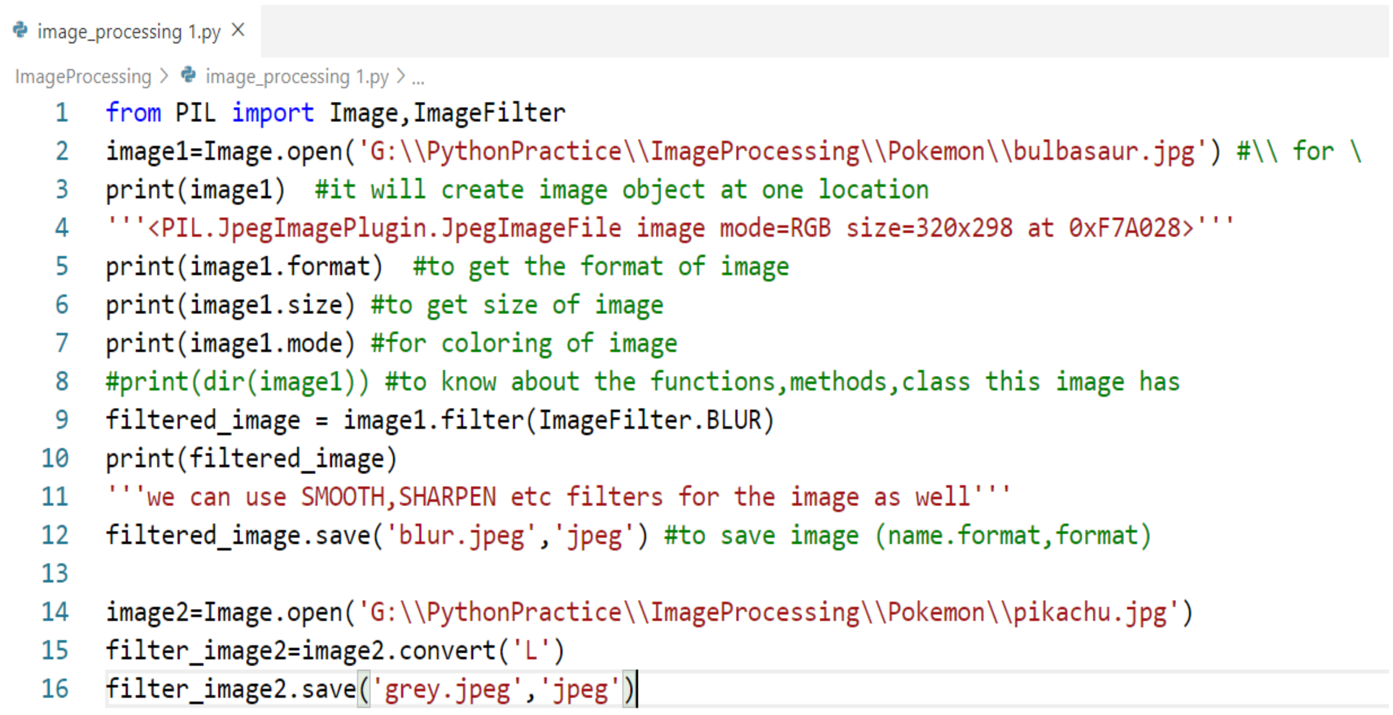
SCRIPTING WITH PYTHON

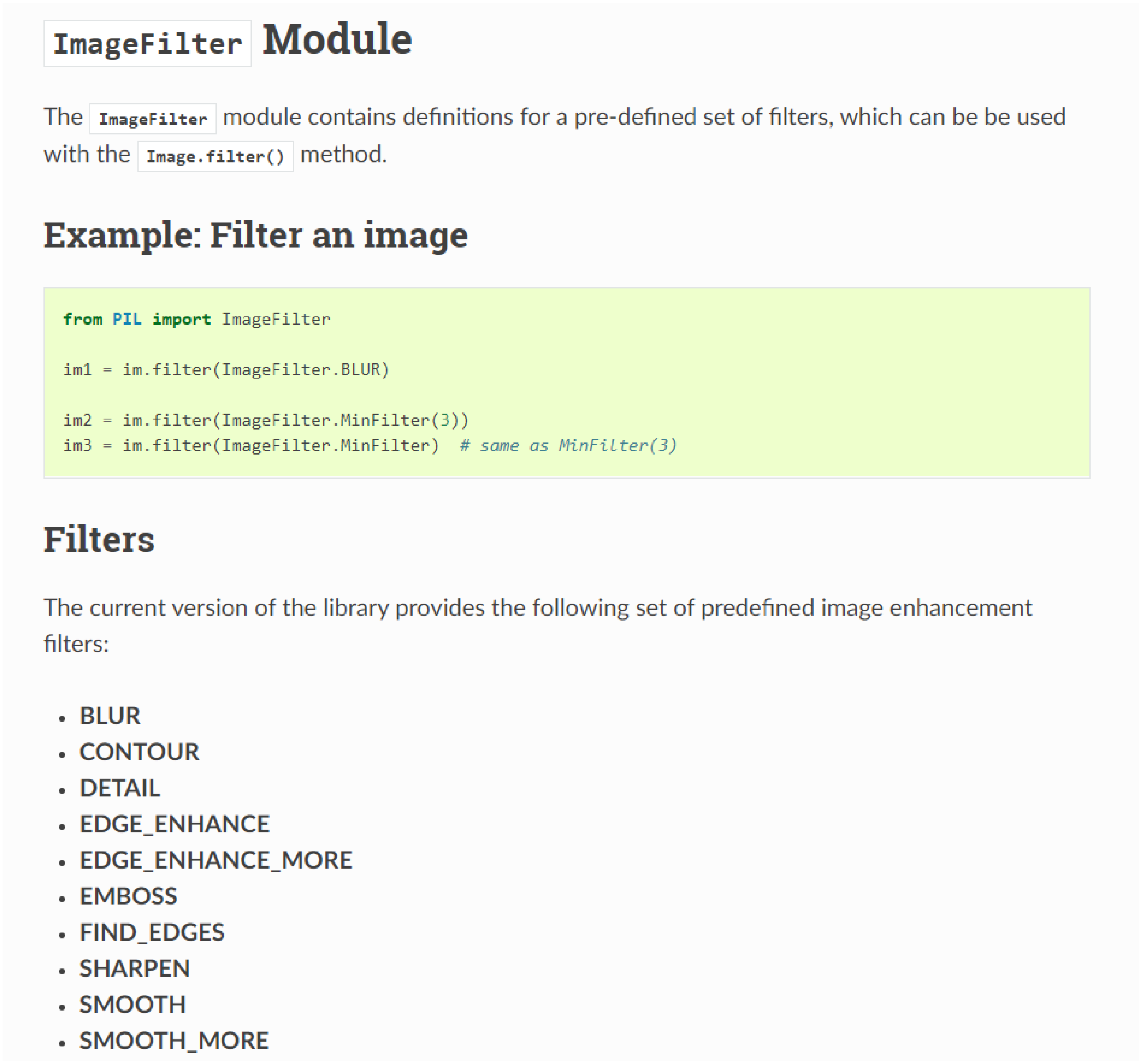
IMAGE PROCESSING

PILLOW LIBRARY

- Pillow Library from PyPI can be used for image processing.
- Link: **https://pypi.org/project/Pillow/**
- You should use libraries which are up to date.



```python
from PIL import Image,ImageFilter
image1=Image.open('G:\\PythonPractice\\ImageProcessing\\Pokemon\\bulbasaur.jpg') #\\ for \
print(image1)  #it will create image object at one location
'''<PIL.JpegImagePlugin.JpegImageFile image mode=RGB size=320x298 at 0xF7A028>'''
print(image1.format)  #to get the format of image
print(image1.size) #to get size of image
print(image1.mode) #for coloring of image
#print(dir(image1)) #to know about the functions,methods,class this image has
filtered_image = image1.filter(ImageFilter.BLUR)
print(filtered_image)
'''we can use SMOOTH,SHARPEN etc filters for the image as well'''
filtered_image.save('blur.jpeg','jpeg') #to save image (name.format,format)

image2=Image.open('G:\\PythonPractice\\ImageProcessing\\Pokemon\\pikachu.jpg')
filter_image2=image2.convert('L')
filter_image2.save('grey.jpeg','jpeg')
```



`ImageFilter` Module

The `ImageFilter` module contains definitions for a pre-defined set of filters, which can be be used with the `Image.filter()` method.

Example: Filter an image

```
from PIL import ImageFilter

im1 = im.filter(ImageFilter.BLUR)

im2 = im.filter(ImageFilter.MinFilter(3))
im3 = im.filter(ImageFilter.MinFilter)  # same as MinFilter(3)
```

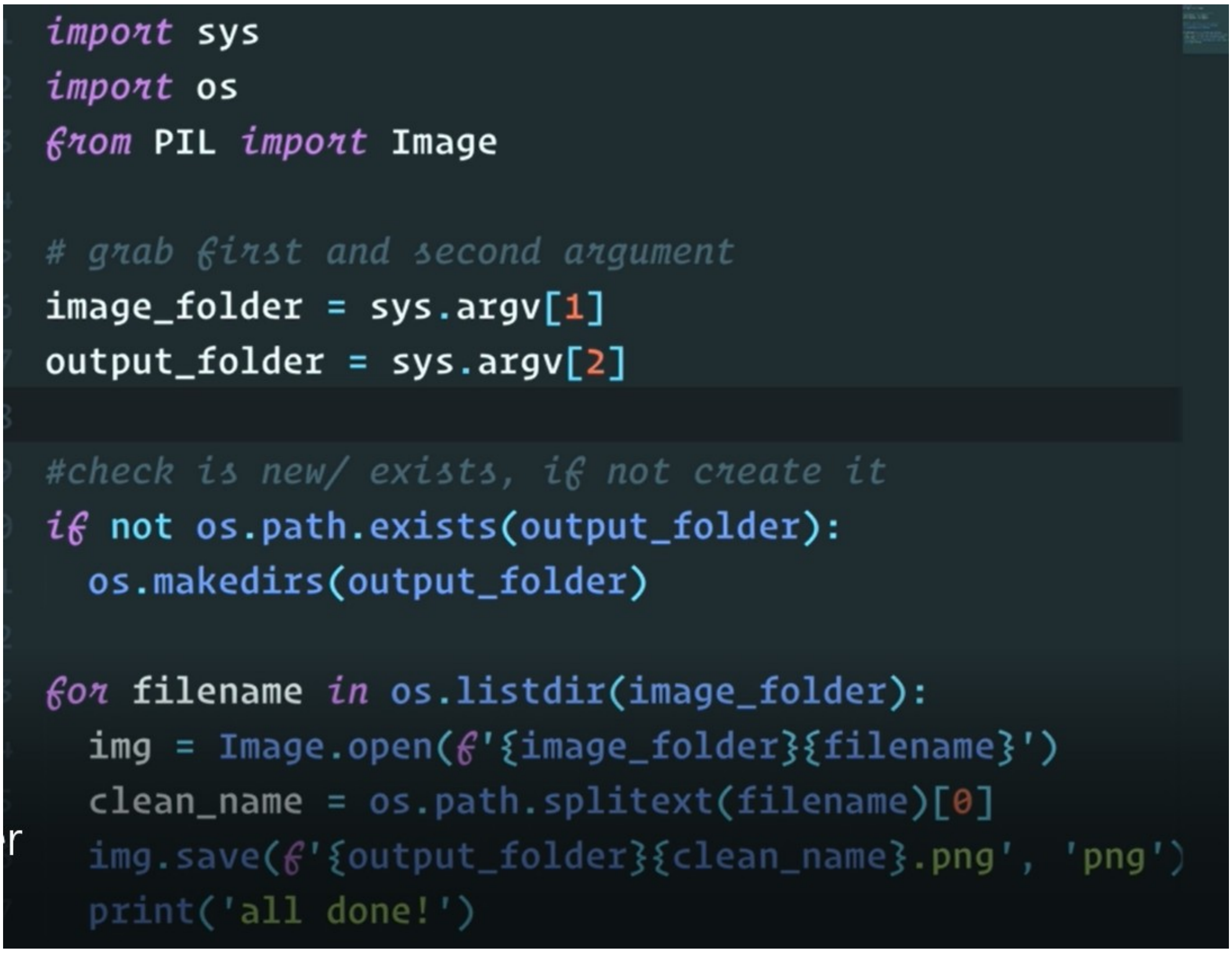
Filters

The current version of the library provides the following set of predefined image enhancement filters:

- BLUR
- CONTOUR
- DETAIL
- EDGE_ENHANCE
- EDGE_ENHANCE_MORE
- EMBOSS
- FIND_EDGES
- SHARPEN
- SMOOTH
- SMOOTH_MORE

Read Here: https://pillow.readthedocs.io/en/stable/reference/ImageFilter.html

Code Exercise: JPEG to PNG Converter



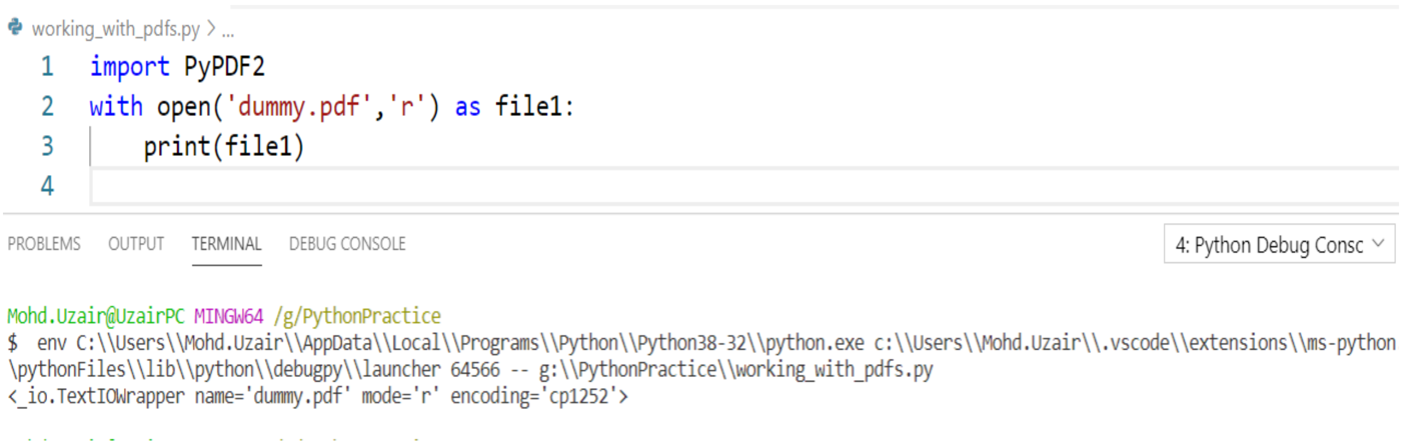
```python
import sys
import os
from PIL import Image

# grab first and second argument
image_folder = sys.argv[1]
output_folder = sys.argv[2]

#check is new/ exists, if not create it
if not os.path.exists(output_folder):
  os.makedirs(output_folder)

for filename in os.listdir(image_folder):
  img = Image.open(f'{image_folder}{filename}')
  clean_name = os.path.splitext(filename)[0]
  img.save(f'{output_folder}{clean_name}.png', 'png')
  print('all done!')
```

OpenCV LIBRARY

- OpenCV used in machine learning

PDFs WITH PYTHON

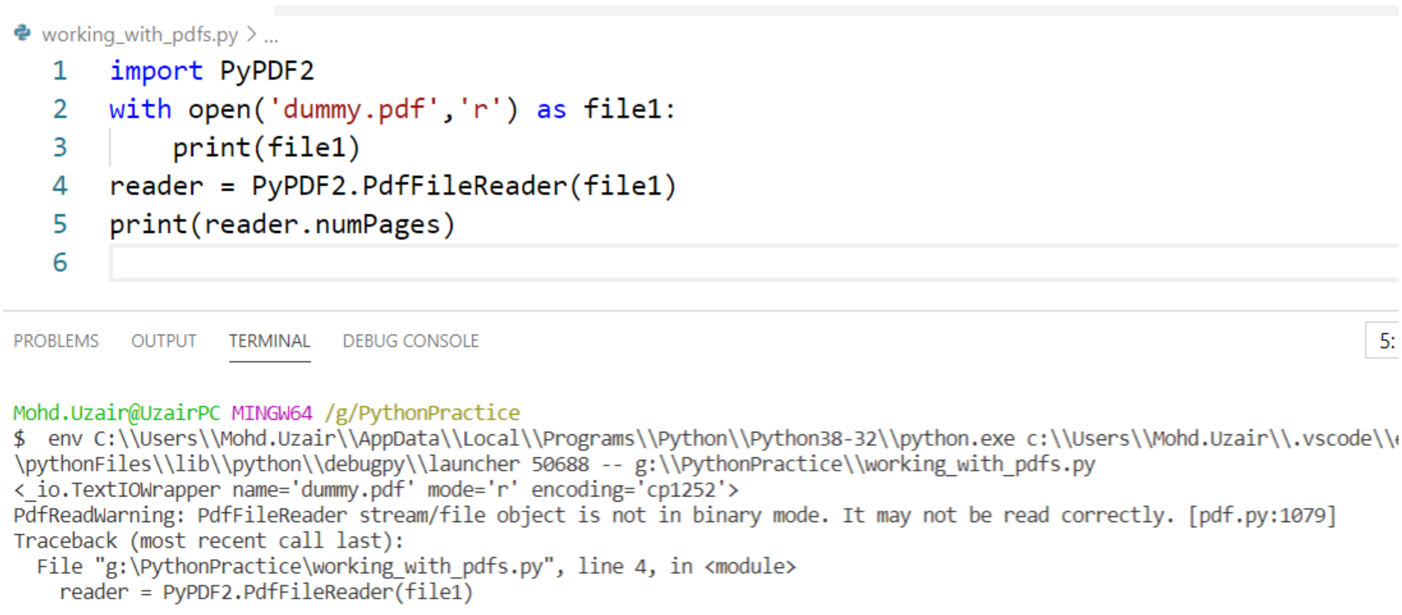
- We can use PyPDF2 package for working with PDFs.
- A Pure-Python library built as a PDF toolkit. It is capable of:
  - document information (title, author, …)
  - splitting documents page by page
  - merging documents page by page
  - cropping pages
  - merging multiple pages into a single page
  - encrypting and decrypting PDF files
- Link: https://pythonhosted.org/PyPDF2/


working_with_pdfs.py > ...

```python
import PyPDF2
with open('dummy.pdf','r') as file1:
    print(file1)

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE 4: Python Debug Consc

```
Mohd.Uzair@UzairPC MINGW64 /g/PythonPractice
$  env C:\\Users\\Mohd.Uzair\\AppData\\Local\\Programs\\Python\\Python38-32\\python.exe c:\\Users\\Mohd.Uzair\\.vscode\\extensions\\ms-python
\pythonFiles\\lib\\python\\debugpy\\launcher 64566 -- g:\\PythonPractice\\working_with_pdfs.py
<_io.TextIOWrapper name='dummy.pdf' mode='r' encoding='cp1252'>
```

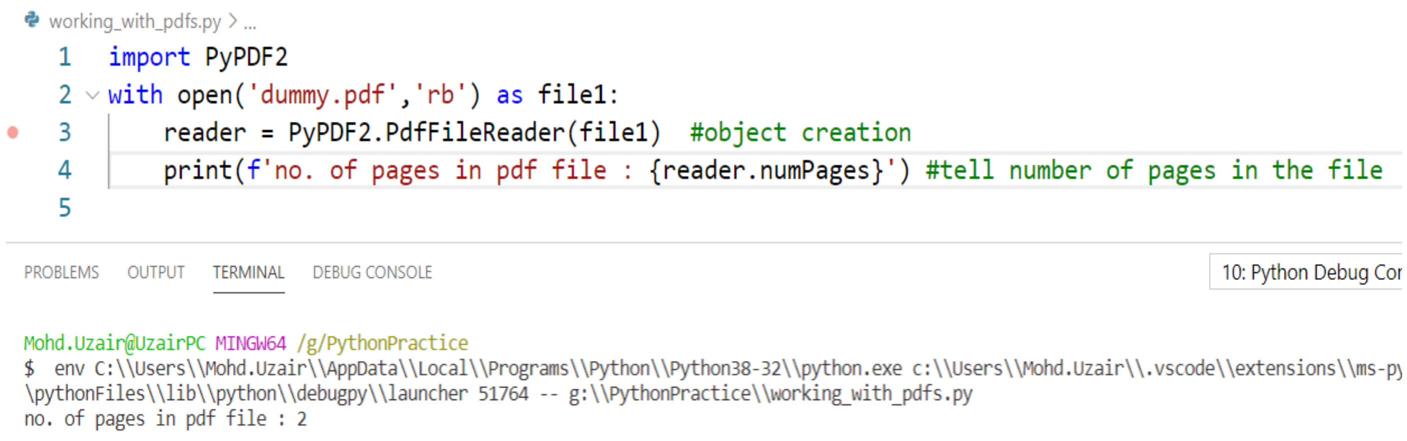

working_with_pdfs.py > ...

```python
import PyPDF2
with open('dummy.pdf','r') as file1:
    print(file1)
reader = PyPDF2.PdfFileReader(file1)
print(reader.numPages)

```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE 5:

```
Mohd.Uzair@UzairPC MINGW64 /g/PythonPractice
$  env C:\\Users\\Mohd.Uzair\\AppData\\Local\\Programs\\Python\\Python38-32\\python.exe c:\\Users\\Mohd.Uzair\\.vscode\\
\pythonFiles\\lib\\python\\debugpy\\launcher 50688 -- g:\\PythonPractice\\working_with_pdfs.py
<_io.TextIOWrapper name='dummy.pdf' mode='r' encoding='cp1252'>
PdfReadWarning: PdfFileReader stream/file object is not in binary mode. It may not be read correctly. [pdf.py:1079]
Traceback (most recent call last):
  File "g:\PythonPractice\working_with_pdfs.py", line 4, in <module>
    reader = PyPDF2.PdfFileReader(file1)
```

- Showing error that PdfFileReader stream/file object is not in binary mode
- So, we have to use mode='rb' for reading in binary mode, this will convert the file object to binary mode.

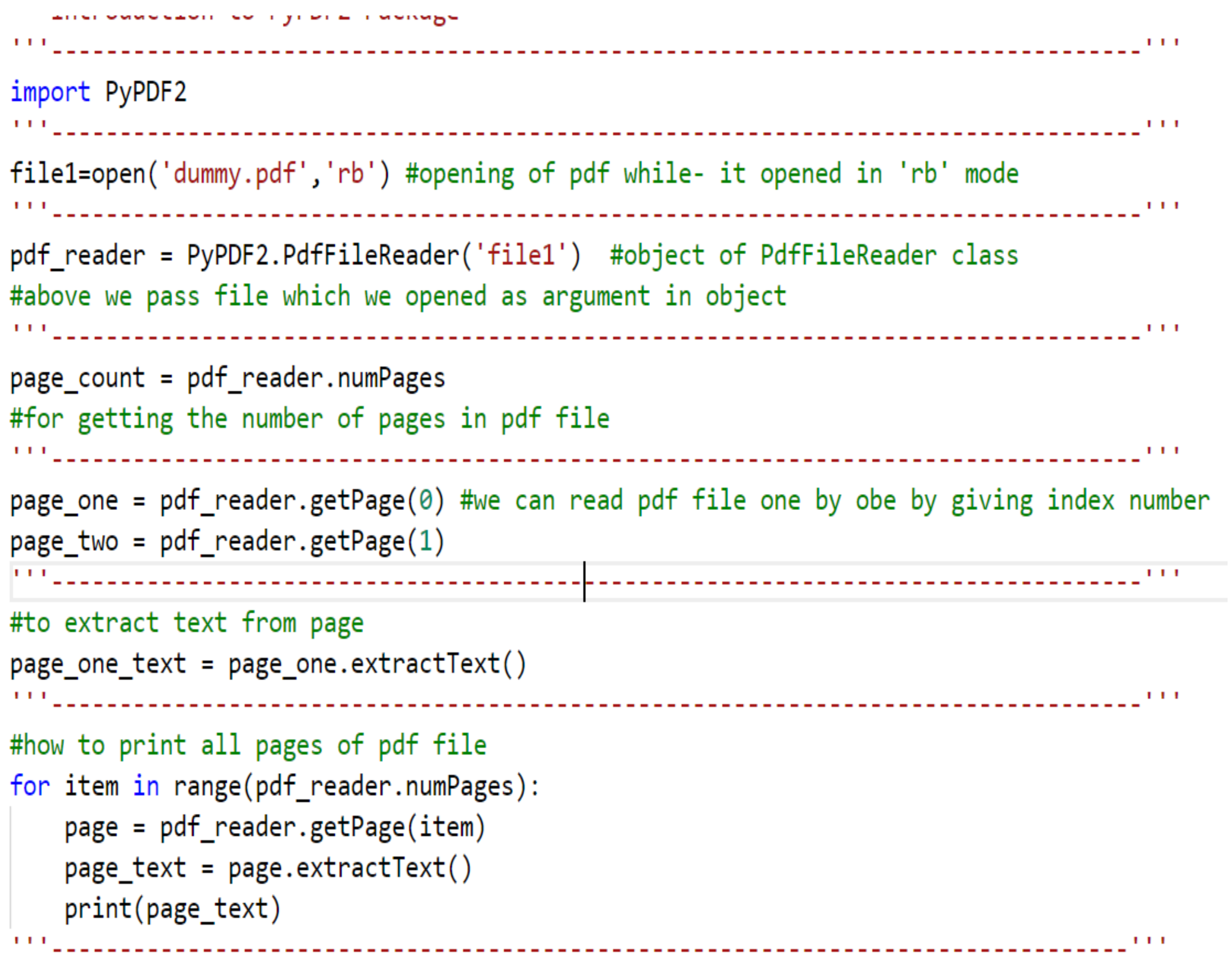


```python
import PyPDF2
with open('dummy.pdf','rb') as file1:
    reader = PyPDF2.PdfFileReader(file1)  #object creation
    print(f'no. of pages in pdf file : {reader.numPages}') #tell number of pages in the file
```

```
Mohd.Uzair@UzairPC MINGW64 /g/PythonPractice
$  env C:\\Users\\Mohd.Uzair\\AppData\\Local\\Programs\\Python\\Python38-32\\python.exe c:\\Users\\Mohd.Uzair\\.vscode\\extensions\\ms-py
\pythonFiles\\lib\\python\\debugpy\\launcher 51764 -- g:\\PythonPractice\\working_with_pdfs.py
no. of pages in pdf file : 2
```

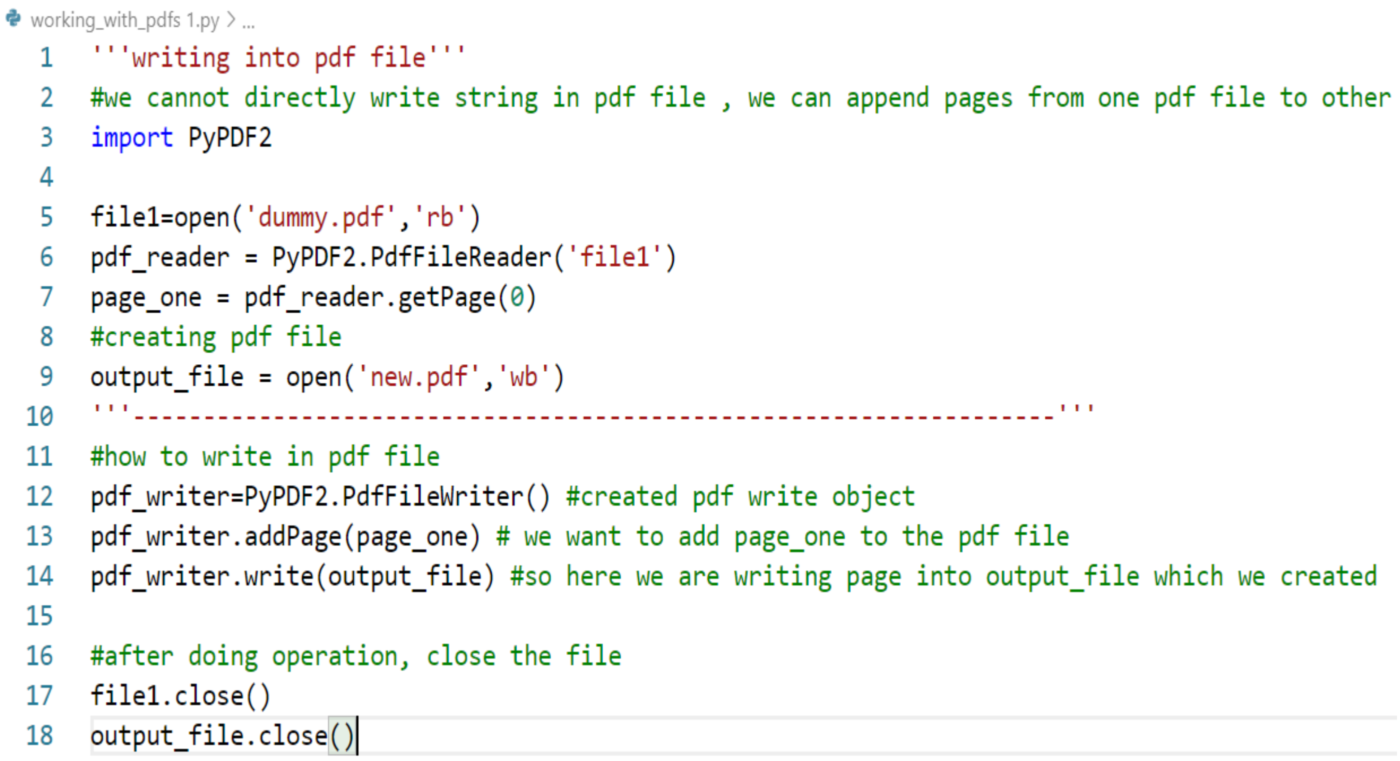
- Output is showing that pages = 2 in the pdf

OPENING AND READING OF PDF FILE



```python
'''-------------------------------------------------------------------------------'''
import PyPDF2
'''-------------------------------------------------------------------------------'''
file1=open('dummy.pdf','rb') #opening of pdf while- it opened in 'rb' mode
'''-------------------------------------------------------------------------------'''
pdf_reader = PyPDF2.PdfFileReader('file1')  #object of PdfFileReader class
#above we pass file which we opened as argument in object
'''-------------------------------------------------------------------------------'''
page_count = pdf_reader.numPages
#for getting the number of pages in pdf file
'''-------------------------------------------------------------------------------'''
page_one = pdf_reader.getPage(0) #we can read pdf file one by obe by giving index number
page_two = pdf_reader.getPage(1)
'''-------------------------------------------------------------------------------'''
#to extract text from page
page_one_text = page_one.extractText()
'''-------------------------------------------------------------------------------'''
#how to print all pages of pdf file
for item in range(pdf_reader.numPages):
    page = pdf_reader.getPage(item)
    page_text = page.extractText()
    print(page_text)
'''------------------------------------------------------------------------------'''
```

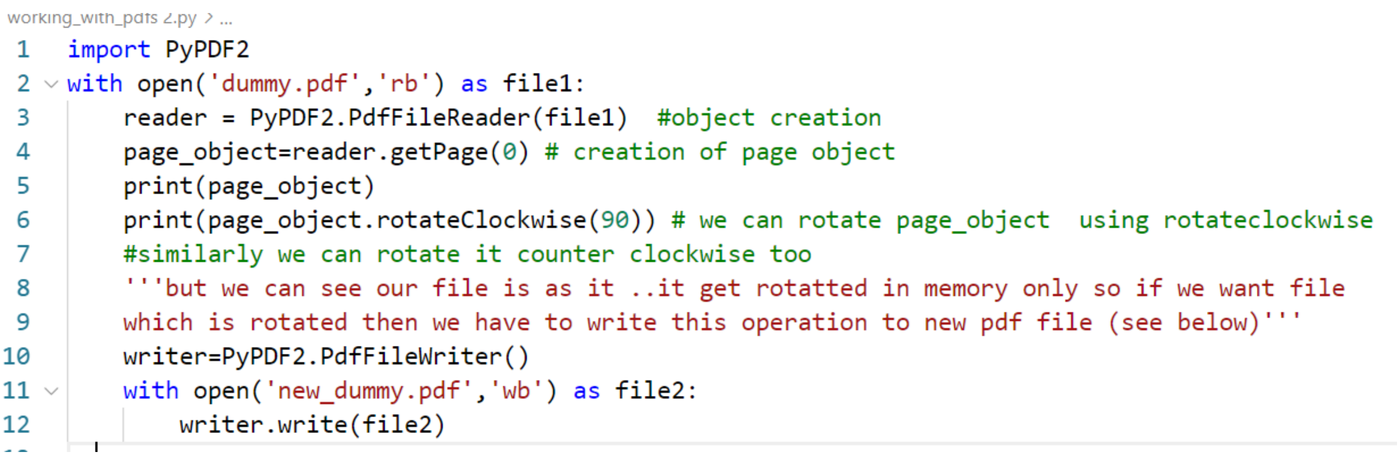
WRITING INTO PDF FILE



working_with_pdfs 1.py > ...

```python
'''writing into pdf file'''
#we cannot directly write string in pdf file , we can append pages from one pdf file to other
import PyPDF2

file1=open('dummy.pdf','rb')
pdf_reader = PyPDF2.PdfFileReader('file1')
page_one = pdf_reader.getPage(0)
#creating pdf file
output_file = open('new.pdf','wb')
'''------------------------------------------------------------------'''
#how to write in pdf file
pdf_writer=PyPDF2.PdfFileWriter() #created pdf write object
pdf_writer.addPage(page_one) # we want to add page_one to the pdf file
pdf_writer.write(output_file) #so here we are writing page into output_file which we created

#after doing operation, close the file
file1.close()
output_file.close()
```

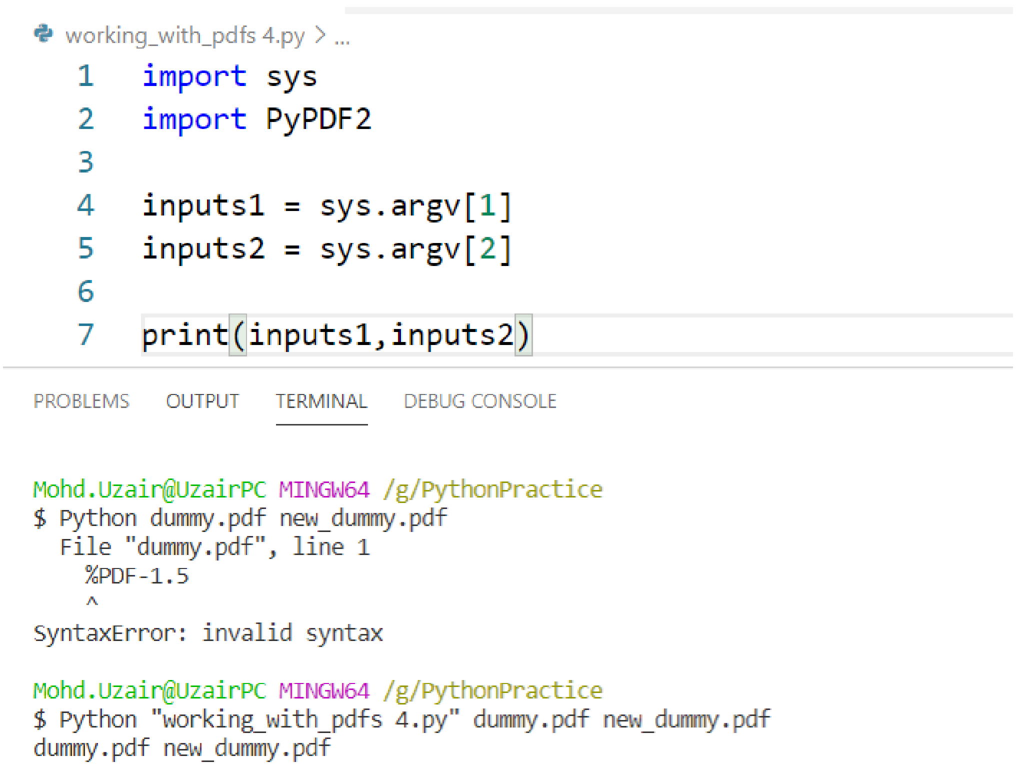
PERFORM ROTATE OPERATION AND SAVE IT IN NEW FILE:



working_with_pdfs 2.py > ...

```python
import PyPDF2
with open('dummy.pdf','rb') as file1:
    reader = PyPDF2.PdfFileReader(file1)  #object creation
    page_object=reader.getPage(0) # creation of page object
    print(page_object)
    print(page_object.rotateClockwise(90)) # we can rotate page_object  using rotateclockwise
    #similarly we can rotate it counter clockwise too
    '''but we can see our file is as it ..it get rotatted in memory only so if we want file
    which is rotated then we have to write this operation to new pdf file (see below)'''
    writer=PyPDF2.PdfFileWriter()
    with open('new_dummy.pdf','wb') as file2:
        writer.write(file2)
```

Learning Command Line Argument (SYS Module)



working_with_pdfs 4.py > ...

```python
import sys
import PyPDF2

inputs1 = sys.argv[1]
inputs2 = sys.argv[2]

print(inputs1,inputs2)
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

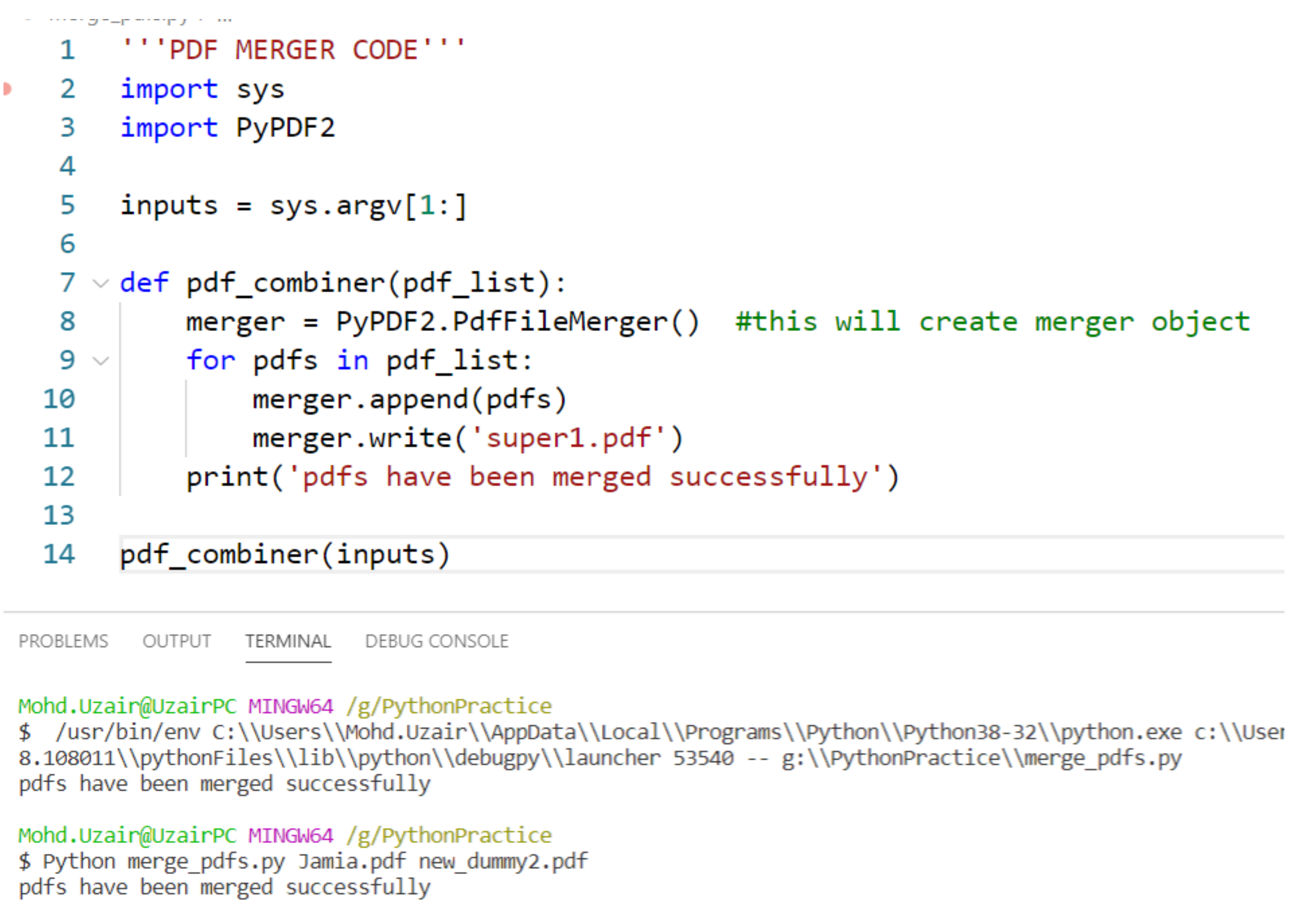
```
Mohd.Uzair@UzairPC MINGW64 /g/PythonPractice
$ Python dummy.pdf new_dummy.pdf
  File "dummy.pdf", line 1
    %PDF-1.5
    ^
SyntaxError: invalid syntax

Mohd.Uzair@UzairPC MINGW64 /g/PythonPractice
$ Python "working_with_pdfs 4.py" dummy.pdf new_dummy.pdf
dummy.pdf new_dummy.pdf
```

Syntax for giving arguments:

[Python] [file_name] [input1] [input2] [input_n]

- If file name or inputs have between names then give names in quotes ‘’

CODE EXERCISE: PDF MERGER CODE

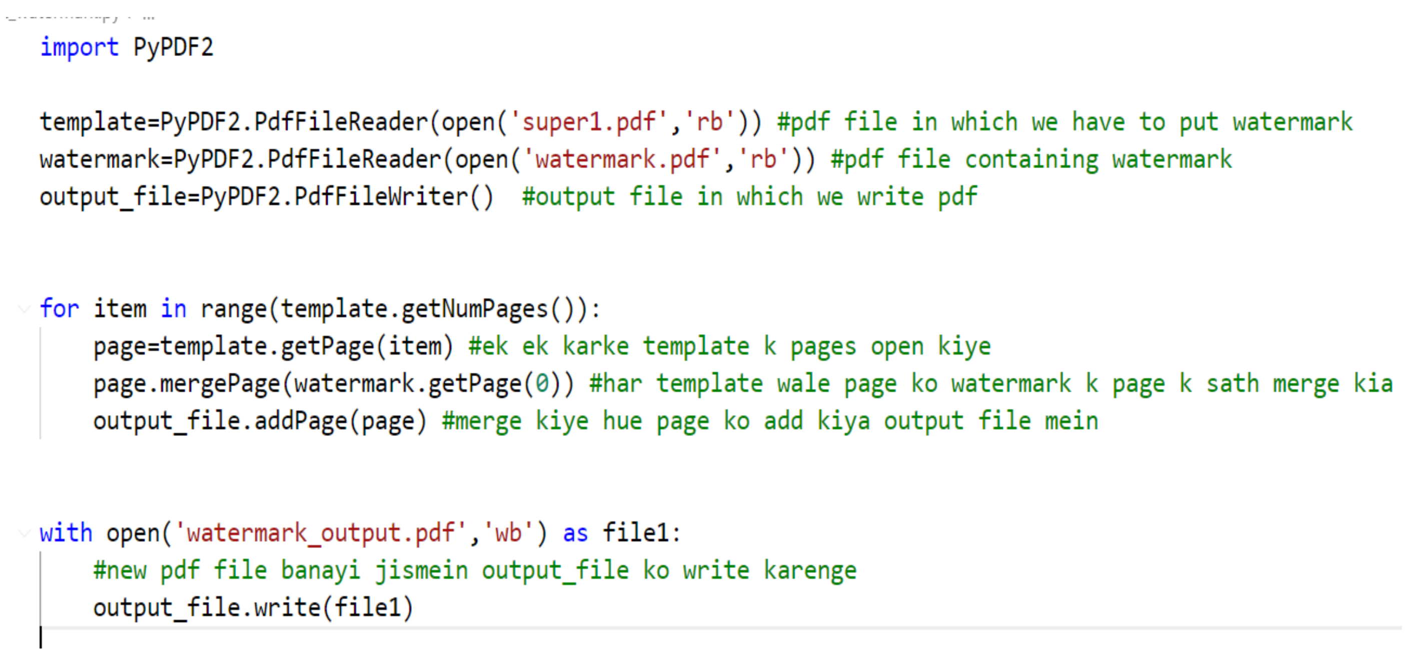


```python
'''PDF MERGER CODE'''
import sys
import PyPDF2

inputs = sys.argv[1:]

def pdf_combiner(pdf_list):
    merger = PyPDF2.PdfFileMerger()  #this will create merger object
    for pdfs in pdf_list:
        merger.append(pdfs)
        merger.write('super1.pdf')
    print('pdfs have been merged successfully')

pdf_combiner(inputs)
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

```
Mohd.Uzair@UzairPC MINGW64 /g/PythonPractice
$  /usr/bin/env C:\\Users\\Mohd.Uzair\\AppData\\Local\\Programs\\Python\\Python38-32\\python.exe c:\\User
8.108011\\pythonFiles\\lib\\python\\debugpy\\launcher 53540 -- g:\\PythonPractice\\merge_pdfs.py
pdfs have been merged successfully

Mohd.Uzair@UzairPC MINGW64 /g/PythonPractice
$ Python merge_pdfs.py Jamia.pdf new_dummy2.pdf
pdfs have been merged successfully
```

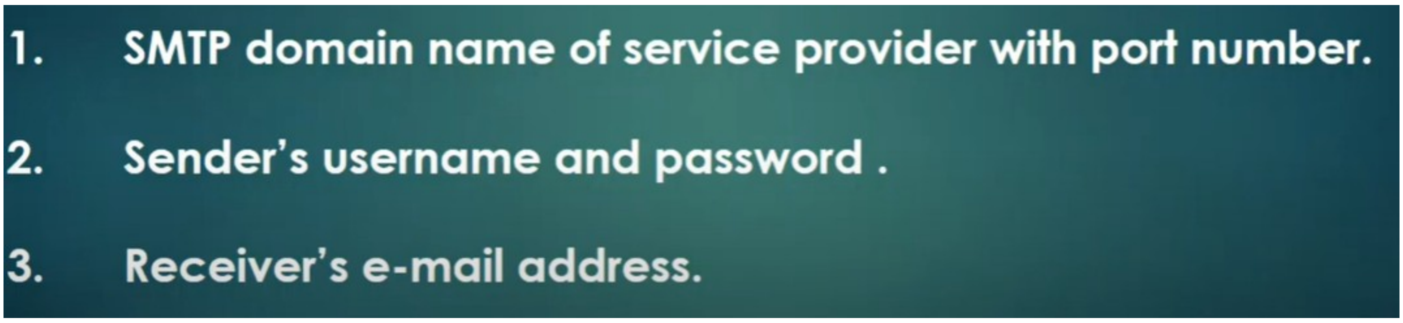
CODE EXERCISE: ADDING A WATERMARK TO A PDF FILE



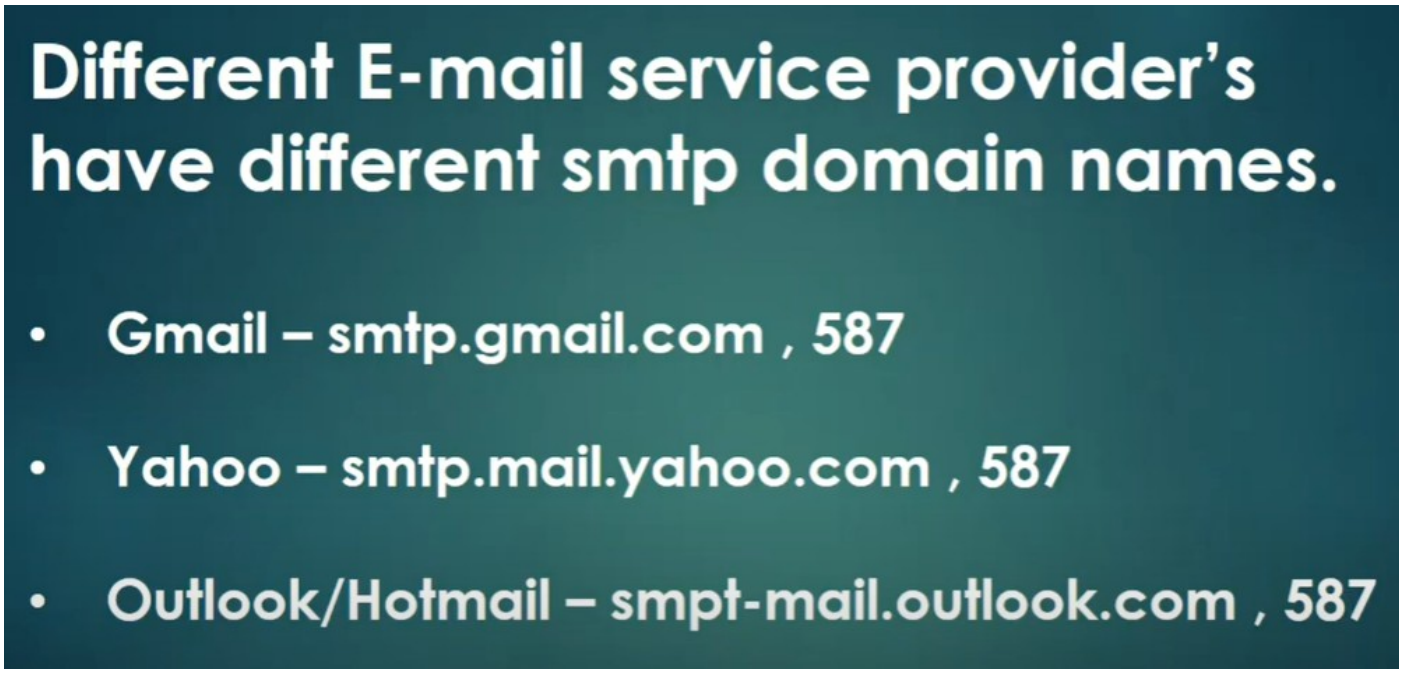
```python
import PyPDF2

template=PyPDF2.PdfFileReader(open('super1.pdf','rb')) #pdf file in which we have to put watermark
watermark=PyPDF2.PdfFileReader(open('watermark.pdf','rb')) #pdf file containing watermark
output_file=PyPDF2.PdfFileWriter()  #output file in which we write pdf


for item in range(template.getNumPages()):
    page=template.getPage(item) #ek ek karke template k pages open kiye
    page.mergePage(watermark.getPage(0)) #har template wale page ko watermark k page k sath merge kia
    output_file.addPage(page) #merge kiye hue page ko add kiya output file mein


with open('watermark_output.pdf','wb') as file1:
    #new pdf file banayi jismein output_file ko write karenge
    output_file.write(file1)
```

SENDING EMAILS USING PYTHON

- email package to read, write, and send simple email messages, as well as more complex MIME messages.
- Python offers a native library to send emails- “SMTP lib”. “smtplib” creates a Simple Mail Transfer Protocol client session object which is used to send emails to any valid email id on the internet.
- SMTP is protocol used for sending emails. (Simple Mail Transfer Protocol)



1. SMTP domain name of service provider with port number.
2. Sender’s username and password .
3. Receiver’s e-mail address.

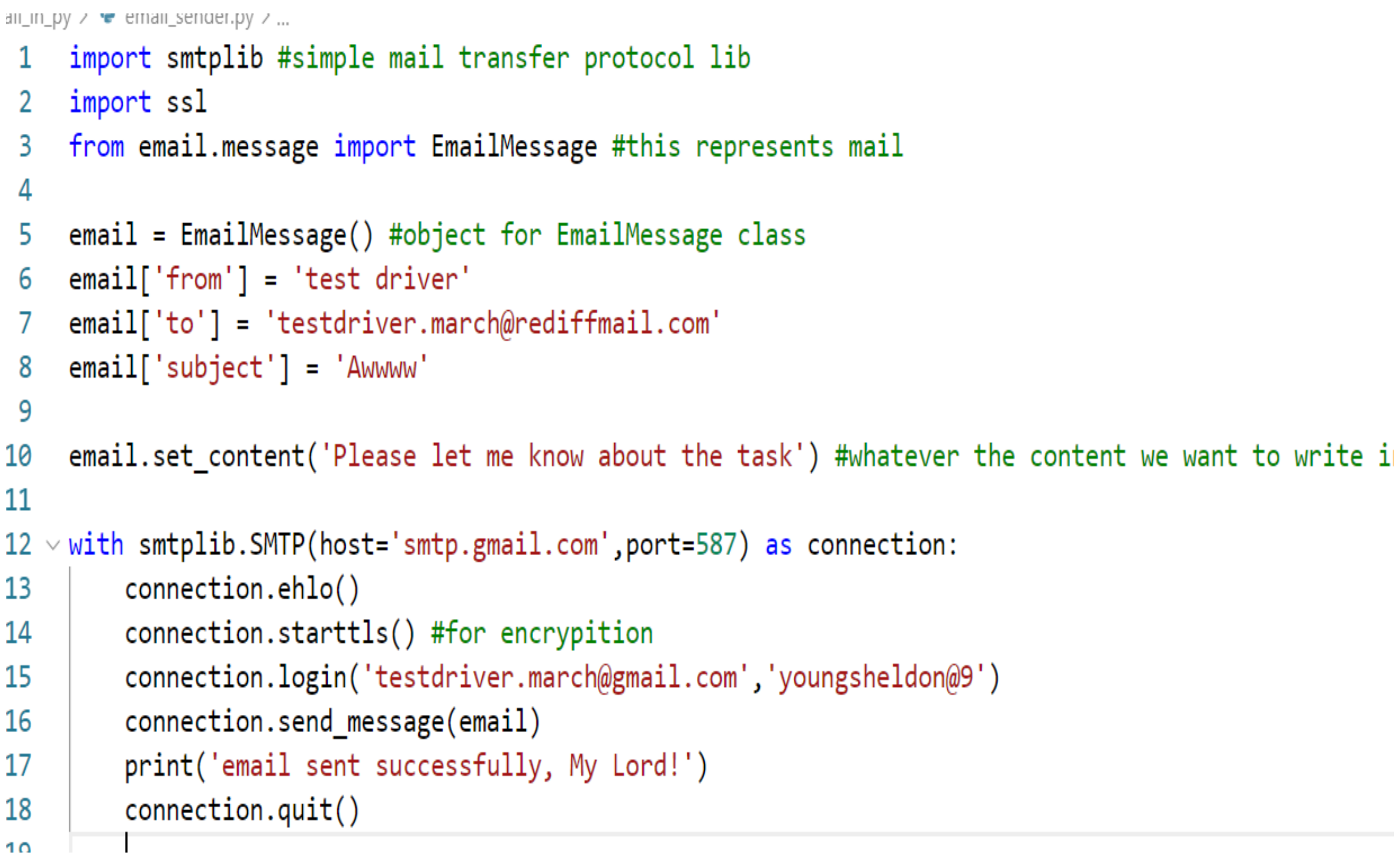


Different E-mail service provider’s have different smtp domain names.

- Gmail – smtp.gmail.com , 587
- Yahoo – smtp.mail.yahoo.com , 587
- Outlook/Hotmail – smpt-mail.outlook.com , 587

- `smtp.elho()`
  The client sends this command to the SMTP server to identify itself and initiate the SMTP conversation. The domain name or IP address of the SMTP client is usually sent as an argument together with the command
- **StartTLS** is a protocol command used to inform the email server that the email client wants to upgrade from an insecure connection to a secure one using TLS or SSL. **StartTLS** is used with **SMTP** and IMAP, while POP3 uses the slightly different command for encryption, STLS.

CODE EXERCISE: SENDING MAIL USING PYTHON



ail_in_py › email_sender.py › ...

```
import smtplib #simple mail transfer protocol lib
import ssl
from email.message import EmailMessage #this represents mail

email = EmailMessage() #object for EmailMessage class
email['from'] = 'test driver'
email['to'] = 'testdriver.march@rediffmail.com'
email['subject'] = 'Awwww'

email.set_content('Please let me know about the task') #whatever the content we want to write i

with smtplib.SMTP(host='smtp.gmail.com',port=587) as connection:
    connection.ehlo()
    connection.starttls() #for encrypition
    connection.login('testdriver.march@gmail.com','youngsheldon@9')
    connection.send_message(email)
    print('email sent successfully, My Lord!')
    connection.quit()
```