
Spis treści

Przedmowa	11
1. Algorytmy i struktury danych	15
1.1. Wypakowywanie sekwencji do odrębnych zmiennych	15
1.2. Wypakowywanie elementów z obiektów iterowalnych o dowolnej długości	16
1.3. Zachowywanie ostatnich N elementów	19
1.4. Wyszukiwanie N największych lub najmniejszych elementów	20
1.5. Tworzenie kolejki priorytetowej	22
1.6. Odwzorowywanie kluczy na różne wartości ze słownika	24
1.7. Określanie uporządkowania w słownikach	25
1.8. Obliczenia na danych ze słowników	26
1.9. Wyszukiwanie identycznych danych w dwóch słownikach	28
1.10. Usuwanie powtórzeń z sekwencji przy zachowaniu kolejności elementów	29
1.11. Nazywanie wycinków	30
1.12. Określanie najczęściej występujących w sekwencji elementów	31
1.13. Sortowanie list słowników według wspólnych kluczy	33
1.14. Sortowanie obiektów bez wbudowanej obsługi porównań	34
1.15. Grupowanie rekordów na podstawie wartości pola	35
1.16. Filtrowanie elementów sekwencji	37
1.17. Pobieranie podzbioru słownika	39
1.18. Odwzorowywanie nazw na elementy sekwencji	40
1.19. Jednoczesne przekształcanie i redukowanie danych	42
1.20. Łączenie wielu odwzorowań w jedno	43
2. Łańcuchy znaków i tekst	47
2.1. Podział łańcuchów znaków po wykryciu dowolnego z różnych ograniczników	47
2.2. Dopasowywanie tekstu do początkowej lub końcowej części łańcucha znaków	48
2.3. Dopasowywanie łańcuchów znaków za pomocą symboli wieloznacznych powłoki	50
2.4. Dopasowywanie i wyszukiwanie wzorców tekstowych	51

2.5. Wyszukiwanie i zastępowanie tekstu	54
2.6. Wyszukiwanie i zastępowanie tekstu bez uwzględniania wielkości liter	55
2.7. Tworzenie wyrażeń regularnych w celu uzyskania najkrótszego dopasowania	56
2.8. Tworzenie wyrażeń regularnych dopasowywanych do wielowierszowych wzorców	57
2.9. Przekształcanie tekstu w formacie Unicode na postać standardową	58
2.10. Używanie znaków Unicode w wyrażeniach regularnych	60
2.11. Usuwanie niepożądanych znaków z łańcuchów	61
2.12. Zapewnianie poprawności i porządkowanie tekstu	62
2.13. Wyrównywanie łańcuchów znaków	64
2.14. Łączenie łańcuchów znaków	66
2.15. Podstawianie wartości za zmienne w łańcuchach znaków	68
2.16. Formatowanie tekstu w celu uzyskania określonej liczby kolumn	70
2.17. Obsługiwanie encji HTML-a i XML-a w tekście	71
2.18. Podział tekstu na tokeny	73
2.19. Tworzenie prostego rekurencyjnego parsera zstępującego	75
2.20. Przeprowadzanie operacji tekstowych na łańcuchach bajtów	83
3. Liczby, daty i czas	87
3.1. Zaokrąglanie liczb	87
3.2. Przeprowadzanie dokładnych obliczeń na liczbach dziesiętnych	88
3.3. Formatowanie liczb w celu ich wyświetlenia	90
3.4. Stosowanie dwójkowych, ósemkowych i szesnastkowych liczb całkowitych	92
3.5. Pakowanie do bajtów i wypakowywanie z bajtów dużych liczb całkowitych	93
3.6. Przeprowadzanie obliczeń na liczbach zespolonych	95
3.7. Nieskończoność i wartości NaN	96
3.8. Obliczenia z wykorzystaniem ułamków	98
3.9. Obliczenia z wykorzystaniem dużych tablic liczbowych	99
3.10. Przeprowadzanie operacji na macierzach i z zakresu algebry liniowej	102
3.11. Losowe pobieranie elementów	103
3.12. Przekształcanie dni na sekundy i inne podstawowe konwersje związane z czasem	105
3.13. Określanie daty ostatniego piątku	107
3.14. Określanie przedziału dat odpowiadającego bieżącemu miesiącowi	108
3.15. Przekształcanie łańcuchów znaków na obiekty typu datetime	110
3.16. Manipulowanie datami z uwzględnieniem stref czasowych	111
4. Iteratory i generatory	113
4.1. Ręczne korzystanie z iteratora	113
4.2. Delegowanie procesu iterowania	114
4.3. Tworzenie nowych wzorców iterowania z wykorzystaniem generatorów	115
4.4. Implementowanie protokołu iteratora	117
4.5. Iterowanie w odwrotnej kolejności	119

4.6. Definiowanie funkcji generatorów z dodatkowym stanem	120
4.7. Pobieranie wycinków danych zwracanych przez iterator	121
4.8. Pomijanie pierwszej części obiektu iterowalnego	122
4.9. Iterowanie po wszystkich możliwych kombinacjach lub permutacjach	124
4.10. Przechodzenie po parach indeks – wartość sekwencji	125
4.11. Jednoczesne przechodzenie po wielu sekwencjach	127
4.12. Przechodzenie po elementach z odrębnych kontenerów	129
4.13. Tworzenie potoków przetwarzania danych	130
4.14. Przekształcanie zagnieźdzonych sekwencji na postać jednowymiarową	133
4.15. Przechodzenie po scalonych posortowanych obiektach iterowalnych zgodnie z kolejnością sortowania	134
4.16. Zastępowanie nieskończonych pętli while iteratorem	135
5. Pliki i operacje wejścia-wyjścia	137
5.1. Odczyt i zapis danych tekstowych	137
5.2. Zapisywanie danych z funkcji print() do pliku	139
5.3. Stosowanie niestandardowych separatorów lub końca wiersza w funkcji print()	140
5.4. Odczyt i zapis danych binarnych	141
5.5. Zapis danych do pliku, który nie istnieje	142
5.6. Wykonywanie operacji wejścia-wyjścia na łańcuchach	143
5.7. Odczytywanie i zapisywanie skompresowanych plików z danymi	144
5.8. Przechodzenie po rekordach o stałej wielkości	145
5.9. Wczytywanie danych binarnych do zmiennego bufora	146
5.10. Odwzorowywanie plików binarnych w pamięci	148
5.11. Manipulowanie ścieżkami	150
5.12. Sprawdzanie, czy plik istnieje	151
5.13. Pobieranie listy zawartości katalogu	152
5.14. Nieuwzględnianie kodowania nazw plików	153
5.15. Wyświetlanie nieprawidłowych nazw plików	154
5.16. Dodawanie lub zmienianie kodowania otwartego pliku	156
5.17. Zapisywanie bajtów w pliku tekstowym	158
5.18. Umieszczanie deskryptora istniejącego pliku w obiekcie pliku	159
5.19. Tworzenie tymczasowych plików i katalogów	160
5.20. Komunikowanie z portami szeregowymi	162
5.21. Serializowanie obiektów Pythona	163
6. Kodowanie i przetwarzanie danych	167
6.1. Wczytywanie i zapisywanie danych CSV	167
6.2. Wczytywanie i zapisywanie danych w formacie JSON	170
6.3. Parsowanie prostych danych w XML-u	174
6.4. Stopniowe parsowanie bardzo dużych plików XML	176

6.5. Przekształcanie słowników na format XML	179
6.6. Parsowanie, modyfikowanie i ponowne zapisywanie dokumentów XML	181
6.7. Parsowanie dokumentów XML z przestrzeniami nazw	183
6.8. Komunikowanie się z relacyjnymi bazami danych	185
6.9. Dekodowanie i kodowanie cyfr w systemie szesnastkowym	187
6.10. Dekodowanie i kodowanie wartości w formacie Base64	188
6.11. Odczyt i zapis tablic binarnych zawierających struktury	188
6.12. Wczytywanie zagnieżdżonych struktur binarnych o zmiennej długości	192
6.13. Podsumowywanie danych i obliczanie statystyk	200
7. Funkcje	203
7.1. Pisanie funkcji przyjmujących dowolną liczbę argumentów	203
7.2. Tworzenie funkcji przyjmujących argumenty podawane wyłącznie za pomocą słów kluczowych	204
7.3. Dołączanie metadanych z informacjami do argumentów funkcji	205
7.4. Zwracanie wielu wartości przez funkcje	206
7.5. Definiowanie funkcji z argumentami domyślnymi	207
7.6. Definiowanie funkcji anonimowych (wewnątrzwerszowych)	210
7.7. Pobieranie wartości zmiennych w funkcjach anonimowych	211
7.8. Uruchamianie n-argumentowej jednostki wywoływalnej z mniejszą liczbą argumentów	212
7.9. Zastępowanie klas z jedną metodą funkcjami	215
7.10. Dodatkowy stan w funkcjach wywoływanych zwrotnie	216
7.11. Wewnątrzwerszowe zapisywanie wywoływanych zwrotnie funkcji	219
7.12. Dostęp do zmiennych zdefiniowanych w domknięciu	221
8. Klasy i obiekty	225
8.1. Modyfikowanie tekstowej reprezentacji obiektów	225
8.2. Modyfikowanie formatowania łańcuchów znaków	226
8.3. Dodawanie do obiektów obsługi protokołu zarządzania kontekstem	228
8.4. Zmniejszanie zużycia pamięci przy tworzeniu dużej liczby obiektów	230
8.5. Hermetyzowanie nazw w klasie	231
8.6. Tworzenie atrybutów zarządzanych	232
8.7. Wywoływanie metod klasy bazowej	236
8.8. Rozszerzanie właściwości w klasie pochodnej	240
8.9. Tworzenie nowego rodzaju atrybutów klasy lub egzemplarza	243
8.10. Stosowanie właściwości obliczanych w leniwy sposób	246
8.11. Upraszczenie procesu inicjowania struktur danych	248
8.12. Definiowanie interfejsu lub abstrakcyjnej klasy bazowej	251
8.13. Tworzenie modelu danych lub systemu typów	254

8.14. Tworzenie niestandardowych kontenerów	259
8.15. Delegowanie obsługi dostępu do atrybutów	262
8.16. Definiowanie więcej niż jednego konstruktora w klasie	266
8.17. Tworzenie obiektów bez wywoływania metody <code>__init__()</code>	267
8.18. Rozszerzanie klas za pomocą klas mieszanych	269
8.19. Implementowanie obiektów ze stanem lub maszyn stanowych	273
8.20. Wywoływanie metod obiektu na podstawie nazwy w łańcuchu znaków	278
8.21. Implementowanie wzorca odwiedzający	279
8.22. Implementowanie wzorca odwiedzający bez stosowania rekurencji	283
8.23. Zarządzanie pamięcią w cyklicznych strukturach danych	288
8.24. Tworzenie klas z obsługą porównań	291
8.25. Tworzenie obiektów zapisywanych w pamięci podręcznej	293
9. Metaprogramowanie	297
9.1. Tworzenie nakładek na funkcje	297
9.2. Zachowywanie metadanych funkcji przy pisaniu dekoratorów	299
9.3. Pobieranie pierwotnej funkcji z nakładki	300
9.4. Tworzenie dekoratorów przyjmujących argumenty	302
9.5. Definiowanie dekoratora z atrybutami dostosowywanymi przez użytkownika	303
9.6. Definiowanie dekoratorów przyjmujących opcjonalny argument	306
9.7. Wymuszanie sprawdzania typów w funkcji za pomocą dekoratora	307
9.8. Definiowanie dekoratorów jako elementów klasy	311
9.9. Definiowanie dekoratorów jako klas	312
9.10. Stosowanie dekoratorów do metod klasy i metod statycznych	315
9.11. Pisanie dekoratorów, które dodają argumenty do funkcji w nakładkach	316
9.12. Stosowanie dekoratorów do poprawiania definicji klas	319
9.13. Używanie metaklasy do kontrolowania tworzenia obiektów	320
9.14. Sprawdzanie kolejności definiowania atrybutów klasy	323
9.15. Definiowanie metaklas przyjmujących argumenty opcjonalne	325
9.16. Sprawdzanie sygnatury na podstawie argumentów <code>*args</code> i <code>**kwargs</code>	327
9.17. Wymuszanie przestrzegania konwencji pisania kodu w klasie	330
9.18. Programowe definiowanie klas	332
9.19. Inicjowanie składowych klasy w miejscu definicji klasy	335
9.20. Przeciążanie metod z wykorzystaniem uwag do funkcji	337
9.21. Unikanie powtarzających się metod właściwości	342
9.22. Definiowanie w łatwy sposób menedżerów kontekstu	344
9.23. Wykonywanie kodu powodującego lokalne efekty uboczne	346
9.24. Parsowanie i analizowanie kodu źródłowego Pythona	348
9.25. Dezasemblacja kodu bajtowego Pythona	351

10. Moduły i pakiety	355
10.1. Tworzenie hierarchicznych pakietów z modułami	355
10.2. Kontrolowanie importowania wszystkich symboli	356
10.3. Importowanie modułów podrzędnych z pakietu za pomocą nazw względnych	357
10.4. Podział modułu na kilka plików	358
10.5. Tworzenie odrębnych katalogów z importowanym kodem z jednej przestrzeni nazw	361
10.6. Ponowne wczytywanie modułów	362
10.7. Umożliwianie wykonywania kodu z katalogu lub pliku zip jako głównego skryptu	364
10.8. Wczytywanie pliku z danymi z pakietu	365
10.9. Dodawanie katalogów do zmiennej sys.path	366
10.10. Importowanie modułów na podstawie nazwy z łańcucha znaków	367
10.11. Wczytywanie modułów ze zdalnego komputera z wykorzystaniem haków w poleceniu importu	368
10.12. Modyfikowanie modułów w trakcie importowania	382
10.13. Instalowanie pakietów tylko na własny użytek	384
10.14. Tworzenie nowego środowiska Pythona	385
10.15. Rozpowszechnianie pakietów	386
11. Sieci i rozwijanie aplikacji sieciowych	389
11.1. Interakcja z usługami HTTP za pomocą kodu klienta	389
11.2. Tworzenie serwera TCP	393
11.3. Tworzenie serwera UDP	395
11.4. Generowanie przedziałów adresów IP na podstawie adresu CIDR	397
11.5. Tworzenie prostego interfejsu opartego na architekturze REST	399
11.6. Obsługa prostych zdalnych wywołań procedur za pomocą protokołu XML-RPC	403
11.7. Prosta komunikacja między interpreterami	405
11.8. Implementowanie zdalnych wywołań procedur	407
11.9. Proste uwierzytelnianie klientów	410
11.10. Dodawanie obsługi protokołu SSL do usług sieciowych	412
11.11. Przekazywanie deskryptora pliku gniazda między procesami	417
11.12. Operacje wejścia-wyjścia sterowane zdarzeniami	422
11.13. Wysyłanie i odbieranie dużych tablic	427
12. Współbieżność	429
12.1. Uruchamianie i zatrzymywanie wątków	429
12.2. Ustalanie, czy wątek rozpoczął pracę	432
12.3. Komunikowanie się między wątkami	434
12.4. Blokowanie sekcji krytycznej	439
12.5. Blokowanie z unikaniem zakleszczenia	441
12.6. Zapisywanie stanu wątku	445

12.7. Tworzenie puli wątków	446
12.8. Proste programowanie równoległe	449
12.9. Jak radzić sobie z mechanizmem GIL (i przestać się nim martwić)	453
12.10. Definiowanie zadań działających jak aktry	456
12.11. Przesyłanie komunikatów w modelu publikuj-subskrybuj	459
12.12. Używanie generatorów zamiast wątków	462
12.13. Odpytywanie wielu kolejek wątków	468
12.14. Uruchamianie procesu demona w systemie Unix	471
13. Skrypty narzędziowe i zarządzanie systemem	475
13.1. Przyjmowanie danych wejściowych skryptu za pomocą przekierowań, potoków lub plików wejściowych	475
13.2. Kończenie pracy programu wyświetleniem komunikatu o błędzie	476
13.3. Parsowanie opcji z wiersza poleceń	477
13.4. Prośba o podanie hasła w czasie wykonywania programu	479
13.5. Pobieranie rozmiarów terminala	480
13.6. Wywoływanie zewnętrznych poleceń i pobieranie danych wyjściowych	481
13.7. Kopiowanie lub przenoszenie plików i katalogów	482
13.8. Tworzenie i wypakowywanie archiwów	484
13.9. Wyszukiwanie plików na podstawie nazwy	485
13.10. Wczytywanie plików konfiguracyjnych	486
13.11. Dodawanie mechanizmu rejestrowania operacji do prostych skryptów	489
13.12. Dodawanie obsługi rejestrowania do bibliotek	491
13.13. Tworzenie stopera	493
13.14. Określanie limitów wykorzystania pamięci i procesora	494
13.15. Uruchamianie przeglądarki internetowej	495
14. Testowanie, debugowanie i wyjątki	497
14.1. Testowanie danych wyjściowych wysyłanych do strumienia stdout	497
14.2. Podstawianie obiektów w testach jednostkowych	498
14.3. Sprawdzanie wystąpienia wyjątków w testach jednostkowych	501
14.4. Zapisywanie danych wyjściowych testu w pliku	503
14.5. Pomijanie testów lub przewidywanie ich niepowodzenia	504
14.6. Obsługa wielu wyjątków	505
14.7. Przechwytywanie wszystkich wyjątków	507
14.8. Tworzenie niestandardowych wyjątków	508
14.9. Zgłaszanie wyjątku w odpowiedzi na wystąpienie innego wyjątku	510
14.10. Ponowne zgłaszanie ostatniego wyjątku	512
14.11. Wyświetlanie komunikatów ostrzegawczych	513
14.12. Debugowanie prostych awarii programu	514
14.13. Profilowanie i pomiar czasu pracy programów	516
14.14. Przyspieszanie działania programów	518

15. Rozszerzenia w języku C	525
15.1. Dostęp do kodu w języku C za pomocą modułu ctypes	526
15.2. Pisanie prostych modułów rozszerzeń w języku C	532
15.3. Pisanie funkcji rozszerzeń manipulujących tablicami	535
15.4. Zarządzanie nieprzejrzystymi wskaźnikami w modułach rozszerzeń w języku C	538
15.5. Definiowanie i eksportowanie interfejsów API języka C w modułach rozszerzeń	540
15.6. Wywoływanie kodu Pythona w kodzie w języku C	544
15.7. Zwalnianie blokady GIL w rozszerzeniach w języku C	548
15.8. Jednoczesne wykonywanie wątków z kodu w językach C i Python	549
15.9. Umieszczanie kodu w języku C w nakładkach opartych na narzędziu Swig	550
15.10. Używanie Cythona do tworzenia nakładek na istniejący kod w języku C	555
15.11. Używanie Cythona do pisania wydajnych operacji na tablicach	560
15.12. Przekształcanie wskaźnika do funkcji w jednostkę wywoływalną	564
15.13. Przekazywanie łańcuchów znaków zakończonych symbolem NULL do bibliotek języka C	565
15.14. Przekazywanie łańcuchów znaków Unicode do bibliotek języka C	569
15.15. Przekształcanie łańcuchów znaków z języka C na ich odpowiedniki z Pythona	573
15.16. Używanie łańcuchów znaków o nieznanym kodowaniu pobieranych z języka C	574
15.17. Przekazywanie nazw plików do rozszerzeń w języku C	577
15.18. Przekazywanie otwartych plików do rozszerzeń w języku C	578
15.19. Wczytywanie w języku C danych z obiektów podobnych do plików	579
15.20. Pobieranie obiektów iterowalnych w języku C	581
15.21. Diagnostowanie błędów segmentacji	582
 A Dalsza lektura	 585
 Skorowidz	 587