

Introductory Programming using Python

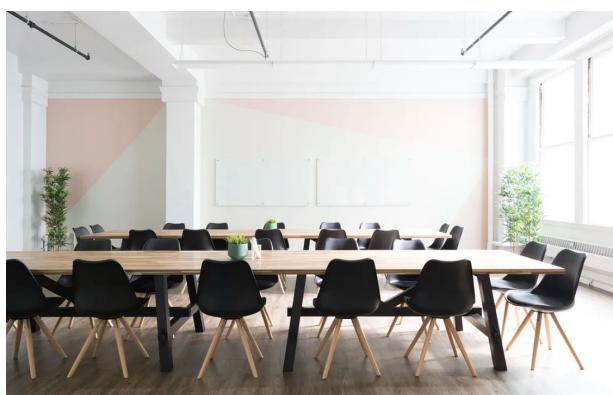
Day 2

Jason Lim, Wong Hau Shian
Republic Polytechnic



Welcome and admin matters

- Please ensure that:
 - your attendance is reflected on the record
 - you have a learning laptop with you
 - you have a good view on the display

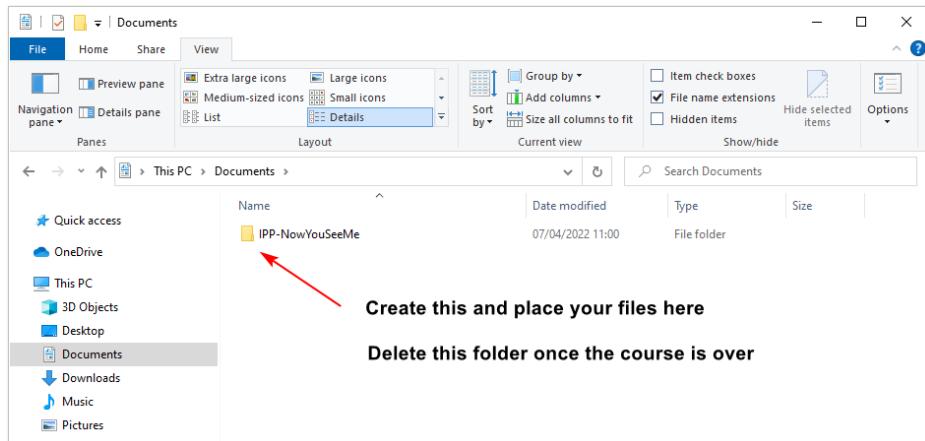


- Course material is also available at <https://bit.ly/IPP-MHA>



Welcome and admin matters

- A tidy/clean laptop is a good laptop for learning
- Created files on the laptop will linger on. To prevent this:
 - Create a folder in the "Documents" location for this workshop
 - E.g. "IPP-Alan" or "IPP-PeterPan"
 - All user created files to be placed in this folder
- Delete that folder after the course



Overview: Day Two

Morning	Afternoon
<ul style="list-style-type: none"> • String functions • String formatting • Dictionary • Read and writing files • Copying, moving and deleting files and folders 	<ul style="list-style-type: none"> • Working with Excel • Image Processing • Connecting to the Web • Demo: Sending Emails (outlook)



String functions

- **Split**

```
>>> a='python or java'
>>> b=a.split(' ')
>>> type(b)
<type 'list'>
>>> b
['python', 'or', 'java']
>>>
```

```
>>> a='python or java'
>>> b=a.split('on')
>>> b
['pyth', ' or java']
>>>
```

- **Join**

```
>>> a=['python','and','java']
>>> b=' '.join(a)
>>> b
'python and java'
>>> c=','.join(a)
>>> c
'python, and, java'
>>>
```

string-functions.py 5



String Slicing

```
>>> s = "freedom"
>>> print(s[:4])
free
>>> print(s[-3:])
dom
>>> |
```

- Slicing works for any sequence (e.g. list), so it works for strings too.

[:4] gets from the start till the fourth character

[-3:] gets the last third till the last character



Exercise – Find Longest Word

- Create the function **findLongestWord** that takes in a sentence and returns the longest word.

Hint: Use `split()`, repetitions



15 mins

findLongestWord.py 7



Strings – Useful functions

- Useful functions for validations
 - `String.isupper()` – Returns True if all characters in `String` are in uppercase.
 - `String.islower()` – Returns True if all characters in `String` are in lowercase.
 - `String.isalpha()` – Returns True if `String` contains alphabets only.
 - `String.isdigit()` – Returns True if all characters in `String` are numbers only.
 - `len(String)` – Returns the total number of characters (including spaces) in `String`
 - `String.count(text)` – Returns the number of times `text` appears in `String`.
 - `String.startswith(text)` – Returns True if `String` starts with `text`.
 - `String.endswith(text)` – Returns True if `String` ends with `text`.
- Useful functions for manipulations
 - `String.upper()` – Returns a String with all characters in the original String converted to uppercase
 - `String.lower()` – Returns a String with all characters in the original String converted to lowercase
 - `String.replace(x, y)` – Returns a String with all occurrences of `x` in the original String replaced with `y`

Extra: See examples on usage in “Additional Resources on Strings.pdf”



String formatting

- **Formatting numbers**

- %d int
- %f float

More about string formatting technique can be found here:

<http://docs.python.org/library/stdtypes.html>

- **Special formatting**

- %.2f float two decimal places
- %+d sign printing (+)
%+f e.g. +5.6

```
>>> import math
>>> print("Pi is " + str(math.pi))
Pi is 3.141592653589793
>>> print("Pi is approx %.2f"%(math.pi))
Pi is approx 3.14
>>> print("Pos or Neg: %+d %+d"%( -5,3))
Pos or Neg: -5 +3
>>> |
```

- **Formatting Strings**

- %15s reserve 15 characters for string, right-justified
- %-15s reserve 15 characters for string, left-justified

In the earlier exercise on temperature_calculator.py, the output can be rewritten as:

```
print(name + "'s temperature is " + str(diff_from_369) + " degree from 36.9 degree celsius")
          ↓
print("%s's temperature is %.2f degree from 36.9 degree celsius"%(name,diff_from_369))
```

☞



Exercise – String formatting (1)

- Try this out yourself!

- Open *string-format1.py*
- Change the values for the value line
- Execute and observe the effect

```
>>> import math
>>> a = math.pi
>>> a
3.141592653589793
>>> b=5
>>> c="python"
>>> line="%s %f %d"%(c,a,b)
>>> line
'python 3.141593 5'
```

```
>>> line="%03d"%(b)
>>> line
'005'
```





Exercise – String formatting (2)

- Given the variable

```
x = "admin:$E*G$@R:/users/root:"
```

Write a program to display the following output:

User	:	admin
Password	:	\$E*G\$@R
Homedir	:	/users/root



string-format2.py 11



Python Dictionary

```
dictionary = {'a':1, 'p':1, 'r':2, 't':1, 'o':1}
```

- A dictionary stores multiple **key-value** pairs
- Each key-value pair are separated by a colon (:)
- Every key is unique; no duplicate key within a dictionary
- A dictionary uses a set of curly brackets { } to store its key-value pairs
 - Contrast with a Python list that uses square brackets []
- To access a value in the dictionary, we use the key as an index
 - e.g. print(dictionary["r"]) displays value of 2



Python Dictionary

- You can *add*, *edit* and *delete* elements from a dictionary

```
# create dictionary with some elements
members = {'mary': 18, 'alan': 20, 'peter': 21}

# add element
members['john'] = 20 ← Key "john" NOT IN the members dict. This ADDS the key "john" and value 20 pair

# edit element
members['mary'] = 25 ← Key "mary" IS IN the members dict. This EDITS the value of key "mary"

# delete element
del members['alan']

# get number of element
print(len(members))

# display all the elements
print(members)
```

13



Python Dictionary

- We can traverse and iterate over a dictionary using *for* loop
 - e.g. assume members contain name (key) and age (value)
To calculate the average age:

```
1 total = 0
2 members = { "mary":18, "alan":20, "peter":21 }
3
4 for key in members: # Iterate the key:value pairs using a for loop
5     age = members[key] # Assign the value for each key in the variable age
6     total = total + age
7
8 average = total / len(members)
9 average = round(average, 2)
10 print("The average age is " + str(average))
```

14



Exercise – Dictionary Operations

- Create a dictionary with the following key:value pairs
 "alan" : 80001234
 "mary": 90004567
 "peter": 61234567
- Update the value for "mary" to 91110000
- Remove the element with "peter" as the key

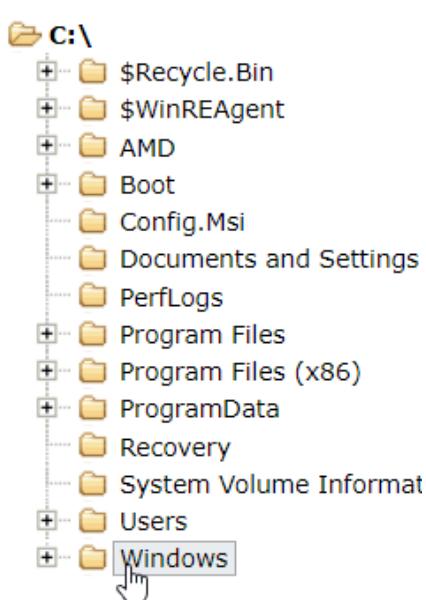


dict-ops.py 15

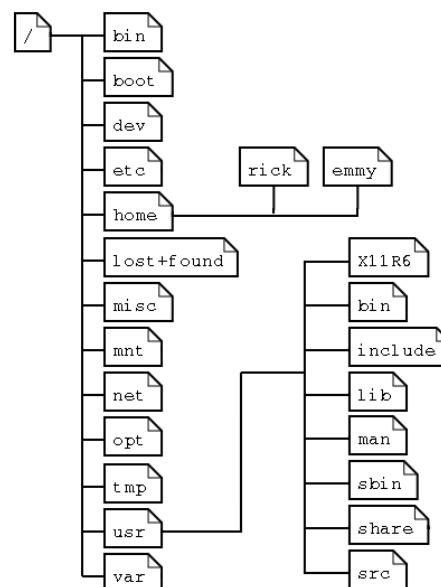


File Tree

• Windows



• Linux

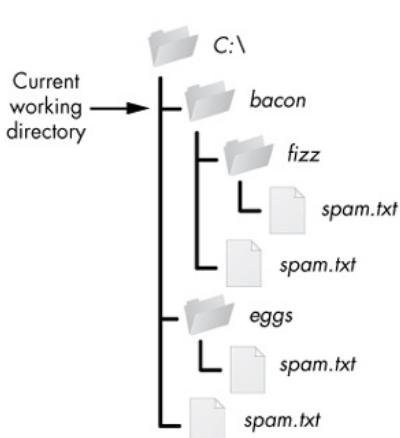




File Paths

An absolute file path describes how to access a given file or directory, starting from the root of the file system.

A relative file path is interpreted from the perspective your current working directory.



Relative Paths	Absolute Paths
..\	C:\
.\	C:\bacon
.\fizz	C:\bacon\fizz
.\fizz\spam.txt	C:\bacon\fizz\spam.txt
.\spam.txt	C:\bacon\spam.txt
..\eggs	C:\eggs
..\eggs\spam.txt	C:\eggs\spam.txt
..\spam.txt	C:\spam.txt

Absolute file paths are notated by a leading forward slash or drive label.

Relative file paths are notated by a **lack of a leading forward slash**.

17

Image credit: <https://automatetheboringstuff.com/chapter8/>



Read files

```
1 # make sure you have a hello.txt in your current working director
2 # same directory as your python script
3 helloFile = open("hello.txt")
4 content = helloFile.read()
5 print(content)
6 helloFile.close()
7
8 # make sure you have a hello.txt in the specified director
9 # same directory as your python script
10 helloFile = open("hello.txt")
11
12 content = helloFile.readlines()
13 print(content)
```

Open() will return a file object which has reading and writing related methods

Pass ‘r’ (or nothing) to open() to open the file in read mode.

Call `read()` to read the contents of a file

Call close() when you are done with the file.

- Call `readLines()` to read the contents of a file

file read.py



Write files

```

1 # make sure you have a hello.txt in your current working director
2 # same directory as your python script
3 helloFile = open("hello.txt", "w")
4 helloFile.write("This is also another line\n")
5 helloFile.close()
6
7 # reopen to display content
8 helloFile = open("hello.txt")
9 print(helloFile.read())
10 helloFile.close()
11
12 # open the file for adding next text
13 helloFile = open("hello.txt", "a")
14 helloFile.write("Hello world again\n")
15 helloFile.close()
16
17 # reopen to display content
18 helloFile = open("hello.txt")
19 print(helloFile.read())
20 helloFile.close()
21

```

Search Stack Data Debug I/O Python Shell Commands execute without debug. Use arrow keys for history.

3.7.4 (tags/v3.7.4:e09359112e, Jul 8 2019, 19:29:22) [MSC ^
Python Type "help", "copyright", "credits" or "license" for
>>> [evaluate file_write.py]
THis is also another line
THis is also another line
Hello world again

file_write.py

Pass 'w' to open() to open the file in write mode or 'a' for append mode.

Opening a non-existent file in write or append mode will create that file

Call write() to write a string to a file.



Copying and moving files

```

import shutil

# copy file, destination must exist
shutil.copy("hello.txt", "folder1")

# copy file, destination must exist, and into a new name
shutil.copy("hello.txt", "folder1/newhello.txt")

# recursively copy an entire directory
# error if the destination folder already exist
shutil.copytree("folder1", "folder_to_delete")
shutil.copytree("folder1", "folder_to_delete2")

# move file, destination must exist
shutil.move("folder1/hello.txt", "folder2")

# move and rename file
shutil.move("folder_to_delete/hello.txt", "folder_to_delete2/newhello.txt")

```

`shutil.copy(src, dst)` – Copy the file `src` to the file or directory `dst`

`shutil.copytree(src, dst)` - Recursively copy an entire directory tree rooted at `src`.

`shutil.move(src, dst)` - Recursively move a file or directory (`src`) to another location (`dst`).

<https://docs.python.org/3/library/shutil.html>



Deleting files

```
import os

# error if file do not exist
os.unlink("folder2/hello.txt")

# get current working directory
print(os.getcwd())

# delete directory (can only delete empty folder)
os.rmdir("folder_to_delete")

import shutil
# delete directory (with content)
# error if folder is not found
shutil.rmtree("folder_to_delete2")
```

e.g. To delete all .docx file in the current folder

```
import os

for filename in os.listdir():
    if filename.endswith(".docx"):
        print(filename)
        os.unlink(filename)
```

- os.unlink() will delete a file
- os.rmdir() will delete a folder (but folder must be empty)
- shutil.rmtree() will delete a folder and all its contents



Deleting can be dangerous, so do a dry run first

file_delete.py 21



Use Case Sharing

- Organizing students' submissions into separate folder
 - Class of 25 students into 5 teams

25 folders,
one for
each
student

- Student 120101
- Student 120897
- Student 120904
- Student 121104
- Student 121243
- Student 121550
- Student 121804
- Student 121938
- Student 122061
- Student 122084
- Student 122152
- Student 122263
- Student 122431
- Student 122868
- Student 123295
- Student 123525
- Student 123534
- Student 123673
- Student 123864
- Student 123900
- Student 124059
- Student 124133
- Student 124990
- Student 128079



- Team 1
- Team 2
- Team 3
- Team 4
- Team 5

Student
submission
sorted by teams

Use_Case01\sort2Teams.py 22



Other Use Cases

- System administrators can use these commands to
 - Copy and backup files to other hard-disks
 - Delete folders/ files at fixed schedules
 - End of financial year
 - End of semester
- Others
 - Check timestamp of files, and delete all files created before a certain date

23



Exercise – File operations

- Write a program to achieve the following:

1. Create a file named: “myfile.txt”.
2. Write the following line of text into the file:
 - Programming is fun!
3. Close the file
4. Create a folder called “myfolder”
 - Use os.mkdir() command
5. Copy myfile.txt to myfolder





Python Package Index

<https://pypi.org/>

- A repository of software for the Python Programming Language
- Python Installation provides the core libraries needed for the common tasks
 - Additional packages can be found at the website and installed as extension
 - E.g. send2trash, openpyxl, pillow etc
- Installation is easy done with the following command
pip install <software_package>
- Installed packages can be found at:
C:\python38\Lib\site-packages

25



Using pip install

- For all windows users by default
 - Open command prompt
pip install <package_name>
- For Mac User
 - Open terminal
pip3 install <package_name>
- For staff using company issued laptop with no Admin rights
 - Open command prompt
pip install --user <package_name>

Double-dash

26

Excel Manipulation using Python

27



Working with Excel

- Install **openpyxl** module using “`pip install openpyxl`”

openpyxl 3.0.3

`pip install openpyxl`

Released: Jan 11, 2020

A Python library to read/write Excel 2010 xlsx/xlsm files

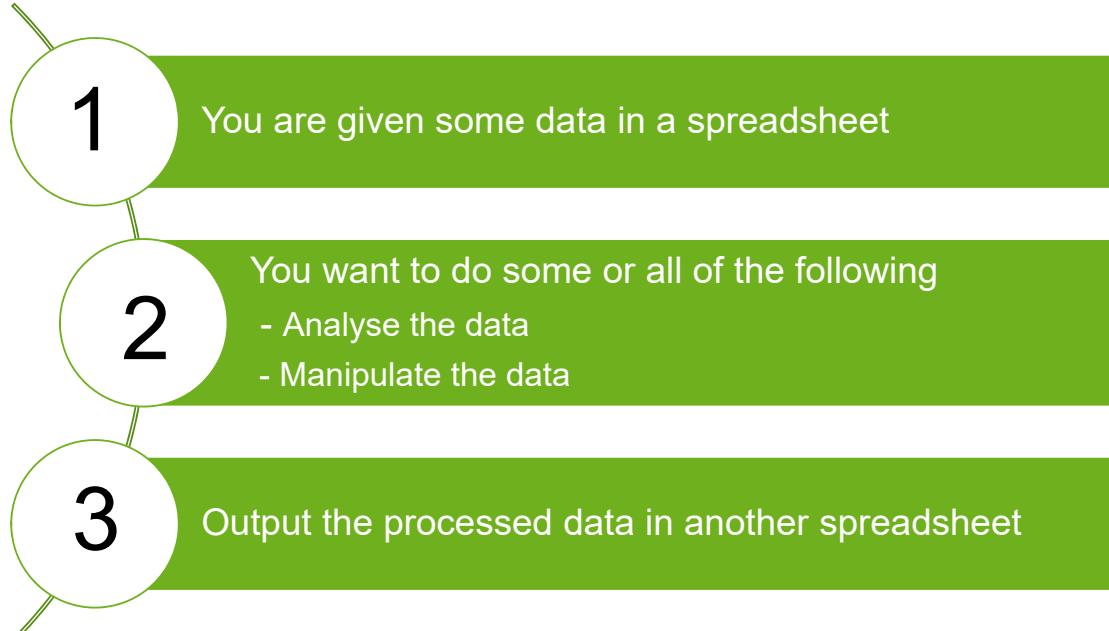
Navigation	Project description
Project description	coverage 99%
Release history	Introduction
Download files	openpyxl is a Python library to read/write Excel 2010 xlsx/xlsm/xltx/xlsm files.
Project links	It was born from lack of existing library to read/write natively from Python the Office Open XML format.
Homepage	All kudos to the PHPExcel team as openpyxl was initially based on PHPExcel.
Tracker	Security
Source	By default openpyxl does not guard against quadratic blowup or billion laughs xml attacks. To guard against these attacks install defusedxml.
Documentation	

- Full **openpyxl** documentation:
<https://openpyxl.readthedocs.io/en/stable/index.html>

28



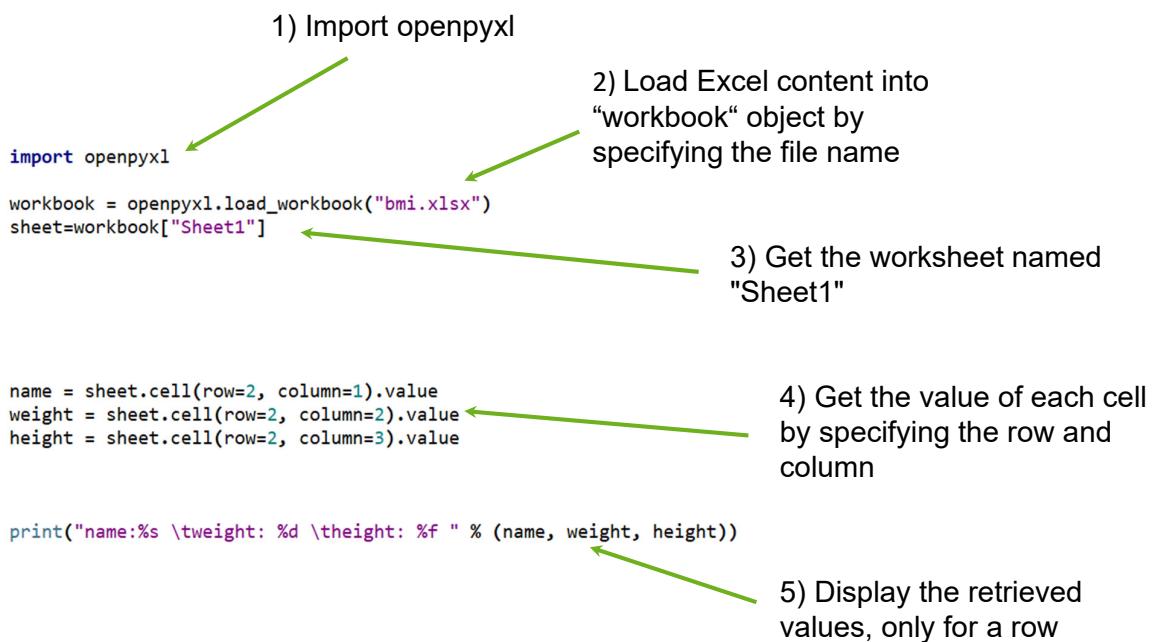
Typical Workflow for Excel Automation



29



Reading Excel file





Reading Excel file

- Reading excel file typically uses a *for* loop to go through each row to read the data

```

import openpyxl
workbook = openpyxl.load_workbook("bmi.xlsx")
sheet=workbook["Sheet1"]

max_row = sheet.max_row # get number of rows
#loop through every row
for i in range(2, max_row + 1):

    #read cell
    name = sheet.cell(row=i, column=1).value
    weight = sheet.cell(row=i, column=2).value
    height = sheet.cell(row=i, column=3).value

    print("name:%s \tweight: %d \theight: %f " % (name, weight, height))

```

1) Get the number of rows and columns

2) Use For loop to go through every row

3) Extract the status at Column C to check for attendance

excel_read_bmi.py 31



Update Excel file

```

import openpyxl
workbook = openpyxl.load_workbook("bmi.xlsx")
sheet=workbook["Sheet1"]

max_row = sheet.max_row # get number of rows
# add a column header for bmi
sheet.cell(row=1, column=4).value = "bmi"
#loop through every row
for i in range(2, max_row + 1):

    #read cell
    name = sheet.cell(row=i, column=1).value
    weight = sheet.cell(row=i, column=2).value
    height = sheet.cell(row=i, column=3).value
    bmi = weight / (height * height)
    sheet.cell(row=i, column=4).value = bmi
    print("name:%s \tBMI: %f" % (name, bmi))

#save the file
workbook.save("bmi_update.xlsx")

```

1) Load file into memory & get the sheet

2) Adds a header at the 4th column

3) Perform calculation with values taken from the excel files

4) Add calculated value to cell

5) Save the spreadsheet

excel_update_bmi.py 32



Create Excel file

- If you have data in nested Python list, you can write the data into an excel file

```

import openpyxl

workbook = openpyxl.Workbook()

#get the default sheet
sheet=workbook["Sheet"]

#create a list of tuples as data source
data = [
    [225.7, 'Gone with the Wind', 'Victor Fleming'],
    [194.4, 'Star Wars', 'George Lucas'],
    [161.0, 'ET: The Extraterrestrial', 'Steven Spielberg']
]

for row in data:
    sheet.append(row)

#save the spreadsheet
workbook.save("movies.xlsx")

```

1) Some data in nested list

2) Using for loop to add each row of data into the excel sheet

3) Save the spreadsheet

excel_create.py 33



Format Excel

```

import openpyxl
from openpyxl.styles import Font, PatternFill, Border, Side

workbook = openpyxl.load_workbook("bmi.xlsx")
sheet=workbook["Sheet1"]

#define the colors to use for styling
BLUE = "#0033CC"
LIGHT_BLUE = "E6ECFF"
WHITE = "FFFFFF"

#define styling
header_font = Font(name="Tahoma", size=14, color=WHITE)
header_fill = PatternFill("solid", fgColor=BLUE)

# format header
for row in sheet["A1:C1"]:
    for cell in row:
        cell.font = header_font
        cell.fill = header_fill

#define styling
white_side = Side(border_style="thin", color=WHITE)
blue_side = Side(border_style="thin", color=BLUE)
alternate_fill = PatternFill("solid", fgColor=LIGHT_BLUE)
border = Border(bottom=blue_side, left=white_side, right=white_side)

# format rows
for row_index, row in enumerate(sheet["A2:C3"]):
    for cell in row:
        cell.border = border
        if row_index %2 :
            cell.fill = alternate_fill

workbook.save("bmi_format.xlsx")

```

1) Import necessary functions

2) Setup colors and styles

3) Loop through cell and set properties

excel_format.py 34



Use Case Sharing

- RPA (Robotic Process Automation) can be used to send WhatsApp messages to students individually
- Write script
 - to split contact list for students based on intake year
 - to add country code: 65 to all the numbers

2 students in 2017 intake

Create worksheets for students of different intake

	A	B	C	D
1	ReqTerm	StudentID	Name	Handphone
2	1610	16011054	MAX TAN	87733828
3	1710	16041191	PETER	87226030
4	1710	17011180	MARY	92266192
5	1810	18034448	ALAN	92336601
6	1910	19043330	JOSEPH	92468837
7	1910	19045104	ERIC	86511160
8	1910	19045784	JAVIER	97937779
9	1910	19047541	JUNE	97277250
10	1910	19011433	APRIL	97661277
11	2010	20041418	MAY	91766557

	A	B	C	D	E	F
1	ReqTerm	StudentID	Name	Handphone		
2	1710	16041191	PETER	6587226030		
3	1710	17011180	MARY	6592266192		
4						
5						
6						
7						

* Data shared here are randomly generated

35



Image Processing with Python



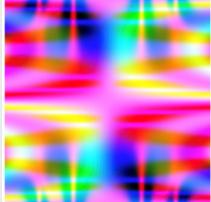
Image Processing

The friendly PIL fork

Welcome
This is the home of Pillow, the friendly PIL fork. PIL is the Python Imaging Library. If you have ever worried or wondered about the future of PIL, please stop. [We're here](#) to save the day.

Code
Our code is [hosted on GitHub](#), tested on Travis CI, AppVeyor, Coveralls, Landscape and released on PyPI.

Documentation
Our documentation is [hosted on readthedocs.io](#) and includes installation instructions, handbook, API reference, release notes and more.

 Random psychedelic art made with PIL.

- For the next section we are going to use the Python Image Library, or in short Pillow.

- Install using the following command:
pip install pillow
- Documentation:
<https://pillow.readthedocs.io/en/stable/handbook/overview.html>

37



Image Processing

```
import os
from PIL import Image

filename = "img/clungup.jpg"

im = Image.open(filename)
print ("%s - %s" % (im.size, im.mode))

# show the image
im.show()

# close the file
im.close()
```



- As a start we need to import it:
import Image
- We can open images with
im = Image.open(fullname)
- Then we can display the image using
im.show()
- Print some info about the image using
im.size and **im.mode**

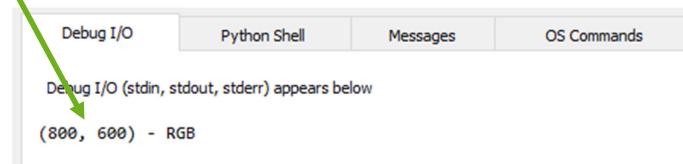




Image Processing

```

import os
from PIL import Image, ImageFilter

filename = "img/clungup.jpg"

im = Image.open(filename)

out = im.filter(ImageFilter.BLUR)

im.show()
out.show()

```



Pillow has many conversion and filters, to use filters we need to extend our import:
 from PIL import Image,
 ImageFilter

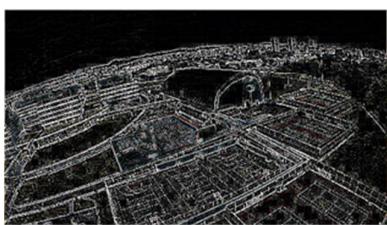
The way you can apply filters is :
 out = im.filter(ImageFilter.BLUR)

Try other different filters!

image.blur_filter.py 39



Image Processing - filters



```
image = ImageOps.grayscale(image)
```



```
image = image.filter(ImageFilter.FIND_EDGES)
```

```
image = image.filter(ImageFilter.CONTOUR)
```



```
image = ImageOps.solarize(image)
```



* Remember to include
 ImageOps in your import statement



Image Processing - filters

```

import os
from PIL import Image, ImageFilter, ImageOps

filename = "img/clungup.jpg"

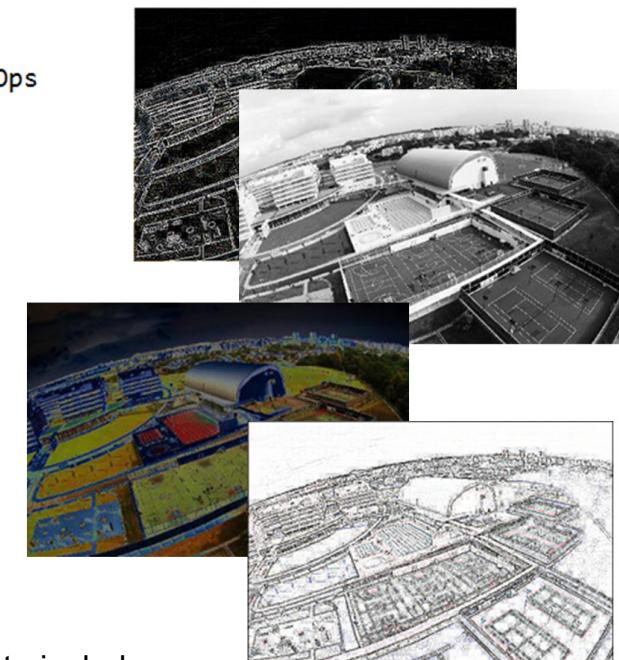
im = Image.open(filename)

# Filter
#out = im.filter(ImageFilter.BLUR)
#out = im.filter(ImageFilter.FIND_EDGES)
#out = im.filter(ImageFilter.CONTOUR)

# ImageOps
out = ImageOps.grayscale(im)
#out = ImageOps.solarize(im)

im.show()
out.show()

```



⚠ * Remember to include
ImageOps in your import statement

image_other_filters.py 41



Image Processing – Rotating

Flipping the image horizontally or vertically
`out = im.transpose(Image.FLIP_LEFT_RIGHT)`
`out = im.transpose(Image.FLIP_TOP_BOTTOM)`

Flip images

Rotating the image
`out = im.transpose(Image.ROTATE_90)`
`out = im.transpose(Image.ROTATE_180)`
`out = im.transpose(Image.ROTATE_270)`

Rotate images

Contrast

First add ImageEnhance to our imports:
`from PIL import Image, ImageFilter, ImageEnhance`

Then:

`enh = ImageEnhance.Contrast(im)` make image brighter by changing the contrast
`out = enh.enhance(1.3)`

image_rotate.py 42



Image Processing - Writing

```

import os
from PIL import Image, ImageFilter, ImageOps

filename = "clungup.jpg"

src_folder = "img/"
out_folder = "out/"

im = Image.open(src_folder + filename) # img/clungup.jpg
out = im.filter(ImageFilter.BLUR)

outfilename = out_folder + filename # out/clungup.jpg

out.save(outfilename)

```

You can see the image, but it's not being saved !

All you need to do to save the images in the "out" folder is:
out.save(the name of the output file)

image_save.py 43



Image Processing - Converting

```

import os
from PIL import Image, ImageFilter, ImageOps

filename = "clungup.jpg"

src_folder = "img/"
out_folder = "out/"

im = Image.open(src_folder + filename) # img/clungup.jpg
out = im.filter(ImageFilter.BLUR)

# split the filename and the extension
f, e = os.path.splitext(filename)

# add the gif extension to the filename
fname2 = f + ".gif"

outfilename = out_folder + fname2 # out/clungup.gif

out.save(outfilename)

```

`os.path.splitext(file)` returns a list.
 We are only interested in `f` which is the first item in the list.

image_convert_ext.py 44



Image Processing – Watermark

Create the mark image →
You can reduce the size to 100,100

```
mark = Image.open("img\\watermark.png")
mark = mark.resize((100,100))
```

Create a new function called

```
def watermark(im, mark, position):
```

...

It takes the original image, the watermark image and the desired position that we want the watermark to appear. The function will return the result.

We can use this function like:

```
watermark(im, mark, (0, 50)).show()
```

or

```
imOut = watermark(im, mark, (0,50))
imOut.save(fileOut)
```

Maybe you want to leave a small footprint on your images, called watermark.

In this case we can use the \\img\\watermark.png and place it in each image on the bottom right.

Copyright
@RP

45



Image Processing - Watermark

```
from PIL import Image

def watermark(im, mark, position):
    layer = Image.new("RGBA", im.size, (0,0,0,0))
    layer.paste(mark, position)
    return Image.composite(layer, im, layer)

im = Image.open("img\\clungup.jpg")
mark = Image.open("img\\watermark.png")
mark = mark.resize((100,100))
mark.putalpha(128)

out = watermark(im, mark, (0,0))
out.show()
```

First we need to create a new layer with the size of the original image.

Then we paste the watermark image at the desired position and we return the composite.

Finally we merge the image and the layer together and return the result.

Then you can use it like this:





Use Case I: Batch Resize

1. Find all the files in “img” folder with “.jpg” extension
2. Resize all the file to 60 x 90.
3. Save all the files to the resized folder

```
import os
from PIL import Image, ImageFilter, ImageOps

files = os.listdir('img')
size = 60, 90

for file in files:
    if file.lower().endswith('.jpg'):
        im = Image.open("img/" + file)
        im.thumbnail(size, Image.ANTIALIAS)
        im.save("resized/" + file, "JPEG")
```

image_batch_resize.py 47



Use Case II: Batch Rename

1. Find all the files in “img” folder with “.jpg” extension
2. Copy all the files to the renamed folder
3. Rename all the files with the “s-” prefix.

```
import os
import shutil

files = os.listdir('img')

for file in files:
    if file.lower().endswith('.jpg'):
        shutil.copyfile("img/" + file, "renamed/s-" + file[:-4] + ".jpg")
```

image_batch_rename.py

48



Exercise on Pillow

- **Write Python code to:**
 - Resize all jpg images in “img” folder to half its original size
 - Place a watermark at the lower right of the image
 - Save/store the modified images into “resized” folder
- **Let's discuss this at 1.55pm 😊**
- **Let me know if you've completed**

49



Web Automation with Python

50



Connecting to the Web

- requests – download files and web pages from the Web

pip install requests

```
import requests

url="https://api.data.gov.sg/v1/environment/24-hour-weather-forecast"
req=requests.get(url)
print(req.text)
```



Get the required information from the given URL

web_request.py 51



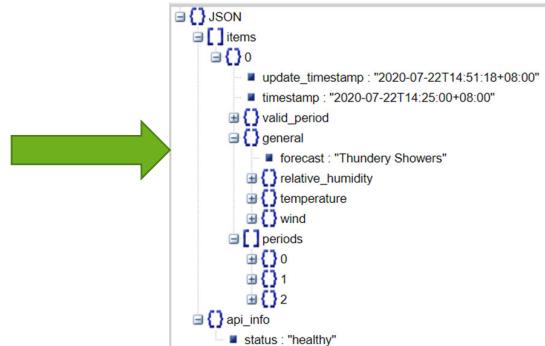
Connecting to the Web

- Data is in JSON format
 - Use a JSON formatter tool to present the data in a nicer form

<http://jsonviewer.stack.hu/>

```
import requests

url="https://api.data.gov.sg/v1/environment/24-hour-weather-forecast"
req=requests.get(url)
print(req.text)
```





Connecting to the Web

- To work with JSON data, import json first
- Use json.loads() to load the data in JSON format
- Extract and retrieve the required data

```
import json
import requests

url="https://api.data.gov.sg/v1/environment/24-hour-weather-forecast"
req=requests.get(url)

data = json.loads(req.text)

# print update timestamp
update_time = data["items"][0]["update_timestamp"]
print("Update time: " + update_time)

# print forecast
forecast = data["items"][0]["general"]["forecast"]
print("Forecast: " + forecast)
```

Update time: 2020-07-22T14:51:18+08:00
Forecast: Thundery Showers

web_json.py 53



Exercise

• Car Park Availability Data:

- 1.url: <https://api.data.gov.sg/v1/transport/carpark-availability>
- 2.Write the code to get the timestamp and the Carpark Number for the first set of carpark data.
- 3.Print out the result as shown.

```
{
  - items: [
    - {
      timestamp: "2021-08-19T13:23:28+08:00",
      - carpark_data: [
        - {
          - carpark_info: [
            - {
              total_lots: "105",
              lot_type: "C",
              lots_available: "0"
            }
          ],
          carpark_number: "HE12",
          update_datetime: "2021-08-19T13:10:03"
        },
        - {
          - carpark_info: [
            -
```

Update time: 2021-08-19T11:49:27+08:00
Carpark number: HE12

web_json_carpark.py 54



Exercise

- URL: <https://wttr.in/Singapore?format=j1>

```

▼ current_condition:
  ▼ 0:
   FeelsLikeC: "43"
   FeelsLikeF: "109"
   cloudcover: "75"
   humidity: "53"
   localObsDateTime: "2021-10-08 02:01 PM"
   observation_time: "06:01 AM"
   precipInches: "0.0"
   precipMM: "0.1"
   pressure: "1009"
   pressureInches: "30"
   temp_C: "33"
   temp_F: "91"
   uvIndex: "6"
   visibility: "10"
   visibilityMiles: "6"
   weatherCode: "116"
   ▼ weatherDesc:
      ▼ 0:
        value: "Partly cloudy"
    ▾ weatherIconUrl:

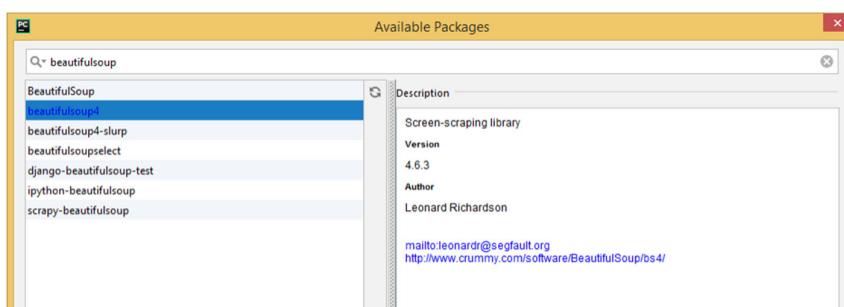
```



Connecting to the Web

- Beautiful Soup – a third party module that parses HTML (web pages)

Web Scraping – download and process Web content
- Install Beautiful Soup 4 - **pip install beautifulsoup4**





Connecting to the Web

- What's the URL?

<https://www.fortytwo.sg/dining/dining-tables/landon-regular-dining-table-coffee.html>

Enquiries: +65 6777 7667 | Mon - Fri (10am - 6pm)

FORTYTWO

Search furniture, mattress, home & decor...

New Furniture Bedding & Mattresses Décor | Essentials Kitchen | Dining Lightings | Fans Sale

Home > Dining Room Furniture > Dining Tables > Landon Regular Dining Table Coffee

Landon Regular Dining Table Coffee

Qty: 1

★★★★★ 6 customer reviews

\$S\$129.90
S\$69.90

Add to Cart

Add to Wishlist Email to a Friend

Warranty: 1 Year

Standard Delivery 42EXPR

100 Day Free Returns Free Assembly Included

57



Connecting to the Web

- Get the url

<https://www.fortytwo.sg/dining/dining-tables/landon-regular-dining-table-coffee.html>

- Select the element to extract
 - right-click -> "Inspect"
 - hover to "Copy"
 - click on "Copy selector"

Responsive 315 x 443

Elements Console Sources

Up To 60% OFF Dining Room Categories

FORTYTWO

Search furniture, mattress, home & decor...

Landon Regular Dining Table Coffee

★★★★★ 6 customer reviews

span.price

Payment Handler debugging

Copy selector

58



Connecting to the Web

```
from urllib.request import Request, urlopen
from bs4 import BeautifulSoup

site= "https://www.fortytwo.sg/dining/dining-tables/landon-regular-dining-table-coffee.html"
hdr = {'User-Agent': 'Mozilla/5.0'}
req = Request(site,headers=hdr)
page = urlopen(req)
soup = BeautifulSoup(page, 'html.parser')

elements = soup.select("#product-price-46728") # $69.90
print(elements)
price = elements[0].text
print("Current Price: " + elements[0].text)

#old-price-46728
elements = soup.select("#old-price-46728") # $129.90
print("\nOld Price: " + elements[0].text)

elements = soup.select('div[class="delivery est-date"]') # Earliest by Sunday, 31 May 2020
print(elements[0].text)
```

Debug I/O Python Shell Messages OS Cc
 Debug I/O (stdin, stdout, stderr) appears below
 Current Price: S\$69.90
 Old Price: S\$129.90
 Delivery Date:
 Earliest by
 Sunday, 31 May 2020

web_scrap.py 59



Exercise

- Table Tennis Bat Price:**

- 1.url: https://www.tabletennis11.com/other_eng/butterfly-viscaria
- 2.Write the code to get the price of the table tennis bat
- 3.Print out the result as shown.

tabletennis11

CRAZY ABOUT TABLE TENNIS

Home > Butterfly Viscaria

Butterfly Viscaria

4.5 stars 20 reviews

€133.25

Handle type

Additional FREE Services

Blade Lacquering

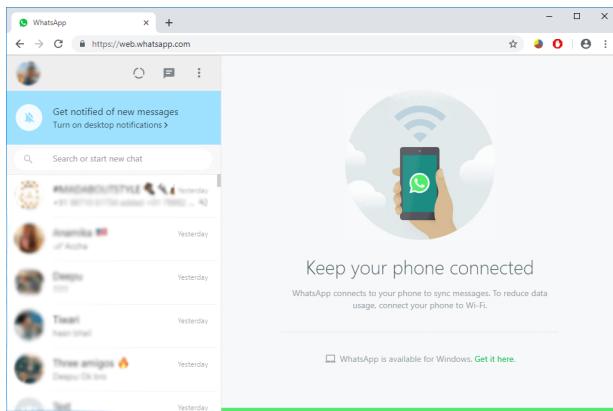
Current Price:
€133.25

web_scrap_tabletennis.py 60



Sharing other Use Cases

- Using another library: selenium
 - Filling up google form
 - Sending WhatsApp message



selenium python automation

Name	<input type="text" value="Your answer"/>
Gender	<input type="radio"/> Male <input type="radio"/> Female

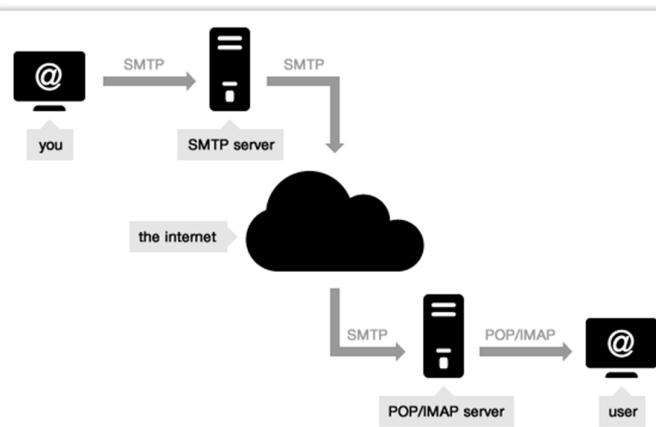
61



Email Automation with Python



Send Email



- SMTP (Simple Mail Transfer Protocol) is used for sending and delivering from a client to a server via port 25, 465 or 587: it's the **outgoing server**.

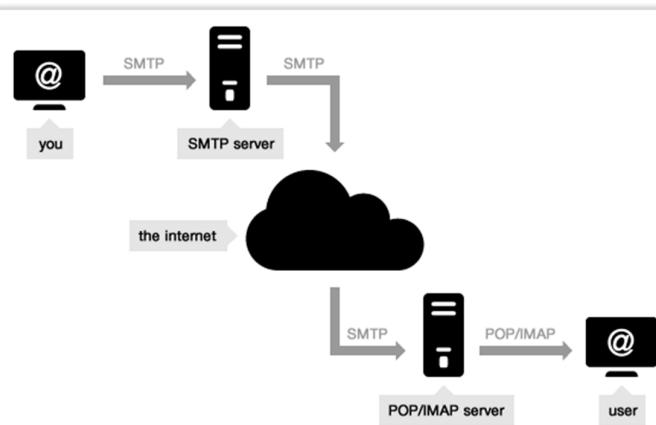
- IMAP and POP are two methods to access email. IMAP is the recommended method when you need to check your emails from several different devices, such as a phone, laptop, and tablet.

<https://www.mailgun.com/blog/which-smtp-port-understanding-ports-25-465-587/>
<https://serverfault.com/what-is-smtp-server/>

63



Send Email



- **Note:** The SMTP servers used when you send your emails- Hotmail, Gmail , Yahoo Mail – are **shared among users**
- Common providers establish some **strict limits** on the number of emails you can send (e.g. Yahoo's restriction is 100 emails per hour).
- If you plan to send a bulk email or set up an email campaign you should opt for a professional outgoing email server like turboSMTP,
- which guarantees a controlled IP and ensure that all your messages reach their destination.

64



Send Email using Gmail

Incoming Mail (IMAP) Server	imap.gmail.com Requires SSL: Yes Port: 993
Outgoing Mail (SMTP) Server	smtp.gmail.com Requires SSL: Yes Requires TLS: Yes (if available) Requires Authentication: Yes Port for SSL: 465 Port for TLS/STARTTLS: 587
Full Name or Display Name	Your name
Account Name, User name, or Email address	Your full email address
Password	Your Gmail password

Note: If you are using your office network, most port numbers, including 587, may be blocked.

65



Send Email using Gmail

- Import smtplib module
- Specify Gmail email & password, receiver's email address, email title & content
- Connect to SMTP server using Port 587
- Call starttls() to enable encryption for your connection
- Login using email and password
- Call sendmail()
- Call quit() to disconnect from the SMTP server

```

import smtplib

sender_email_address = "your_email_address@gmail.com"
sender_email_password = "xxxxxxxxxxxxxx"
receiver_email_address = "another_email_address@gmail.com"
email_title_content = "Subject: Sending Email Using Python\nThis is a test email."
email_title_content = "Subject: Sending Email Using Python\nThis is a test email."

print("Trying to connect to Gmail SMTP server")
smtpObj = smtplib.SMTP("smtp.gmail.com", 587)
smtpObj.starttls()

print("Connected. Logging in...")
smtpObj.login(sender_email_address, sender_email_password)

smtpObj.sendmail(sender_email_address, receiver_email_address, email_title_content)
print("Email sent successfully...")

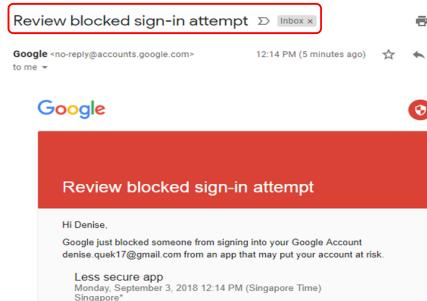
smtpObj.quit()
    
```

➤ The start of the email body must begin with "Subject: " for the subject line. The "\n" newline character separates the subject line from the main body content.



Send Email using Gmail

- Google may block attempted sign-in from unknown devices that don't meet their security standards!



```
C:\Users\denise quek\AppData\Local\Programs\Python\Python37\python.exe D:/CET_Python/Denise/TestEmail.py
Trying to connect to Gmail SMTP server
Connected. Logging in...
Traceback (most recent call last):
  File "D:/CET_Python/Denise/TestEmail.py", line 13, in <module>
    smtpObj.login(sender_email_address, sender_email_password)
  File "C:\Users\denise quek\AppData\Local\Programs\Python\Python37\lib\smtplib.py", line 730, in login
    raise last_exception
  File "C:\Users\denise quek\AppData\Local\Programs\Python\Python37\lib\smtplib.py", line 721, in login
    initial_response_ok=initial_response_ok)
  File "C:\Users\denise quek\AppData\Local\Programs\Python\Python37\lib\smtplib.py", line 642, in auth
    raise SMTPAuthenticationError(code, resp)
smtplib.SMTPAuthenticationError: (534, b'5.7.9 Application-specific password required. Learn more at\nhttps://support.google.com/accounts/answer/1858331?hl=en')

Process finished with exit code 1
```

67



Send Email using Gmail

Steps To Create Google App Password

Step 1: Login to Gmail. Go to Account → Signing in to Google

Step 2: Make sure that 2-Step Verification is on

Step 3: Create an App password

The screenshot shows two pages of the Google 'App passwords' interface:

- Left Page (App passwords):**
 - Header: App passwords
 - Text: App passwords let you sign in to your Google Account from apps on devices that don't support 2-Step Verification. You'll only need to enter it once so you don't need to remember it. [Learn more](#)
 - Form fields: You don't have any app passwords. Select the app and device you want to generate the app password for. Options: Mail (selected) and Windows Computer.
 - Buttons: GENERATE
- Right Page (Generated app password):**
 - Header: Generated app password
 - Text: Your app password for Windows Computer: **applepig@applepig-laptop:123456**
 - Text: How to use it:
 - Open the "Mail" app.
 - Open the "Settings" menu.
 - Select "Accounts" and then select your Google Account.
 - Replace your password with the 16-character password shown above.
 - Text: Just like your normal password, this app password grants complete access to your Google Account. You won't need to remember it, so don't write it down or share it with anyone. [Learn more](#)
 - Buttons: DONE

68



Send Email using Gmail

The screenshot shows the Google Account Security settings page. On the left, a sidebar lists options: Home, Personal info, Data & personalisation, **Security** (which is selected and highlighted with a red box), People & sharing, Payments & subscriptions, Help, and Send feedback. The main content area is titled 'Security' with the subtitle 'Settings and recommendations to help you keep your account secure'. It features two sections: 'Security issues found' (with a warning icon) and 'Secure account'. Below these is a section titled 'Signing in to Google' which includes 'Password' (last changed 10 Jan 2018) and '2-Step Verification' (which is turned on, indicated by a checked checkbox and the word 'On'). There is also a section for 'App passwords'. At the bottom, there's a link to 'Ways that we can verify that it's you'.

69



Send Email using Gmail

- Replace your actual password with the App password

```
import smtplib

sender_email_address = "your_email_address@gmail.com"
sender_email_password = "xxxxxxxxxxxxxx"
receiver_email_address = "another_email_address@gmail.com"
email_title_content = "Subject: Sending Email Using Python\nThis is a test email."
```

- Run your email program

```
C:\Users\denise_quek\AppData\Local\Programs\Python\Python37\python.exe D:/CET_Python/Denise/TestEmail.py
Trying to connect to Gmail SMTP server
Connected. Logging in...
Email sent successfully...

Process finished with exit code 0
```

70



Use Case: Send emails to students

- Send email to students who were absent

```

1  #! python3
2
3  import openpyxl, smtplib
4
5  def sendEmail(name, emailTo):
6      email_body = "Subject: Your attendance. \nDear %s, \nYou were absent for class.\n" %(name)
7
8      smtpObj = smtplib.SMTP("smtp.gmail.com", 587)
9      smtpObj.starttls()
10     smtpObj.login("code.musically@gmail.com", "xxxxxxxxxxxx")
11     smtpObj.sendmail('code.musically@gmail.com', emailTo, email_body)
12
13     smtpObj.quit()
14
15
16     workbook = openpyxl.load_workbook("D:\CET_Python\students_attendance.xlsx")
17     sheet = workbook["Sheet1"]
18
19     max_row = sheet.max_row
20     max_column = sheet.max_column
21
22     for i in range(1, max_row+1):
23
24         attendance = sheet.cell(row=i, column=3).value
25
26         if attendance == "Absent":
27             name = sheet.cell(row=i, column=1).value
28             email = sheet.cell(row=i, column=2).value
29
30             print(name + " is absent.")
31             sendEmail(name, email)
32             print("Email sent to " + email)
33             print()
34

```

	A	B	C
1	Student	Email	Status
2	Alicia	code.musically@gmail.com	Present
3	Bryan	code.musically@gmail.com	Present
4	Carol	code.musically@gmail.com	Absent
5	David	code.musically@gmail.com	Absent
6	Evelyn	code.musically@gmail.com	Present
7			

71



Sharing other Use Cases

- Sending Emails using Outlook
- Create Appointment using Outlook

72



Other Python Libraries

- Play music using winsound
- Generate QR code using qrcode
- Face detection using opencv



73



Where to go from here ?

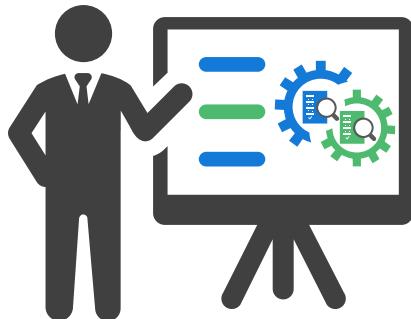


Think Python is an introduction to Python programming for beginners. It starts with basic concepts of programming, and is carefully designed to define all terms when they are first used and to develop each new concept in a logical progression. Larger pieces, like recursion and object-oriented programming are divided into a sequence of smaller steps and introduced over the course of several chapters.

Think Python is a Free Book. It is available under the [Creative Commons Attribution-NonCommercial 3.0 Unported License](#), which means that you are free to copy, distribute, and modify it, as long as you attribute the work and don't use it for commercial purposes.
<http://greenteapress.com/thinkpython/thinkpython.pdf>



Thank you



Email:

jason_lim@rp.edu.sg

wong_hau_shian@rp.edu.sg

Learning material & source code:

<https://bit.ly/IPP-MHA>

OFFICIAL (CLOSED) \ NON-SENSITIVE



End of Course Survey



<https://for.edu.sg/mha-survey-python>

**Please kindly complete this survey before leaving
the class**