
Spis treści

Przedmowa	29
Część I. Wprowadzenie	35
1. Pytania i odpowiedzi dotyczące Pythona	37
Dlaczego ludzie używają Pythona?	37
Jakość oprogramowania	39
Wydażność programistów	39
Czy Python jest językiem skryptowym?	40
Jakie są zatem wady języka Python?	41
Kto dzisiaj używa Pythona?	43
Co mogę zrobić za pomocą Pythona?	43
Programowanie systemowe	44
Graficzne interfejsy użytkownika (GUI) i interfejsy użytkownika (UI)	44
Skrypty internetowe	44
Integracja komponentów	45
Dostęp do baz danych	45
Szybkie prototypowanie	46
Programowanie numeryczne i naukowe	46
I więcej: sztuczna inteligencja, gry, przetwarzanie obrazu, wyszukiwanie danych, testowanie, Excel, aplikacje...	47
Jakie są techniczne mocne strony Pythona?	47
Jest zorientowany obiektowo i funkcyjny	47
Jest darmowy i otwarty	48
Jest przenośny	48
Ma duże możliwości	49
Można go łączyć z innymi językami	50
Jest względnie łatwy w użyciu.	51
Jest względnie łatwy do nauczenia się	51
Podsumowanie rozdziału	52
Sprawdź swoją wiedzę — quiz	52
Sprawdź swoją wiedzę — odpowiedzi	53

2. Jak Python wykonuje programy?	54
Wprowadzenie do interpretera Pythona	54
Wykonywanie programu	55
Z punktu widzenia programisty	55
Z punktu widzenia Pythona	56
Warianty modeli wykonywania	60
Alternatywne implementacje Pythona	60
Samodzielne pliki wykonywalne	64
Przyszłe możliwości	65
Podsumowanie rozdziału	65
Sprawdź swoją wiedzę — quiz	65
Sprawdź swoją wiedzę — odpowiedzi	66
3. Jak uruchamia się programy?	67
Instalacja Pythona	67
Interaktywny kod	68
Uruchamianie interaktywnego środowiska REPL	68
Gdzie uruchamiać programy — katalogi z kodem źródłowym	70
Czego nie wpisywać — znaki zachęty i komentarze	71
Inne środowiska REPL Pythona	72
Interaktywne wykonywanie kodu	73
Do czego służy sesja interaktywna?	74
Pliki źródłowe	76
Pierwszy skrypt	77
Wykonywanie plików z poziomu wiersza poleceń powłoki	78
Sposoby użycia wiersza poleceń	79
Inne sposoby uruchamiania plików	81
Klikanie ikon plików	81
Interfejs użytkownika środowiska IDLE	82
Inne środowiska IDE	84
Aplikacje na smartfony	84
WebAssembly dla przeglądarek	84
Notatniki Jupyter do celów naukowych	85
Kompilatory AOT dla zwiększenia szybkości	85
Uruchamianie kodu w kodzie	86
Inne opcje wykonywania kodu	91
Jaką opcję wybrać?	92
Podsumowanie rozdziału	92
Sprawdź swoją wiedzę — quiz	93
Sprawdź swoją wiedzę — odpowiedzi	93
Sprawdź swoją wiedzę — ćwiczenia do części I	94

4. Wprowadzenie do obiektów Pythona	101
Hierarchia pojęć w Pythonie	101
Dlaczego korzystamy z obiektów wbudowanych?	102
Najważniejsze typy danych w Pythonie	103
Liczby	105
Łańcuchy znaków	107
Operacje na sekwencjach	107
Niezmienność	109
Metody specyficzne dla typu	110
Uzyskiwanie pomocy	112
Inne sposoby kodowania łańcuchów znaków	113
Ciągi znaków w formacie Unicode	114
Listy	116
Operacje na typach sekwencyjnych	116
Operacje specyficzne dla typu	117
Sprawdzanie granic	117
Zagnieżdżanie	118
Listy składane	119
Słowniki	120
Operacje na odwzorowaniach	121
Zagnieżdżanie raz jeszcze	122
Brakujące klucze — testowanie za pomocą if	124
Sortowanie kluczy — pętle for	126
Krotki	127
Do czego służą krotki?	128
Pliki	128
Pliki tekstowe Unicode i binarne	130
Inne narzędzia podobne do plików	131
Inne typy podstawowe	131
Zbiory	131
Wartości logiczne i obiekt None	132
Typy	132
Podpowiedzi typów	133
Klasy definiowane przez użytkownika	134
I wszystko inne	135
Podsumowanie rozdziału	135
Sprawdź swoją wiedzę — quiz	136
Sprawdź swoją wiedzę — odpowiedzi	136

5. Typy liczbowe	138
Podstawy typów liczbowych Pythona	138
Literały liczbowe	139
Wbudowane narzędzia liczbowe	141
Operatory wyrażeń Pythona	141
Połączone operatory stosują się do priorytetów	143
Podwyrażenia grupowane są w nawiasach	144
Pomieszane typy poddawane są konwersji	144
Wprowadzenie: przeciążanie operatorów i polimorfizm	145
Liczby w akcji	146
Zmienne i podstawowe wyrażenia	146
Formaty wyświetlania liczb	148
Operatory porównania	149
Operatory dzielenia	151
Precyzja liczb całkowitych	153
Liczby zespolone	154
Notacja szesnastkowa, ósemkowa i dwójkowa	154
Operacje na poziomie bitów	156
Znaki podkreślenia jako separatory w liczbach	158
Inne wbudowane narzędzia numeryczne	160
Inne typy liczbowe	162
Typ Decimal (liczby dziesiętne)	162
Typ Fraction (liczby ułamkowe)	164
Zbiory	166
Wartości Boolean	173
Rozszerzenia numeryczne	174
Podsumowanie rozdziału	174
Sprawdź swoją wiedzę — quiz	175
Sprawdź swoją wiedzę — odpowiedzi	175
6. Wprowadzenie do typów dynamicznych	177
Sprawa brakujących deklaracji typu	177
Zmienne, obiekty i referencje	178
Typy powiązane są z obiektami, a nie ze zmiennymi	180
Obiekty są uwalniane	181
Referencje współdzielone	183
Referencje współdzielone a modyfikacje w miejscu	184
Referencje współdzielone a równość	186
Typy dynamiczne są wszędzie	188
Podpowiedzi typów: opcjonalne, nieużywane i po co?	189
Podsumowanie rozdziału	190
Sprawdź swoją wiedzę — quiz	191
Sprawdź swoją wiedzę — odpowiedzi	191

7. Łańcuchy znaków	193
Łańcuchy znaków — podstawy	194
Literały łańcuchów znaków	196
Łańcuchy znaków w apostrofach i cudzysłowach są tym samym	196
Sekwencje ucieczki reprezentują znaki specjalne	197
Surowe łańcuchy znaków blokują sekwencje ucieczki	201
Potrójne cudzysłowy i apostrofy kodują łańcuchy znaków będące wielowierszowymi blokami	202
Łańcuchy znaków w akcji	204
Podstawowe operacje	205
Indeksowanie i wycinki	206
Narzędzia do konwersji łańcuchów znaków	211
Modyfikowanie łańcuchów znaków — część I: działania na sekwencjach	213
Metody łańcuchów znaków	214
Składnia wywoływania metod	214
Metody typów znakowych	215
Modyfikowanie łańcuchów znaków — część II: metody łańcuchów znaków	216
Więcej metod łańcuchów znaków — analiza składniowa tekstu	219
Inne często używane metody łańcuchów znaków	220
Formatowanie łańcuchów znaków — triathlon	221
Opcje formatowania łańcuchów znaków	221
Formatowanie z użyciem wyrażeń formatujących	222
Formatowanie łańcuchów z użyciem metody format	227
Podstawy	227
Zaawansowana składnia wywołań metody format	230
Literal formatowania f-string	234
A zwycięzcą jest...	239
Generalne kategorie typów	241
Typy z jednej kategorii współdzielą zbiory operacji	241
Typy mutowalne można modyfikować w miejscu	242
Podsumowanie rozdziału	242
Sprawdź swoją wiedzę — quiz	243
Sprawdź swoją wiedzę — odpowiedzi	243
8. Listy i słowniki	245
Listy	245
Listy w akcji	248
Podstawowe operacje na listach	248
Indeksowanie i wycinki	249
Modyfikacja list w miejscu	250
Iteracje po listach i składanie list	255
Słowniki	258

Słowniki w akcji	260
Podstawowe operacje na słownikach	260
Modyfikacja słowników w miejscu	262
Inne metody słowników	263
Inne sposoby tworzenia słowników	264
Słowniki składane	266
Kolejność wstawiania kluczy	268
Operator „sumy” dla słowników	269
Przykład: baza danych o książkach	270
Uwagi na temat korzystania ze słowników	272
Podsumowanie rozdziału	280
Sprawdź swoją wiedzę — quiz	281
Sprawdź swoją wiedzę — odpowiedzi	281
9. Krotki, pliki i wszystko inne	284
Krotki	285
Krotki w akcji	285
Dlaczego istnieją listy i krotki?	289
Repetitorium: rekordy — krotki nazwane	290
Pliki	291
Otwieranie plików	292
Wykorzystywanie plików	293
Pliki w akcji	295
Pliki tekstowe i binarne — krótka historia	297
Przechowywanie obiektów Pythona w plikach i ich przetwarzanie	299
Przechowywanie natywnych obiektów Pythona — moduł pickle	301
Przechowywanie obiektów Pythona w formacie JSON	302
Przechowywanie obiektów za pomocą innych narzędzi	303
Menedżery kontekstu plików	304
Inne narzędzia powiązane z plikami	304
Przegląd i podsumowanie podstawowych typów obiektów	306
Elastyczność obiektów	307
Referencje a kopie	308
Porównania, testy równości i prawda	310
Porównywanie słowników	313
Prawda czy fałsz, czyli znaczenie True i False w Pythonie	314
Hierarchie typów Pythona	316
Obiekty typów	316
Inne typy w Pythonie	318
Pułapki typów wbudowanych	319
Przypisanie tworzy referencje, nie kopie	319
Powtórzenie dodaje jeden poziom zagłębienia	319

Uwaga na cykliczne struktury danych	320
Typów niemutowalnych nie można modyfikować w miejscu	321
Podsumowanie rozdziału	321
Sprawdź swoją wiedzę — quiz	322
Sprawdź swoją wiedzę — odpowiedzi	322
Sprawdź swoją wiedzę — ćwiczenia do części II	323

Część III. Instrukcje i składnia **327**

10. Wprowadzenie do instrukcji Pythona	329
Raz jeszcze o hierarchii pojęciowej języka Python	329
Instrukcje Pythona	330
Historia dwóch if	332
Co dodaje Python?	332
Co usuwa Python?	333
Skąd bierze się składnia z użyciem wcięć?	335
Kilka przypadków specjalnych	337
Szybki przykład: interaktywne pętle	340
Prosta pętla interaktywna	340
Wykonywanie obliczeń na danych wpisywanych przez użytkownika	342
Obsługa błędów poprzez sprawdzanie danych wejściowych	342
Obsługa błędów za pomocą instrukcji try	344
Kod zagnieżdżony na trzy poziomy głębokości	346
Podsumowanie rozdziału	346
Sprawdź swoją wiedzę — quiz	347
Sprawdź swoją wiedzę — odpowiedzi	347
11. Przypisania, wyrażenia i wyświetlanie	348
Instrukcje przypisania	348
Formy instrukcji przypisania	349
Podstawowe przypisanie	351
Przypisanie sekwencji	351
Rozszerzona składnia rozpakowania sekwencji w Pythonie 3.x	354
Przypisanie z wieloma celami	360
Przypisania rozszerzone	361
Wyrażenia przypisania nazwanego	364
Reguły dotyczące nazw zmiennych	367
Instrukcje wyrażeń	371
Instrukcje wyrażeń i modyfikacje w miejscu	372
Polecenia print	373
Funkcja print	373
Przekierowanie strumienia wyjściowego	376

Podsumowanie rozdziału	380
Sprawdź swoją wiedzę — quiz	381
Sprawdź swoją wiedzę — odpowiedzi	381
12. Testy if i reguły składni	384
Instrukcje if	384
Ogólny format	384
Proste przykłady	385
Instrukcja wielokrotnego wyboru	386
Instrukcje match	388
Podstawowe użycie match	389
Zaawansowane użycie match	391
Reguły składni Pythona raz jeszcze	393
Ograniczniki bloków — reguły tworzenia wcięć	395
Ograniczniki instrukcji — wiersze i znaki kontynuacji	397
Kilka przypadków specjalnych	398
Testy prawdziwości i testy logiczne	399
Wyrażenie trójargumentowe if/else	401
Podsumowanie rozdziału	403
Sprawdź swoją wiedzę — quiz	403
Sprawdź swoją wiedzę — odpowiedzi	404
13. Pętle while i for	407
Pętle while	407
Ogólny format	408
Przykłady	408
Instrukcje break, continue, pass oraz else w pętli	409
Ogólny format pętli	409
Instrukcja pass	410
Instrukcja continue	411
Instrukcja break	412
Klauzula else pętli	413
Pętle for	415
Ogólny format	415
Przykłady	416
Techniki tworzenia pętli	421
Pętle z licznikami — range	422
Skanowanie sekwencji — while, range, for	423
Przetaskowania sekwencji — funkcje range i len	424
Przechodzenie niewyczerpujące — range kontra wycinki	425
Modyfikowanie list — range kontra listy składane	426
Przechodzenie równoległe — zip	427
Generowanie wartości przesunięcia i elementów — enumerate	430

Podsumowanie rozdziału	431
Sprawdź swoją wiedzę — quiz	432
Sprawdź swoją wiedzę — odpowiedzi	432
14. Iteracje i listy składane	435
Iteracje	436
Protokół iteracyjny	437
Wbudowane funkcje iter i next	439
Inne wbudowane typy iterowalne	443
Listy składane	449
Podstawy list składanych	449
Wykorzystywanie list składanych w plikach	450
Rozszerzona składnia list składanych	452
Listy składane — zawieszenie tematu	454
Narzędzia iteracyjne	454
Inne zagadnienia związane z iteracjami	459
Podsumowanie rozdziału	459
Sprawdź swoją wiedzę — quiz	459
Sprawdź swoją wiedzę — odpowiedzi	460
15. Wprowadzenie do dokumentacji	461
Źródła dokumentacji Pythona	461
Komentarze ze znakami #	462
Funkcja dir	462
Notki dokumentacyjne — __doc__	464
PyDoc — funkcja help	468
PyDoc — raporty HTML	472
Nie tylko notki docstrings — pakiet Sphinx	476
Zbiór standardowej dokumentacji	476
Zasoby internetowe	478
Często spotykane problemy programistyczne	478
Podsumowanie rozdziału	480
Sprawdź swoją wiedzę — quiz	481
Sprawdź swoją wiedzę — odpowiedzi	481
Ćwiczenia do części III	482

Część IV. Funkcje i generatory **485**

16. Podstawy funkcji	487
Dlaczego używamy funkcji?	488
Tworzenie funkcji	489
Podstawowe narzędzia funkcji	489
Zaawansowane narzędzia funkcji	490

Ogólne koncepcje związane z funkcjami	491
Instrukcje def	492
Instrukcje return	492
Instrukcja def uruchamiana jest w czasie wykonywania	493
Wyrażenie lambda tworzy funkcje anonimowe	494
Pierwszy przykład: definicje i wywoływanie	495
Definicja	495
Wywołanie	495
Polimorfizm w Pythonie	496
Drugi przykład: przecinające się sekwencje	497
Definicja	497
Wywołania	498
Raz jeszcze o polimorfizmie	499
Zmienne lokalne	500
Podsumowanie rozdziału	500
Sprawdź swoją wiedzę — quiz	501
Sprawdź swoją wiedzę — odpowiedzi	501
17. Zasięgi	503
Podstawy zasięgów w Pythonie	503
Reguły dotyczące zasięgów	505
Rozwiązywanie nazw — reguła LEGB	506
Przykład zasięgu	509
Zasięg wbudowany	511
Instrukcja global	513
Projektowanie programów: minimalizowanie stosowania zmiennych globalnych	515
Projektowanie programów: minimalizacja modyfikacji dokonywanych pomiędzy plikami	516
Inne metody dostępu do zmiennych globalnych	518
Zasięgi a funkcje zagnieżdżone	519
Szczegóły dotyczące zasięgów zagnieżdżonych	519
Przykłady zasięgów zagnieżdżonych	520
Funkcje fabrykujące: domknięcia	521
Instrukcja nonlocal	523
Podstawy instrukcji nonlocal	524
Instrukcja nonlocal w akcji	524
Przypadki graniczne	526
Opcje zachowania stanu	527
Zmienne nonlocal — modyfikowalne, na wywołanie, LEGB	527
Zmienne globalne — modyfikowalne, ale współdzielone	528
Atrybuty funkcji — modyfikowalne, na wywołanie, jawne	529

Klasy — modyfikowalne, na wywołanie, programowanie zorientowane obiektowo	531
A zwycięzcą jest...	531
Zakresy i domyślne wartości argumentów	532
Pętle wymagają wartości domyślnych, nie zasięgów	533
Podsumowanie rozdziału	536
Sprawdź swoją wiedzę — quiz	536
Sprawdź swoją wiedzę — odpowiedzi	537
18. Argumenty	540
Podstawy przekazywania argumentów	540
Argumenty a współdzielone referencje	541
Unikanie modyfikacji argumentów mutowalnych	543
Symulowanie parametrów wyjścia i wielu wyników działania	545
Specjalne tryby dopasowywania argumentów	545
Podstawy dopasowywania argumentów	546
Składnia dopasowania argumentów	547
Dopasowywanie argumentów — szczegóły	548
Przykłady ze słowami kluczowymi i wartościami domyślnymi	549
Przykłady dowolnych argumentów	552
Argumenty tylko ze słowami kluczowymi	557
Argumenty tylko pozycyjne	559
Kolejność argumentów — szczegóły techniczne	561
Kolejność w definicji funkcji	561
Kolejność wywoływania	563
Przykład z funkcją obliczającą minimum	564
Pełne rozwiązanie	565
Bonus	567
Puenta	568
Przykład z uogólnionymi funkcjami działającymi na zbiorach	568
Testowanie kodu	569
Przykład z utworzeniem własnej funkcji print	570
Wykorzystywanie argumentów ze słowami kluczowymi	572
Podsumowanie rozdziału	574
Sprawdź swoją wiedzę — quiz	574
Sprawdź swoją wiedzę — odpowiedzi	575
19. Zaawansowane zagadnienia dotyczące funkcji	577
Koncepcje projektowania funkcji	577
Funkcje rekurencyjne	579
Sumowanie z użyciem rekurencji	580
Implementacje alternatywne	581

Pętle a rekurencja	582
Obsługa dowolnych struktur	583
Obiekty funkcji — atrybuty, adnotacje i tym podobne	589
Obiekty „pierwszej klasy”	589
Introspekcja funkcji	590
Atrybuty funkcji	591
Adnotacje funkcji i dekoratory	592
Funkcje anonimowe — lambda	596
Podstawy wyrażeń lambda	596
Po co używamy wyrażeń lambda?	597
Jak (nie) zaciemniać kodu napisanego w Pythonie?	600
Zasięgi: wyrażenia lambda również można zagnieżdżać	601
Narzędzia programowania funkcyjnego	602
Odwzorowywanie funkcji na obiekty iterowalne — map	602
Wybieranie elementów obiektów iterowalnych — funkcja filter	604
Łączenie elementów obiektów iterowalnych — funkcja reduce	605
Podsumowanie rozdziału	606
Sprawdź swoją wiedzę — quiz	606
Sprawdź swoją wiedzę — odpowiedzi	607
20. Listy składane i generatory	610
Listy składane — akt końcowy	610
Powtórka z list składanych	611
Przykład: listy składane i macierze	615
Funkcje i wyrażenia generatorów	618
Funkcje generatorów — yield kontra return	619
Wyrażenia generatorów — obiekty iterowalne spotykają złożenia	625
Różne ciekawostki dotyczące generatorów	632
Przykład: generowanie mieszanych sekwencji	637
Sekwencje mieszające	638
Permutacje: wszystkie możliwe kombinacje	641
Przykład: emulowanie funkcji zip i map	646
Tworzymy własną implementację funkcji map	647
Własna wersja funkcji zip i map z Pythona 2.X	648
Funkcje asynchroniczne — krótka historia	651
Podstawy funkcji asynchronicznych	652
Podsumowanie funkcji asynchronicznych	659
Podsumowanie rozdziału	660
Sprawdź swoją wiedzę — quiz	660
Sprawdź swoją wiedzę — odpowiedzi	660

21. Wprowadzenie do pomiarów wydajności	664
Pomiary wydajności domowymi sposobami	664
Moduł pomiaru czasu — ujęcie pierwsze	665
Moduł pomiaru czasu — ujęcie drugie	666
Skrypt mierzący wydajność	669
Wyniki pomiarów czasu iteracji	671
Inne rozwiązania dla modułu do pomiaru czasu	675
Mierzenie czasu iteracji z wykorzystaniem modułu timeit	677
Podstawowe reguły korzystania z modułu timeit	678
Automatyczne testy wydajnościowe z użyciem modułu timeit	682
Pułapki związane z funkcjami	688
Lokalne nazwy są wykrywane w sposób statyczny	688
Wartości domyślne i obiekty mutowalne	690
Funkcje, które nie zwracają wyników	692
Różne problemy związane z funkcjami	692
Podsumowanie rozdziału	693
Sprawdź swoją wiedzę — quiz	694
Sprawdź swoją wiedzę — odpowiedzi	694
Sprawdź swoją wiedzę — ćwiczenia do części IV	694

Część V. Moduły i pakiety **699**

22. Moduły — wprowadzenie	701
Moduły — podstawy	701
Dlaczego używamy modułów?	702
Architektura programu w Pythonie	703
Struktura programu	703
Importowanie i atrybuty	704
Moduły biblioteki standardowej	706
Jak działa importowanie?	707
1. Odszukanie modułu	707
2. Kompilowanie (o ile jest to potrzebne)	708
3. Wykonanie	710
Ścieżka wyszukiwania modułów	711
Elementy ścieżki wyszukiwania	711
Konfigurowanie ścieżki wyszukiwania	714
Lista sys.path	715
Wybór pliku modułu	717
Nietypowe ścieżki — pliki samodzielne i pakiety	718
Podsumowanie rozdziału	718
Sprawdź swoją wiedzę — quiz	719
Sprawdź swoją wiedzę — odpowiedzi	719

23. Podstawy tworzenia modułów	721
Tworzenie modułów	721
Nazwy modułów	722
Inne rodzaje modułów	722
Używanie modułów	723
Instrukcja import	723
Instrukcja from	723
Instrukcja from *	724
Operacja importowania jest przeprowadzana tylko raz	725
Instrukcje import są przypisaniami	725
Równoważność instrukcji import oraz from	727
Potencjalne pułapki związane z użyciem instrukcji from	728
Przestrzenie nazw modułów	730
Pliki generują przestrzenie nazw	730
Słowniki przestrzeni nazw: __dict__	731
Kwalifikowanie nazw atrybutów	732
Importowanie a zasięgi	733
Zagnieżdżanie przestrzeni nazw	734
Przeładowywanie modułów	736
Podstawy przeładowywania modułów	737
Przykład przeładowywania z użyciem reload	738
Różne aspekty przeładowywania	739
Podsumowanie rozdziału	739
Sprawdź swoją wiedzę — quiz	740
Sprawdź swoją wiedzę — odpowiedzi	740
24. Pakiety modułów	742
Stosowanie pakietów	742
Podstawy importowania pakietów	743
Pakiety a ustawienia ścieżki wyszukiwania	743
Tworzenie pakietów	744
Podstawowa struktura pakietów	744
Pliki __init__.py	747
Pliki __main__.py	748
Do czego służą pakiety?	750
Historia dwóch systemów	750
Role pliku inicjalizacji pakietu	753
Względne importowanie pakietów	755
Względne i bezwzględne importowanie pakietów	755
Importowanie względne — za i przeciw	756
Importy względne w działaniu	757

Pakiety przestrzeni nazw	760
Modele importowania w Pythonie	761
Uzasadnienie dla pakietów przestrzeni nazw	762
Algorytm wyszukiwania modułu	762
Pakiety przestrzeni nazw w akcji	764
Podsumowanie rozdziału	766
Sprawdź swoją wiedzę — quiz	767
Sprawdź swoją wiedzę — odpowiedzi	767
25. Zaawansowane zagadnienia związane z modułami	769
Koncepcje związane z projektowaniem modułów	769
Ukrywanie danych w modułach	771
Minimalizacja niebezpieczeństw użycia <code>from *</code> — <code>_X</code> oraz <code>__all__</code>	771
Zarządzanie dostępem do atrybutów — <code>__getattr__</code> i <code>__dir__</code>	772
Włączanie opcji z przyszłych wersji Pythona: <code>__future__</code>	774
Mieszane tryby użycia — <code>__name__</code> i <code>__main__</code>	775
Przykład: testy jednostkowe z wykorzystaniem atrybutu <code>__name__</code>	776
Rozszerzenie <code>as</code> dla instrukcji <code>import</code> oraz <code>from</code>	778
Introspekcja modułów	779
Przykład: wyświetlanie modułów za pomocą <code>__dict__</code>	780
Importowanie modułów z użyciem nazwy w postaci ciągu znaków	782
Uruchamianie ciągów znaków zawierających kod	783
Bezpośrednie wywołania: dwie opcje	783
Przykład: przechodnie przeładowywanie modułów	784
Pułapki związane z modułami	794
Kolizje nazw modułów: pakiety i importowanie względne w pakietach	794
W kodzie najwyższego poziomu kolejność instrukcji ma znaczenie	794
Instrukcja <code>from</code> kopiuje nazwy, jednak łączy już nie	795
Instrukcja <code>from *</code> może zaciemnić znaczenie zmiennych	796
Funkcja <code>reload</code> może nie mieć wpływu na obiekty importowane za pomocą <code>from</code>	797
Funkcja <code>reload</code> i instrukcja <code>from</code> a testowanie interaktywne	797
Rekurencyjne importowanie za pomocą <code>from</code> może nie działać	799
Podsumowanie rozdziału	800
Sprawdź swoją wiedzę — quiz	800
Sprawdź swoją wiedzę — odpowiedzi	801
Sprawdź swoją wiedzę — ćwiczenia do części V	801

Część VI. Klasy i programowanie zorientowane obiektowo 805

26. Programowanie zorientowane obiektowo — wprowadzenie	807
Po co używa się klas?	808
Programowanie zorientowane obiektowo z dystansu	809
Wyszukiwanie atrybutów dziedziczonych	810
Klasy a instancje	812
Wywołania metod klasy	813
Tworzenie drzew klas	814
Przeciążanie operatorów	816
Programowanie zorientowane obiektowo oparte jest na ponownym wykorzystaniu kodu	817
Podsumowanie rozdziału	820
Sprawdź swoją wiedzę — quiz	820
Sprawdź swoją wiedzę — odpowiedzi	821
27. Podstawy tworzenia klas	823
Klasy generują wiele obiektów instancji	823
Obiekty klas udostępniają zachowania domyślne	824
Obiekty instancji są rzeczywistymi elementami	825
Pierwszy przykład	825
Klasy dostosowujemy do własnych potrzeb przez dziedziczenie	828
Drugi przykład	829
Klasy są atrybutami w modułach	830
Klasy mogą przechwytywać operatory Pythona	832
Trzeci przykład	833
Najprostsza klasa Pythona na świecie	835
Klasy od kuchni	837
Jeszcze kilka słów o rekordach: klasy kontra słowniki	839
Podsumowanie rozdziału	841
Sprawdź swoją wiedzę — quiz	841
Sprawdź swoją wiedzę — odpowiedzi	842
28. Bardziej realistyczny przykład	844
Krok 1. — tworzenie instancji	845
Tworzenie konstruktorów	845
Testowanie w miarę pracy	847
Wykorzystywanie kodu na dwa sposoby	848
Krok 2. — dodawanie metod	849
Tworzenie kodu metod	851
Krok 3. — przeciążanie operatorów	853
Udostępnienie sposobów wyświetlania	853

Krok 4. — dostosowywanie zachowania za pomocą klas podrzędnych	855
Tworzenie klas podrzędnych	855
Rozszerzanie metod — niepoprawny sposób	856
Rozszerzanie metod — poprawny sposób	857
Polimorfizm w akcji	860
Dziedziczenie, dostosowanie do własnych potrzeb i rozszerzenie	860
Programowanie zorientowane obiektowo — idea	861
Krok 5. — dostosowanie do własnych potrzeb także konstruktorów	862
Programowanie zorientowane obiektowo jest prostsze, niż się wydaje	864
Inne sposoby łączenia klas	864
Krok 6. — wykorzystywanie narzędzi do introspekcji	868
Specjalne atrybuty klas	869
Uniwersalne narzędzie do wyświetlania	870
Atrybuty instancji a atrybuty klas	872
Nazwy w klasach narzędziowych	872
Ostateczna postać naszych klas	873
Krok 7. i ostatni — przechowanie obiektów w bazie danych	875
Obiekty pickle i shelve	876
Przechowywanie obiektów w bazie danych za pomocą shelve	877
Interaktywna eksploracja obiektów shelve	878
Uaktualnianie obiektów w pliku shelve	880
Przyszłe kierunki rozwoju	881
Podsumowanie rozdziału	882
Sprawdź swoją wiedzę — quiz	883
Sprawdź swoją wiedzę — odpowiedzi	883
29. Szczegóły kodowania klas	886
Instrukcja class	886
Ogólna forma	887
Przykład: atrybuty klasy	887
Metody	890
Przykład metody	891
Inne możliwości wywoływania metod	892
Dziedziczenie	892
Tworzenie drzewa atrybutów	892
Szczegóły dziedziczenia	893
Specjalizacja odziedziczonych metod	893
Techniki interfejsów klas	896
Abstrakcyjne klasy nadrzędne	897
Przestrzeń nazw — cała historia	900
Proste nazwy — globalne, o ile nie są przypisane	900
Nazwy atrybutów — przestrzeń nazw obiektów	901

Zen przestrzeni nazw Pythona — przypisanie klasyfikują zmienne	902
Klasy zagnieżdżone — jeszcze kilka słów o regule LEGB	904
Słowniki przestrzeni nazw — przegląd	907
Łączy przestrzeni nazw — przechodzenie w górę drzewa klas	909
Raz jeszcze o notkach dokumentacyjnych	911
Klasy a moduły	913
Podsumowanie rozdziału	913
Sprawdź swoją wiedzę — quiz	914
Sprawdź swoją wiedzę — odpowiedzi	914
30. Przeciążanie operatorów	915
Podstawy	915
Konstruktory i wyrażenia — <code>__init__</code> i <code>__sub__</code>	916
Często spotykane metody przeciążania operatorów	917
Indeksowanie i wycinanie — <code>__getitem__</code> i <code>__setitem__</code>	919
Wycinki	919
Przechwytywanie przypisań elementu	921
Metoda <code>__index__</code> nie służy do indeksowania!	921
Iteracja po indeksie — <code>__getitem__</code>	922
Obiekty iteratorów — <code>__iter__</code> i <code>__next__</code>	923
Iteratory zdefiniowane przez użytkownika	924
Wiele iteracji po jednym obiekcie	927
Alternatywa: metoda <code>__iter__</code> i instrukcja <code>yield</code>	929
Test przynależności — <code>__contains__</code> , <code>__iter__</code> i <code>__getitem__</code>	934
Dostęp do atrybutów — <code>__getattr__</code> i <code>__setattr__</code>	937
Odwołania do atrybutów	937
Przypisywanie wartości i usuwanie atrybutów	938
Inne narzędzia do zarządzania atrybutami	940
Emulowanie prywatności w atrybutach instancji	941
Reprezentacje łańcuchów — <code>__repr__</code> i <code>__str__</code>	942
Po co nam dwie metody wyświetlania?	943
Uwagi dotyczące wyświetlania	944
Dodawanie prawostronne i miejscowa modyfikacja:	
metody <code>__radd__</code> i <code>__iadd__</code>	946
Dodawanie prawostronne	946
Dodawanie w miejscu	950
Wywołania — <code>__call__</code>	951
Interfejsy funkcji i kod oparty na wywołaniach zwrotnych	953
Porównania — <code>__lt__</code> , <code>__gt__</code> i inne	955
Testy logiczne — <code>__bool__</code> i <code>__len__</code>	956
Destrukcja obiektu — <code>__del__</code>	957
Uwagi dotyczące stosowania destruktorów	958

Podsumowanie rozdziału	959
Sprawdź swoją wiedzę — quiz	960
Sprawdź swoją wiedzę — odpowiedzi	960
31. Projektowanie z użyciem klas	962
Python a programowanie zorientowane obiektowo	962
Polimorfizm to interfejsy, a nie sygnatury wywołań	963
Programowanie zorientowane obiektowo i dziedziczenie — związek „jest”	964
Programowanie zorientowane obiektowo i kompozycja — związki typu „ma”	966
Raz jeszcze procesor strumienia danych	968
Programowanie zorientowane obiektowo a delegacja — obiekty „opakowujące”	971
Pseudoprywatne atrybuty klas	973
Przegląd zniekształcania nazw zmiennych	974
Po co używa się atrybutów pseudoprywatnych?	974
Metody są obiektami — z wiązaniem i bez wiązania	977
Metody związane w akcji	978
Klasy są obiektami — uniwersalne fabryki obiektów	981
Do czego służą fabryki?	982
Dziedziczenie wielokrotne i MRO	983
Jak działa dziedziczenie wielokrotne?	985
Jak działa MRO?	986
Rozwiązywanie konfliktów atrybutów	988
Przykład: listowanie atrybutów klas mieszanych	990
Przykład: wyświetlanie atrybutów ze źródłem dziedziczenia	1000
Inne zagadnienia związane z projektowaniem	1004
Podsumowanie rozdziału	1005
Sprawdź swoją wiedzę — quiz	1005
Sprawdź swoją wiedzę — odpowiedzi	1005
32. Zaawansowane zagadnienia związane z klasami	1007
Rozszerzanie typów wbudowanych	1008
Rozszerzanie typów za pomocą osadzania	1008
Rozszerzanie typów za pomocą klas podrzędnych	1009
Model obiektowy Pythona	1012
Klasy są typami, a typy są klasami	1012
Niektóre instancje są równiejsze od innych	1013
Rozgałęzienie dziedziczenia	1014
Różnica między metaklasą a klasą	1016
I jeden object rządzi wszystkim	1016
Zaawansowane narzędzia do obsługi atrybutów	1017
Sloty: deklaracje atrybutów	1017
Właściwości klas: dostęp do atrybutów	1027
Implementacje atrybutów: __getattr__ i deskryptory	1030

Metody statyczne oraz metody klasy	1031
Do czego potrzebujemy metod specjalnych?	1031
Metody statyczne	1032
Alternatywy dla metod statycznych	1033
Używanie metod statycznych i metod klas	1034
Zliczanie instancji z użyciem metod statycznych	1036
Zliczanie instancji z metodami klas	1037
Dekoratory i metaklasy	1040
Podstawowe informacje o dekoratorach funkcji	1041
Pierwsze spojrzenie na funkcję dekoratora zdefiniowaną przez użytkownika	1042
Pierwsze spojrzenie na dekoratory klas i metaklasy	1044
Dalsza lektura	1046
Funkcja super	1047
Podstawy funkcji super	1047
Szczegóły dotyczące funkcji super	1048
Posumowanie funkcji super	1056
Pułapki związane z klasami	1057
Modyfikacja atrybutów klas może mieć efekty uboczne	1058
Modyfikowanie mutowalnych atrybutów klas również może mieć efekty uboczne	1059
Dziedziczenie wielokrotne — kolejność ma znaczenie	1060
Zakresy w metodach i klasach	1061
Różne pułapki związane z klasami	1063
Przesadne opakowywanie	1063
Podsumowanie rozdziału	1064
Sprawdź swoją wiedzę — quiz	1064
Sprawdź swoją wiedzę — odpowiedzi	1065
Sprawdź swoją wiedzę — ćwiczenia do części VI	1065

Część VII. Wyjątki 1073

33. Podstawy wyjątków	1075
Po co używa się wyjątków?	1075
Role wyjątków	1076
Wyjątki w skrócie	1077
Domyślny program obsługi wyjątków	1077
Przechwytywanie wyjątków	1078
Zgłaszanie wyjątków	1080
Wyjątki zdefiniowane przez użytkownika	1081
Działania końcowe	1081

Podsumowanie rozdziału	1083
Sprawdź swoją wiedzę — quiz	1084
Sprawdź swoją wiedzę — odpowiedzi	1084
34. Szczegółowe informacje dotyczące wyjątków	1086
Instrukcja try	1086
Klauzule instrukcji try	1086
Klauzule except i else	1087
Klauzula finally	1094
Połączone klauzule instrukcji try	1097
Instrukcja raise	1101
Zgłaszanie wyjątków	1101
Klauzula except jako punkt zaczepienia	1102
Zakresy widoczności zmiennych i except as	1102
Przekazywanie wyjątków za pomocą raise	1103
Łańcuchy wyjątków — raise from	1104
Instrukcja assert	1106
Przykład: wychwytywanie ograniczeń (ale nie błędów!)	1107
Instrukcja with i menedżery kontekstu	1108
Podstawowe zastosowanie with	1108
Protokół zarządzania kontekstem	1110
Kilka menedżerów kontekstu	1112
Obsługa zakończenia	1112
Podsumowanie rozdziału	1114
Sprawdź swoją wiedzę — quiz	1115
Sprawdź swoją wiedzę — odpowiedzi	1115
35. Obiekty wyjątków	1116
Klasy wyjątków	1117
Tworzenie klas wyjątków	1117
Do czego służą hierarchie wyjątków?	1119
Wbudowane klasy wyjątków	1122
Kategorie wbudowanych wyjątków	1123
Domyślne wyświetlanie oraz stan	1124
Własne sposoby wyświetlania	1126
Własne dane oraz zachowania	1127
Udostępnianie szczegółów wyjątku	1127
Udostępnianie metod wyjątków	1128
Grupy wyjątków — kolejna gwiazdka!	1130
Podsumowanie rozdziału	1133
Sprawdź swoją wiedzę — quiz	1133
Sprawdź swoją wiedzę — odpowiedzi	1134

36. Projektowanie z wykorzystaniem wyjątków	1135
Zagnieżdżanie programów obsługi wyjątków	1135
Przykład: zagnieżdżanie przebiegu sterowania	1137
Przykład: zagnieżdżanie składniowe	1138
Zastosowanie wyjątków	1140
Wychodzenie z głęboko zagnieżdżonych pętli: instrukcja go to	1140
Wyjątki nie zawsze są błędami	1141
Funkcje mogą sygnalizować warunki za pomocą raise	1142
Zamykanie plików oraz połączeń z serwerem	1143
Debugowanie z wykorzystaniem zewnętrznych instrukcji try	1143
Testowanie kodu wewnątrz tego samego procesu	1144
Więcej informacji na temat funkcji sys.exc_info	1145
Wyświetlanie błędów i śladów stosu	1146
Wskazówki i pułapki dotyczące projektowania wyjątków	1147
Co powinniśmy opakować w try?	1147
Jak nie przechwytywać zbyt wiele — unikanie pustych except i wyjątków	1148
Jak nie przechwytywać zbyt mało — korzystanie z kategorii opartych na klasach	1150
Podsumowanie podstaw języka Python	1151
Zbiór narzędzi Pythona	1151
Narzędzia programistyczne przeznaczone do większych projektów	1152
Podsumowanie rozdziału	1156
Sprawdź swoją wiedzę — quiz	1157
Sprawdź swoją wiedzę — odpowiedzi	1157
Sprawdź swoją wiedzę — ćwiczenia do części VII	1157

Część VIII. Zagadnienia zaawansowane **1159**

37. Łącuchy znaków Unicode oraz łącuchy bajtowe	1161
Podstawy Unicode	1162
Kodowanie znaków	1162
Kodowanie znaków	1164
Wprowadzenie do narzędzi łańcuchów znaków w Pythonie	1167
Obiekt str	1167
Obiekt bytes	1167
Obiekt bytearray	1168
Pliki binarne i tekstowe	1168
Wykorzystanie ciągów znaków	1169
Literały tekstowe i podstawowe właściwości	1170
Konwersje typów ciągów	1171
Kodowanie łańcuchów znaków Unicode w Pythonie	1173
Deklaracje typu kodowania znaków pliku źródłowego	1178

Wykorzystywanie łańcuchów bajtowych	1181
Wywołania metod	1181
Operacje na sekwencjach	1182
Formatowanie	1183
Inne sposoby tworzenia obiektów bytes	1183
Mieszanie typów łańcuchów znaków	1184
Obiekt bytearray	1185
Wykorzystywanie plików tekstowych i binarnych	1188
Podstawy plików tekstowych	1188
Tryby tekstowy i binarny	1189
Pliki tekstowe Unicode	1191
Unicode, obiekt bytes i inne narzędzia łańcuchów znaków	1194
Moduł dopasowywania wzorców re	1194
Moduł danych binarnych struct	1195
Moduł serializacji obiektów pickle i json	1196
Nazwy plików w funkcji open i inne narzędzia dla nazw plików	1197
Zmierz Unicode	1201
Obsługa BOM w Pythonie	1201
Normalizacja Unicode — dokąd zmierza ten standard?	1205
Podsumowanie rozdziału	1208
Sprawdź swoją wiedzę — quiz	1208
Sprawdź swoją wiedzę — odpowiedzi	1209
38. Zarządzane atrybuty	1211
39. Dekoratory	1212
40. Metaklasy i dziedziczenie	1214
Tworzyć metaklasy czy tego nie robić?	1215
Wady funkcji pomocniczych	1215
Metaklasy a dekoratory klas — runda 1.	1218
Model metaklasy	1219
Klasy są instancjami obiektu type	1219
Metaklasy są klasami podrzędnymi klasy type	1220
Instrukcje class wywołują typ	1221
Instrukcje class mogą wybierać typ	1222
Protokół metod metaklas	1223
Tworzenie metaklas	1224
Prosta metaklasa	1224
Dostosowywanie tworzenia do własnych potrzeb oraz inicjalizacja	1225
Pozostałe sposoby tworzenia metaklas	1226
Zarządzanie klasami za pomocą metaklas i dekoratorów	1229

Dziedziczenie — finał	1235
Metaklasy a klasy nadrzędne	1237
Dziedziczenie metaklas	1239
Algorytm dziedziczenia Pythona — prosta wersja	1240
Algorytm dziedziczenia w Pythonie — trudniejsza wersja	1243
Podsumowanie dziedziczenia	1246
Metody metaklas	1247
Metody metaklasy a metody klasy	1248
Przeciążanie operatorów w metodach metaklasy	1249
Metody metaklas a metody instancji	1250
Podsumowanie rozdziału	1252
Sprawdź swoją wiedzę — quiz	1252
Sprawdź swoją wiedzę — odpowiedzi	1252
41. Wszystko, co najlepsze	1254
Fala zmian w Pythonie	1254
Piaskownica Pythona	1256
Zalety Pythona	1257
Końcowe wnioski	1257
Dokąd dalej?	1258
Na bis: wydrukuj swój certyfikat!	1258
 Dodatki	 1263
A. Wskazówki dotyczące użytkowania platformy	1265
Korzystanie z Pythona w systemie Windows	1266
Korzystanie z Pythona w systemie macOS	1272
Korzystanie z Pythona w systemie Linux	1277
Korzystanie z Pythona w systemie Android	1280
Korzystanie z Pythona w systemie iOS	1284
Samodzielne programy i pliki wykonywalne	1285
I tak dalej	1288
B. Rozwiązania ćwiczeń podsumowujących poszczególne części książki	1289
Część I. Wprowadzenie	1289
Część II. Typy i operacje	1292
Część III. Instrukcja i składnia	1298
Część IV. Funkcje i generatory	1301
Część V. Moduły i pakiety	1311
Część VI. Klasy i programowanie zorientowane obiektowo	1316
Część VII. Wyjątki	1324
Skorowidz	1332