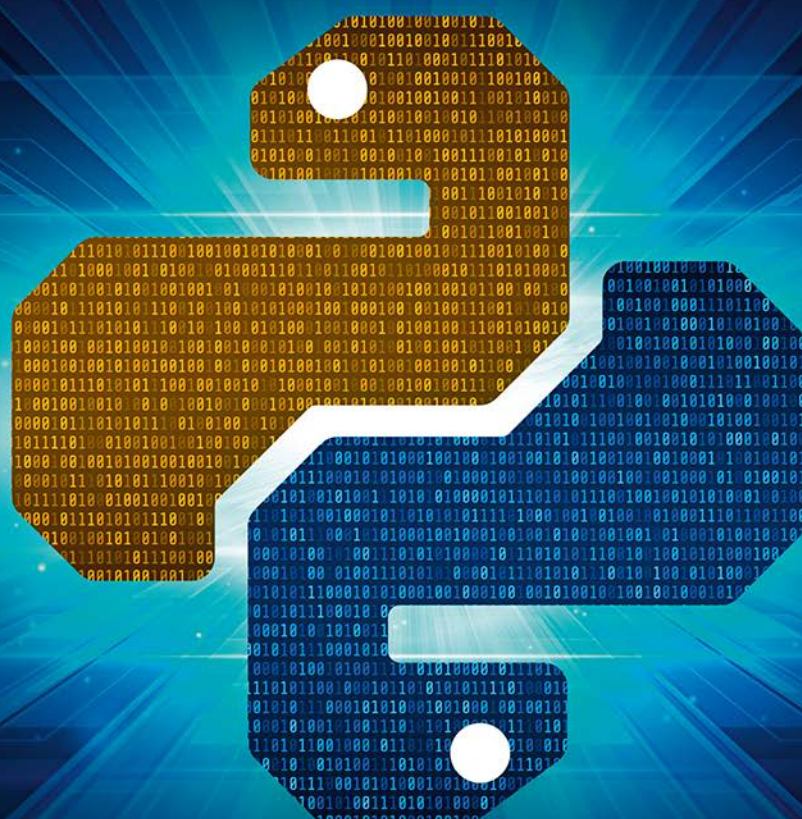


Gniewomir Sarbicki



Python

Kurs dla **nauczycieli** i **studentów**

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Wydawnictwo HELION nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Opieka redakcyjna: Ewelina Burska

Projekt okładki: Studio Gravite/Olsztyn

Obarek, Pokoński, Pazdrijowski, Zaprucki

Materiały graficzne na okładce zostały wykorzystane za zgodą Shutterstock.

Wydawnictwo HELION

ul. Kościuszki 1c, 44-100 GLIWICE

tel. 32 231 22 19, 32 230 98 63

e-mail: helion@helion.pl

WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<http://helion.pl/user/opinie/pytkna>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-283-4566-9

Copyright © Helion 2019

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

Spis treści

Wstęp	9
1 Wprowadzenie	12
1.1 Interaktywna powłoka, interpreter skryptów, edytory i środowiska	12
1.2 Typy liczbowe	14
1.3 Typy sekwencyjne	16
1.3.1 Łańcuchy znaków i łańcuchy bajtów	18
1.3.2 Listy	20
1.3.3 Krotki	22
1.4 Instrukcje warunkowe	22
1.4.1 Trójargumentowa instrukcja logiczna	23
1.4.2 Kwantyfikatory	24
1.5 Pętle for i while	24
1.6 Listy składane	25
1.7 Słowniki	27
1.8 Funkcje	28
1.8.1 Zmienna liczba argumentów	31
1.8.2 Dokumentacja funkcji	32

1.8.3	Zmienne globalne w funkcjach	33
1.8.4	Funkcje anonimowe	34
1.9	Programowanie funkcyjne	34
1.10	Formatowanie łańcuchów	37
1.10.1	Formatowanie z użyciem operatora %	37
1.10.2	Formatowanie z użyciem metody format	38
1.11	Importowanie modułów	40
1.12	Funkcje matematyczne i liczby pseudolosowe	42
1.13	Pobieranie argumentów ze standardowego wejścia	43
1.14	Pobieranie argumentów z linii poleceń. Tworzenie aplikacji konsolowych.	44
1.15	Obsługa wyjątków	44
1.16	Praca z plikami	45
1.17	Porównywanie wydajności rozwiązań	48
1.18	Data i czas	49
1.19	Serializacja*	50
1.20	Współpraca z systemem operacyjnym	52
1.21	Dostęp do zasobów WWW	53
2	Programowanie obiektowe	56
2.1	Klasy i instancje, atrybuty i metody	56
2.2	Konstruktor klasy	59
2.3	Dziedziczenie i przysłanianie	59
2.4	Przeciążanie operatorów	60
2.5	Wywoływanie wyjątków	67

3	Graficzny interfejs użytkownika	70
3.1	Pierwszy program w GTK	70
3.2	Umieszczanie w oknie jego obiektów składowych	72
3.3	Obsługa zdarzeń	75
3.4	Metody elementów okna	79
4	Wielowątkowość	83
4.1	Pierwszy program wielowątkowy	84
4.2	Blokady	85
4.3	Porównanie wydajności	87
4.4	Kolejki	90
5	Komunikacja sieciowa	93
5.1	Pierwszy program	94
5.2	Serwer wielowątkowy	97
5.3	Serwer dyskusyjny	98
5.4	Klient usługi TCP*	101
5.5	Serwer i klient UDP*	102
6	Obsługa baz danych	103
6.1	SQLite	103
6.1.1	Dostęp do bazy z linii poleceń	103
6.1.2	Polecenia SQL w SQLite	104
6.1.3	Moduł sqlite3	104
6.2	MySQL*	108
6.2.1	Dostęp do serwera z linii poleceń i tworzenie kont użytkowników	108
6.2.2	Polecenia SQL w MySQL	109

6.2.3	Moduł <code>mysql.connector</code>	109
6.3	Ćwiczenia	110
7	Współpraca z serwerem Apache	113
7.1	Skrypty CGI	114
7.2	<code>mod_python.publisher</code>	117
7.3	Aplikacje WWW korzystające z bazy danych	119
7.4	Prosty mechanizm uwierzytelniania	133
8	Obliczenia numeryczne	136
8.1	Tablice jednowymiarowe	136
8.2	Wykresy funkcji jednej zmiennej	139
8.3	Tablice wielowymiarowe	145
8.4	Wykresy trójwymiarowe	147
8.5	Pola wektorowe	154
8.6	Wykresy animowane	155
8.7	Równania różniczkowe zwyczajne	157
8.8	Równania różniczkowe cząstkowe	164
9	Iteratory, generatory, koprocedury	172
9.1	Funkcje generatorów	175
9.2	Wyrażenia generatorów i odwzorowywanie generatorów	176
9.3	Działania na iteratorach	177
9.4	Koprocedury	182
9.5	Obsługa wyjątków w generatorze	184
9.6	Algorytm roju cząstek realizowany przez mikrowątki	186
9.7	Nieblokujące operacje wejścia-wyjścia	191

9.8	Wielozadaniowość kooperatywna*	193
10	Funkcje wyższych rzędów	201
10.1	Dekoratory funkcji	202
10.2	Atrybuty funkcji	207
10.3	Dekoratory jako klasy	209
10.4	Dekoratory klas	210
10.5	Menedżery kontekstu	214
11	Zarządzanie atrybutami w klasach	218
11.1	Niskopoziomowe zarządzanie atrybutami	219
11.2	Właściwości	224
11.3	Deskryptory	227
11.4	Metody statyczne i metody klas	229
12	Współbieżność wykorzystująca podprocesy	232
12.1	Operacje na tablicach NumPy	235
12.2	Pula podprocesów	238
12.3	Obiekt podprocesu	239
12.4	Komunikacja międzyprocesowa	241
12.5	Synchronizacja podprocesów	246
13	Rozwiązania	260
	Rozdział 1.	260
	Rozdział 2.	272
	Rozdział 3.	288
	Rozdział 4.	302
	Rozdział 5.	302

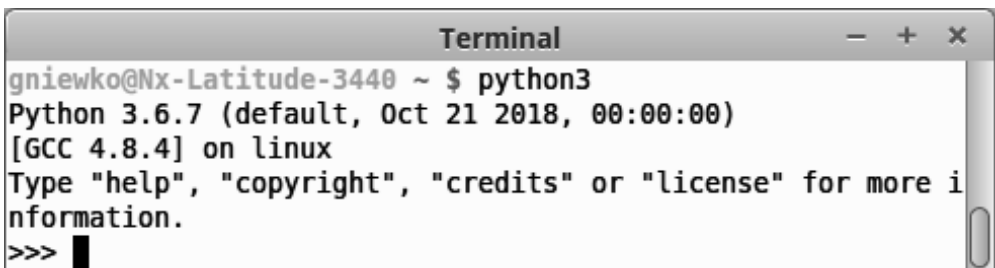
Rozdział 6.	310
Rozdział 7.	316
Rozdział 8.	338
Rozdział 9.	356
Rozdział 10.	387
Rozdział 11.	407
Rozdział 12.	412
Skorowidz	425

Rozdział 1.

Wprowadzenie

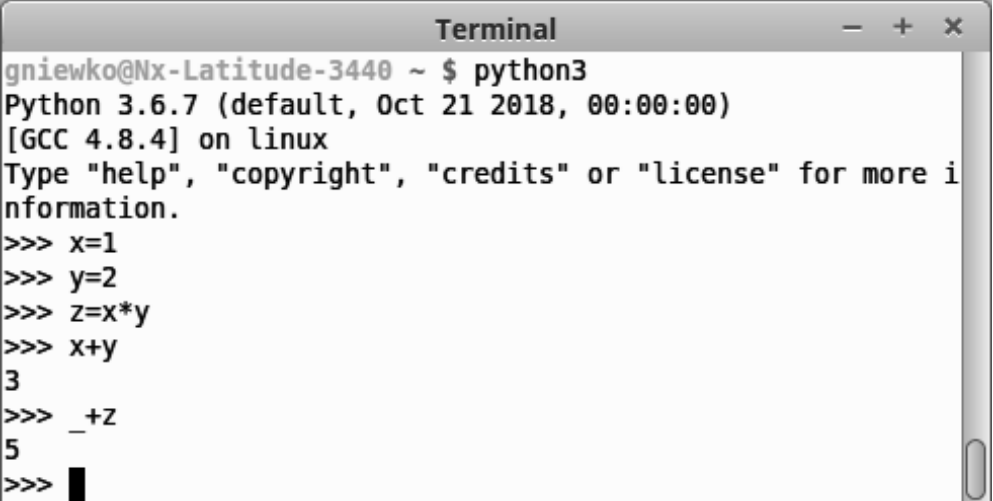
1.1 Interaktywna powłoka, interpreter skryptów, edytory i środowiska

Wywołując polecenie `python3`, przechodzimy do interaktywnej powłoki Pythona.



```
Terminal
gniewko@Nx-Latitude-3440 ~ $ python3
Python 3.6.7 (default, Oct 21 2018, 00:00:00)
[GCC 4.8.4] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> █
```

W sesji powłoki pamiętane są zdefiniowane zmienne, możemy wykonywać na nich operacje. Jeżeli wyniku operacji nie przypiszemy, zostanie on wypisany. Żeby uzyskać dostęp do ostatniego wypisanego wyniku, używamy zmiennej `_`:

A screenshot of a terminal window titled "Terminal". The prompt is "gniewko@Nx-Latitude-3440 ~ \$". The user has entered "python3". The terminal output shows "Python 3.6.7 (default, Oct 21 2018, 00:00:00)", "[GCC 4.8.4] on linux", and instructions to type "help", "copyright", "credits", or "license" for more information. The user then enters a series of Python commands: ">>> x=1", ">>> y=2", ">>> z=x*y", ">>> x+y", which outputs "3". Then ">>> _+z", which outputs "5". The prompt ">>>" is followed by a cursor.

```
Terminal
gniewko@Nx-Latitude-3440 ~ $ python3
Python 3.6.7 (default, Oct 21 2018, 00:00:00)
[GCC 4.8.4] on linux
Type "help", "copyright", "credits" or "license" for more i
nformation.
>>> x=1
>>> y=2
>>> z=x*y
>>> x+y
3
>>> _+z
5
>>> █
```

Powłoki interaktywne oferujące więcej funkcjonalności to m.in. `ipython`, `bpython`.

Używanie powłoki jest wygodne, gdy wykonujemy po sobie jednolinijkowe instrukcje. W przypadku bardziej złożonych programów umieszczamy je w skrypcie — pliku tekstowym, który następnie przekazujemy jako polecenie dla interpretera. Zapiszmy do pliku `pierwszy.py` następujący ciąg instrukcji:

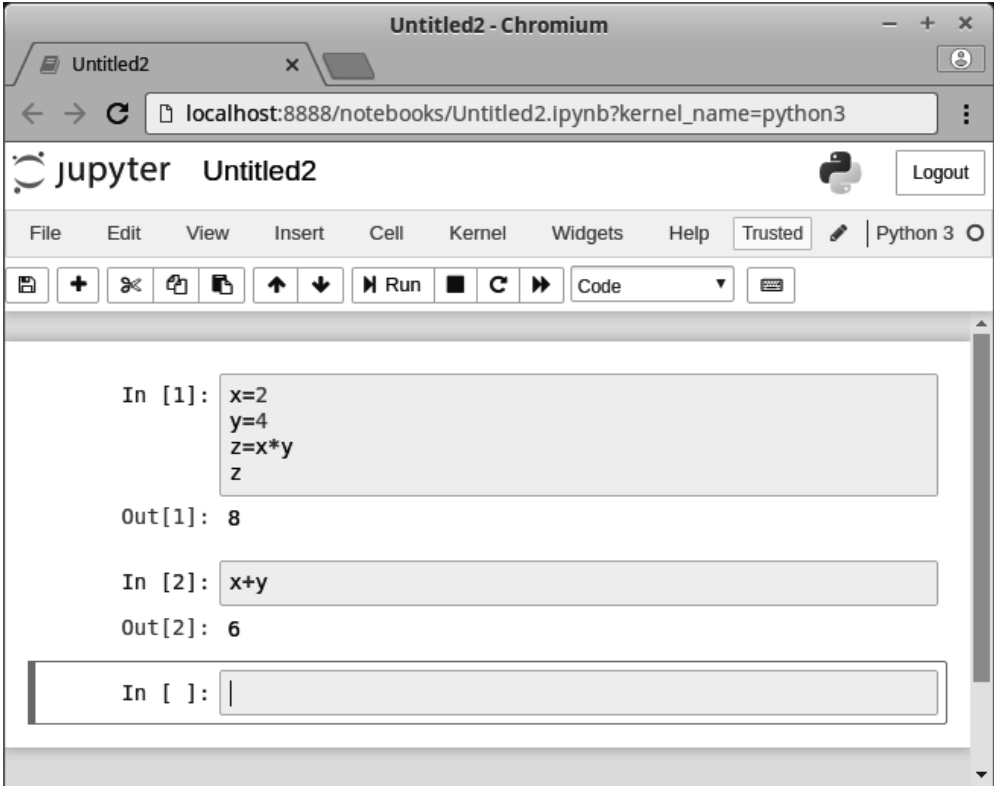
```
x=2
y=4
z=x*y
print(z)
```

i wywołajmy go poleceniem `python3 pierwszy.py`. Inaczej niż podczas pracy w trybie powłoki teraz, żeby wyprowadzić wynik operacji na ekran, trzeba użyć funkcji wbudowanej `print`¹.

Do tworzenia skryptów Pythona można użyć zwykłego edytora tekstu lub narzędzi bardziej zaawansowanych — środowisk programistycznych, które pozwalają edytować kod, uruchamiać go, korzystać z podpowiedzi, wstępnie analizują kod pod kątem błędów i pozwalają zarządzać projektami złożonymi z wielu plików. Najprostszym takim środowiskiem, dającym już wiele funkcjonalności, jest `geany`, a najbardziej zaawansowanym — `pycharm`. Środowisko `geany` jest wystarczająco funkcjonalne dla programów z tej książki.

¹ W Pythonie 2 poprawną składnią będzie `print z`.

Tryb pracy, który łączy zalety pisania wielolinijkowych skryptów i interaktywnej powłoki, oferuje środowisko `jupyter notebook`. Opiera się ono na koncepcji wielolinijkowych komórek — wpisujemy do komórki wielolinijkowy kod, wywołujemy całą komórkę i przechodzimy do następnej.



1.2 Typy liczbowe

Można zauważyć, że w powyższych przykładach nie definiowaliśmy typu zmiennej, jak należałoby zrobić np. w C. W odróżnieniu od C, `x` i `y` nie są obszarami w pamięci, ale nazwami, które wskazują na obiekty w pamięci. O typie obiektu informuje polecenie `type`. Wykonajmy w powłoce następujące instrukcje:

```
>>> x=2
>>> type(x)
<class 'int'>
>>> x=2.0
>>> type(x)
<class 'float'>
```

Stworzony został nowy obiekt 2.0 typu float. Zmienna `x` przestała wskazywać na obiekt 2, a zaczęła na obiekt 2.0 i w ten sposób zmieniła swój typ. Obiektowi 2 zmniejszyła się liczba dowiązań o jeden i ponieważ żadna zmienna już na niego nie wskazywała, został on usunięty z pamięci. Właściwość języka polegająca na tym, że zmienna może wskazywać na obiekty różnego typu, nazywamy *słabym typowaniem*.

Na obiektach typów liczbowych możemy wykonywać następujące operacje liczbowe: `+`, `-`, `*`, `/`, `//` (dzielenie całkowitoliczbowe), `%` (reszta z dzielenia), `**` (potęgowanie)². W Pythonie 3 dzielenie liczb całkowitych prowadzi w ogólności do liczby wymiernej: $1/2 = 0.5$. Dzielenie całkowitoliczbowe realizujemy przez `//`³. Wszystkie te działania mają odpowiadające im operatory przypisania:

```
a += b  ⇔  a = a + b
a -= b  ⇔  a = a - b
a *= b  ⇔  a = a * b
a /= b  ⇔  a = a / b
a //= b ⇔  a = a // b
a %= b  ⇔  a = a % b
a **= b ⇔  a = a ** b
```

Zmienne typu `int` są liczbami całkowitymi o nieograniczonym zakresie (ich rozmiar w bajtach ogranicza tylko ilość dostępnej pamięci operacyjnej)⁴.

Zmienne typu float odpowiadają typowi `double` w C i są zapisywane na 64 bitach (1 bit znaku, 11 bitów wykładnika, 52 bity mantysy).

Ostatnim typem liczbowym jest typ zespolony `complex`, przechowujący część rzeczywistą i urojoną jako zmienne typu float:

```
>>> type(1.0+0.2j)
<class 'complex'>
```

² Potęgowanie realizuje również funkcja wbudowana `pow`, która oprócz argumentów *podstawa*, *wykładnik* może przyjmować trzeci argument, *m*, informujący o tym, że potęgowanie wykonujemy modulo *m*. Gdy potęgujemy modulo, `pow(x,y,z)` jest bardziej wydajne niż `x**y % z`.

³ W Pythonie 2 dzielenie liczb całkowitych jest dzieleniem całkowitoliczbowym. Żeby wymusić wynik wymierny, należy wykonać je jako `1./2` (jeden z argumentów musi być liczbą wymierną).

⁴ W Pythonie 2 zmienne typu `int` są zapisywane na 8 bajtach i przyjmują wartości od -2^{63} do $2^{63} - 1$. W przypadku przekroczenia tego zakresu wynik przyjmuje typ `long`, odpowiadając typowi `int` w Pythonie 3. Typ `long` poznajemy po literze `L` na końcu zapisu liczby:

```
>>> 2**64
18446744073709551616L
>>> type(_)
<type 'long'>
```

Zmienna typu `complex` ma atrybuty: `real`, `imag`, oznaczające jej część rzeczywistą i część urojoną. Wywołanie `z.conjugate()` zwraca liczbę sprzężoną do `z`.

Funkcja wbudowana `abs` zwraca wartość bezwzględną argumentu (lub jego moduł, gdy jest zespolony).

Operatory rzutowania jednego typu na drugi nazywają się tak jak typ, na który rzutują: `int`, `float`, `complex`.

Ćwiczenie 1. *Zapisz wzór na rozwiązanie równania kwadratowego o współczynnikach zapisanych w zmiennych `a`, `b`, `c`. Wzór powinien działać dla dowolnych typów danych (całkowitych, wymiernych, zespolonych).*

1.3 Typy sekwencyjne

Jest 7 typów sekwencyjnych w Pythonie 3, z których najważniejsze to:

- łańcuch znaków (`str`)

```
>>> s='żółw'
```

- łańcuch bajtów (`bytes`)

```
>>> b=b'żółw'
```

- lista (`list`)

```
>>> l=[1,2,3]
```

- krotka (`tuple`)

```
>>> k=(1,2,3)
```

Wspólne operacje dla wszystkich typów sekwencyjnych to:

- Indeksowanie: `s[0]`, `b[1]`, `l[-1]`, `k[-2]`. Wartości ujemne oznaczają indeksy liczone od końca (-1 to ostatni indeks).
- Branie wycinków: `s[x:y]` lub `s[x:y:z]`, gdzie `x` oznacza początek wycinka, `y` oznacza koniec wycinka. Pominięcie wartości `x` oznacza od początku, pominięcie wartości `y` oznacza do końca. `z` oznacza, co który element, wartość ujemna oznacza odliczanie od końca.

- Mnożenie przez liczbę całkowitą dodatnią — powtarzanie:
`2*[1,2]=[1,2,1,2]`.
- Dodawanie — oznacza konkatencję: `'ab'+'cd'='abcd'`.
- Operator członkostwa `in`:

```
>>> 'a' in ('a','b','c')
True
>>> 'a' not in ('a','b','c')
False
```

- Długość: `len(s)`.
- Elementy minimalny i maksymalny: `min(u)`, `max(u)`.
- Wyszukiwanie pierwszego indeksu:

```
>>> 'Ala_ma_kota'.index('a')
2
```

- Zliczanie elementów:

```
>>> 'Ala_ma_kota'.count('a')
3
```

Operatorami rzutowania na typy sekwencyjne są: `str`, `list`, `tuple`. Na typ `bytes` nie możemy rzutować — przekształcając ciąg znaków na ciąg bajtów, musimy podać kodowanie, tak samo, gdy chcemy rzutować w drugą stronę, czyli zdekodować ciąg bajtów do ciągu znaków. Do konwersji ciągu bajtów na ciąg znaków służy metoda `decode`. Do konwersji ciągu znaków na ciąg bajtów służy metoda `encode`. W metodach tych za argument podajemy kodowanie, np. `'utf8'`:

```
>>> b='żółw'.encode('utf8')
>>> type(b)
<class 'bytes'>
>>> b
b'\xc5\xbc\xc3\xb3\xc5\x82w'
>>> s=b.decode('utf8')
>>> type(s)
<class 'str'>
>>> s
'żółw'
```


Skorowidz

- * (operator mnożenia), 15
- * (operator rozpakowania sekwencji), 31
- ** (operator potęgowania), 15
- ** (operator rozpakowania słownika), 32
- **=, 15
- *=, 15
- +=, 15
- =, 15
- //, 15
- //=, 15
- /=, 15
- @, 203
- @classmethod, 230
- @property, 225
- @staticmethod, 230
- % (formatowanie łańcuchów), 38
- % (reszta z dzielenia), 15
- %=, 15
- %d, 38
- %e, 38
- %f, 38
- %s, 37
- # coding, 19
- #!, 44
- #, 18
- &, 262
- <<, 262
- [:], 16
- [], 16
- \n, 18
- \t, 18
- \v, 18
- _, 12
- __add__, 61
- __auth__, 133
- __auth_realm__, 134
- __call__, 64
- __class__, 213
- __delattr__, 222
- __delete__ (metoda deskryptora), 227
- __dict__, 212
- __doc__, 32, 58
- __enter__ (metoda menedżera kontekstu), 215
- __eq__, 67
- __exit__ (metoda menedżera kontekstu), 215
- __floordiv__, 61
- __ge__, 67
- __get__ (metoda deskryptora), 227
- __getattr__, 219
- __getattribute__, 221
- __getitem__, 65
- __gt__, 67
- __init__, 59
- __iter__, 172
- __le__, 67

- __lt__, 67
- '__main__', 40
- __mod__, 61
- __mul__, 61
- __name__, 40
- __name__ (atrybut funkcji), 203
- __ne__, 67
- __neg__, 68
- __next__, 172
- __pow__, 61
- __radd__, 63
- __rfloordiv__, 63
- __rmod__, 63
- __rmul__, 62, 63
- __rpow__, 63
- __rsub__, 63
- __rtruediv__, 63
- __set__ (metoda deskryptora), 227
- __setattr__, 221
- __str__, 61
- __sub__, 61
- __truediv__, 61
- _pickle (moduł Pythona 3), 50

- abs, 138
- abs (funkcja wbudowana), 16
- accept (metoda klasy socket), 94
- access (funkcja z modułu os), 52
- acos (funkcja z modułu math), 42
- add (metoda klasy Gtk.Window), 73
- AF_INET (stała z modułu socket), 93
- AF_INET6 (stała z modułu socket), 93
- AF_UNIX (stała z modułu socket), 93
- all, 24
- and, 23
- animation (moduł z pakietu matplotlib), 155
- any, 24
- Apache (serwer HTTP), 113
- append (metoda typu list), 20
- arange (funkcja z pakietu numpy), 137
- args (atrybut wyjątku), 186
- argument opcjonalny (o domyślnej wartości), 29
- argument pozycyjny, 29
- argument przekazany przez klucz, 29
- argv (zmienna z modułu sys), 44
- Array (klasa z modułu multiprocessing), 245
- as, 40
- asin (funkcja z modułu math), 42
- atan (funkcja z modułu math), 42
- atan2 (funkcja z modułu math), 223
- atomowość instrukcji, 85
- atrybut generowany dynamicznie, 218
- atrybuty prywatne, 223
- auto_scale_xyz (metoda typu Axes3D), 153
- Axes3D (typ z modułu mpl_toolkits.mplot3d), 150
- axis (funkcja z modułu pyplot), 141

- biblioteka standardowa, 41
- bind (metoda klasy socket), 94
- blok instrukcji, 22
- bpython, 13
- branie wycinków, 16
- break, 25
- Button (klasa z modułu Gtk), 73
- 'button-press-event' (sygnał emitowany przez elementy okna), 76
- 'button-release-event' (sygnał emitowany przez elementy okna), 78

- bytes (typ wbudowany), 16
- cgi, 113
- cgi (moduł), 116
- chain (funkcja z modułu `itertools`), 179
- chdir (funkcja z modułu `os`), 52
- check_output (fun. z modułu `subprocess`), 52
- chmod (funkcja z modułu `os`), 52
- chown (funkcja z modułu `os`), 52
- chr (funkcja wbudowana), 20
- class, 57
- close, 184
- close (metoda generatora), 184
- close (metoda klasy `socket`), 94
- close (metoda typu `sqlite3.Connection`), 105
- close (metoda typu `file`), 46
- colorbar (funkcja z modułu `matplotlib.pyplot`), 149
- combinations (funkcja z modułu `itertools`), 178
- combinations_with_replacement (funkcja z modułu `itertools`), 178
- commit (metoda typu `sqlite3.Connection`), 105
- complex (typ wbudowany), 15
- CONCAT (funkcja MySQL), 109
- concatenate (funkcja z modułu `numpy`), 236
- 'configure-event' (sygnał `Gtk.Window`), 78
- conjugate (metoda typu `complex`), 16
- connect (funkcja z modułu `mysql.connector`), 109
- connect (funkcja z modułu `sqlite3`), 104
- connect (metoda klas modułu `Gtk`), 75
- connect (metoda klasy `socket`), 101
- Connection (klasa z modułu `multiprocessing`), 243
- contextlib (moduł), 216
- contextmanager (dekorator z modułu `contextlib`), 216
- continue, 25
- contour (funkcja z modułu `matplotlib.pyplot`), 147
- contour (metoda typu `Axes3D`), 152
- contourf (funkcja z modułu `matplotlib.pyplot`), 148
- cos (funkcja z modułu `math`), 42
- count (funkcja z modułu `itertools`), 177
- count (metoda typu sekwencyjnego), 17
- cPickle (moduł Pythona 2), 50
- CREATE DATABASE (polecenie MySQL), 108
- CREATE TABLE (polecenie MySQL), 109
- CREATE TABLE (polecenie SQLite), 104
- CREATE USER (polecenie MySQL), 108
- cursor (metoda typu `sqlite3.Connection`), 105
- cycle (funkcja z modułu `itertools`), 177
- date (typ z modułu `datetime`), 49
- datetime (moduł), 49
- datetime (typ z modułu `datetime`), 50
- deadlock, 87
- decode (metoda typu `str`), 17
- def, 28
- dekorator funkcji, 202

- del, 28
- DELETE FROM (polecenie SQL), 104
- 'delete-event' (sygnał
 Gtk.Window), 75
- deleter (atrybut deskryptora), 226
- deskryptor, 227
- dict, 27
- dir, 58
- divmod, 262
- dokumentacja funkcji, 32
- dokumentacja klasy, 58
- domknięcie funkcji, 202
- DROP TABLE (polecenie SQL), 104
- dropwhile (funkcja z modułu
 itertools), 180
- dump (funkcja z modułu cPickle),
 51
- dumps (funkcja z modułu cPickle),
 51
- dziedziczenie, 60

- e (stała z modułu math), 42
- elif, 23
- else (dla instrukcji if), 23
- else (dla instrukcji try), 45
- else (dla pętli), 25
- empty (metoda klasy
 multiprocessing.Queue),
 241
- empty (metoda klasy
 Queue.Queue), 90
- encode (metoda typu str), 17
- endwith (metoda typu str), 19
- 'enter_notify_event' (sygnał
 Gtk.Window), 79
- Entry (klasa z modułu Gtk), 73
- environ (zmienna z modułu os), 52
- except, 45
- Exception (klasa nadrzędna
 wyjątków), 185
- execute (metoda typu
 sqlite3.Cursor), 105
- exp (funkcja z modułu math), 42
- fetchall (metoda typu
 sqlite3.Cursor), 105
- fetchone (metoda typu
 sqlite3.Cursor), 105
- FieldStorage (klasa z modułu
 cgi), 116
- figure (funkcja z modułu pyplot),
 150
- file (typ wbudowany), 46
- filter, 35
- filterfalse (funkcja z modułu
 itertools), 179
- finally, 45
- float (typ wbudowany), 15
- float128 (typ z pakietu numpy),
 137
- float64 (typ z pakietu numpy), 137
- for, 24
- format, 38
- from, 40
- FuncAnimation (klasa z modułu
 animation), 155
- funkcja generatora, 175
- funkcja, 28
- funkcja fabryki, 202

- GdkPixbuf (moduł z pakietu gi),
 81
- geany, 13
- generator, 175
- GeneratorExit, 184
- get (funkcja z modułu requests),
 53
- get (metoda klasy
 multiprocessing.Queue),
 241
- get (metoda klasy Queue.Queue),
 90
- get (metoda typu dict), 27
- get_opacity (metoda klasy
 Gtk.Window), 72

- get_position (metoda klasy `Gtk.Window`), 71
- get_size (metoda klasy `Gtk.Window`), 72
- get_text (metoda klasy `Gtk.Entry`), 79
- get_title (metoda klasy `Gtk.Window()`), 71
- getcwd (funkcja z modułu `os`), 52
- getenv (funkcja z modułu `os`), 52
- getfirst (metoda klasy `cgi.FieldStorage`), 116
- getlist (metoda klasy `cgi.FieldStorage`), 116
- gi (pakiet modułów), 70
- global, 33
- gniazdo sieciowe, 93
- grab_focus (metoda elementów okna), 80
- GRANT ALL ON ... TO (polecenie MySQL), 108
- Gtk (moduł z pakietu `gi`), 70
- `Gtk.gdk.pixbuf_new_from_file`, 81

- hasattr, 213
- HBox (klasa z modułu `Gtk`), 73
- hstack (funkcja z modułu `numpy`), 235

- if, 22
- ifilter (funkcja z modułu `itertools`), 177
- ifilterfalse (funkcja z modułu `itertools`), 179
- imag (atrybut typu `complex`), 16
- Image (klasa z modułu `Gtk`), 81
- imap (funkcja z modułu `itertools`), 177
- import, 40
- imshow (funkcja z modułu `pyplot`), 149

- in (członkostwo w typie sekwencyjnym), 17
- in (członkostwo w słowniku), 28
- indeksowanie, 16
- index (metoda typu sekwencyjnego), 17
- input, 43
- insert (metoda typu `list`), 20
- INSERT INTO (polecenie SQL), 104
- instancja, 57
- int (typ wbudowany), 15
- integrate (moduł z pakietu `scipy`), 158
- interaktywna powłoka, 12
- ipython, 13
- isalnum (metoda typu `str`), 19
- isalpha (metoda typu `str`), 19
- isdigit (metoda typu `str`), 19
- isinstance, 60
- islice (funkcja z modułu `itertools`), 179
- isnan (funkcja z pakietu `numpy`), 170
- iter, 174
- iter_lines (metoda obiektu `Response`), 54
- iterator, 172
- itertools (moduł), 177
- izip (funkcja z modułu `itertools`), 180

- join (metoda klasy `Process`), 240
- join (metoda klasy `Thread`), 85
- join (metoda typu `str`), 19
- jupyter notebook, 14

- 'key-press-event' (sygnał `Gtk.Window`), 78
- 'key-release-event' (sygnał `Gtk.Window`), 79
- keys (metoda typu `dict`), 28
- klasa, 57

- klasa w starym stylu (Python 2), 57
- kodowanie znaków, 19
- komentarze, 18
- konkatenacja sekwencji, 17
- konstruktor, 59
- krotka, 16
- kwantyfikatory, 24

- lambda, 34
- laplasjan, 164
- 'leave_notify_event' (sygnał
Gtk.Window), 79
- legend (funkcja z modułu pyplot),
140
- len(s), 17
- liczby zespolone, 15
- LifoQueue (klasa z modułu Queue),
92
- LIKE (operator w SQL), 104
- linspace (funkcja z pakietu
numpy), 137
- list (typ wbudowany), 16
- lista, 16
- lista składana, 25
- listdir (funkcja z modułu os), 52
- listen (metoda klasy socket), 94
- load (funkcja z modułu cPickle),
51
- loads (funkcja z modułu cPickle),
51
- Lock (funkcja z modułu
threading), 86
- log (funkcja z modułu math), 42
- log10 (funkcja z modułu math), 42
- logical_and (funkcja pakietu
numpy), 138
- logical_not (funkcja pakietu
numpy), 138
- logical_or (funkcja pakietu
numpy), 138
- long (typ wbudowany w Pythonie
2), 15
- lstrip (metoda typu str), 20
- łańcuch znaków, 16
- łańcuch bajtów, 16

- main (funkcja z modułu Gtk), 70
- main_quit (funkcja z modułu Gtk),
75
- map, 34
- map (metoda obiektu Pool), 238
- math (moduł), 42
- matplotlib (pakiet modułów), 139
- max, 17
- max (funkcja z modułu numpy), 168
- menedżer kontekstu, 214
- meshgrid (funkcja z pakietu
numpy), 146
- metoda klasy, 230
- metoda statyczna, 230
- mikrowątki, 190
- min, 17
- min (funkcja z modułu numpy), 168
- mkdir (funkcja z modułu os), 52
- mod_python, 114
- mod_python.publisher, 114
- mode (atrybut zmiennej plikowej),
46
- moduł, 40
- modyfikowalny w miejscu, 22
- move (metoda klasy Gtk.Window),
71
- multiprocessing (moduł), 238
- mysql.connector (moduł), 109
- mysqlclient (moduł), 109
- MySQLdb (moduł), 109

- name (atrybut zmiennej plikowej),
46
- name (funkcja z modułu os), 52
- NaN (stała w pakiecie numpy), 142
- nanmax (funkcja z pakietu numpy),
150
- nanmin (funkcja z pakietu numpy),
150

- ndarray (typ z pakietu numpy), 136
- new_subpixbuf (metoda obiektu Pixbuf), 81
- next (funkcja wbudowana), 174
- next (metoda iteratora), 172
- NotImplementedError, 196
- np (skrót od numpy), 136
- numpy (pakiet modułów), 136
- numpy.random (moduł), 188
- object, 57
- odeint (funkcja z modułu scipy.integrate), 158
- ones (funkcja z pakietu numpy), 137, 145
- open, 45
- operator członkostwa, 17, 28
- operatory logiczne, 23
- operatory porównania, 23
- or, 23
- ord (funkcja wbudowana), 20
- ORDER BY ... (polecenie SQL), 104
- os (moduł), 52
- pack_end (metoda klas Gtk.VBox i Gtk.HBox), 73
- pack_start (metoda klas Gtk.VBox i Gtk.HBox), 73
- pakiety modułów, 41
- pass, 25
- permutations (funkcja z modułu itertools), 178
- pi (stała z modułu math), 42
- Pipe (funkcja z modułu multiprocessing), 243
- Pixbuf (obiekt z modułu GdkPixbuf), 81
- plot (funkcja z modułu pyplot), 139
- plot_surface (metoda typu Axes3D), 150
- plt (skrót od matplotlib.pyplot), 139
- pochodna numeryczna druga, 164
- pochodna numeryczna pierwsza, 164
- polar (funkcja z modułu pyplot), 141
- Pool (funkcja z modułu multiprocessing), 238
- pop (metoda typu list), 20
- pop (metoda typu dict), 28
- Popen, 52
- popen (funkcja z modułu os), 52
- powłoka, 12
- powtarzanie sekwencji, 17
- print, 13
- PriorityQueue (klasa z modułu Queue), 92
- problem płytkiej kopii, 22
- Process (klasa z modułu multiprocessing), 239
- product (funkcja z modułu itertools), 178
- property, 224
- przeciążanie, 61
- przysyłanie, 60
- put (metoda klasy multiprocessing.Queue), 241
- put (metoda klasy Queue.Queue), 90
- pycharm, 13
- PyMySQL (moduł), 109
- pyplot (moduł z pakietu matplotlib), 139
- pyswarm (moduł), 191
- query string, 116
- Queue (klasa z modułu multiprocessing), 241
- Queue (klasa z modułu queue/Queue), 90
- Queue (moduł w Pythonie 2), 90
- queue (moduł w Pythonie 3), 90

- quiver (funkcja z modułu `pyplot`), 154
- quote (funkcja z modułu `urllib.parse`), 55
- raise, 67
- rand (funkcja z modułu `numpy.random`), 188
- randint (funkcja z modułu `random`), 43, 111
- randn (funkcja z modułu `numpy.random`), 188
- random (funkcja z modułu `random`), 43
- random (moduł), 43, 111
- range, 173
- range (typ wbudowany), 24
- raw_input (funkcja wbudowana w Pythonie 2), 43
- read (metoda typu `file`), 46
- readline (metoda typu `file`), 46
- readlines (metoda typu `file`), 46
- real (atrybut typu `complex`), 16
- recv (metoda klasy `socket`), 94
- recv (metoda obiektu `multiprocessing.Connection`), 243
- recvfrom (metoda klasy `socket`), 94
- reduce, 36
- remove (funkcja z modułu `os`), 52
- remove (metoda typu `list`), 20
- rename (funkcja z modułu `os`), 52
- repeat (funkcja z modułu `itertools`), 177
- replace (metoda typu `str`), 20
- requests (moduł), 53
- resize (metoda klasy `Gtk.Window`), 71
- Response (obiekt z modułu `requests`), 53
- reszta z dzielenia, 15
- return, 28
- rmdir (funkcja z modułu `os`), 52
- rollback (metoda typu `sqlite3.Connection`), 105
- rsplit (metoda typu `str`), 19
- rstrip (metoda typu `str`), 20
- scale_simple (metoda obiektu `Pixbuf`), 81
- scheduler, 193
- scipy (pakiet modułów), 158
- select (funkcja z modułu `select`), 191
- select (moduł), 191
- SELECT ... FROM (polecenie SQL), 104
- self, 58
- Semaphore (funkcja z modułu `threading`), 89
- send (metoda generatora), 182
- send (metoda obiektu `multiprocessing.Connection`), 243
- sendall (metoda klasy `socket`), 94
- sendto (metoda klasy `socket`), 94
- set_buffer (metoda obiektu `Gtk.TextView`), 79
- set_editable (metoda klasy `Gtk.TextBuffer`), 80
- set_from_file (metoda klasy `Gtk.Image`), 81
- set_from_pixbuf (metoda klasy `Gtk.Image`), 81
- set_opacity (metoda klasy `Gtk.Window`), 71
- set_text (metoda klasy `Gtk.Entry`), 79
- set_text (metoda klasy `Gtk.TextBuffer`), 80
- set_title (metoda klasy `Gtk.Window`), 71
- set_xdata (funkcja uaktualniająca

- dane na wykresie), 156
- set_xlim (funkcja ustalająca zakres osi *x* wykresu), 161
- set_ydata (funkcja uaktualniająca dane na wykresie), 156
- set_ylim (funkcja ustalająca zakres osi *y* wykresu), 161
- setenv (funkcja z modułu `os`), 52
- setsockopt (metoda klasy `socket`), 94
- setter (atrybut deskryptora), 226
- shape (atrybut typu `numpy.array`), 233
- shebang, 44
- show (funkcja z modułu `pyplot`), 140
- show (metoda klas z modułu `Gtk`), 70
- show_all (metoda klasy `Gtk.Window`), 75
- sin (funkcja z modułu `math`), 42
- skrypt, 13
- skrypt cgi, 113
- słownik, 27
- słabe typowanie, 15
- SOCK_DGRAM (stała z modułu `socket`), 93
- SOCK_STREAM (stała z modułu `socket`), 93
- socket (klasa z modułu `socket`), 93
- socket (moduł), 93
- sort (metoda typu `list`), 35
- sorted, 35
- split (metoda typu `str`), 19
- SQL injection, 107
- sqlite3 (moduł), 104
- starmap (funkcja z modułu `itertools`), 179
- start (metoda klasy `Process`), 240
- start (metoda klasy `Thread`), 84
- startswith (metoda typu `str`), 19
- StopIteration, 172
- str (typ wbudowany), 16
- strip (metoda typu `str`), 20
- subplots (funkcja z modułu `pyplot`), 156
- subprocess (moduł), 52
- sum, 20
- super, 66
- sys (moduł), 44
- tablica asocjacyjna, 27
- takewhile (funkcja z modułu `itertools`), 179
- tan (funkcja z modułu `math`), 42
- text (atrybut obiektu `Response`), 53
- TextBuffer (klasa z modułu `Gtk`), 79
- TextView (klasa z modułu `Gtk`), 73
- Thread (moduł), 84
- thread-safe, 84
- threading (klasa z modułu `threading`), 84
- throw (metoda generatora), 186
- time (funkcja z modułu `time`), 48
- time (moduł), 48
- time (typ z modułu `datetime`), 50
- tmpfile (funkcja z modułu `os`), 52
- trójargumentowa instrukcja logiczna, 23
- try, 45
- tryb binarny, 46
- tryb tekstowy, 46
- tuple (typ wbudowany), 16
- typ mapujący, 27
- type, 14
- uname (funkcja z modułu `os`), 52
- unichr (funkcja wbudowana w Pythonie 2), 20
- unicode (typ wbudowany w Pythonie 2), 18

- update (metoda typu dict), 28
- UPDATE ... SET (polecenie SQL), 104
- urllib (moduł w Pythonie 2), 55
- urllib.parse (moduł w Pythonie 3), 55
- urllib.request (moduł w Pythonie 3), 55
- urlopen (funkcja z modułu urllib/urllib.request), 55
- USE (polecenie MySQL), 109
- Value (klasa z modułu multiprocessing), 245
- values (metoda typu dict), 28
- VBox (klasa z modułu Gtk), 73
- vstack (funkcja z modułu numpy), 235

- wątek demona, 85
- wątek roboczy, 90
- WHERE (słowo kluczowe SQL), 104
- while, 24

- widok, 28
- Window (klasa z modułu Gtk), 70
- with, 47
- wraps (dekorator z modułu functools), 204
- write (metoda typu file), 46
- writelines (metoda typu file), 46
- wycinki tablic, 146
- wyrażenie generatora, 176

- xrange (typ wbudowany w Pythonie 2), 24

- yield, 175
- yield from, 176

- zakleszczenie, 87
- zarządca procesów, 193
- zeros (funkcja z pakietu numpy), 137, 145
- zip, 21
- znak ucieczki, 18

PROGRAM PARTNERSKI

— GRUPY HELION —



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

GRUPA
Helion

Python

Kurs dla **nauczycieli** i **studentów**

Weź Pythona na uczelnię!

Python to jeden z tych języków programowania, bez których trudno się dziś obejść. Dzięki słynnej elastyczności oraz rozbudowanemu zestawowi narzędzi i bibliotek można wykorzystywać go w zróżnicowanych projektach i na najróżniejszych platformach, a łatwość opanowania podstaw zachęca do nauki. Sprawdź sam, jak wygodnie i prosto da się w nim pisać własny kod, opracowywać aplikacje desktopowe, projektować strony WWW czy przeprowadzać obliczenia numeryczne.

Książka *Python. Kurs dla nauczycieli i studentów* powstała z myślą o wszystkich, którzy chcą opanować podstawy Pythona i praktycznie wykorzystywać go na co dzień. Kurs dzieli się na dwie części: pierwsza zawiera wprowadzenie do języka, pozwalające ruszyć z własnymi projektami; druga zaś wprowadza ważne, bardziej rozbudowane konstrukcje językowe. Niezwykle istotnym atutem tego podręcznika są ćwiczenia wraz z rozwiązaniami, opracowane na podstawie wieloletnich doświadczeń autora. Można tu znaleźć zarówno materiały do samodzielnej nauki, jak i inspiracje do prowadzenia własnych zajęć z uczniami czy ze studentami. Jeśli chcesz zacząć aktywnie korzystać z niesamowitych możliwości Pythona, ta publikacja jest dla Ciebie!

- **Podstawowe elementy, operacje i funkcje**
- **Programowanie obiektowe i graficzny interfejs użytkownika**
- **Wielowątkowość i komunikacja sieciowa**
- **Obsługa baz danych i współpraca z serwerem Apache**
- **Obliczenia numeryczne**
- **Iteratory, generatory, koprocedury**
- **Współbieżność wykorzystująca podprocesy**

Moc Pythona w Twoich rękach!

 Helion	<i>Sprawdź nasze szkolenia!</i>  SZKOLENIA AKADEMIA IT & BUSINESS WWW.SZKOLENIA.HELION.PL	KOD KORZYŚCI <i>Sięgnij po więcej!</i>  
 helion.pl  HELION SA ul. Kościuszki 1c 44-100 Gliwice tel.: 32 230 98 63 helion@helion.pl		ISBN 978-83-283-4566-9  9 788328 345669
INFORMATYKA W NAJLEPSZYM WYDANIU		Cena: 59,00 zł