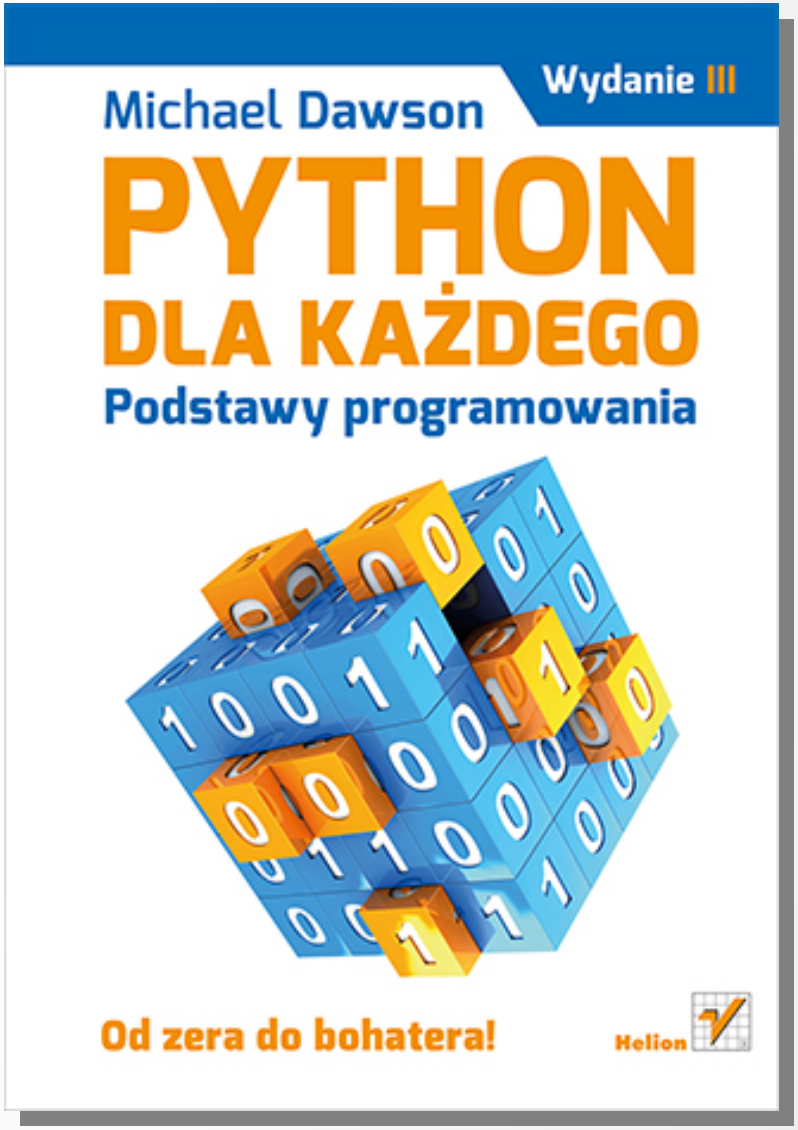
rozdział 2:

PYTHON

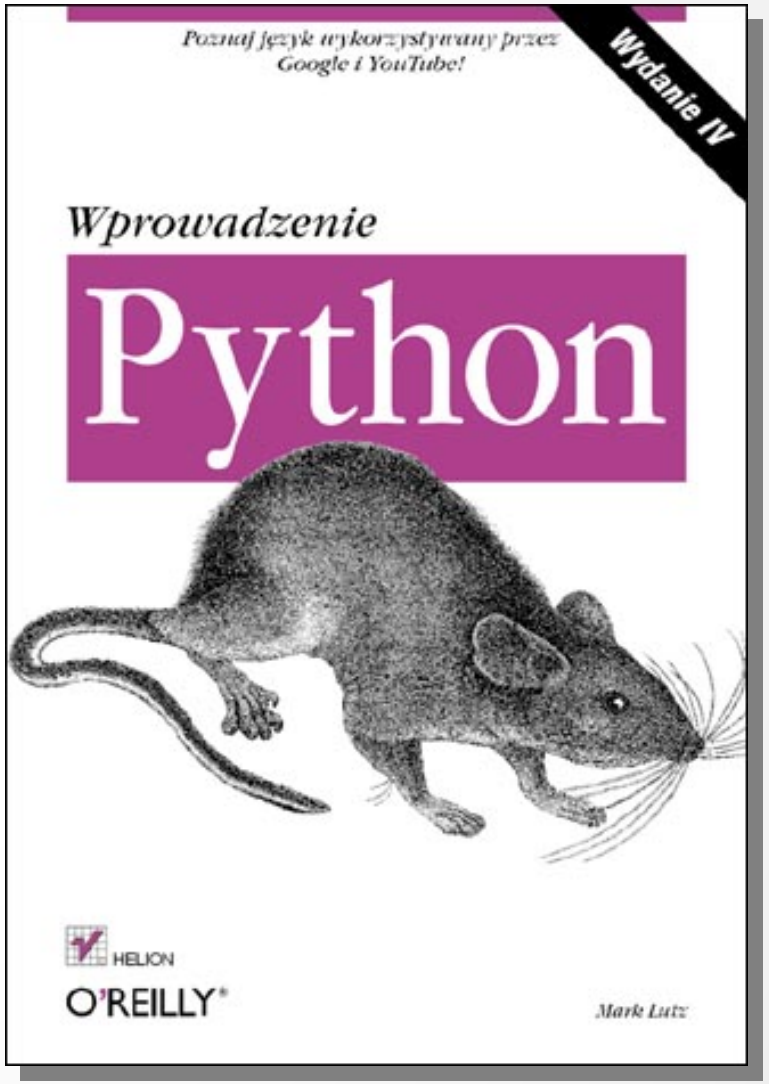
Podstawy programowania

Michael Dawson
„Python dla każdego“
Helion, 2014



Podstawy programowania

Mark Lutz
„Python, wprowadzenie“
Helion, 2011



Podstawy programowania

Python: skąd, jak i kto

- **Python** to nie gad – to **Monty Python** z Latającego Cyrku
- Guido van Rossum (ur. 1960, Holender) – stworzył Pythona w grudniu 1989, jak sam twierdzi, właściwie z nudów
- w roku 2005 Guido został zatrudniowy przez Google, a w 2013 przeniósł się do Dropboksa.

Podstawy programowania

Python: jaki jest?

- łatwy w nauce i użytkowaniu, intuicyjny, o dużej sile ekspresji
- publikowany jako *open source*
- do zastosowań ogólnych, ale użyteczny również w zastosowaniach niszowych
- skraca czas programowania i podnosi komfort pracy programisty

Podstawy programowania

Python: do czego?

- do nauki programowania
- do obliczeń naukowych
- do aplikacji internetowych
- do aplikacji systemowych
- do wszelakich aplikacji
- do pisania cracków
- po prostu do (niemal) wszystkiego

Podstawy programowania

Python: zalety

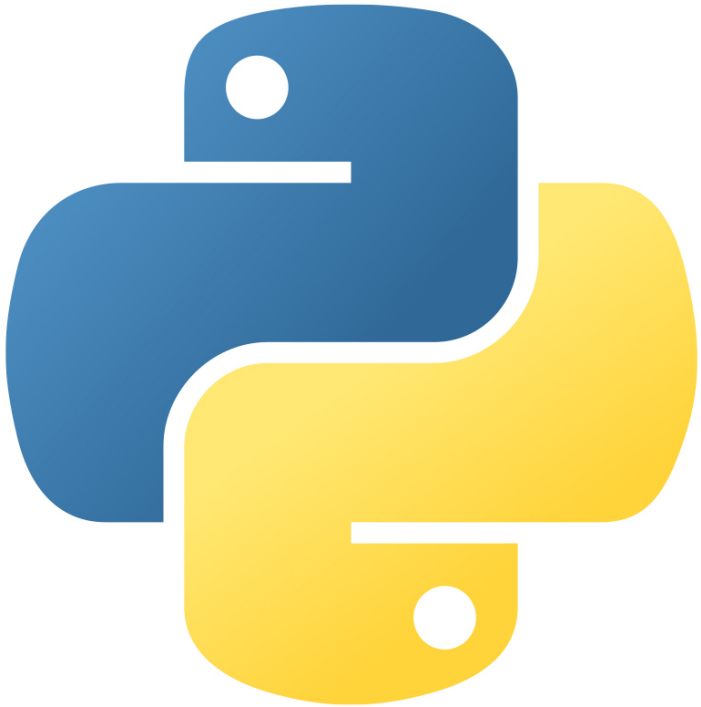
- programy w Pythonie łatwo się czyta (!)
- programy w Pythonie pisze się łatwo i wydajnie
- Pythona łatwo się nauczyć
- Pythona łatwo nauczać
- Python jest wszędzie (no, prawie...)

Python: wady

- programy w Pythonie nie są demonami szybkości
- ze względu na przyjętą filozofię, może wymagać dokładniejszego testowania niż programy w innych językach, a szukanie błędów może być kosztowniejsze

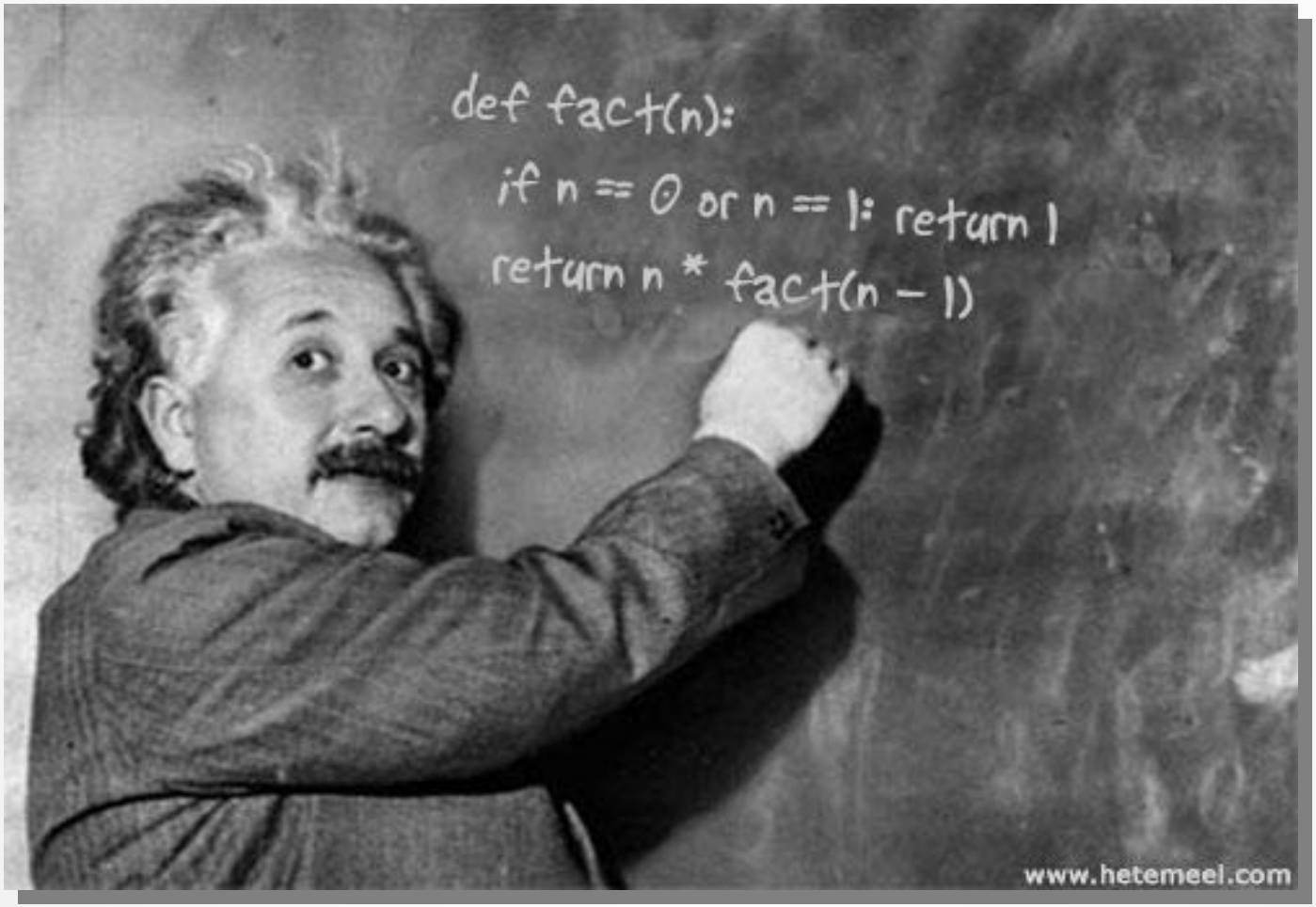
Podstawy programowania

Python: jak wygląda?



Podstawy programowania

Dlaczego Python?



Podstawy programowania

Dlaczego Python?

- w obrocie towarowym:

  netto (waga produktu)
  +
  tara (waga opakowania)
  =
  brutto (to, co się kupuje, sprzedaje i transportuje)

Podstawy programowania

Dlaczego Python?

- w tworzeniu kodu:

  netto (sedno kodu)
  +
  tara (wszelkie niezbędne dodatki)
  =
  brutto (kompletny, użyteczny kod)

Podstawy programowania

Klasyka: język „C“

```
#include <stdio.h>
int main(int argc, char *argv[])
{
    puts("Witaj!");
    return 0;
}
```

Podstawy programowania

Klasyka: język Java

```
class Witaj {
   public static void main(String[] args)
   {
      System.out.println("Witaj!");
   }
}
```

Podstawy programowania

A teraz – Python!

```
print("Witaj!")
```

Podstawy programowania

Python: niejedno ma imię

- w chwili obecnej w użyciu są dwie niezależne i niekompatybilne wersje Pythona:
  - **Python 2** – podtrzymywany przy życiu ze względu na ogromną liczbę programów w nim napisanych (obecna wersja to 2.7)
  - **Python 3** – stosowany do nowych projektów i traktowany jako przyszłościowy (obecna wersja to 3.7)

Podstawy programowania

Python 3

- wszystkie nasze programy i wszelkie rozważania będą dotyczyć programowania w Pythonie 3
- ewentualne wyjątki będziemy czytelnie sygnalizować

Podstawy programowania

Python: skąd go wziąć?

- to proste – stąd:

  `https://www.python.org/downloads/`

Podstawy programowania

Python: co nam daje?

- interpreter
- biblioteki gotowych rozwiązań, których możemy użyć w naszym kodzie
- proste środowisko programisty o nazwie IDLE
- czyli wszystko, czego potrzebujemy, żeby rozpocząć przygodę z Pythonem

Podstawy programowania

Python: od słów do czynów

- pokażmy więc, jak przygotować się do korzystania z Pythona...

Podstawy programowania

Python:

- dwa sposoby pracy:
  - **interakcyjny** – wpisujemy polecenia, a Python je natychmiast wykonuje
  - **nieinterakcyjny** – piszemy kod źródłowy w pliku, a następnie nakazujemy Pythonowi taki plik wykonać

Podstawy programowania

Pierwszy program:

- nie oczekujemy wiele – niech nas przywita i powie „dzień dobry“ :)
- zrobimy to na dwa sposoby:
  - najpierw **interakcyjnie** wpisując polecenie w konsoli Pythona
  - a potem **nieinterakcyjnie**, pisząc prawdziwy (choć bardzo krótki) program

Podstawy programowania

Jak nakłonić Pythona, żeby coś do nas powiedział?

- to proste – służy do tego funkcja o nazwie:

```
print
```

Podstawy programowania

Dygresja – co to jest *funkcja*?

- cofnijmy się w czasie do strasznej szkolnej matematyki i przypomnijmy sobie np. funkcję *sinus (brrr….)*

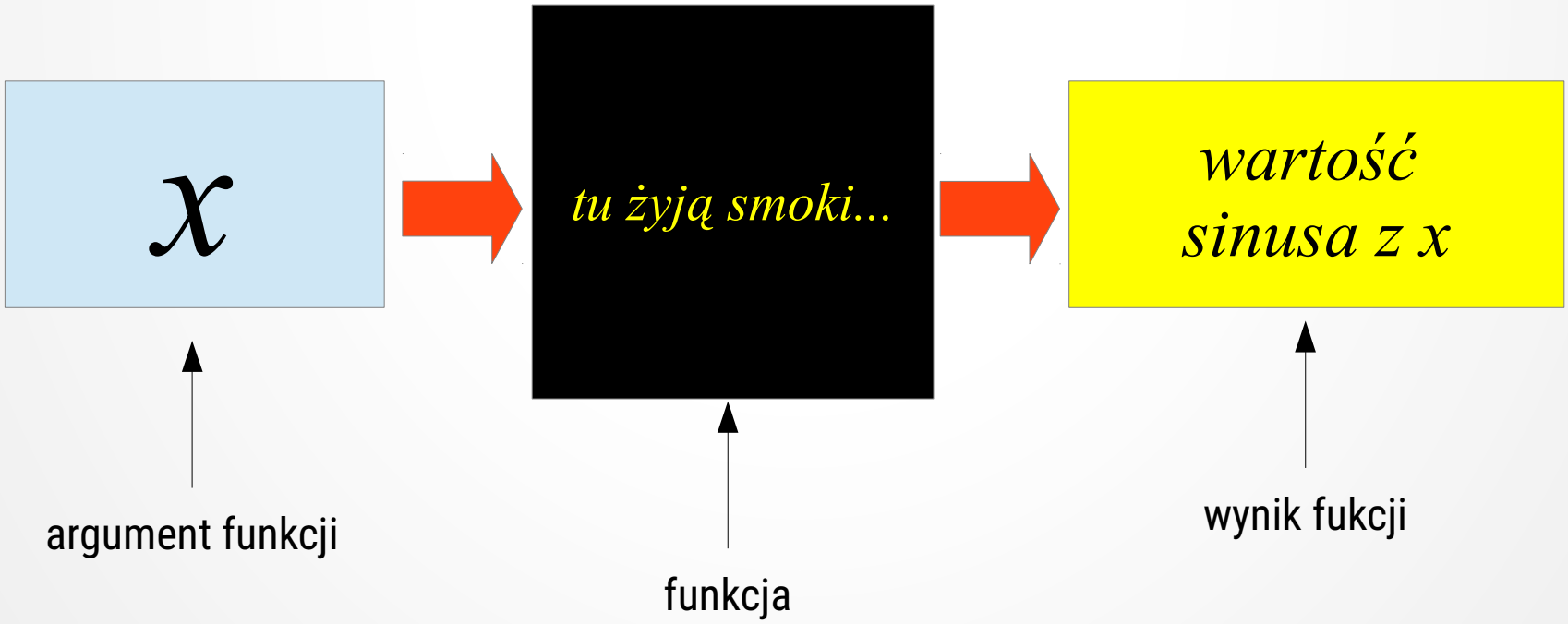
$$sin(x)$$

Podstawy programowania

$sin(x)$

Podstawy programowania

$sin(x)$



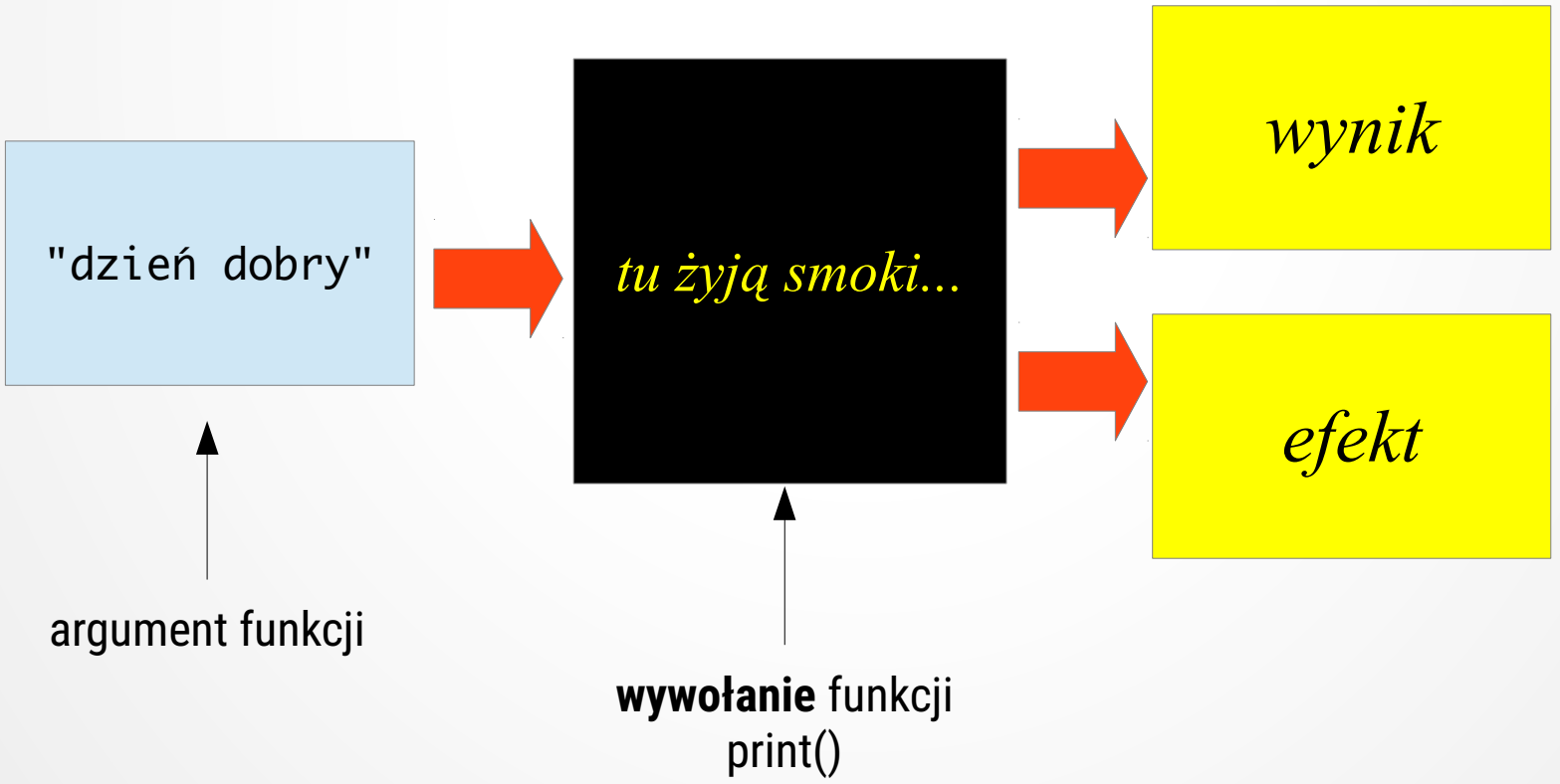
Podstawy programowania

- dokładnie tak samo jest z funkcją **print**, z tą jednak różnicą, że funkcje w Pythonie oprócz wyniku mogą również mieć efekt, tzn. mogą wykonać jakąś pożyteczną czynność, np.:

```
print("dzień dobry!")
```

Podstawy programowania

```
print("dzień dobry")
```



Podstawy programowania

dygresja o cudzysłowach:

- ujęcie pewnego tekstu w cudzysłowy powoduje, że Python będzie go brał **dosłownie** i nie będzie próbował dociec, czy przypadkiem nie jest on fragmentem kodu, np. nazwą jakiejś funkcji
- używamy cudzysłowów anglosaskich ("), a nie pisarskich („“)

Podstawy programowania

No to do dzieła!

Podstawy programowania

definicja (dość intuicyjna):

- kompletna czynność, wykonywana przez Pythona to **instrukcja**
- wywołanie funkcji jest **instrukcją**
- w jednej instrukcji można wywołać kilka funkcji (będziemy to robić)

Podstawy programowania

ważne:

- w Pythonie piszemy **jedną** instrukcję w jednym **wierszu** pliku
- możemy wstawiać do kodu puste linie, jeśli poprawi to czytelność kodu
- jeśli zdarzy się tak, że instrukcja naszego kodu nie zmieści się w jednej linii (mało prawdopodobne, ale niewykluczone), to można ją **złamać**, umieszczając na końcu linii znak **** (*backslash*)

Podstawy programowania

kolejne trzy postaci kodu są równoważne:

- **#1**

```
print("Dzien dobry")
print("Nazywam się Python")
```

Podstawy programowania

- #2

```
print("Dzien dobry")

print("Nazywam się Python")
```

Podstawy programowania

- #3

```
print("Dzien dobry")

print \
(\
"Nazywam się Python"\
)
```

Podstawy programowania

ważne:

- **nie wolno** ci wstawiać ***białych znaków*** na początku linii, jeśli nie wiesz, po co to robisz!
- linia wcięta względem lewego marginesu jest dla Pythona czymś innym, niż linia zaczynająca się od pierwszej kolumny
- wytłumaczenie tego fenomenu nastąpi trochę później – prosimy o cierpliwość :)

Podstawy programowania

ważne:

- Python rozróżnia wielkość liter: *X* jest dla niego czymś zupełnie innym niż *x*
- w konsekwencji nazwy funkcji musisz pisać dokładnie tak, jak to zapisano w dokumentacji, a więc:
- to jest OK → `print`
- to nie jest OK → ~~`Print PRINT PrInT`~~

Podstawy programowania

komentarz – ważna sprawa:

- **komentarz** to część kodu źródłowego, która jest **ignorowana** przez Pythona
- komentarze wstawia się do kodu, żeby… **skomentować** to, co się w nim dzieje
- czasami komentarzy używa się po to, żeby tymczasowo **zablokować** wykonanie pewnych części programu

Podstawy programowania

komentarz:

- **komentarz** zaczyna się tam, gdzie w linii stoi znak **#** (hash), a kończy się tam, gdzie kończy się ta linia
- ale **hash** umieszczony wewnątrz cudzysłowów nie rozpoczyna komentarza – dlaczego?

Podstawy programowania

- **wytęż wzrok i znajdź wszystkie komentarze:**

```
# to jest nasz pierwszy program
print("Dzien dobry") # powitanie

# to jest komentarz
print("# a to nie jest komentarz")
```

Podstawy programowania

definicja:

- **literał** to dana, która oznacza **sama siebie**
- wbrew narzucającemu się wrażeniu, to wcale nie jest masło maślane

Podstawy programowania

przykład:

- to **jest** literał → 3,1415926535
- to **nie jest** literał → π

Podstawy programowania

literały całkowite:

- zapisywane jako ciąg cyfr arabskich, bez jakichkolwiek wtrąceń
- to jest OK → `3000000000`
- to nie jest OK → ~~`3_000_000_000`~~

Podstawy programowania

literały całkowite:

- taką liczbę wolno nam poprzedzić znakiem:
  - – (wtedy zmieni znak)

  oraz znakiem:
  + (wtedy nie zmieni znaku)

Podstawy programowania

literały całkowite:

- poniższe literały, mimo naszych obaw, są poprawnymi liczbami:

```
-----123
+-+-+-+123
```

Podstawy programowania

literały całkowite:

- jeżeli literał całkowity zaczyna się od **przedrostka `0o`** (zero o), oznacza to, że został zapisany w systemie **ósemkowym**, np:
  `0o20`
  ma wartość...

Podstawy programowania

16

Podstawy programowania

literały całkowite:

- jeżeli literał całkowity zaczyna się od **przedrostka `0x`** (zero iks), oznacza to, że został zapisany w systemie **szesnastkowym**, np:
  `0x20`
  ma wartość...

32

Podstawy programowania

literały całkowite:

- jeżeli literał całkowity zaczyna się od **przedrostka `0b`** (zero be), oznacza to, że został zapisany w systemie **dwójkowym**, np:
  `0b20`
  ma wartość...

Nie ma wartości, bo to błąd!

Podstawy programowania

literały całkowite:

- to już jest ok:
  `0b110`
  ma wartość...

Podstawy programowania

6

Podstawy programowania

uwaga:

- literki o, x i b mogą być wielkie, czyli to też są poprawne literały:
  - `0XFF`
  - `0B0110`
  - `0O777`
- chociaż ostatni z nich wygląda dość ryzykownie (zauważ, jak wiele zależy tu od zastosowanej czcionki – dlatego istnieją specjalne czcionki dla programistów – w tej prezentacji używamy czcionki Monaco)

Podstawy programowania

literały rzeczywiste (zmiennopozycyjne):

- zapisywane prawie jak w normalnie (przynajmniej w Polsce), ale…
- zamiast przecinka dziesiętnego stanowczo używamy **kropki**!

Podstawy programowania

literały rzeczywiste (zmiennopozycyjne):

- a więc liczbę „dwa i czterdzieści dziewięć setnych“ należy zapisać jako:

```
2.49
```

Podstawy programowania

literały rzeczywiste (zmiennopozycyjne):

- jeżeli **przed** kropką byłoby tylko zero, to można je pominąć – oba poniższe literały oznaczają tę samą liczbę:

```
0.49
 .49
```

Podstawy programowania

literały rzeczywiste (zmiennopozycyjne):

- jeżeli **po** kropce byłoby tylko zero, to można je pominąć – oba poniższe literały oznaczają tę samą liczbę:

```
49.0
49.
```

Podstawy programowania

literały rzeczywiste (zmiennopozycyjne):

- pominięcie kropki – chociaż wydaje się bez znaczenia – zmienia charakter literału i dlatego wymaga chwili refleksji

49 → to jest literał całkowity

49. → to jest literał rzeczywisty

Podstawy programowania

literały rzeczywiste (zmiennopozycyjne):

- różnica w rodzaju literału skutkuje innym sposobem **reprezentacji** liczby w pamięci, a co za tym idzie, zupełnie inną **arytmetyką** stosowaną przez komputer
- dalsze szczegóły niebawem

Podstawy programowania

literały rzeczywiste (zmiennopozycyjne):

- bardzo duże i bardzo małe (co do modułu) liczby rzeczywiste można zapisywać w tzw. notacji naukowej, np.

`2.89E10` → $2{,}89 \cdot 10^{10}$

`0.342E-20` → $0{,}342 \cdot 10^{-20}$

Podstawy programowania

literały rzeczywiste (zmiennopozycyjne):

- w notacji naukowej:
  - można używać e albo E
  - liczba stojąca za e musi być całkowita
  - to nie jest ok → ~~1.24E2.5~~
  - liczba stojąca za e może mieć znak
  - to jest ok → 1.24E-2

Podstawy programowania

literały zespolone:

- służą do zapisu liczb zespolonych, a więc takich, które składają się z części rzeczywistej i urojonej
- nie będziemy się nimi zajmować
- Uff….

Podstawy programowania

literały logiczne:

- służą do zapisu dwóch elementarnych wartości logicznych, czyli prawdy i fałszu
- prawda → `True`
- fałsz → `False`
- literały te należy zapisywać dosłownie, tzn np. nie tak: ~~`TRUE`~~ ~~`false`~~