

# Python

## Pierwszy wykład

dr inż. Marcin Ciura, Uniwersytet Komisji Edukacji Narodowej

Nazywam się  
Marcin Ciura

Jestem programistą od  
30 lat

UKEN to moja czwarta  
uczelnia



# Plan na dziś

Przedstawię wam:

- + cel tego przedmiotu
- + plan wykładów
- + kilka spraw :-)
- + historię Pythona
- + i zen Pythona

A potem omówię:

- + instrukcje
- + i nazwy obiektów

*Cel tego przedmiotu*

# Cel tego przedmiotu

- Cel tego przedmiotu to:
  - + nauczyć was myśleć jak programiści
  - + nauczyć was poprawnych nazw
  - + nauczyć was podstaw języka Python
  - + nauczyć was programować w dobrym stylu

# Plan wykładów

# Plan wykładów

Na drugim wykładzie omówię:

- + Funkcje
- + Funkcję wbudowaną `print`
- + Funkcję wbudowaną `input`
- + Łańcuchy znaków, czyli teksty
- + Słowa kluczowe
- + Indeksowanie łańcuchów

- + Wycinki łańcuchów
- + Dodawanie łańcuchów
- + Metody łańcuchów
- + Instrukcję warunkową `if`
- + Operatory porównania
- + Operatory `in` i `not in`
- + Operatory logiczne
- + I pliki tekstowe

# Plan wykładów

Na trzecim wykładzie omówię:

- + Trzy typy proste:

- tekst (**str**)
- liczby całkowite (**int**)
- liczby rzeczywiste (**float**)

- + f-łańcuchy

- + łańcuchy wieloliniowe

- + instrukcje if...else i if...elif...else

- + to, jak definiować własne funkcje

- + i dwa typy złożone:

- listę (**list**)
- słownik (**dict**)



# Plan wykładów

Na czwartym wykładzie omówię:

- + Kilka sztuczek

- + Krotki

- + Funkcje:

- sum
- sorted
- range
- enumerate

- + Zbiory

- + Importowanie modułów

- + I wybrane funkcje czterech modułów:

- math
- statistics
- random
- urllib

# Plan wykładów

Na piątym wykładzie  
omówię wyrażenia  
regularne:

- + kropkę .
- + zbiory znaków [...]
- + daszek ^
- + dolar \$

+ zachłanną gwiazdkę

Kleene'a \*

+ i niezachłanną gwiazdkę

Kleene'a \*?

# Plan wykładów

Po piątym wykładzie  
rozwiążecie zadanie  
w zespole

# Plan wykładów

Na szóstym wykładzie  
przedstawię wam język R:

- + jego historię
- + powiem wam, dlaczego R jest fajny
- + i omówię elementy języka R:
  - zmienne
  - wektory
  - listy

- i ramki danych

A potem opowiem wam o tym:

- + jak statystycy używają R
- + jak programiści piszą funkcje w R
- + i jak programiści piszą testy  
(właściwie powinienem powiedzieć wam o tym na początku :-)

# Plan wykładów

Na siódmym wykładzie  
omówię kolejne elementy  
języka Python:

- klasy
- instrukcję pass
- instrukcję break
- instrukcję continue

- pętle while
- wyjątki
- funkcje jako argumenty  
funkcji
- i funkcje anonimowe  
lambda

# Plan wykładów

Na ósmym wykładzie omówię  
zewnętrzną bibliotekę

Pythona **pandas**:

- + jak odczytywać dane z serii
- + jak zmieniać dane w serii
- + działania arytmetyczne na seriach
- + jak tworzyć ramki danych

+ jak odczytywać dane  
z ramek danych

+ i jak zmieniać ramki danych

na podstawie wykładu  
dr. hab. Grzegorza  
Góralskiego  
z Uniwersytetu  
Jagiellońskiego

Kilka spraw :-)

# Wyjaśnijmy sobie kilka spraw :-)

- Każdy z was może mówić do mnie tak, jak chce
- Na przykład:
  - + proszę pana
  - + panie doktorze
  - + albo szefie :-)



Wyjaśnijmy sobie kilka spraw :-)

- Każdy student, który chce opuścić wykład, może wyjść z niego w dowolnej chwili
- Nie obrażę się
- Naprawdę się nie obrażę

# Wyjaśnijmy sobie kilka spraw :-)

- Po każdym wykładzie będzie 10 minut przerwy
- Po przerwie będą ćwiczenia
- Na ćwiczeniach każdy student rozwiąże kilka zadań
- Nie przerobimy na zajęciach całego Pythona, tylko jego najpotrzebniejsze części

# Historia Pythona



Wyobraźmy sobie Amsterdam w grudniu 1989 roku





Przedstawiam wam  
Guido van Rossuma  
holenderskiego programistę





Guido

jest fanem serialu  
komedijnego Latający Cyrk  
Monty Pythona





# Guido

miął wolny czas w ferie  
świąteczne, więc zaczął  
pracować nad nowym  
językiem programowania





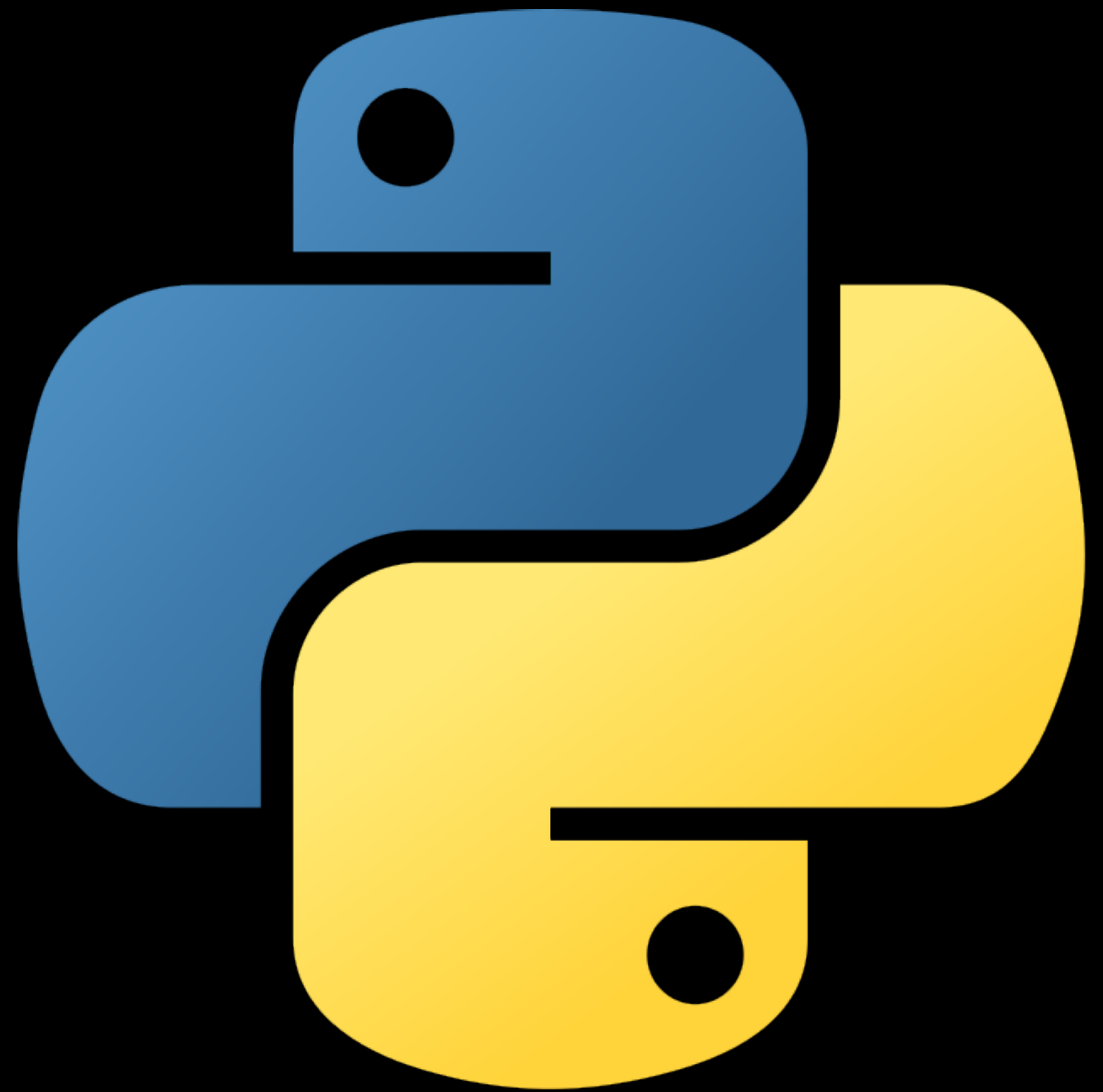
Guido

nazwał ten język Python  
na część Latającego Cyrku  
Monty Pythona



To jest stare logo języka Python





To jest aktualne logo Pythona

Guido jeszcze nie wiedział, że  
język Python zmieni świat,  
a Guido zostanie kiedyś jego  
Dobrotliwym Dożywotnim  
Dyktatorem\* (chodzi  
o Pythona, nie o świat :-)

\* Benevolent Dictator for  
Life

# Python zmienił świat:

- Bo nowi programiści łatwo mogą nauczyć się Pythona
- Bo w Pythonie łatwo przetwarzać teksty
- Bo „baterie są w zestawie”\*, czyli do wielu zadań wystarcza standardowa biblioteka funkcji Pythona  
\* “Batteries included”
- Bo Python działa na wielu rodzajach komputerów od telefonów komórkowych po superkomputery (aplikacje to też programy :-)

# Programiści rozwijają język Python\*

Jeśli trzeba, programiści tworzą coraz nowsze wersje programów, czyli je rozwijają. W nowszych wersjach:

- + poprawiają stare błędy
- + czasem wprowadzają nowe błędy :-)
- + i wprowadzają nowe udogodnienia dla użytkowników

\* programmers develop Python

# Programiści rozwijają Python

- + Wersja 1.0 Pythona ukazała się w 1994 roku
- + Wersja 2.0 ukazała się w 2000 roku
- + Wersja 3.0 ukazała się w 2008 roku
- + Potem ukazywały się wersje 3.0.1, 3.1, 3.1.1, 3.1.2, 3.1.3, 3.1.4, 3.1.5, 3.2, potem wersje od 3.2.1 do 3.2.6, od 3.3.0 do 3.3.7 i tak dalej

W sierpniu 2025 roku ukazała się wersja 3.13.7

Wystarczy nam wersja 3.10.0 lub nowsza

# Ojej, zapomniałem wam powiedzieć...

- Czym się różnią języki programowania od języków naturalnych
- Nigdy bym się nie domyślił, jak to wytłumaczyć, gdybym nie rozmawiał z pewnym językoznawcą, który jest dobrym matematykiem
- Ten językoznawca powiedział tak:
- Języki naturalne mają dużo słów. Te słowa można wypowiadać w różnej kolejności.  
Języki programowania mają mało słów. Te słowa trzeba ustawiać w określonej kolejności.

Zen Pythona

# Zen Pythona\* czyli 20 zasad, z których zapisano tylko 19

\* Wersję angielską tych zasad możecie przeczytać, wydając Pythonowi instrukcję `import this`

1. Piękne jest lepsze od brzydkiego
2. Jawne jest lepsze od niejawnego
3. Proste jest lepsze od złożonego
4. Złożone jest lepsze od skomplikowanego
5. Płaskie jest lepsze od zagnieżdżonego
6. Rzadkie jest lepsze od zagęszczonego



# Zen Pythona

20 zasad, z których zapisano tylko 19

7. Liczy się czytelność

8. Przypadki szczególne nie są aż tak wyjątkowe, żeby łamać reguły

9. Ale praktyczne zwycięża nad nieskazitelnym

10. Błędy nigdy nie powinny mijać bez echa

11. Chyba że są celowo wyciszone

# Zen Pythona

20 zasad, z których zapisano tylko 19

12. Gdy widzisz coś niejednoznacznego, odrzuć pokusę zgadywania

13. Powinien być jeden — i najlepiej tylko jeden — oczywisty sposób, żeby coś zrobić

14. Chociaż na pierwszy rzut oka ten sposób może nie być oczywisty, chyba że jesteś Holendrem

# Zen Pythona

20 zasad, z których zapisano tylko 19

15. Teraz jest lepsze niż nigdy

16. Chociaż nigdy jest często lepsze niż natychmiast

17. Jeśli implementację trudno wyjaśnić, to zły pomysł

18. Jeśli implementację łatwo wyjaśnić, to może być dobry pomysł

19. Przestrzenie nazw to rewelacyjny pomysł — róbmy ich więcej!

Przedstawiłem wam:

- + cel tego przedmiotu
- + plan wykładów
- + kilka spraw
- + historię Pythona
- + i zen Pythona

Teraz omówię:

- + instrukcje
- + i nazwy obiektów

Proszę robić notatki :-)

Instrukcje

# Instrukcje

Komputer wykonuje instrukcje programu

Każda instrukcja programu napisanego w Pythonie znajduje się w innej linii tego programu

Nazwy obiektów



# O czym dobrze jest pamiętać, gdy programujemy

Gdy programujemy, dobrze jest pamiętać o tym, że nazwy obiektów są ważne

Gdy raz nadaliśmy obiektowi nazwę, a potem wpisujemy tę nazwę jeszcze raz, powinniśmy dbać o to, żeby wpisać ją poprawnie

Nazwy obiektów  
w Pythonie są jak  
naklejki przyklejane  
do obiektów





Operator  
przypisania =  
nadaje nazwę  
obiektowi





Instrukcja

`fruit = '🍏'`

przypisuje '🍏' do zmiennej  
`fruit`

Programiści czytają instrukcję

`fruit = '🍏'`

tak:

„Przypisz '🍏' do zmiennej `fruit`”





# Różne obiekty nie mogą mieć tej samej nazwy

Gdy przypisujemy nazwę `fruit` najpierw do obiektu '🍏', a potem do obiektu '🍐':

```
fruit = '🍏'
```

```
fruit = '🍐'
```

jest tak, jakbyśmy przykleili naklejkę `fruit` najpierw na '🍏', a potem na '🍐'





Jeden obiekt może  
mieć wiele nazw

fruit = '🍐'

pear = '🍐'



Przedstawiłem wam:

- + cel tego przedmiotu
- + plan wykładów
- + kilka spraw
- + historię Pythona
- + i zen Pythona

A potem omówiłem:

- + instrukcje
- + i nazwy obiektów

To wszystko na dziś  
Proszę mi zadać pytanie :-)



Czy każdy z was zrozumiał  
wszystko z tego wykładu?

Dziękuję za uwagę  
Zapraszam na następny wykład,  
na którym dowiemy się, jak wygląda plik

# Źródła zdjęć

Anne Helmond: Snow in Amsterdam, CC BY-NC-ND 2.0

Guido van Rossum, Wikimedia Commons, CC BY-SA 2.0

Alastair Campbell: Monty Python banner, CC BY-SA 2.0

Logo języka Python w latach 1996-2006, Wikimedia Commons,  
licencja BSD

Logo języka Python, Wikimedia Commons, licencja GPL