## Zaszyfruj się z Pythonem!

inż. Laura Dymarczyk inż. Adrian Rupala



- Język wysokiego poziomu, ogólnego zastosowania
- Prosta i czytelna składnia
- Darmowe biblioteki, menedżer pakietów, Open Source



- Nie wymusza jednego stylu programowania
- Pozwala programować obiektowo, strukturalnie oraz funkcyjnie
- Język interpretowany

## Napiszmy trochę kodu!



Witaj świecie! A może coś innego?

>>> print("Hello world!")



 Przypisanie do określonej wartości czegoś innego

>>> dzban = "słowo roku 2019"

>>> print(dzban)



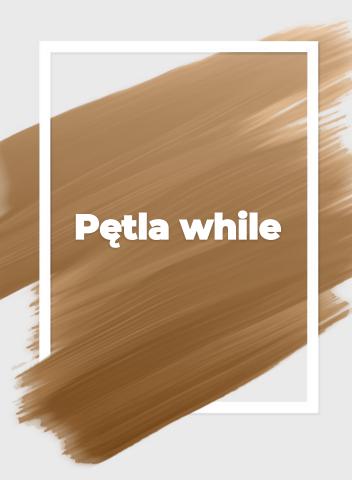
 Stwórzmy coś bardziej zaawansowanego.

>>> print(nazwa\_funkcji(parms))

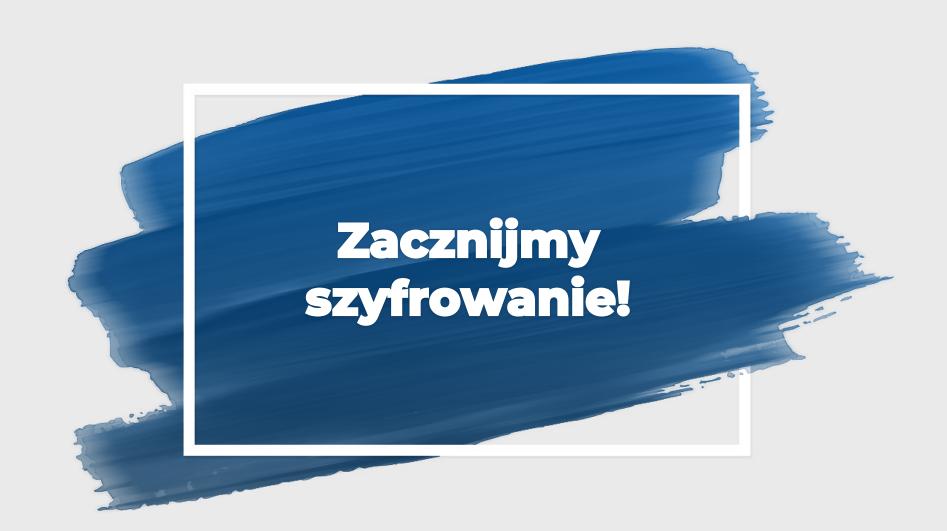


- >>> for <zmienna> in <wartość>: <twój fajny kod który coś robi>

>>> ... in range(start, stop, krok):



```
 - >>> i = 1
 - >>> while i = <wartość>:
 <dalej fajny kod>
 i += 1
```





- Przykład szyfru podstawieniowego
- Nazwa pochodzi od Juliusza
  Cezara, który używał tej techniki do komunikowania się z przyjaciółmi
- Każda litera tekstu zaszyfrowana jest inną, oddaloną od niej o stałą liczbę pozycji w alfabecie



Parametr przesunięcia 3:

Alfabet:

AĄBCĆDEĘFGHIJKLŁMNŃOÓPRSŚTUWYZŹŻ

Szyfr:

CĆDEĘFGHIJKLŁMNŃOÓPRSŚTUWYZŹŻAĄB

Tekst jawny:

MĘŻNY BĄDŹ, CHROŃ PUŁK TWÓJ I SZEŚĆ FLAG Tekst zaszyfrowany:

OHBÓŻ DĆFĄ, EKTRP ŚZŃM YŹSŁ L UAGWĘ INCJ



- Musimy zdefiniować nasz alfabet
- Musimy określić ilość liter w alfabecie
- Odczytać i zapisać plik tekstowy z tekstem
- Napisać funkcję odpowiedzialną za szyfrowanie tekstu (odpowiednie przesuwanie liter)



- >>> plik = open(nazwa\_pliku, 'parametr')
  - >>> plik.read()
  - >>> plik.write()
  - >>> plik.close()
- Parametr może przyjmować różne wartości, np:
  - r dla odczytu
  - w dla zapisu
  - b- w postaci binarnej



- ->> len(dane) zwraca ilość (długość) obiektów w strukturze
- >>> wartosc.find(litera) zwraca indeks pierwszej szukanej wartości w strukturze











koło naukowe informatyków