Note of **Python Course** from "https://automatetheboringstuff.com/"

*__Python Download__ at https://python.org.

```
>where python
>python -m pip list
>python -m pip install …
>python -m pip list
```

Virtal env:

```
C:\Users\al\Scripts>cd .venv\Scripts
C:\Users\al\Scripts\.venv\Scripts>activate.bat
```

*__Mu editor__ download from https://codewith.mu.

*highly recommend Simon Willison, Python Software Foundation board member and co-creator of the **Django Web Framework**, for his writing about AI at https://simonwillison.net/about. You can watch his sober and illuminating **PyCon 2024 keynote** speech on LLMs at https://autbor.com/pycon2024keynote.

*asking people in a **web forum** such as Stack Overflow (https://stackoverflow.com) or the "learn programming" subreddit at https://reddit.com/r/learnprogramming.

*__Share your messa__ge / code to public from https://pastebin.com or https://gist.github.com.

*__If-else example__:

```
>>> print('Enter your name:')
>>> name = input('>')
>>> age = 33
>>> if name == 'Alice':
>>>     print('Hi, Alice.')
>>> elif age < 12:
>>>     print('You are not Alice, kiddo.')
>>> else:
>>>     print('You are neither Alice nor a little kid.')
```

*__Loop example:__

```
>>> print('Hello!')
>>> i = 0
>>> while i < 5:
>>>     print('On this iteration, i is set to ' + str(i))
>>>     i = i + 1
>>> print('Goodbye!')
>>> for i in range(0, 10, 2):
>>>     print(i)
```

Chapter 8 "pyperclip" copy and paste

```
>>> import pyperclip
>>> pyperclip.copy('Hello, world!')
>>> pyperclip.paste()
```

Chapter 9 "regexes" for pattern of text.

```
>>> import re
>>> phone_num_pattern_obj = re.compile(r'\d{3}-\d{3}-\d{4}')
>>> match_obj = phone_num_pattern_obj.search('My number is 415-555-4242.')
>>> match_obj.group()
'415-555-4242'
```

Chapter 10 "pathlib" path and read/write files example

```
>>> from pathlib import Path
>>> import os
>>> Path.cwd()
WindowsPath('C:/Users/Al/AppData/Local/Programs/Python/Python313')
>>> os.chdir('C:\\Windows\\System32')
>>> os.makedirs('C:\\delicious\\walnut\\waffles')
>>> bacon_file = open('bacon.txt', 'w', encoding='UTF-8')
>>> bacon_file.write('Hello, world!\n')
14
```

```
>>> bacon_file.close()
>>> bacon_file = open('bacon.txt', 'a', encoding='UTF-8')
>>> bacon_file.write('Bacon is not a vegetable.')
25
>>> bacon_file.close()
>>> bacon_file = open('bacon.txt', encoding='UTF-8')
>>> content = bacon_file.read()
>>> bacon_file.close()
>>> print(content)
Hello, world!
Bacon is not a vegetable.
file_obj = open('data.txt', encoding='utf-8')
content = file_obj.read()
file_obj.close()
```

“Shelve” lets to restore that data to the variables the next time it is run

```
>>> import shelve
>>> shelf_file = shelve.open('mydata')

>>> shelf_file['cats'] = ['Zophie', 'Pooka', 'Simon']
>>> shelf_file.close()
```

Chapter 11

“shutil” has functions to let you copy, move, rename, and delete files

```
>>> import shutil
>>> from pathlib import Path
>>> h = Path.home()❶ >>> shutil.copy(h / 'spam/file1.txt', h)
'C:\\Users\\Al\\file1.txt'❷ >>> shutil.copy(h / 'spam/file1.txt', h / 'spam/file2.txt')
WindowsPath('C:/Users/Al/spam/file2.txt')
```

“send2trash” safer than Python’s regular delete functions

```
>>> import send2trash
>>> send2trash.send2trash('file1.txt')
```

ZipFile to compress files

```
>>> import zipfile
>>> with open('file1.txt', 'w', encoding='utf-8') as file_obj:
...     file_obj.write('Hello' * 10000)
...
>>> with zipfile.ZipFile('example.zip', 'w') as example_zip:
...     example_zip.write('file1.txt', compress_type=zipfile.ZIP_DEFLATED, compresslevel=9)
```

Chapter 12

12.1 Colorful Text with “Bext” object

```
>>> import bext
>>> bext.fg('red')>>> print('This text is red.')
import os
def clear():
   os.system('cls' if os.name == 'nt' else 'clear')
```

“playsound3” to play a simple audio file

```
>>> import playsound3
>>> playsound3.playsound('hello.mp3')
```

12.2 “PyInstaller” to compile a Python script (e.g. named yourScript.py)

```
C:\Users\al>python -m PyInstaller --onefile yourScript.py
378 INFO: PyInstaller: X.X.X     378 INFO: Python: 3.XX.XX
392 INFO: Platform: Windows-XX-XX.X.XXXX
393 INFO: wrote C:\Users\al\Desktop\hello-test\hello.spec
399 INFO: UPX is not available.--snip--
11940 INFO: Appending PKG archive to EXE
11950 INFO: Fixing EXE headers
13622 INFO: Building EXE from EXE-00.toc completed successfully.
```

Chapter 13

13.1 Web browsing using “Import requests, os, bs4, time” to download page & image.

```
>>> import webbrowser
>>> webbrowser.open('https://inventwithpython.com/')
```

13.2 requests.get() function takes a string representing a URL to download

```
>>> import requests
>>> response = requests.get('https://automatetheboringstuff.com/files/rj.txt')
>>> type(response)
<class 'requests.models.Response'>
>>> response.status_code == requests.codes.ok
True
>>> len(response.text)
178978
>>> print(response.text[:210])
The Project Gutenberg EBook of Romeo and Juliet, by William Shakespeare

This eBook is for the use of anyone anywhere at no cost and with
almost no restrictions whatsoever.  You may copy it, give it away or
```

“requests” to download file from url

```
>>> import requests
>>> response = requests.get('https://automatetheboringstuff.com/files/rj.txt')
>>> response.raise_for_status()
>>> with open('RomeoAndJuliet.txt', 'wb') as play_file:
...     for chunk in response.iter_content(100000):
...         play_file.write(chunk)
...
100000
78978
```

13.3 to operate an API

```
>>> import requests
>>> city_name = 'San Francisco'
>>> state_code = 'CA'
>>> country_code = 'US'
>>> API_key = '30ee784a80d81480dab1749d33980112'  # Not a real API key
>>> response =
requests.get(f'https://api.openweathermap.org/geo/1.0/direct?q={city_name},{state_code},{country
_code}&appid={API_key}')
>>> response.text  # This is a Python string.
'[{"name":"San Francisco","local_names":{"id":"San Francisco",--snip--
,"lat":37.7790262,"lon":-122.419906,"country":"US","state":"California"}]'
>>> import json
>>> response_data = json.loads(response.text)
>>> response_data  # This is a Python data structure.
[{"name":"San Francisco","local_names":{"id":"San Francisco",--snip--
,"lat":37.7790262,"lon":-122.419906,"country":"US","state":"California"}]
```

13.4 Beautiful Soup is a package for extracting information from an HTML page

```
>>> import requests, bs4
>>> res = requests.get('https://autbor.com/example3.html')
>>> res.raise_for_status()
>>> example_soup = bs4.BeautifulSoup(res.text, 'html.parser')
>>> type(example_soup)
<class 'bs4.BeautifulSoup'>
```

13.5 Selenium to launch the Firefox browser and operate its interactive shell:

```
>>> from selenium import webdriver
>>> browser = webdriver.Firefox()
>>> type(browser)
<class 'selenium.webdriver.firefox.webdriver.WebDriver'>
>>> browser.get('https://inventwithpython.com')
```

Chapter 14

14.1 openpyxl to operate .xlsx files

```
>>> import openpyxl
>>> wb = openpyxl.load_workbook('example3.xlsx')
>>> sheet = wb['Sheet1']  # Get a sheet from the workbook.
>>> sheet['A1']  # Get a cell from the sheet.
<Cell 'Sheet1'.A1>
>>> sheet['A1'].value  # Get the value from the cell.
datetime.datetime(2035, 4, 5, 13, 34, 2)
>>> c = sheet['B1']  # Get another cell from the sheet.
>>> c.value
'Apples'
>>>  # Get the row, column, and value from the cell.
>>> f'Row {c.row}, Column {c.column} is {c.value}'
'Row 1, Column 2 is Apples'
>>> f'Cell {c.coordinate} is {c.value}'
'Cell B1 is Apples'
>>> sheet['C1'].value
73
```

14.2 Write the data structure to a text file with the readCensusExcel.py extension using pprint

```
import openpyxl, pprint
print('Opening workbook...')
wb = openpyxl.load_workbook('censuspopdata.xlsx')
sheet = wb['Population by Census Tract']
county_data = {}
# TODO: Fill in county_data with each county's population and tracts.
print('Reading rows...')
for row in range(2, sheet.max_row + 1):
    # Each row in the spreadsheet has data for one census tract.
    state  = sheet['B' + str(row)].value
    county = sheet['C' + str(row)].value
    pop    = sheet['D' + str(row)].value
    # Make sure the key for this state exists.
    county_data.setdefault(state, {})
    # Make sure the key for this county in this state exists.
    county_data[state].setdefault(county, {'tracts': 0, 'pop': 0})
    # Each row represents one census tract, so increment by one.
    county_data[state][county]['tracts'] += 1
    # Increase the county pop by the pop in this census tract.
    county_data[state][county]['pop'] += int(pop)
# Open a new text file and write the contents of county_data to it.
print('Writing results...')
result_file = open('census2010.py', 'w')
result_file.write('allData = ' + pprint.pformat(county_data))
result_file.close()
print('Done.')
>>> import census2010
>>> census2010.allData['AK']['Anchorage']
{'pop': 291826, 'tracts': 55}
>>> anchorage_pop = census2010.allData['AK']['Anchorage']['pop']
```

```
>>> print('The 2010 population of Anchorage was ' + str(anchorage_pop))
The 2010 population of Anchorage was 291826
```

Chapter 15 EZSheets to access and edit your Google Sheets spreadsheets

In your browser, go to https://console.cloud.google.com and sign in to your Google account with your username and password. You will be taken to a Getting Started page.

```
>>> import ezsheets
>>> ss = ezsheets.Spreadsheet()
>>> ss.title = 'Title of My New Spreadsheet'
>>> ss.title
'Title of My New Spreadsheet'
>>> ss.url
'https://docs.google.com/spreadsheets/d/1gxz-Qr2-RNtqi_d7wWlsDlbtPLRQigcEXvCtdVwmH40/'
>>> ss.id
'1gxz-Qr2-RNtqi_d7wWlsDlbtPLRQigcEXvCtdVwmH40'
```

Chapter 16 sqlite3 working with other SQL databases, create index, drop table, backup database, etc.

```
>>> import sqlite3
>>> sqlite3.sqlite_version
'3.xx.xx'
>>> conn = sqlite3.connect('example.db', isolation_level=None)
>>> conn.execute('CREATE TABLE IF NOT EXISTS cats (name TEXT NOT NULL,birthdate TEXT, fur TEXT,
weight_kg REAL) STRICT')
<sqlite3.Cursor object at 0x00000201B2399540>
>>> conn = sqlite3.connect('example.db', isolation_level=None)
>>> conn.execute('SELECT name FROM sqlite_schema WHERE type="table"').fetchall()
>>> import sqlite3, pprint
>>> conn = sqlite3.connect('sweigartcats.db', isolation_level=None)
>>> pprint.pprint(conn.execute('SELECT * FROM cats ORDER BY fur').fetchall())
[('Iris', '2017-07-13', 'bengal', 6.8),
 ('Ruby', '2023-12-22', 'bengal', 5.0),
```

Chapter 17 PyPDF extract the images from a PDF document, edit it, add watermark, encrypt

```
>>> import pypdf
>>> reader = pypdf.PdfReader('Recursion_Chapter1.pdf')
>>> len(reader.pages)
18
import pypdf
PDF_FILENAME = 'Recursion_Chapter1.pdf'
reader = pypdf.PdfReader(PDF_FILENAME)
image_num = 0
for i, page in enumerate(reader.pages):
    print(f'Reading page {i+1} - {len(page.images)} images found...')
    try:
      for image in page.images:
          with open(f'{image_num}_page{i+1}_{image.name}', 'wb') as file:
              file.write(image.data)
          print(f'Wrote {image_num}_page{i+1}_{image.name}...')
          image_num += 1
    except Exception as exc:
      print(f'Skipped page {i+1} due to error: {exc}')
```

install Python-Docx (not docx) for create and modify Microsoft Word documents

```
import docx
def get_text(filename):
    doc = docx.Document(filename)
    full_text = []
    for para in doc.paragraphs:
        full_text.append(para.text)
    return '\n'.join(full_text)
```

Chapter 18

18.1 Csv extension

```
>>> import csv
>>> example_file = open('example3.csv')
>>> example_reader = csv.reader(example_file)
>>> example_data = list(example_reader)
>>> example_data
[['4/5/2035 13:34', 'Apples', '73'], ['4/5/2035 3:41', 'Cherries', '85'],
['4/6/2035 12:46', 'Pears', '14'], ['4/8/2035 8:59', 'Oranges', '52'],
['4/10/2035 2:07', 'Apples', '152'], ['4/10/2035 18:10', 'Bananas', '23'],
['4/10/2035 2:40', 'Strawberries', '98']]
>>> example_file.close()
```

18.2 Json

```
>>> import json
>>> json_string = '{"name": "Alice Doe", "age": 30, "car": null, "programmer": true, "address":
{"street": "100 Larkin St.", "city": "San Francisco", "zip": "94102"}, "phone": [{"type": "mobile",
"number": "415-555-7890"}, {"type": "work", "number": "415-555-1234"}]}'
>>> python_data = json.loads(json_string)
>>> python_data
{'name': 'Alice Doe', 'age': 30, 'car': None, 'programmer': True, 'address':
{'street': '100 Larkin St.', 'city': 'San Francisco', 'zip': '94102'},
'phone': [{'type': 'mobile', 'number': '415-555-7890'}, {'type': 'work',
'number': '415-555-1234'}]}
>>> import json
>>> python_data = {'name': 'Alice Doe', 'age': 30, 'car': None, 'programmer': True, 'address':{'street':
'100 Larkin St.', 'city': 'San Francisco', 'zip': '94102'}, 'phone': [{'type': 'mobile', 'number': '415-555-
7890'}, {'type': 'work', 'number': '415-555-1234'}]}
>>> json_string = json.dumps(python_data)
>>> print(json_string)
{"name": "Alice Doe", "age": 30, "car": null, "programmer": true, "address": {"street":
"100 Larkin St.", "city": "San Francisco", "zip": "94102"}, "phone": [{"type": "mobile",
"number": "415-555-7890"}, {"type": "work", "number": "415-555-1234"}]}
>>> json_string = json.dumps(python_data, indent=2)
>>> print(json_string)
{
  "name": "Alice Doe",
  "age": 30,
  "car": null,
  "programmer": true,
  "address": {
    "street": "100 Larkin St.",
    "city": "San Francisco",--snip--
}
```

18.3 Xml

```
>>> import xml.etree.ElementTree as ET
>>> xml_string = """<person><name>Alice
Doe</name><age>30</age><programmer>true</programmer><car xsi:nil="true"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance”/><address><street>100 Larkin
St.</street><city>San
Francisco</city><zip>94102</zip></address><phone><phoneEntry><type>mobile</type><number>4
15-555-7890</number></phoneEntry><phoneEntry><type>work</type><number>415-555-
1234</number></phoneEntry></phone></person>"""
>>> root = ET.fromstring(xml_string)
>>> root
<Element 'person' at 0x000001942999BBA0>
```

Chapter 19

Time object (import time)
Datetime object (import datetime)
run() function in the built-in subprocess

```
>>> import subprocess
>>> subprocess.run(['C:\\Windows\\System32\\calc.exe'])
CompletedProcess(args=['C:\\Windows\\System32\\calc.exe'], returncode=0)
>>> file_obj = open('hello.txt', 'w')  # Create a hello.txt file.
>>> file_obj.write('Hello, world!')
13
>>> file_obj.close()
>>> import subprocess
>>> subprocess.run(['start', 'hello.txt'], shell=True)
```

Chapter 20 Gmail API

Because of Gmail's additional security and anti-spam measures, controlling a Gmail account is better done through the EZGmail module than through the smtplib and imaplib modules in Python's standard library.
set up the Gmail API for your account through the Google Cloud console at https://console.cloud.google.com. (refer to Chapter 15)

```
>>> import ezgmail
>>> ezgmail.init()
>>> ezgmail.EMAIL_ADDRESS
'example@gmail.com'
>>> import ezgmail
>>> ezgmail.send('recipient@example.com', 'Subject line', 'Body of the email',
cc='friend@example.com', bcc='otherfriend@example.com,someoneelse@example.com')
>>> unread_threads = ezgmail.unread()  # List of GmailThread objects
>>> ezgmail.summary(unread_threads)
Al, Jon - Do you want to watch RoboCop this weekend? - Dec 09
Jon - Thanks for stopping me from buying Bitcoin. - Dec 09
```

Sending texts via an SMS gateway at https://ntfy.sh. This app is free and can be found in the app stores for Android and iPhone. You can also receive notifications in your web browser by going to https://ntfy.sh/app. ("requests" module refer to Chapter 13)

```
>>> import requests
>>> requests.post('https://ntfy.sh/AlSweigartZPgxBQ42', 'Hello, world!')
<Response [200]>
```

Chapter 21

21.1 work with colors and edit image file in Pillow

```
>>> from PIL import ImageColor
>>> ImageColor.getcolor('red', 'RGBA')
(255, 0, 0, 255)
>>> from PIL import Image
>>> im = Image.new('RGBA', (100, 200), 'purple')
>>> im.save('purpleImage.png')
>>> im2 = Image.new('RGBA', (20, 20))
>>> im2.save('transparentImage.png')
>>> cat_im = Image.open('zophie.png')
>>> cropped_im = cat_im.crop((335, 345, 565, 560))
>>> cropped_im.save('cropped.png')
```

21.2 draw lines, rectangles, circles, or other simple shapes on an image, use Pillow's ImageDraw

```
>>> from PIL import Image, ImageDraw
>>> im = Image.new('RGBA', (200, 200), 'white')
>>> draw = ImageDraw.Draw(im)
>>> draw.line([(0, 0), (199, 0), (199, 199), (0, 199), (0, 0)], fill='black')
>>> draw.rectangle((20, 30, 60, 60), fill='blue')
>>> draw.ellipse((120, 30, 160, 60), fill='red')
```

```
>>> draw.polygon(((57, 87), (79, 62), (94, 85), (120, 90), (103, 113)), fill='brown')
>>> for i in range(100, 200, 10):
...     draw.line([(i, 0), (200, i - 100)], fill='green')
>>> im.save('drawing.png')
```

21.3 use text() to draw text onto an image

```
>>> from PIL import Image, ImageDraw, ImageFont
>>> import os
>>> im = Image.new('RGBA', (200, 200), 'white')
>>> draw = ImageDraw.Draw(im)
>>> draw.text((20, 150), 'Hello', fill='purple')
>>> arial_font = ImageFont.truetype('arial.ttf', 32)
>>> draw.text((100, 150), 'Howdy', fill='gray', font=arial_font)
>>> im.save('text.png')
```

21.4 create line/bar graphs, scatter plots, pie charts and 3D graph with Matplotlib

```
>>> import matplotlib.pyplot as plt
>>> x_values = [0, 1, 2, 3, 4, 5]
>>> y_values1 = [10, 13, 15, 18, 16, 20]
>>> y_values2 = [9, 11, 18, 16, 17, 19]
>>> plt.plot(x_values, y_values1)
[<matplotlib.lines.Line2D object at 0x000002501D9A7D10>]
>>> plt.plot(x_values, y_values2)
[<matplotlib.lines.Line2D object at 0x00000250212AC6D0>]
>>> plt.savefig('linegraph.png')  # Saves the plot as an image file
>>> plt.show()  # Opens a window with the plot
>>> plt.show()  # Does nothing
```

Chapter 22

22.1 Tesseract OCR engine (PyTesseract)

```
>>> import pytesseract as tess
>>> from PIL import Image
>>> img = Image.open('ocr-example.png')
>>> text = tess.image_to_string(img)
>>> print(text)
This book provides you with practice examples of how programming
concepts are applied, with a collection of over 80 games, simulations, and dig-
ital art programs. These aren't code snippets; they're full, runnable Python
programs. You can copy their code to become familiar with how they work,
experiment with your own changes, and then attempt to re-create them on
your own as practice. After a while, you'll start to get ideas for your own pro-
grams and, more importantly, know how to go about creating them.--snip--
```

22.2 NAPS2 can combine several images into a PDF file with embedded text without being connected to a physical scanner

```
>>> import subprocess
>>> naps2_path = [r'C:\Program Files\NAPS2\NAPS2.Console.exe']  # Windows
>>> proc = subprocess.run(naps2_path + ['-i', 'frankenstein.png', '-o', 'output.pdf', '--install', 'ocr-eng',
'--ocrlang', 'eng', '-n', '0', '-f', '-v'], capture_output=True)
```

22.3 Usage of ChatGPT to fine tune the OCR result:

The following is the text output from an OCR scan. Correct any spacing, missing characters, or inaccurately recognized characters from it. Do not correct spelling or grammar mistakes that exist in the original text. Put paragraphs on a single line and undo the hyphenated words that are broken across the end of a line. Only give the corrected text without explaining what OCR is or any other preface. Here is the text: …

Chapter 23

23.1 move the mouse and track its position on the screen using PyAutoGUI (PyAutoGUI's window features work only on Windows, not on macOS or Linux.)

```
>>> import pyautogui
>>> screen_size = pyautogui.size()  # Obtain the screen resolution.
```

```
>>> screen_size
Size(width=1920, height=1080)
>>> screen_size[0], screen_size[1]
(1920, 1080)
>>> import pyautogui
>>> for i in range(10):  # Move the mouse in a square.
...     pyautogui.moveTo(100, 100, duration=0.25)
...     pyautogui.moveTo(200, 100, duration=0.25)
...     pyautogui.moveTo(200, 200, duration=0.25)
...     pyautogui.moveTo(100, 200, duration=0.25)
...
>>> pyautogui.click(10, 5)  # Move the mouse to (10, 5) and click.
>>> pyautogui.mouseInfo()   #into the interactive shell
>>> im = pyautogui.screenshot()
>>> import pyautogui
>>> box = pyautogui.locateOnScreen('submit.png')
>>> box
Box(left=643, top=745, width=70, height=29)
>>> box[0]
643
>>> box.left
643
>>> list(pyautogui.locateAllOnScreen('submit.png'))
[(643, 745, 70, 29), (1007, 801, 70, 29)]
>>> pyautogui.click('submit.png')
>>> pyautogui.click(100, 200); pyautogui.write('Hello, world!')
>>> pyautogui.write(['a', 'b', 'left', 'left', 'X', 'Y'])
```

Chapter 24

24.1 Usage of “pyttsx3” text to speech

```
>>> import pyttsx3
>>> engine = pyttsx3.init()
>>> engine.save_to_file('Hello. How are you doing?', 'hello.wav')
>>> engine.say('Hello. How are you doing?')
>>> engine.runAndWait()  # The computer speaks.
```

24.2 Usage of “Whisper” speech to text (only for python 3.10)

Start window cmd -> cd C:\Users\user\Documents\wa-doc-2022\Wealth\Job_2022\python-automatetheboringstuff.com

```
> py -3.10 -m venv whisperenv
> whisperenv\Scripts\activate
> pip install openai-whisper
> Python
>>> import whisper
>>> model = whisper.load_model('base')
>>> result = model.transcribe('hello.wav')
>>> print(result['text'])
```

Error 1 on “model = whisper.load_model('base')“

Solution: whisper for python <= 3.10, so create virtual env with python 3.10

Error 2: Whisper requires ffmpeg -> RuntimeError: Failed to load audio: ffmpeg not found

Solution: > winget install ffmpeg, add it to PATH: Start → type Environment Variables, then restart

Error 3: FileNotFoundError: [Errno 2] No such file or directory: 'audio.mp3'

Solution: model.transcribe(r"C:\Users\user\Desktop\audio.mp3")

24.3 Usage of “yt-dlp” download youtube video

```
>>> import yt_dlp
>>> video_url = 'https://www.youtube.com/watch?v=kSrnLbioN6w'
>>> with yt_dlp.YoutubeDL() as ydl:
...     ydl.download([video_url])
...
```

```
>>> import yt_dlp
>>> video_url = 'https://www.youtube.com/watch?v=kSrnLbioN6w'
>>> options = {
...     'quiet': True,  # Suppress the output.
...     'no_warnings': True,  # Suppress warnings.
...     'outtmpl': 'downloaded_content.%(ext)s',
...     'format': 'm4a/bestaudio/best',
...     'postprocessors': [{  # Extract audio using ffmpeg.
...         'key': 'FFmpegExtractAudio',
...         'preferredcodec': 'm4a',
...         }]
...     }
...
>>> with yt_dlp.YoutubeDL(options) as ydl:
...     ydl.download([video_url])
...
>>> from pathlib import Path
>>> matching_filenames = list(Path().glob('downloaded_content.*'))
>>> downloaded_filename = str(matching_filenames[0])
>>> downloaded_filename
'downloaded_content.m4a'
```

Appendix: Necessary sample data file can be download from https://automatetheboringstuff.com/