

Urszula Wiejak, Adrian Wojciechowski



PYTHON W ZADANIACH

PROGRAMOWANIE DLA MŁODZIEŻY

poziom podstawowy

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.

Autorzy oraz wydawca 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. Autorzy oraz wydawca nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Redaktor prowadzący: Małgorzata Kulik

Projekt okładki: Studio Gravite / Olsztyn
Obarek, Pokoński, Pazdrijowski, Zaprucki

Grafika na okładce została wykorzystana za zgodą Shutterstock.com

Helion S.A.

ul. Kościuszki 1c, 44-100 Gliwice

tel. 32 231 22 19, 32 230 98 63

e-mail: helion@helion.pl

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

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<https://helion.pl/user/opinie/prpypv>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-283-9196-3

Copyright © Urszula Wiejak 2020, 2022

Printed in Poland.

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

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

Spis treści

Wstęp	9
Ważne informacje	11
Instalacja potrzebnych elementów	13
Rozdział 1. Trochę o Pythonie.	
Operacje wejścia i wyjścia w konsoli	21
Co to jest programowanie i czym zajmują się programiści	21
Interpreter	22
Operatory	24
Napisy	25
Edytor programów Pythona	27
Operacje wyjścia	31
Operacje wejścia	34
Typy danych	35
Wypisywanie kilku argumentów dla funkcji print()	36

Konwersja typów danych	37
Metody string — upper() i lower()	39
Podsumowanie	40
Rozdział 2. Operatory porównania i instrukcja if	43
Operatory porównania	43
Wartości True i False	44
Warunki	46
Instrukcja warunkowa if	47
Podsumowanie	50
Rozdział 3. Zmienne	51
Zmienne liczbowe i zmiana wartości zmiennej	52
Zmienne napisowe	54
Zmienne dla wartości logicznych	55
Podsumowanie	56
Rozdział 4. Biblioteki	57
Instrukcja import	58
Biblioteka time	58
Funkcja sleep()	58
Funkcje gmtime() i typ struct_time	60
Funkcja time()	62
Funkcja strftime()	64
Podsumowanie	65
Rozdział 5. Instrukcja else i liczby losowe	67
Instrukcja else	67
Liczby losowe	70
Zagnieżdżanie warunków, czyli warunki w warunkach	72
Podsumowanie	74

Spis treści

Rozdział 6. Operatory logiczne i instrukcja else if	75
Operator logiczny AND	76
Operator logiczny OR	77
Operator logiczny NOT	79
Instrukcja elif	79
Podsumowanie	81
Rozdział 7. Debugowanie	83
Błędy	84
Podsumowanie	88
Rozdział 8. Biblioteka dźwiękowa pydub	89
Przyspieszanie dźwięków	94
Eksportowanie dźwięków	95
Podsumowanie	96
Rozdział 9. Pętla while i komentarze	97
Pętla while	97
Nieskończone pętle while	100
Instrukcja break	101
Komentarze	102
Podsumowanie	104
Rozdział 10. Zmienne tablicowe (listy)	105
Tablice jednowymiarowe	106
Metoda append() i funkcja len()	108
Mieszanie tablicy	109
Sortowanie tablicy	110
Algorytmy	111
Tablice dwuwymiarowe	113
Biblioteka progressbar	115
Podsumowanie	117

Rozdział 11. Biblioteka graficzna pygame	119
Biblioteka pygame	120
Okno graficzne	120
Funkcja pygame.draw.rect()	122
Funkcja circle()	124
Funkcja polygon()	125
Wypełnianie ekranu kolorem i tworzenie animacji ruchu	127
Wyświetlanie obrazów	130
Wyświetlanie tekstu	133
Podsumowanie	136
Rozdział 12. Pętla for i funkcje wbudowane Pythona	137
Pętla for	138
Pętla w pętli	140
Wbudowane funkcje Pythona	142
Podsumowanie	145
Rozdział 13. Pliki	147
Operacje na plikach	147
Tablica ASCII	151
Pisanie do pliku	153
Podsumowanie	156
Rozdział 14. Wyjątki i instrukcja with	157
Instrukcje try i except	158
Wyjątki	158
Instrukcja finally	160
Instrukcja with	161
Podsumowanie	162

Spis treści

Rozdział 15. Pliki ZIP	163
Biblioteka zipfile	164
Podsumowanie	167
Rozdział 16. Własne funkcje	169
Tworzenie własnych funkcji	169
Argumenty funkcji	171
Importowanie własnych plików	174
Zwracanie wartości przez funkcję	175
Podsumowanie	177
Rozdział 17. Klasy i obiekty	179
Obiekty	179
Klasy	180
Metody klas	182
Metoda <code>_init_()</code>	184
Biblioteka <code>math</code>	185
Pierwiastkowanie — funkcja <code>sqrt()</code>	186
Funkcja <code>sin()</code>	187
Podsumowanie	188
Rozdział 18. Wątki	191
Wątki	192
Pobieranie wartości z wątków	194
Podsumowanie	195
Rozdział 19. Biblioteka filmowa <code>moviepy</code>	197
Tworzenie gifa z filmu	198
Przycinanie filmu i eksportowanie	200
Zmiana właściwości filmu	202
Obracanie filmu	203
Podsumowanie	205

Rozdział 20. GUI	207
GUI — graficzny interfejs użytkownika	207
Prosty graficzny interfejs użytkownika EasyGUI	207
Biblioteka PyQt5	212
Widżety i dziedziczenie	213
Okna dialogowe	220
Podsumowanie	222
Rozdział 21. Projekt gry	223
Szablon projektu	224
Klasa Element	225
Klasa Board	226
Klasa Game	231
Uruchomienie gry	233
Zakończenie	235
Rozwiązania	237
Rozwiązania zadań	237
Rozwiązania zadań powtórzeniowych	269
Twoja lista wykonanych zadań	273
Zadania	273
Zadania powtórzeniowe	276

ROZDZIAŁ 8.

Biblioteka dźwiękowa pydub

Zadanie powtórzeniowe 6

Spójrz na poniższe programy i powiedz, jakie zawierają błędy.

```
File Edit Format Run Options Window Help
a = input("Podaj liczbę: ")
if a == 2
print("tekst")
```

Rysunek 8.1

```
File Edit Format Run Options Window Help
random.randint(1, 3.5)
if a == 2
print("oliwki")
```

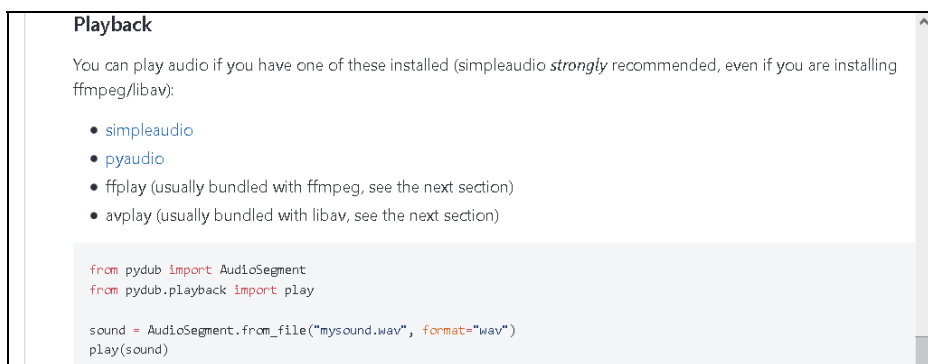
Rysunek 8.2

```
File Edit Format Run Options Window Help
print("xxx")
    print("yyy")
a = int(input("float: "))
```

Rysunek 8.3

Za pomocą biblioteki pydub będziemy odtwarzać, edytować i eksportować dźwięki. Aby z niej skorzystać, zaimportujemy różne jej moduły.

Obecnie (2020 rok) biblioteka pydub nie posiada pełnej dokumentacji. Aby zrozumieć działanie niektórych funkcji, należy zapoznać się z ich kodem w plikach biblioteki albo znaleźć odpowiednie tematy na forach w internecie. Zaczniemy od odtwarzania dźwięku. Działanie funkcji `play()` możemy sprawdzić na stronie GitHub autora biblioteki: <https://github.com/jiaaro/pydub>.



Rysunek 8.4

Dla przykładu prześledźmy, jak odtworzyć dźwięk `sound1.mp3`. Żeby program zadziałał, należy przenieść plik `sound1.mp3` do katalogu, w którym jest nasz program.

W bibliotece pydub możemy odtwarzać dźwięki ze zmiennych nazywanych segmentami dźwiękowymi. Do tworzenia tych segmentów posłużymy się modułem `AudioSegment` i znajdującą się w nim funkcją `from_file()`.

`AudioSegment` jest modułem biblioteki pydub. Żeby zaimportować coś z jakiegś biblioteki, piszemy:

```
from nazwa_biblioteki
```

Następnie, żeby wybrać konkretny moduł, piszemy:

```
import nazwa_modułu
```

Możemy też z wybranego modułu biblioteki importować wybrane funkcje. Żeby z pydub wybrać moduł `playback`, używamy kropki. Dlatego piszemy najpierw:

```
from nazwa_biblioteki.nazwa_modułu
```

Następnie importujemy wybraną funkcję. W naszym przypadku jest to funkcja `play()`.

```
import nazwa_funkcji
```

Zwróć uwagę na ostatni argument funkcji `from_file()` znajdującej się w module `AudioSegment`. W Pythonie nie trzeba podawać argumentów w określonej kolejności. Jeśli odwołamy się do nazwy argumentu, w tym przypadku argumentu `format`, oraz przypiszemy mu wartość, którą chcemy przekazać, to wartość zostanie przypisana właśnie temu argumentowi bez względu na jego miejsce w nawiasach.



```
File Edit Format Run Options Window Help
from pydub import AudioSegment
from pydub.playback import play
sound = AudioSegment.from_file("sound1.mp3", format="mp3")
play(sound)
```

Rysunek 8.5

Teraz na stronie <https://freesound.org> (wymagana rejestracja) możesz znaleźć dźwięki, na których będziesz wykonywać zadania.

Możliwe jest odtwarzanie dźwięków także w innych formatach, np. WAV, OGG.

Zadanie 52

Napisz program, który pobiera od użytkownika nazwę dźwięku do odtworzenia i odtwarza go.

Dodatkowo program sprawdza, czy w nazwie pliku znajduje się fragment napisu `".mp3"`.

Jeśli tak, to próbuje otworzyć plik w formacie MP3.

W przeciwnym wypadku, jeśli w nazwie znajduje się fragment napisu `".wav"`, to próbuje otworzyć plik w formacie WAV.

W przeciwnym wypadku informuje użytkownika, że nie rozpoznaje formatu pliku.

Zadanie 53

Za pomocą operatora + zwiększamy lub zmniejszamy głośność dźwięku. Dopisz do poprzedniego programu kod, który pozwala użytkownikowi wpisać liczbę decybeli, o jaką ma być zmieniony dźwięk, a następnie dodaj tę liczbę do zmiennej przechowującej dane o dźwięku zwrócone przez funkcję `from_file()`.

```
nazwa pliku: sound1.mp3
dB: 20
>>> |
```

Rysunek 8.6

Przycinanie elementów — napisów i dźwięków

Przejdźmy do konsoli Pythona. Przecwiczymy tam przycinanie napisów.

Napis składa się z połączonych ze sobą znaków. Żeby zostawić tylko początek napisu, po napisie wstawiamy w nawiasy kwadratowe dwukropek i liczbę znaków, które mają zostać.

```
>>> "KomputriloEdu"[:3]
'Kom'
>>>
```

Rysunek 8.7

Jeżeli dwukropek damy po liczbie, to wytniemy z napisu znaki od początku do liczby, którą mamy w nawiasie.

```
>>> "KomputriloEdu"[3:]
'putriloEdu'
>>> |
```

Rysunek 8.8

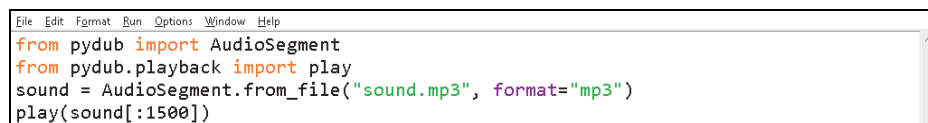
Możemy też wycinać litery od początku do numeru znaku podanego od końca. Wtedy wpisujemy liczbę z minusem. Spróbujmy dla przykładu wyświetlić ostatnie 3 litery.

```
>>> "komputriloEdu"[-3:]
'Edu'
>>> |
```

Rysunek 8.9

Tak samo będziemy przycinać dźwięki. Różnica będzie taka, że w zmiennej z dźwiękiem jest bardzo dużo danych. Te dane są podzielone na milisekundy. 1000 milisekund to jedna sekunda.

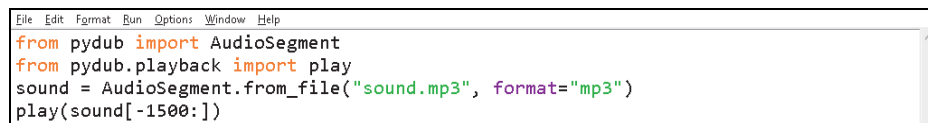
Dla przykładu odtworzmy pierwsze 1,5 sekundy dźwięku *sound.mp3*.



```
File Edit Format Run Options Window Help
from pydub import AudioSegment
from pydub.playback import play
sound = AudioSegment.from_file("sound.mp3", format="mp3")
play(sound[:1500])
```

Rysunek 8.10

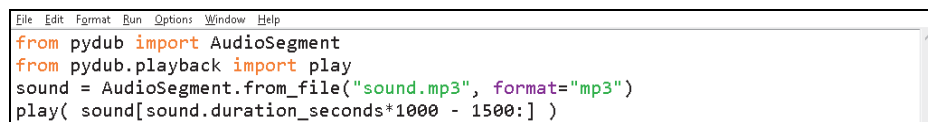
Możemy też odtworzyć ostatnie 1,5 sekundy dźwięku na dwa sposoby. Pierwszy sposób:



```
File Edit Format Run Options Window Help
from pydub import AudioSegment
from pydub.playback import play
sound = AudioSegment.from_file("sound.mp3", format="mp3")
play(sound[-1500:])
```

Rysunek 8.11

Drugi sposób na to, żeby zostały ostatnie 1,5 sekundy dźwięku, wymaga wycięcia do wartości równej długości dźwięku minus 1500 milisekund. Najpierw trzeba wiedzieć, jak długo trwa dźwięk. Możemy zrobić to za pomocą `duration_seconds`, które przechowuje długość dźwięku w sekundach. Po pomnożeniu przez 1000 otrzymujemy milisekundy.



```
File Edit Format Run Options Window Help
from pydub import AudioSegment
from pydub.playback import play
sound = AudioSegment.from_file("sound.mp3", format="mp3")
play( sound[sound.duration_seconds*1000 - 1500:] )
```

Rysunek 8.12

Zadanie 54

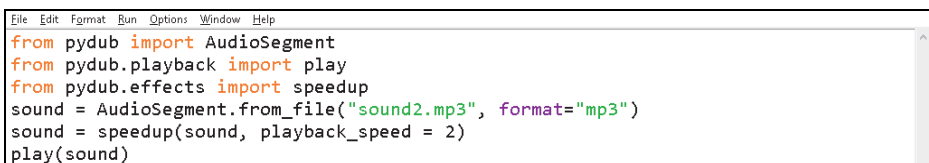
Napisz program, który pobiera od użytkownika nazwę dźwięku, odtwarza go w takim formacie jak ostatnie 3 znaki napisu oraz pobiera, od której milisekundy do której milisekundy ma być odtworzony dźwięk, i odtwarza go.

Przyspieszanie dźwięków

Żeby przyspieszać nagrania, musimy zaimportować funkcję `speedup`:

```
from pydub.effects import speedup
```

Funkcja `speedup` służy do zmiany prędkości odtwarzania dźwięku. Jako pierwszy argument podajemy dźwięk, a jako drugi argument — o nazwie `playback_speed` — wartość przyspieszenia, gdzie 1 to normalna prędkość, 0.5 to zwolnione dwukrotnie, a 2 to przyspieszone dwukrotnie. Przyspieszmy dwukrotnie dźwięk `sound2.mp3`. Przyspieszony dźwięk zapisujemy w zmiennej.



```
File Edit Format Run Options Window Help
from pydub import AudioSegment
from pydub.playback import play
from pydub.effects import speedup
sound = AudioSegment.from_file("sound2.mp3", format="mp3")
sound = speedup(sound, playback_speed = 2)
play(sound)
```

Rysunek 8.13

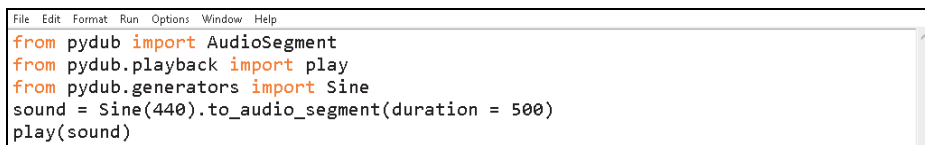
Zadanie 55

Zmodyfikuj program tak, żeby przyspieszał dźwięk o losową wartość od 1 do 15.

Generowanie tonu

Możemy za pomocą klasy `Sine` generować ton. Importujemy ją z `pydub.generators`. Przy tworzeniu tonu w nawiasach za `Sine` podajemy częstotliwość (człowiek słyszy od 20 Hz do 20 000 Hz), a następnie konwertujemy wygenerowany dźwięk na segment dźwiękowy funkcją `to_audio_segment()`. Jako argument o nazwie `duration` możemy podać długość dźwięku w milisekundach.

Wygenerujmy dźwięk A.



```
File Edit Format Run Options Window Help
from pydub import AudioSegment
from pydub.playback import play
from pydub.generators import Sine
sound = Sine(440).to_audio_segment(duration = 500)
play(sound)
```

Rysunek 8.14

Zadanie 56

Zmodyfikuj poprzedni program tak, żeby dowolną liczbę razy odtwarzał dźwięk o losowej częstotliwości od 20 Hz do 2000 Hz o długości 50 ms.

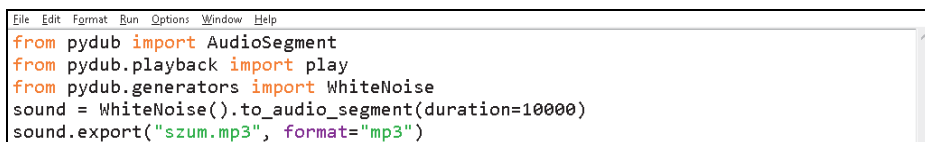
Zadanie 57

Za pomocą klasy `WhiteNoise` stworzymy biały szum. Biały szum to jednostajny, monotoniczny dźwięk. Usłyszysz, czym on jest, po jego wygenerowaniu.

Klasa `WhiteNoise` w momencie tworzenia obiektu nie pobiera argumentu z częstotliwością. Spróbuj sam wygenerować biały szum, tak jak generowaliśmy tony.

Eksportowanie dźwięków

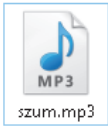
Spróbujmy wygenerować biały szum o długości 10 sekund i wyeksportować go jako plik. Żeby generować biały szum, musimy importować `WhiteNoise`. Eksportujemy dźwięk metodą `export`, która jako argument przyjmuje przede wszystkim nazwę pliku, oraz możemy podać rozszerzenie pliku, z jakim ma zostać on wyeksportowany.



```
File Edit Format Run Options Window Help
from pydub import AudioSegment
from pydub.playback import play
from pydub.generators import WhiteNoise
sound = WhiteNoise().to_audio_segment(duration=10000)
sound.export("szum.mp3", format="mp3")
```

Rysunek 8.15

Plik zostanie wyeksportowany do katalogu z programem. Odtwarzamy go.



Rysunek 8.16

Zadanie 58

Napisz program, który generuje ton o długości i wysokości takiej, o jaką poprosi użytkownik, i eksportuje z nazwą taką, o jaką poprosi użytkownik, i w formacie WAV.

```
Nazwa pliku: Szumek  
Hz: 880  
długość: 500  
>>> |
```

Rysunek 8.17

Podsumowanie

W rozdziale ósmym nauczyłeś się:

- dzielić tablice,
- korzystać z biblioteki pydub.

Odpowiedz na pytania:

- W jaki sposób możemy dzielić tablice?
- Do czego służy biblioteka pydub?

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

OPANUJ PYTHONA OD STRONY PRAKTYCZNEJ!

- **Poznaj konstrukcje języka Python**
- **Naucz się stosować je w praktyce**
- **Wykorzystaj standardowe biblioteki**
- **Rozwiąż 139 zadań i sprawdź swoje rozwiązania**

Python to bardzo popularny, wydajny i elastyczny język programowania ogólnego przeznaczenia, którego prosta, czytelna i zwięzła składnia zdobyła uznanie rzesz programistów na całym świecie. Duże możliwości, przenośność kodu, pisanie skryptów — wszystko to sprawia, że Python znajduje zastosowanie w najróżniejszych dziedzinach i wspiera rozwój wielu popularnych serwisów internetowych. Bez wątpienia warto się go nauczyć!

Jeśli chcesz poznać ten język, jesteś na dobrej drodze! Książka omawia najważniejsze elementy programowania w Pythonie. Co ważne, wiedza została usystematyzowana w formie konkretnych zadań do wykonania. Poznasz dzięki nim typy danych i składnię języka, dowiesz się, jak debugować programy, nauczysz się korzystać z funkcji oferowanych przez biblioteki i opanujesz podstawy tworzenia graficznych interfejsów użytkownika. To podręcznik przeznaczony dla młodzieży, ale doskonale sprawdzi się również w przypadku osób dorosłych, które rozpoczynają przygodę z programowaniem.

- **Instalowanie oraz konfigurowanie środowiska i niezbędnych narzędzi**
- **Podstawowe informacje na temat języka i programowania**
- **Typy danych, zmienne proste i tablicowe**
- **Operatory, instrukcje warunkowe oraz instrukcje pętli**
- **Operacje wejścia-wyjścia w konsoli i na plikach**
- **Biblioteki graficzne, dźwiękowe i wideo**
- **Korzystanie z funkcji wbudowanych i tworzenie własnych**
- **Programowanie obiektowe, obsługa wątków i wyjątków**
- **Tworzenie graficznych interfejsów użytkownika**
- **Zadania do samodzielnego wykonania i praktyczny projekt gry**

POZNAJ PYTHONA, ROZWIĄZUJ ZADANIA, ZOSTAŃ PROGRAMISTĄ!

  helion.pl  HELION SA ul. Kościuszki 1c 44-100 Gliwice tel.: 32 230 98 63 helion@helion.pl	<i>Sprawdź nasze szkolenia!</i>  SZKOLENIA AKADEMIA IT & BUSINESS HELIONSZKOLENIA.PL	KOD KORZYŚCI <i>Sięgnij po więcej!</i>   ISBN 978-83-283-9196-3  9 788328 391963 INFORMATYKA W NAJLEPSZYM WYDANIU Cena: 59,00 zł
--	--	--