprzedniego zadania.

# Zadania do samodzielnego rozwiązania

**Zadanie 1**
Napisz funkcję, która jako argument przyjmuje listę z liczbami całkowitymi. Za pomocą algorytmu sortowania bąbelkowego funkcja ma zwracać posortowaną listę.

**Zadanie 2**
Napisz funkcję, która zostanie opatrzona w dekorator (*wrapper*) zapamiętujący jej wyniki (*cache*). Użyj funkcji kilka razy dla różnych argumentów i wyświetl zapisane w *cache*'u informacje.

**Zadanie 3**
Napisz dekorator, który sprawi, że podczas uruchamiania funkcji pokazywany jest komunikat mówiący o tym, jaka funkcja została uruchomiona (pokazywana jest nazwa funkcji) i o której to było godzinie (podpowiedź znajdziesz w [8] linku na końcu prezentacji)

```
procedure bubbleSort( A : lista elementów do posortowania )
  n = liczba_elementów(A)
   do
    for (i = 0; i < n-1; i++) do:
      if A[i] > A[i+1] then
        swap(A[i], A[i+1])
      end if
    end for
    n = n-1
  while n > 1
end procedure
```

*Pseudokod sortowania bąbelkowego, źródło: [7]*

# Zadania do samodzielnego rozwiązania

**Zadanie 4**

Z biblioteki math zaimportuj **tylko trzy** wybrane funkcje trygonometryczne. Następnie napisz funkcję, która zwróci jak wynik inną funkcję wykonującą dowolne działanie z wybraną, jedną z trzech funkcji trygonometrycznych. Na przykład, jeśli ktoś wybrał jako funkcję sinus, to Twoja funkcja powinna móc przyjąć argument z nazwą wybranej przez użytkownika funkcji i za pomocą instrukcji warunkowej zwrócić jakąś inną, wymyśloną przez Ciebie funkcję, która używa w sobie funkcji sinus.

**Zadanie 5**

Napisz funkcję, która oblicza pole trapezu i oferuje możliwość przeskalowania obliczonego pola o dowolną liczbę, przekazywaną w argumencie *keyword* o nazwie „skaluj”.

# Przydatne linki


[1] Installing with conda — conda 4.9.2.post24+e37cf84a documentation
[2] Python Functions (w3schools.com)
[3] Python args and kwargs: Demystified – Real Python
[4] Function Wrappers in Python – GeeksforGeeks
[5] Function Wrappers in Python. Putting a Wrapper Around a Function | by Sadrach Pierre, Ph.D. | Towards Data Science
[6] Function wrapper and python decorator - Blog - Amaral Lab (northwestern.edu)
[7] Sortowanie bąbelkowe – Wikipedia, wolna encyklopedia
[8] Python dekoratory - Analityk.edu.pl – podpowiedź do zadania 3

SMARTFACTOR

+48 798 622 487

ul. Poselska 29
03-931 Warszawa

mail@smartfactor.pl

www.smartfactor.pl