Introductory Programming using Python
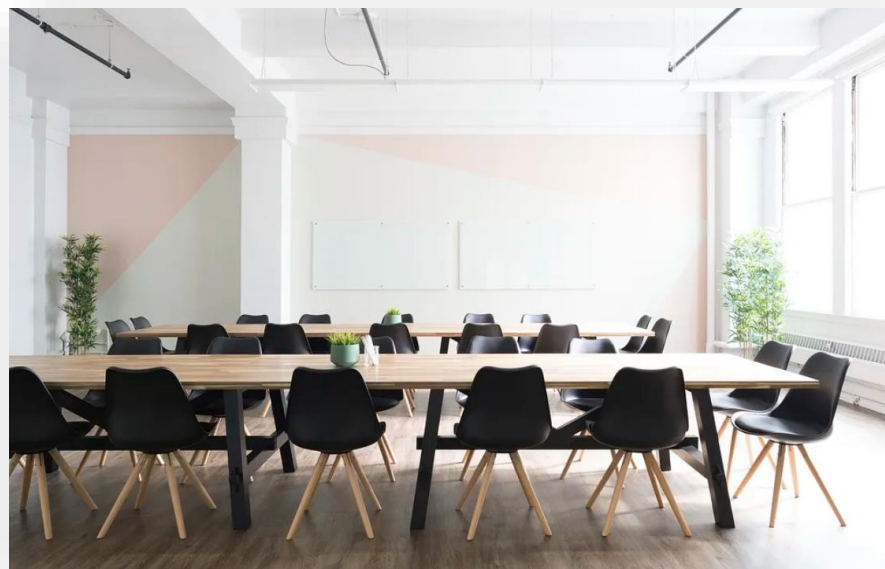
Day 2

Republic Polytechnic

Ver 3.3

Welcome and admin matters

- Please ensure that:
  - your attendance has been captured (via QR code scanning)
  - 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
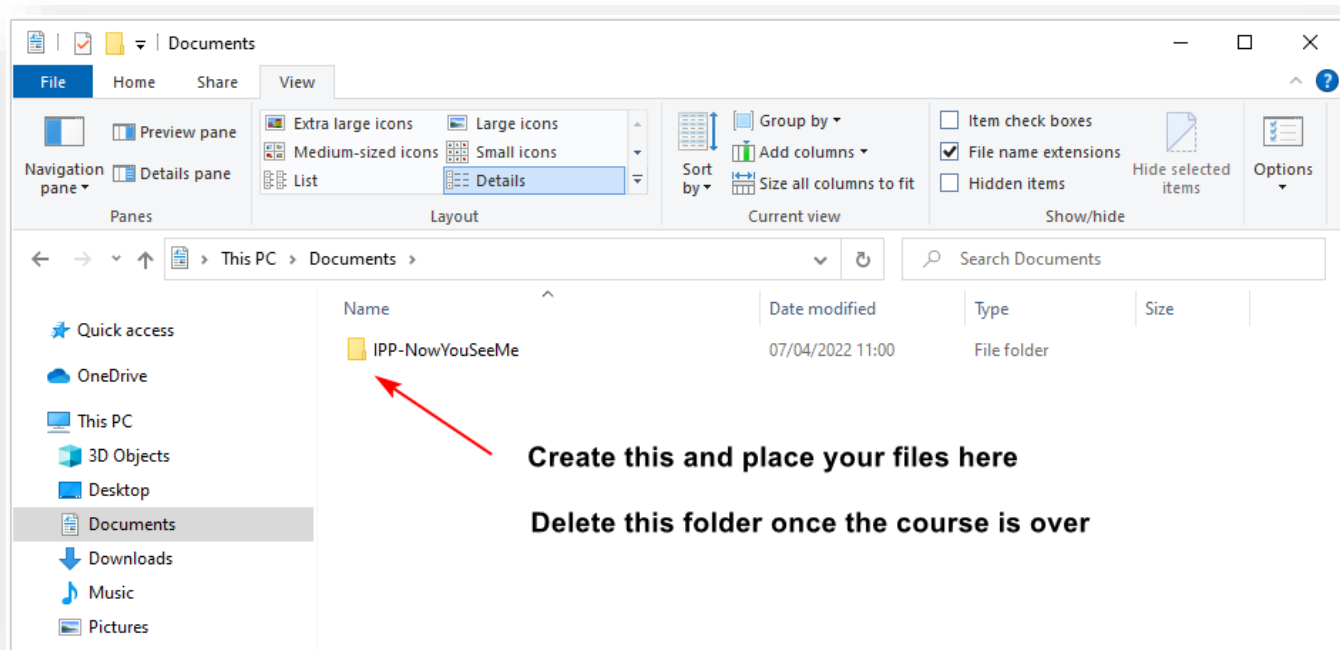
Objectives of MHQ Digital Upskilling

1. **Gain awareness of digital tools**, as they become essential in the future

2. Equip officers with skills and knowledge to **digitalise our work processes** where applicable

MHQ officers should:

1. Be **OPEN** to **POSSIBILITIES** and explore use cases

2. **PARTICIPATE ACTIVELY** in activities and discussions to maximise learning time

Overview: Day Two

Morning	Afternoon
• String functions • String formatting • Dictionary • Working with Excel	• 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   7

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

FIX_ME exercise

split or slice?

- Walkthrough on the following use case:
  - Extract the digit part of a Singapore NRIC number
  - Extract the user id from an email address

- Let's solve it together

01_split_or_slice.py   9

Exercise – Find Longest Word

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

  Hint: Use split(), repetitions

15 mins

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 uppercase
    - *String*.replace(*x*, *y*) – Returns a String with all occurrences of *x* in the original String replaced with *y*

String formatting

- **String format() method**
    - Use {} as a place holder for the text to be printed.
    - Call the string dot format method
    - Pass the desired value into the method

More about string formatting technique can be found here: https://docs.python.org/3/library/string.html

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

```python
print(name +"'s temperature is " + str(diff_from_369) + " degree from 36.9 degree celsius")
```

```python
#Using String formatting:
print("{}'s temperature is {:.2f} degree from 36.9 degree celsius".format(name, diff_from_369))
```

- **Formatting types**
    - :d   Decimal
    - :f   Float
    - :+   plus sign if results is positive
    - :<   Left aligns the result

12

Exercise – String formatting (1)

- Try this out yourself!
  - Open *string-format1.py*
  - Change the values for the value line
  - Execute and observe the effect

```python
import math

a = math.pi
b = 5
c = 'python'

line = "{} {:.2f} {}".format(c, a , b)
print(line)

line = "{:0>5}".format(b)
print(line)
```

```
python 3.14 5
00005
```

5 mins

string-format1.py   13

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
```

10 mins

string-format2.py

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

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

# add element
members['john'] = 20

# edit element
members['mary'] = 25

# delete element
del members['alan']

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

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

Key "john" **NOT IN** the *members* dict. This **ADDS** the key "john" and value 20 pair

Key "mary" **IS IN** the *members* dict. This **EDITS** the value of key "mary"

16

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:

```python
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) )
```

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

5 mins

dict-ops.py   18

Excel Automation using Python

Typical Workflow for Excel Automation

1 You are given some data in a spreadsheet

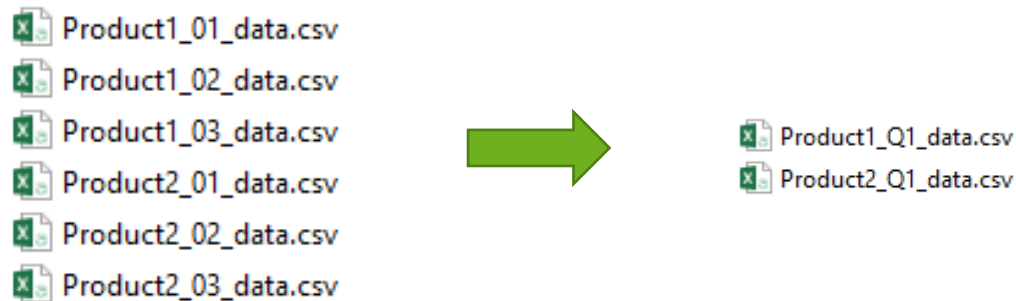**2** You want to do some or all of the following
- Analyse the data
- Manipulate the data

3 Output the processed data in another spreadsheet

Excel Use Case (Walkthrough)

- Finance department receives monthly sales data from the various product lines.

- Finance department would like to:
  - ➤ Task 1: Merge the product specific monthly sales files into quarterly files to observe the trends

Product1_01_data.csv
Product1_02_data.csv
Product1_03_data.csv
Product2_01_data.csv
Product2_02_data.csv
Product2_03_data.csv

→

Product1_Q1_data.csv
Product2_Q1_data.csv

OFFICIAL (CLOSED) \ NON-SENSITIVE

Excel Use Case (Walkthrough)

- Finance department would like to:

  ➢ Task 2: Simple data cleansing to remove duplicates or empty lines

Date	Customer	Order #	Revenue	Sales Rep Name	Commission Rate
1/1/2019	Alpha Soft	i-100001	$202.33	Jimena Carrillo	10%
2/1/2019	Soul Soluti	i-100002	$592.22	Emma Ford	10%
3/1/2019	Ironavigati	i-100003	$59.44	Lily Grant	4%
4/1/2019	Ironavigati	i-100004	$283.33	Jamie Webb	10%
5/1/2019	Soul Soluti	i-100005	$19.00	Jamie Webb	0%
6/1/2019	Alpire	i-100006	$43.03	Emma Ford	0%
7/1/2019	Oystertain	i-100007	$23.34	Jimena Carrillo	0%
8/1/2019	Titanium S	i-100008	$230.30	Lily Grant	10%
9/1/2019	Surprise M	i-100009	$430.33	Lily Grant	10%
10/1/2019	Apexshow	i-100010	$492.32	Emma Ford	10%
10/1/2019	Apexshow	i-100010	$492.32	Emma Ford	10%
10/1/2019	Ironavigati	i-100011	$23.34	Jimena Carrillo	0%

Excel Use Case (Walkthrough)

- Finance department would like to:

  ➢ Task 3: Perform mathematical calculations and create new columns to store the calculated information.

Date	Customer Na	Order #	Revenue	Sales Rep Name	Commission Rate	Total Commission
1/1/2019	Alpha Softwa	i-100001	$202.33	Jimena Carrillo	10%	$20.23
2/1/2019	Soul Solutior	i-100002	$592.22	Emma Ford	10%	$59.22
3/1/2019	Ironavigatior	i-100003	$59.44	Lily Grant	4%	$2.38
4/1/2019	Ironavigatior	i-100004	$283.33	Jamie Webb	10%	$28.33
5/1/2019	Soul Solutior	i-100005	$19.00	Jamie Webb	0%	$0.00
6/1/2019	Alpire	i-100006	$43.03	Emma Ford	0%	$0.00
7/1/2019	Oystertainm	i-100007	$23.34	Jimena Carrillo	0%	$0.00

Excel Use Case (Walkthrough)

- Finance department would like to:

  ➢ Task 4: Glean some simple insights.

    ➢ Total sales of product

    ➢ Highest performing Sales rep

    ➢ Customers with highest purchase value

Excel: Task 1

- **Merge the monthly files into quarterly files to observe trends**

- New concepts:
  - Libraries:
    - glob and pandas (https://pypi.org/project/pandas/)

  - Specifying the location of the data files:

Python Package Index

- Many of the tasks to be performed using a programming language are non unique.

- To make programming more efficient, there exists many pre-written, reusable chunks of codes that allow a programmer to quickly get its code up and working.

- These pre-written, reusable chunks of codes are known as modules (e.g. single functionality) or library (e.g., collection of functionalities)

- https://pypi.org/ : is a repository of such codes for the Python Programming Language
    - E.g. send2trash, pillow, pandas, numpy etc

- Installation is easy done with the following command
    pip install <software_package>

- Installed packages can be found at:
    C:\python38\Lib\site-packages

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

Alternative

- If command prompt is inaccessible, we can try to install the packages programmatically

- Example below shows package installation via Python code

```
import subprocess
import sys

subprocess.check_call([sys.executable, "-m", "pip", "install", "openpyxl"])
subprocess.check_call([sys.executable, "-m", "pip", "install", "pillow"])
```

File Tree

- **Windows**
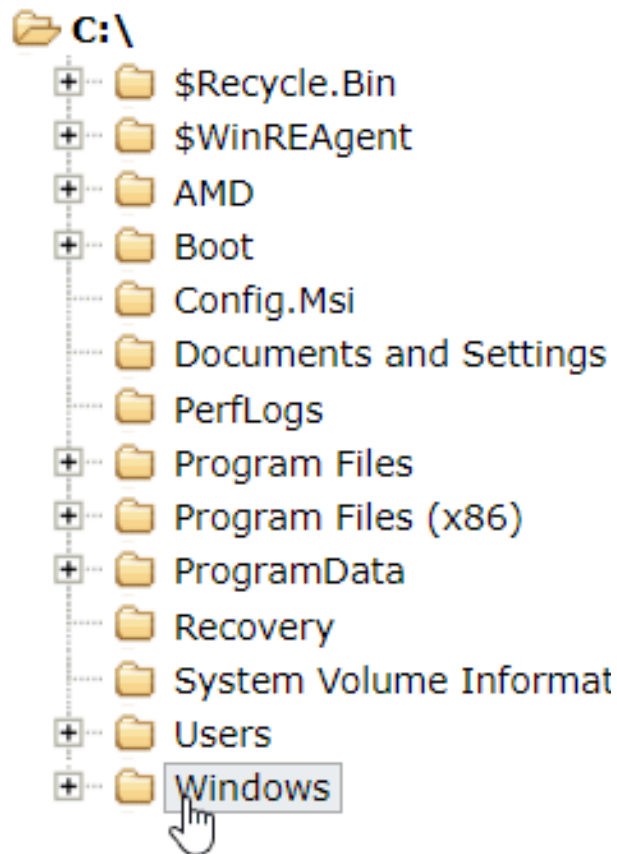
- **Linux**
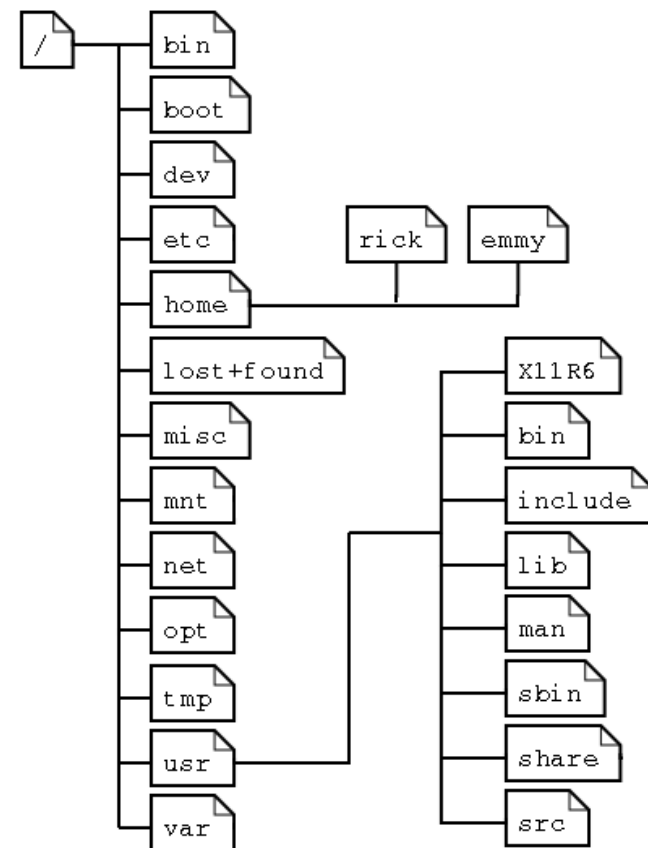
File Paths

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.

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**.

Img credit: https://automatetheboringstuff.com/chapter8/

Excel: Task 1 (Code)

```python
#1. Import the necessary libraries
import glob    #search files that matches required patterns
import pandas as pd  #analyze data

# 2. Define path to the csv data files
path = './Data/'

# 3. Create a list using glob to store file names starting with
#     Product1 and ending with .csv
fileList = glob.glob(path + "Product1*.csv")
print (fileList)

# 4. Create empty list to store the csv file as pandas dataframes
dfList = []
```
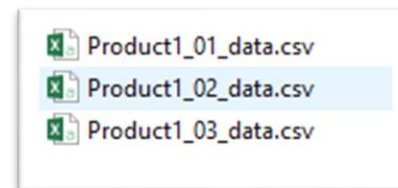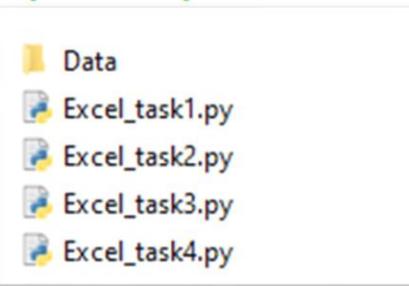
Data
Excel_task1.py
Excel_task2.py
Excel_task3.py
Excel_task4.py

Product1_01_data.csv
Product1_02_data.csv
Product1_03_data.csv

Excel_task1.py

Excel: Task 1 (Code)

```python
# 5. Cycle through the files
for file in fileList:

    #read the file and convert to pandas dataframe
    data = pd.read_csv(file, encoding='utf-8')

    #append to the list of dataframes
    dfList.append(data)

# merge the data in the csvList into a single pandas dataframe
csvMerged = pd.concat(dfList)

# 6. Output the pandas dataframes as csv
csvMerged.to_csv(path+ 'Product1_Q1_data.csv', index=False)
```

Excel: Task 2

- **Clean data to remove duplicates or empty lines**

- New concepts:
  - Reading the documentations to identify useful functions

- Useful Pandas functions:
  - dropna() to remove rows that are empty
  - https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.dropna.html#pandas.DataFrame.dropna

  - drop_duplicates() to remove rows that are exactly the same
  - https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.drop_duplicates.html

Excel: Task 2 (Code)

```python
# 5. Cycle through the files
for file in fileList:

    #read the file and convert to pandas dataframe
    data = pd.read_csv(file, encoding='utf-8')

    #Task 2: remove empty lines
    data.dropna(axis = 0, how='all', inplace=True)

    #Task 2: remove duplicates entries based on names
    data.drop_duplicates(inplace=True)

    #append to the list of dataframes
    dfList.append(data)

# merge the data in the csvList into a single pandas dataframe
csvMerged = pd.concat(dfList)
```

Excel_task2.py

Excel: Task 3

- **Perform mathematical calculations and create new columns to store the calculated information.**

- New concepts:
  - Specifying the required columns
  - Striping the data of special characters
  - Converting the data type of the columns
  - Performing simple mathematical functions on column values
  - Creating a new column to store the values of the calculations

Excel: Task 3 (Code)

```python
#Task 3: Calculate Total Commission
data['Revenue'] = data['Revenue'].str.replace('$', '')
data['Revenue'] = data['Revenue'].astype(float)

data['Commission Rate'] = data['Commission Rate'].str.replace('%', '')
data['Commission Rate'] = data['Commission Rate'].astype(float)

#create a new column and perform the mathematical calculations
data['Total Commission'] = data['Revenue']*(data['Commission Rate']/100)
```

Excel_task3.py

Excel: Task 4

- **Task 4: Glean some simple insights.**

  ➢ Total sales of a product

  ➢ Single largest customer order

  ➢ Highest performing Sales rep of a product

- New concepts:

  - Extract values based on conditions

  - Group data by conditions

Excel: Task 4 (Code)

```python
#Task 4: Generating insights

# Total sales of a product in the quarter
totalSales=sum(data['Revenue'])
print ("Total sales of product 1 in this quarter is ${:.2f} \n".format(totalSales))

# What is the single largest customer order?
largestOrder = data['Revenue'].max()
customer = data[data['Revenue']==largestOrder]['Customer Name'].to_string(index=False)
salesRep = data[data['Revenue']==largestOrder]['Sales Rep Name'].to_string(index=False)
print("Customer with singles largest order is {} by {} served by {}\n".format(largestOrder, customer, salesRep))

#Which sales rep brought in the highest revenue
salesByRep = data.groupby('Sales Rep Name')['Revenue'].sum()
salesByRepByProportion = salesByRep/totalSales
print(salesByRepByProportion)
```

Excel_task4.py

Summary

Lunch Break

What is Python?

Src: https://youtu.be/Y8Tko2YC5hA
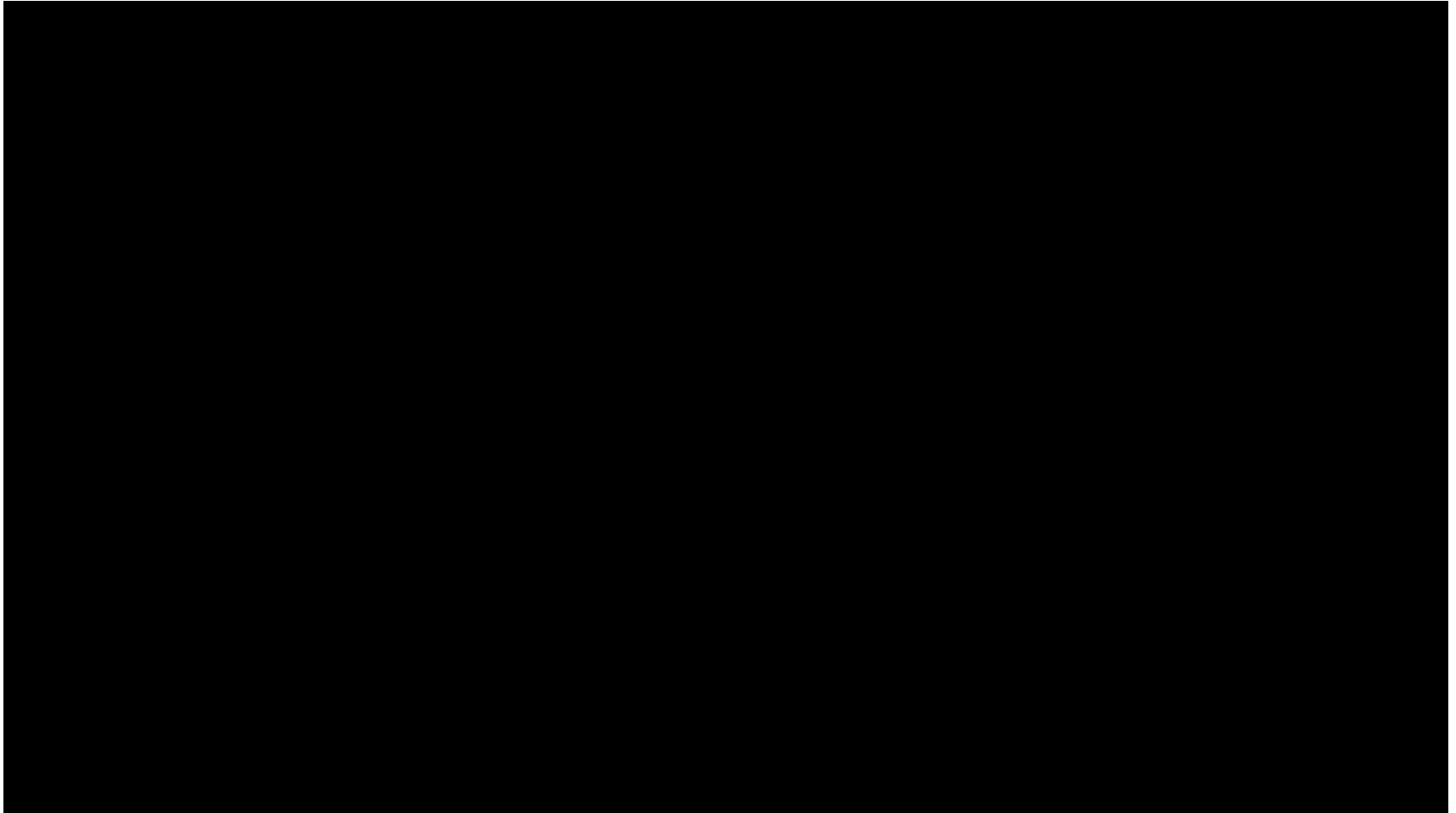
Web Automation with Python

Connecting to the Web

- requests – download files and web pages from the Web

pip install requests

Get the required information from the given URL

```
import requests

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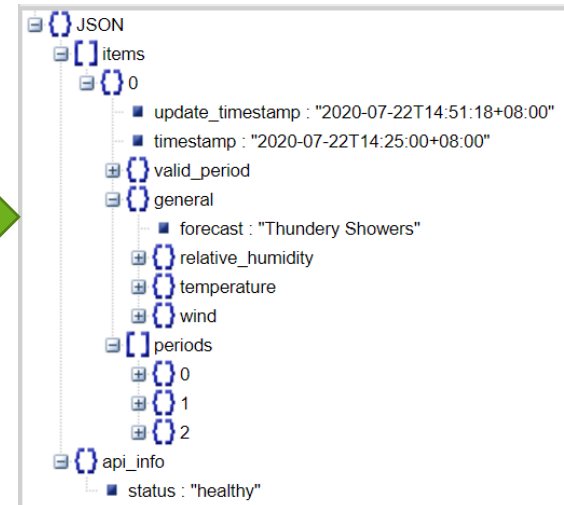

web_request.py   43

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/

```python
import requests

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

{"items":[{"update_timestamp":"2020-07-22T14:51:18+08:00","timesta
"2020-07-22T14:25:00+08:00","valid_period":{"start":"2020-07-22T12
"end":"2020-07-23T12:00:00+08:00"},"general":{"forecast":"Thundery
"relative_humidity":{"low":70,"high":95},"temperature":{"low":22,'
"wind":{"speed":{"low":10,"high":20},"direction":"ESE"}},"periods'
"start":"2020-07-22T12:00:00+08:00","end":"2020-07-22T18:00:00+08:
:{"west":"Moderate Rain","east":"Moderate Rain","central":"Light H
"Light Rain","north":"Light Rain"}},{"time":{"start":"2020-07-22T1
,"end":"2020-07-23T06:00:00+08:00"},"regions":{"west":"Partly Clou
"east":"Partly Cloudy (Night)","central":"Partly Cloudy (Night)","
"Partly Cloudy (Night)","north":"Partly Cloudy (Night)"}},{"time":
"2020-07-23T06:00:00+08:00","end":"2020-07-23T12:00:00+08:00"},"re

JSON tree view:
- JSON
  - items
    - 0
      - update_timestamp : "2020-07-22T14:51:18+08:00"
      - timestamp : "2020-07-22T14:25:00+08:00"
      - valid_period
      - general
        - forecast : "Thundery Showers"
        - relative_humidity
        - temperature
        - wind
      - periods
        - 0
        - 1
        - 2
  - api_info
    - status : "healthy"

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

```python
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   45

Get weather info

FIX_ME exercise

- Get the weather information for "Ang Mo Kio"
- Produce the output as below

```
name : Ang Mo Kio, forecast : Cloudy
```

- Try to fix it, could you?

05_read_weather.py   46

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   47

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**
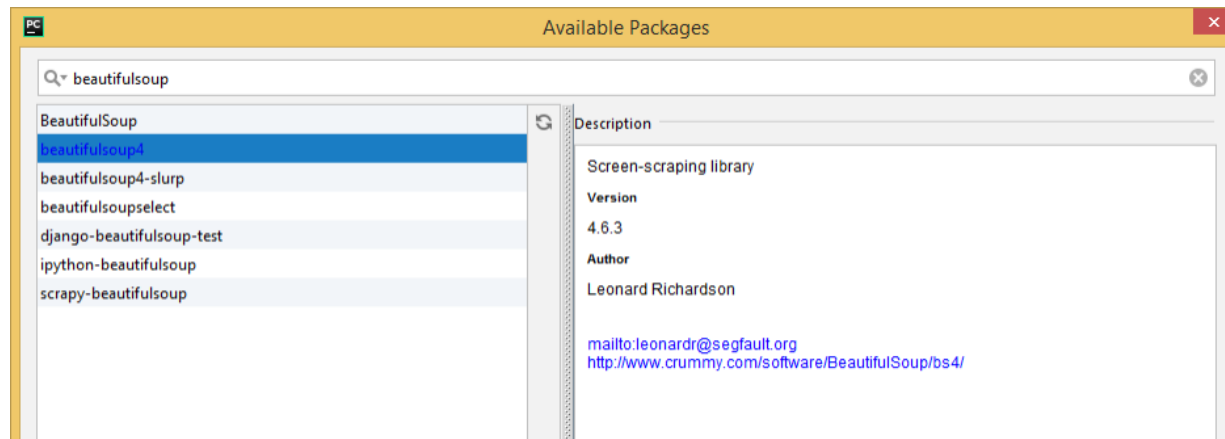
Connecting to the Web

- What's the URL?

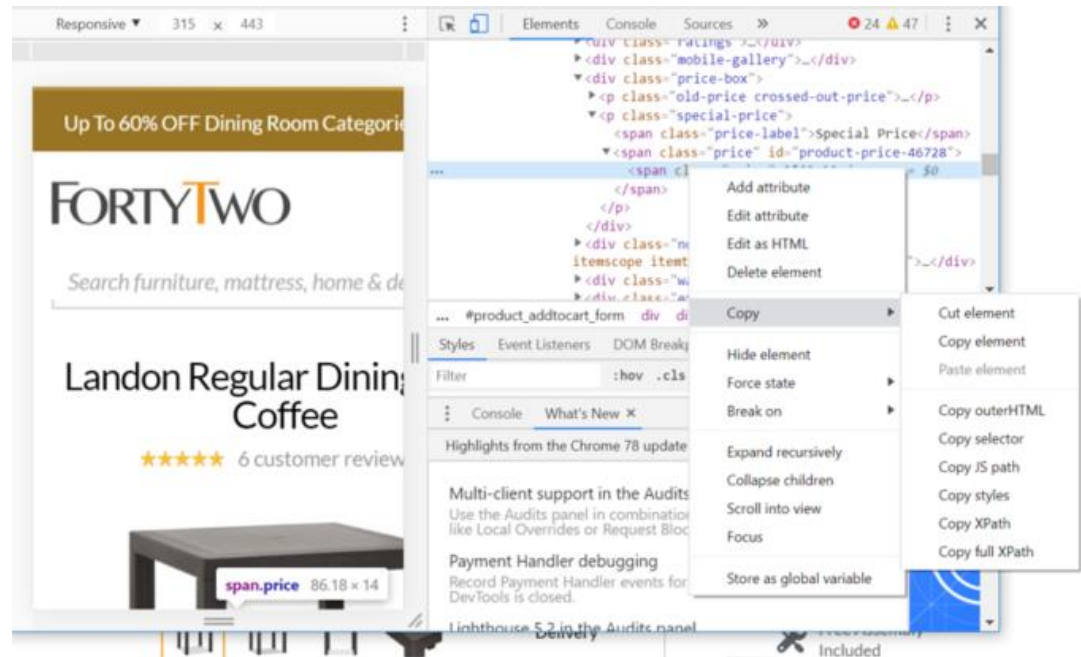https://www.fortytwo.sg/dining/dining-tables/landon-regular-dining-table-coffee.html

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"
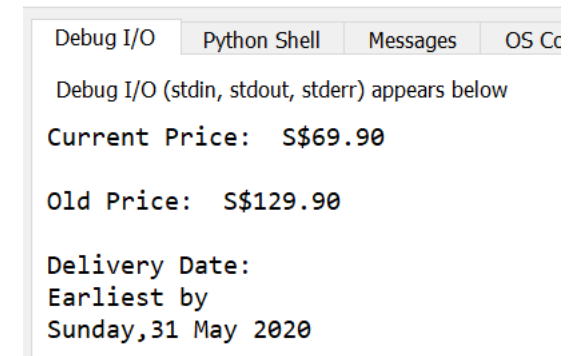 - click on "Copy selector"

Connecting to the Web

```python
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)
```
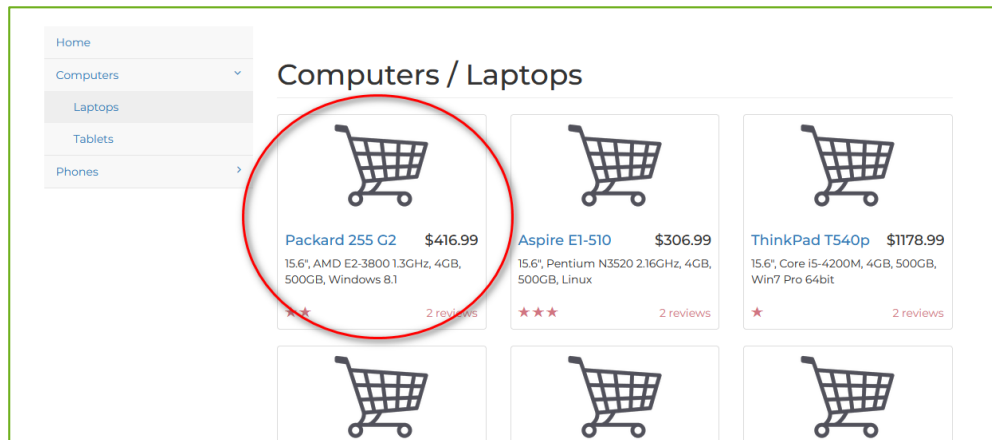
```
Debug I/O   Python Shell   Messages   OS Co

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_scrape.py   51

FIX_ME exercise

Getting item information

- Extract the item name and price, and produce the output below:


```
Item: Packard 255 G2
Price: $416.99
```

- Try to fix it, could you?

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
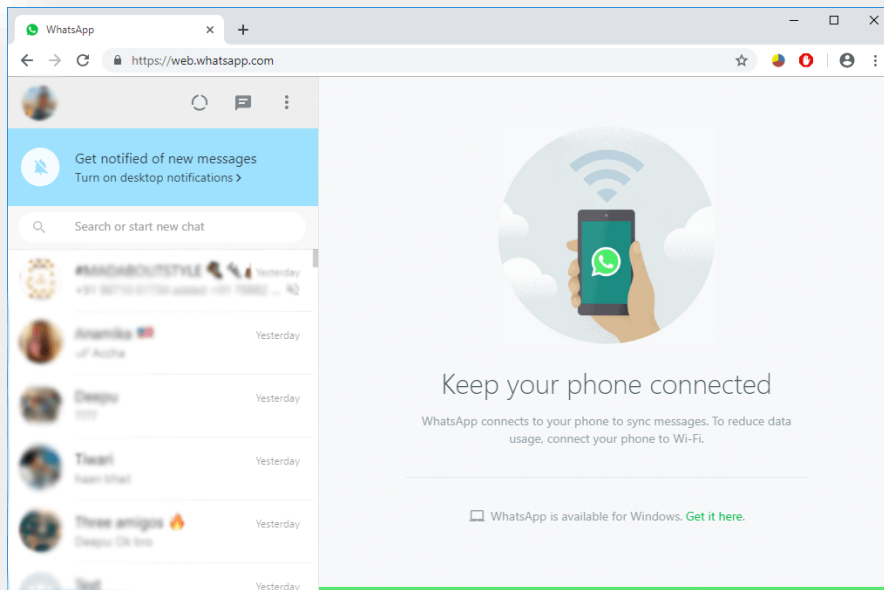   3. Print out the result as shown.

web_scrape_tabletennis.py

Sharing other Use Cases

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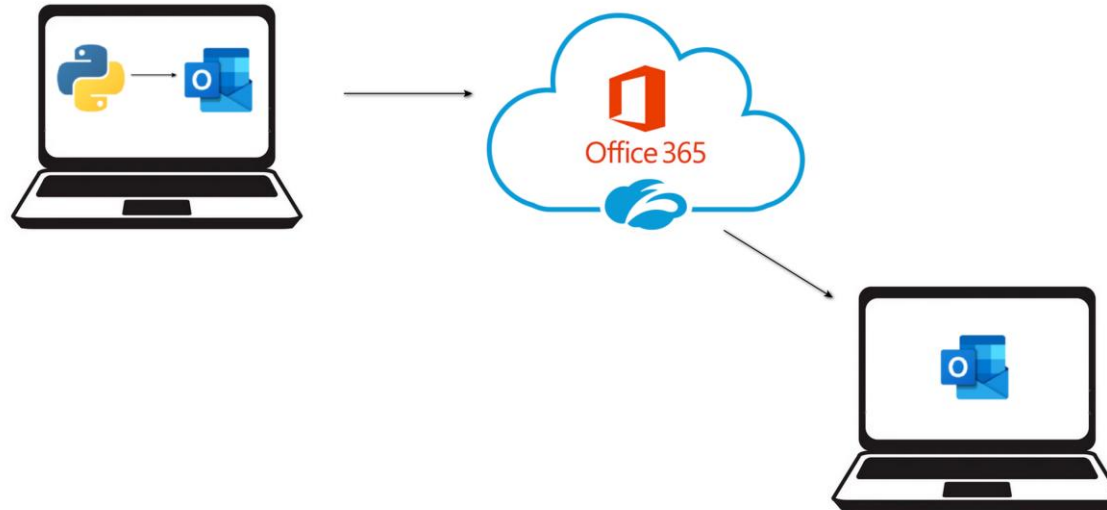 - Filling up google form
  - Sending WhatsApp message

Email Automation with Python

pywin32 package

- This assumes that we are using MS Outlook on the machine the python code is run

- pywin32 package is used to allow Python to access MS Outlook

- Python is not accessing the Outlook server directly, but it is using existing account set up in the MS Outlook

Send Email using Outlook

```python
import win32com.client

subject = 'email subject'
body = '<html><body>' + 'This is a test email. Oct Python 2021<br />' + '</body></html>'
recipient = 'jason_lim@rp.edu.sg'


#Create and send email
obj = win32com.client.Dispatch("Outlook.Application")
newMail = obj.CreateItem(0)
newMail.Subject = subject
newMail.HTMLBody = body
newMail.To = recipient


newMail.Send()
```

- The code above sends a simple HTML (meaning we can have tables, text formatting in it)
- No credentials are needed for the Python program as Python uses MS Outlook

Create appt using Outlook

```python
import win32com.client

outlook = win32com.client.Dispatch("Outlook.Application")
appt = outlook.CreateItem(1)
appt.Start = "2021-10-08 15:30"
appt.Subject = "Meeting 1"
appt.Duration = 60
appt.Location = "Library"
appt.MeetingStatus = 0
appt.display()
appt.Save()
```

- The code above creates a calendar appointment
- No credentials are needed for the Python program as Python uses MS Outlook

Sharing Use Cases

- Sending Emails using Outlook to students
- Create Appointments using Outlook

Other Python Libraries

- Play music using winsound

- Generate QR code using qrcode

- Face detection using opencv

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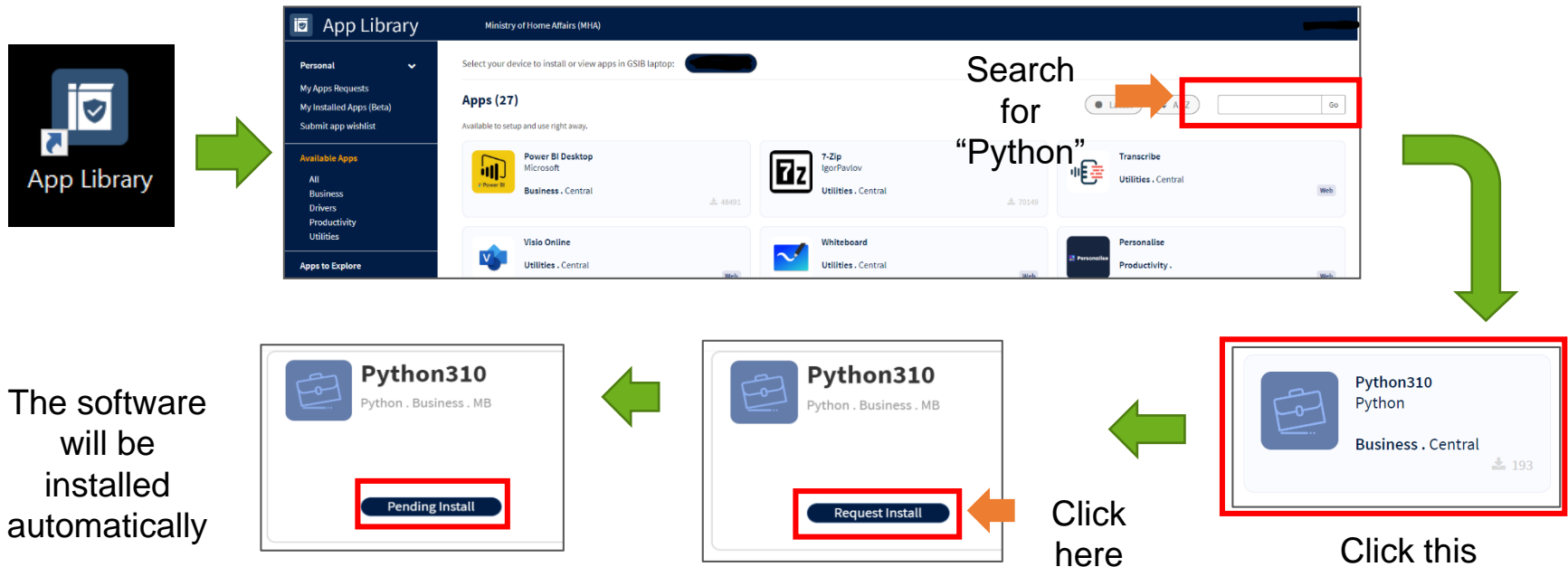
Think Python is a Free Book. It is available under the Creative Commons Attribution-NonCommercial 3.0 Unported License, which means that you are free to copy, distribute, and modify it, as long as you attribute the work and don't use it for commercial purposes. http://greenteapress.com/thinkpython/thinkpython.pdf

Interested to apply Python at work?

Follow the steps below:

App Library (Desktop) > Search bar: "Python" > "Python310" > "Request Install"

Search for "Python"

The software will be installed automatically

Click here

Click this

Please log a case with AFM Helpdesk if you face technical difficulties

End of Course Survey

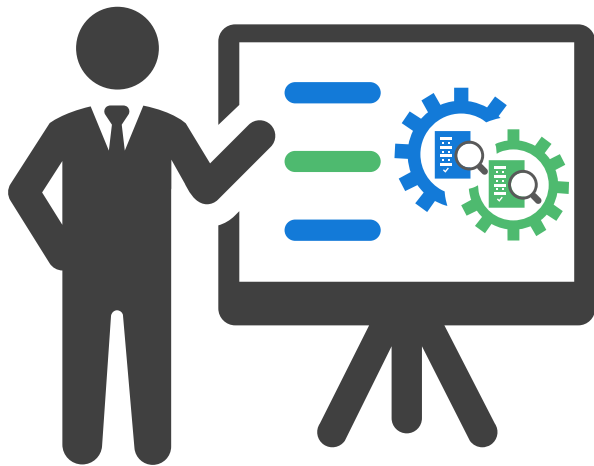https://for.edu.sg/mha-survey-python

Please kindly complete this survey before leaving the class

Thank you

Email us at:

Learning material & source code:
https://bit.ly/IPP-MHA