

# **Business Process Automation with VBA and Python**

Lecture 5 – Business Process Automation with Python (Part 3)

Jackie Liu - Jul 2024

College of Business and Finance (CBF)



# Course Outline

Lecture	Topics	Activities
#1 07-Sep	<b>(3) Business process automation with Python</b> <ul style="list-style-type: none"><li>• Fundamentals of Python syntax and related packages</li><li>• Development of user cases<ul style="list-style-type: none"><li>• Data source management</li><li>• Automation development for browser-based functions</li><li>• Automation development for application-based functions</li><li>• Automated reporting: tabular and graphic formats</li><li>• Common algorithm used in automation: fuzzy matching and standard deviation</li></ul></li><li>• Business and finance applications</li></ul>	In-class Practice
#2 14-Sep		
#3 21-Sep		
#4 28-Sep		
#5 05-Oct		



# Browser Automation

# Business Process Automation with Python

## Browser Automation

- Motivation
  - Modern websites may use JavaScript
  - Content are “injected” when browser runs the scripts

The screenshot illustrates the process of extracting data from a dynamic website using browser automation. On the left, a code editor shows a Python script using BeautifulSoup to find all div elements with the class 'price'. A blue arrow points from this code to the product page. The product page shows a package of pork slices with a price of \$61.90. Another blue arrow points from the product page to the Chrome DevTools Elements tab on the right. The Elements tab displays the HTML structure of the page, highlighting the div element containing the price. The bottom status bar of the browser indicates 'Highlights from the Chrome 112 update'.

```
# Price
[div_tag for div_tag in voucher_soup.find_all('div') if div_tag.attrs.get('class') == ['price']]
```

疯台味 - Taiwan Thick Pork Slices 240g...  
2,000+ Sold  
★★★★★ (13)  
Deep Cut! Original:\$78  
**\$ 61.90**

Elements Console Sources Network Performance

```
<!-- react-text: 25 -->
<!-- /react-text -->
</div>
<div class="info-wrapper">
  <div class="upper-wrapper" style="height: 94px;">...</div>
  <div class="lower-wrapper" style="height: 121px;">
    <div class="price-label">
      <div class="promotional">
        <span class="red">Deep Cut! Original:$78</span>
      </div>
      <div class="price">
        <span class="discount">$ 61.90</span> == $0
      </div>
    </div>
  </div>
</div>
```

... wrapper div.lower-wrapper div.price-label div.price span.discount

61.90 1 of 2 Cancel

Console What's New Network conditions

Highlights from the Chrome 112 update

CSS property documentation in the Styles pane

# Business Process Automation with Python

## Browser Automation

- Motivation
  - Replicating what the JavaScript does → troublesome

The screenshot shows the Network tab of a browser developer tools interface. It displays two requests:

- Request 1:** **Name:** search\_products  
**Headers:** Headers tab selected. **Request URL:** https://www.hktvmall.com/hktv/en/ajax/search\_products  
**Request Method:** POST  
**Status Code:** 200
- Request 2:** **Name:** ?id=761332883983543.  
**Headers:** Headers tab selected. **Request URL:** https://www.hktvmall.com/hktv/en/ajax/search\_products?id=761332883983543.  
**Request Method:** GET  
**Status Code:** 200

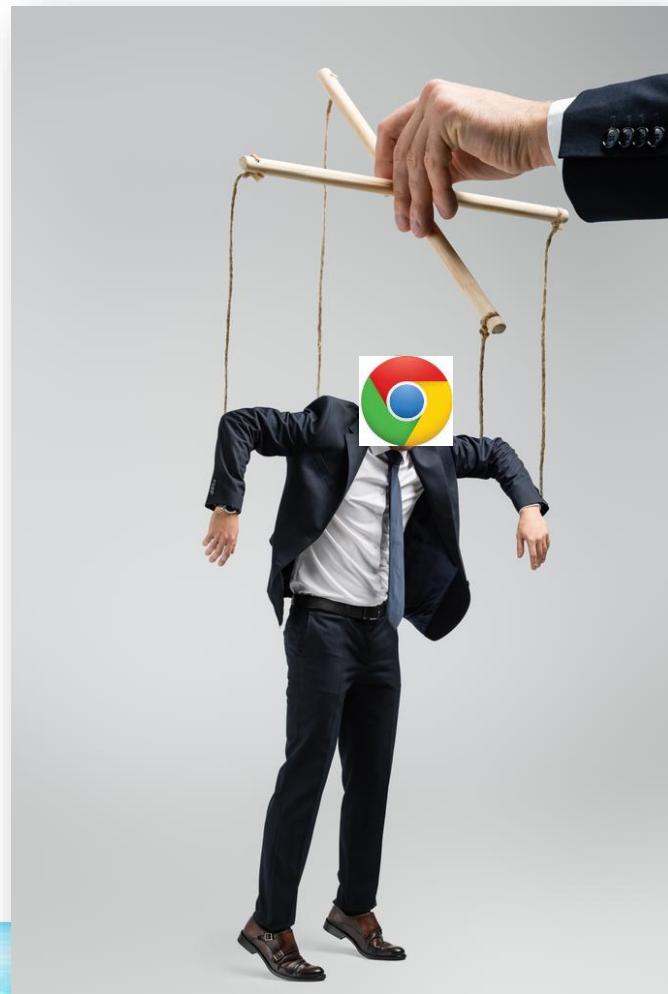
The second request's Headers tab is also selected, showing the following form data:

Name	Type	Value
query	Text	:relevance:street:main:category:AA1103700001
currentPage	Text	0
pageSize	Text	60
pageType	Text	searchResult
categoryCode	Text	AA1103700001
lang	Text	en
CSRFToken	Text	c2c880d5-d2fb-461f-84cd-af254e108ad5

# Business Process Automation with Python

## Browser Automation

- Q: If browsers can run JavaScript, can I control the browser with Python?



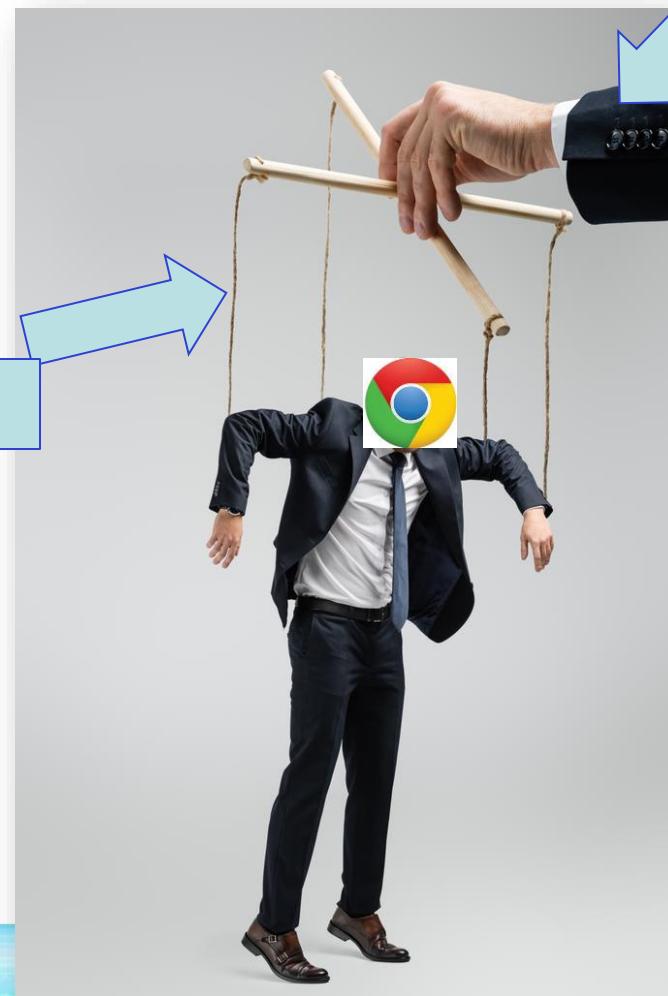
# Business Process Automation with Python

## Browser Automation

- Yes!
  - Selenium talks to drivers
  - Drivers control browsers

Selenium

ChromeDriver



# Business Process Automation with Python

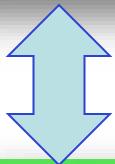
## Browser Automation (Chrome)

- Install the latest Chrome driver
  - <https://chromedriver.chromium.org/downloads>
- **ChromeDriver** version should match **Chrome** version
  - <https://googlechromelabs.github.io/chrome-for-testing/#stable>

chrome

win64

<https://storage.googleapis.com/chrome-for-testing-public/127.0.6533.119/win64/chrome-win64.zip>



chromedriver

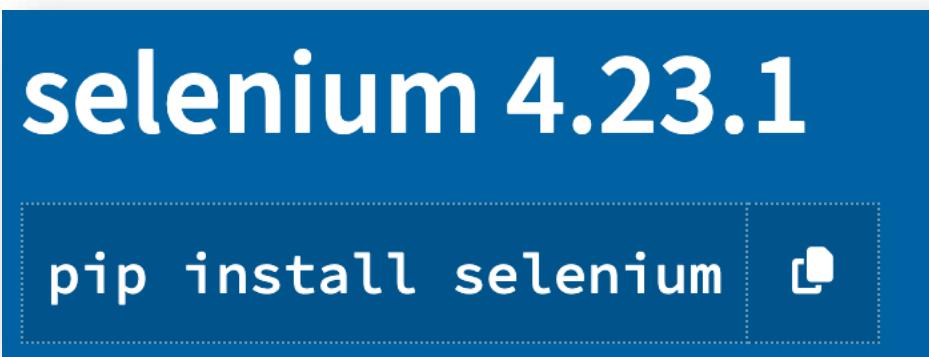
win64

<https://storage.googleapis.com/chrome-for-testing-public/127.0.6533.119/win64/chromedriver-win64.zip>

# Business Process Automation with Python

## Browser Automation

- Install Selenium
  - <https://pypi.org/project/selenium/>



- Import before use

```
▶ from selenium import webdriver  
from selenium.webdriver.common.by import By
```

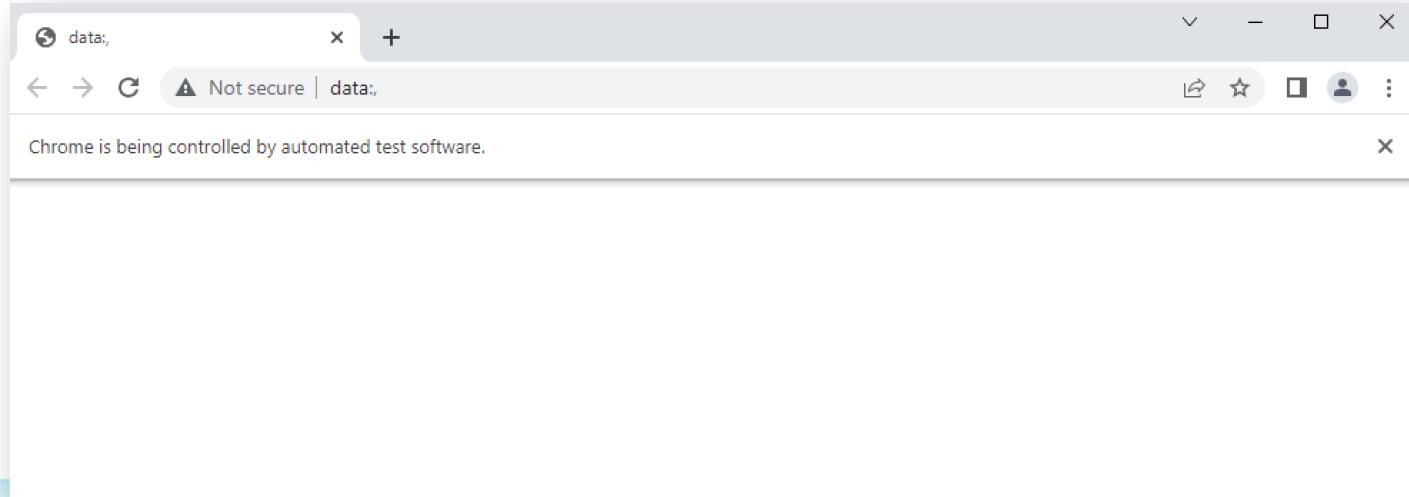
# Business Process Automation with Python

## Browser Automation

- Creating the driver object
  - `driver = webdriver.Chrome()`

```
# Create the driver
driver = webdriver.Chrome()
```

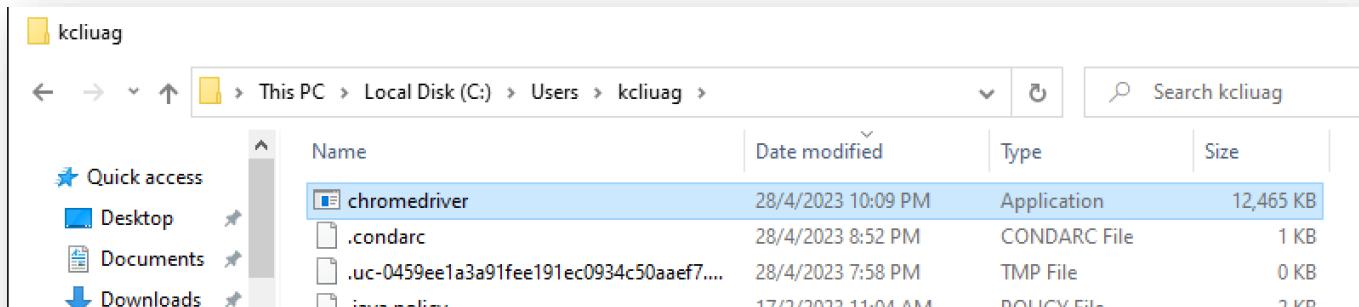
- This launches a Chrome browser!



# Business Process Automation with Python

## Browser Automation

- For Windows
  - If it cannot detect your chrome driver
  - Specify the **path** of the driver
    - `from selenium.webdriver.chrome.service import Service`
    - `driver = webdriver.Chrome(service=Service(path))`



A screenshot of a Windows File Explorer window. The path shown is `This PC > Local Disk (C:) > Users > kcliuag >`. The left sidebar shows 'Quick access' with links to Desktop, Documents, Downloads, and Pictures. The main pane displays a list of files and folders:

Name	Date modified	Type	Size
chromedriver	28/4/2023 10:09 PM	Application	12,465 KB
.condarc	28/4/2023 8:52 PM	CONDARC File	1 KB
.uc-0459ee1a3a91fee191ec0934c50aaef7....	28/4/2023 7:58 PM	TMP File	0 KB
.java.policy	17/2/2023 11:04 AM	POLICY File	2 KB
.conda	28/4/2023 10:29 PM	File folder	

At the bottom of the slide, there is a terminal window showing Python code running. The code imports the webdriver module from selenium, imports the Service class from selenium.webdriver.chrome.service, creates a Chrome service object pointing to the chromedriver executable, and then prints a message indicating that DevTools are listening on ws://127.0.0.1:60342/devtools/browser/053f33c4-cce4-4cd1-baac-98b1ab5996c4.

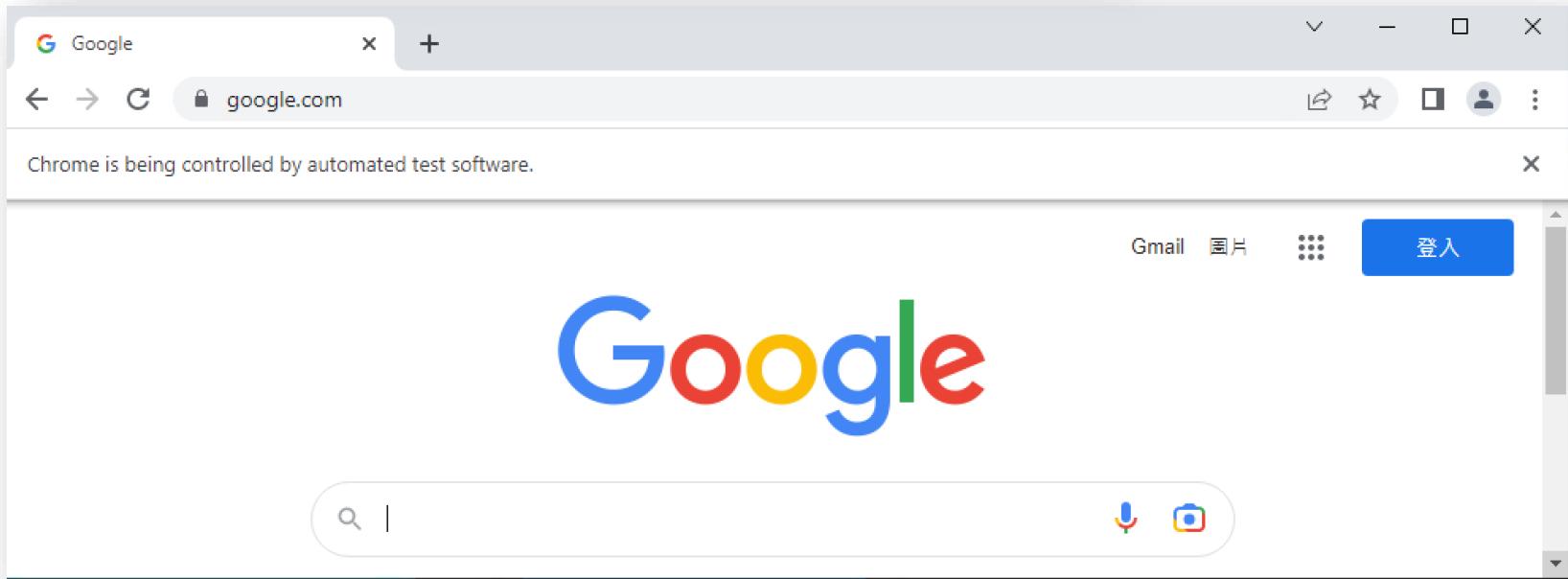
```
>>> from selenium import webdriver
>>> from selenium.webdriver.chrome.service import Service
>>> driver = webdriver.Chrome(service=Service(r'C:\Users\kcliuag\chromedriver.exe'))
DevTools listening on ws://127.0.0.1:60342/devtools/browser/053f33c4-cce4-4cd1-baac-98b1ab5996c4
>>>
```

# Business Process Automation with Python

## Browser Automation

- Basics - Getting a webpage
  - `driver.get(url)`

```
▶ driver.get('http://www.google.com')
```



# Business Process Automation with Python

## Browser Automation

- Basics - Turning it off
  - `driver.close()`
    - Close current tab (page), exit if only 1 tab exists
  - `driver.quit()`
    - Close all tabs and exit browser



# Business Process Automation with Python

## Browser Automation

- Basics - Options
  - For automation → no need to see the browser
    - options = webdriver.ChromeOptions()
    - options.add\_argument('--headless')
    - driver = webdriver.Chrome(options=options)
  - The browser will silently run in the background



Naruto – Disguise Jutsu

# Business Process Automation with Python

## Browser Automation

- Basics - Options
  - Other options

```
# Set options
options = webdriver.ChromeOptions()
options.add_argument('--disable-gpu')
options.add_argument("--window-size=1920, 1200")
options.add_argument('--disable-dev-shm-usage')

# Create the driver
driver = webdriver.Chrome(options=options)
```

1. List of Chromium Command Line Switches. Peter Beverloo. <https://peter.sh/experiments/chromium-command-line-switches>

# Business Process Automation with Python

## Browser Automation

- Getting the HTML
  - Requests → response.content
  - Selenium → driver.page\_source

```
▶ driver.get(voucher_url)

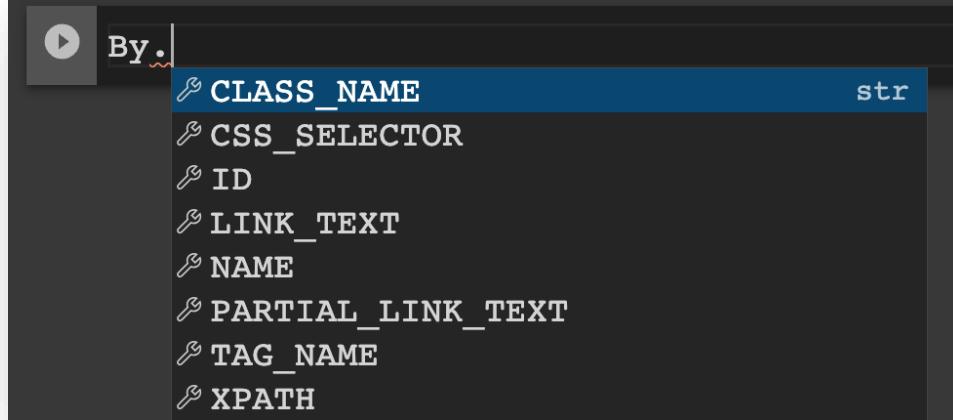
[59] driver.page_source[:100]

'<html lang="en" class=" svgfilters inlinesvg supports cssfilters flexbox flexboxlegacy flexwrap no-f'
```

# Business Process Automation with Python

## Browser Automation

- Navigation
  - BeautifulSoup → `find()` and `find_all()`
  - Selenium → `find_element()` and `find_elements()`
  - Must specify “By”



# Business Process Automation with Python

## Browser Automation

- Navigation – example

- e.g., <div id="product-result">



```
▼<div id="product-result">
  ▶<div id="product-result-header" class="product-result-navbar">...
```

- driver.find\_element(By.ID, value="product-result")

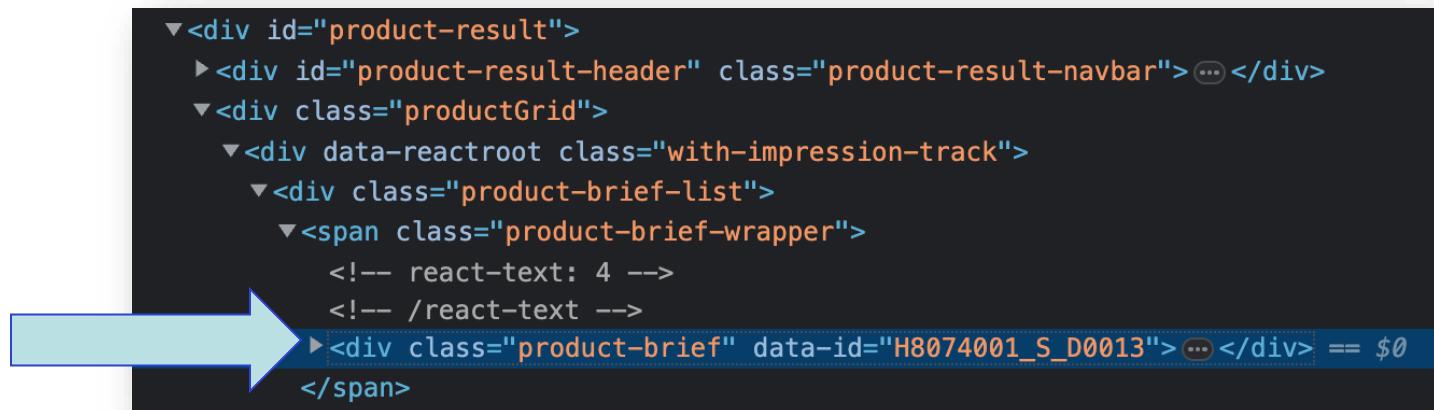
```
▶ product_result_tag = driver.find_element(By.ID, value='product-result')
product_result_tag
[] <selenium.webdriver.remote.webelement.WebElement (session="32f8a759bde20942680ad468d9c7b285", element=
```

# Business Process Automation with Python

## Browser Automation

- Navigation – More example

- e.g., <div class="product-brief">



- `element.find_element(By.CLASS_NAME, value="product-brief")`

```
▶ product_result_tag.find_element(By.CLASS_NAME, value='product-brief')
◀ <selenium.webdriver.remote.webelement.WebElement (session="32f8a759bde20942680ad468d9c7b285", element="55a237db-4f4b-42b5-8e9e-658a4adc9f72")>
```

# Business Process Automation with Python

## Browser Automation

- WebElement - Attributes
  - element.get\_attribute(name)

```
<div id="product-result">
  <div id="product-result-header" class="product-result-navbar">...</div>
  <div class="productGrid">
    <div data-reactroot class="with-impression-track">
      <div class="product-brief-list">
        <span class="product-brief-wrapper">
          <!-- react-text: 4 -->
          <!-- /react-text -->
          <div class="product-brief" data-id="H8074001_S_D0013">...</div> == $0
        </span>
```

- Getting data-id attribute
  - element.get\_attribute('data-id')

```
brief_element = product_result_tag.find_element(By.CLASS_NAME, value='product-brief')
brief_element.get_attribute('data-id')

' H8074001_S_D0013'
```

# Business Process Automation with Python

## Browser Automation

- WebElement - Text
  - `.text`
  - Simple, right?



A screenshot of the Chrome DevTools Elements tab. It shows the HTML structure of the product page. The discounted price '\$ 61.90' is highlighted in the DOM tree under the class 'discount'. The status bar at the bottom of the DevTools shows the text 'CSS property documentation in the Styles pane'.

```
▶ e.find_element(By.CLASS_NAME, value='discount').text  
⇨ '$ 61.90'
```

# Business Process Automation with Python

## Extended Learning - Auto scroll

- Some websites only load more if you scroll down
  - Can simulate that by running a script
  - `driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")`



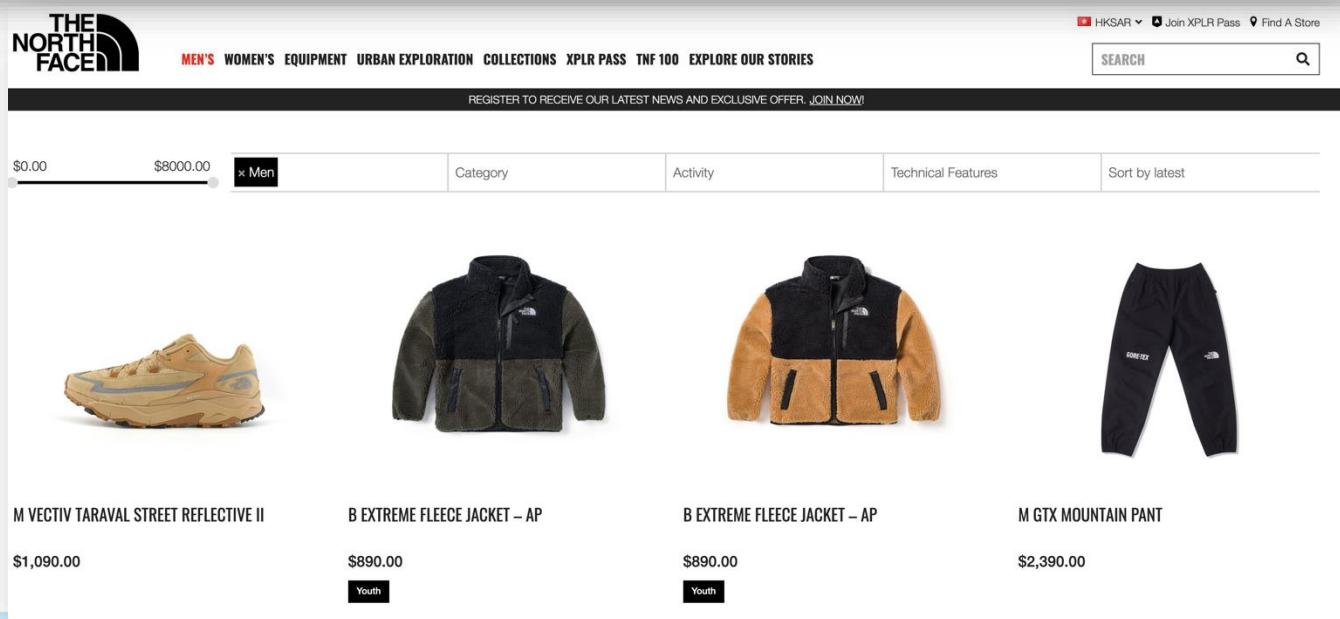
# Business Process Automation with Python

## Extended Learning - Auto scroll

- Example - Before

```
▶ nf_url = 'https://www.thenorthface.com.hk/shop/?filters=gender%5Bmen%5D'
driver.get(nf_url)

[41] len(driver.page_source)
739503
```



# Business Process Automation with Python

## Extended Learning - Auto scroll

- Example - After

```
▶ nf_url = 'https://www.thenorthface.com.hk/shop/?filters=gender%5Bmen%5D'  
driver.get(nf_url)
```

```
[41] len(driver.page_source)
```

```
739503
```

```
▶ # Scroll down  
driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")  
# Wait for browser to load  
time.sleep(3)  
# Check length again  
len(driver.page_source)
```

```
871579
```

# Business Process Automation with Python

## Extended Learning - Auto scroll

- Looping till end of page
  - Use the `break` statement if there is no height change

```
▶ for i in range(100):  
    # Record current page height  
    current_height = driver.execute_script("return document.body.scrollHeight")  
    # Scroll down  
    driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")  
    time.sleep(3)  
    # Check new page height  
    new_height = driver.execute_script("return document.body.scrollHeight")  
    # Stop if there is no change  
    if new_height == current_height:  
        break  
  
    # Check length again  
    len(driver.page_source)
```

2343197

# Business Process Automation with Python

## Browser Automation

- **Q: Why not just use Selenium?**
  - Performance → Slow / require more resources
  - Driver compatibility → Lose support / require updates



- Example notebook
  - <https://colab.research.google.com/drive/1JTNPmRV63eVE1GZve2PmVenBNgGAoBHI?usp=sharing>



# Basic Visualization

# Business Process Automation with Python

## Charting

- Motivation
  - Datasets could be large / boring
  - Charts help to draw insights<sup>1</sup>



Salary trend according to the *Job Seeker Salary Report 2023*



1. *Job seeker salary report 2023 (2023) JobsDB Hong Kong.*  
<https://hk.jobsdb.com/en-hk/pages/job-seeker-salary-report-2023/>

# Business Process Automation with Python

## Charting

- From DataFrame
  - Try → `df.plot()`
- “**Matplotlib** is used”<sup>1</sup>
  - What is that?



## pandas.DataFrame.plot

`DataFrame.plot(*args, **kwargs)`

[source]

Make plots of Series or DataFrame.

Uses the backend specified by the option `plotting.backend`

By default, matplotlib is used.

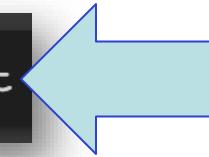
1. `pandas.DataFrame.plot` - pandas 2.1.0 documentation.  
<https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.plot.html>

# Business Process Automation with Python

## Matplotlib

- Import before usage
- It is a package with many indirectly referenced modules
- Main module → **PyPlot**
  - Commonly imported as “**plt**”

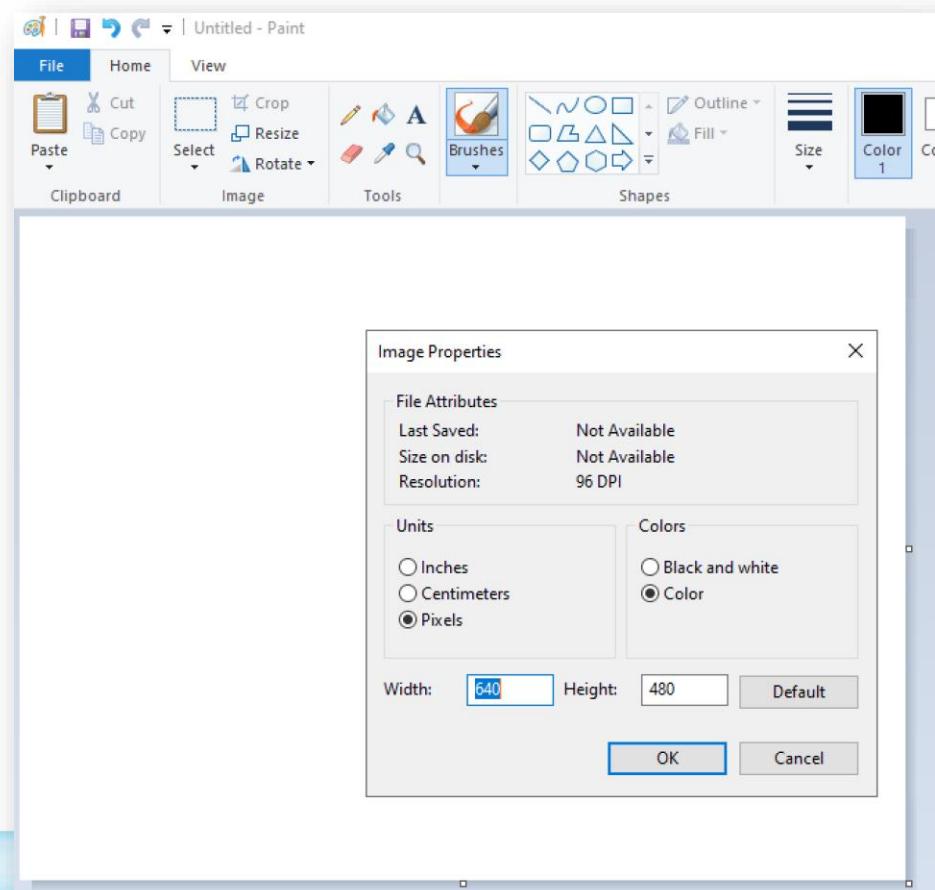
```
import matplotlib.pyplot as plt
```



# Business Process Automation with Python

## Matplotlib - Figure

- Imagine charting → drawing
- Start with a canvas / drawing board



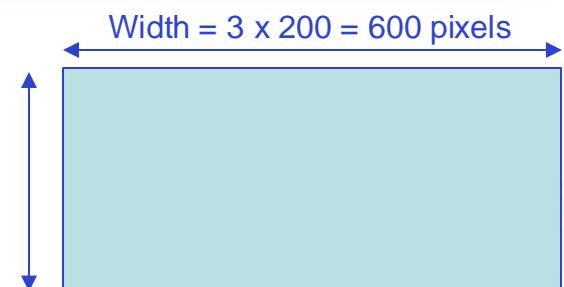
# Business Process Automation with Python

## Matplotlib - Figure

- Create → **plt.figure()**

- **figsize**: Tuple of width and height (Required)
- **dpi**: dot per inch (default: 100)
- **facecolor**: Background colour (default: white)

```
▶ # Canvas with 600 x 400 pixels
    plt.figure(figsize=(3.0, 2.0), dpi=200, facecolor='white')
↳ <Figure size 600x400 with 0 Axes>
```



1. *matplotlib.figure - Matplotlib 3.7.3 documentation.*  
[https://matplotlib.org/stable/api/figure\\_api.html](https://matplotlib.org/stable/api/figure_api.html)

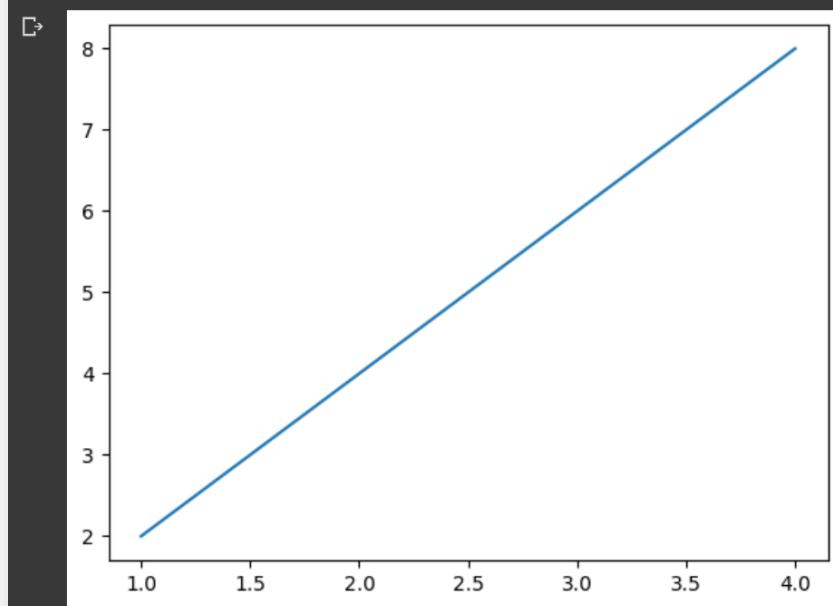
# Business Process Automation with Python

## Matplotlib - Plot

- Drawing a line graph
  - `plt.plot(x_axis, y_axis)`
  - In IDEs → needs `plt.show()`

```
# Plotting a line
x_axis = [1, 2, 3, 4]
y_axis = [2, 4, 6, 8]
plt.plot(x_axis, y_axis)

plt.show()
```



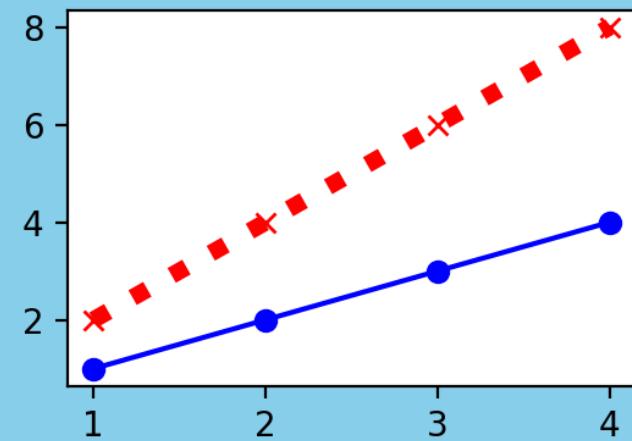
1. `matplotlib.pyplot.plot` - Matplotlib 3.8.0 documentation.  
[https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.plot.html](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html)

# Business Process Automation with Python

## Matplotlib - Plot

- Styling (Format string)
  - Marker
    - x / o / v
  - Line
    - - (solid) / : (dotted)
  - Color
    - g (green)
    - r (red)
    - k (black)

```
fmt = '[marker][line][color]'
```



```
▶ # More options
plt.figure(figsize=(3.0, 2.0), dpi=200, facecolor='skyblue')

# : => Dotted (Line)
# x => x (Marker)
# r => Red colour
plt.plot([1, 2, 3, 4], [2, 4, 6, 8], ':xr', linewidth=5)
plt.plot([1, 2, 3, 4], [1, 2, 3, 4], '-ob')

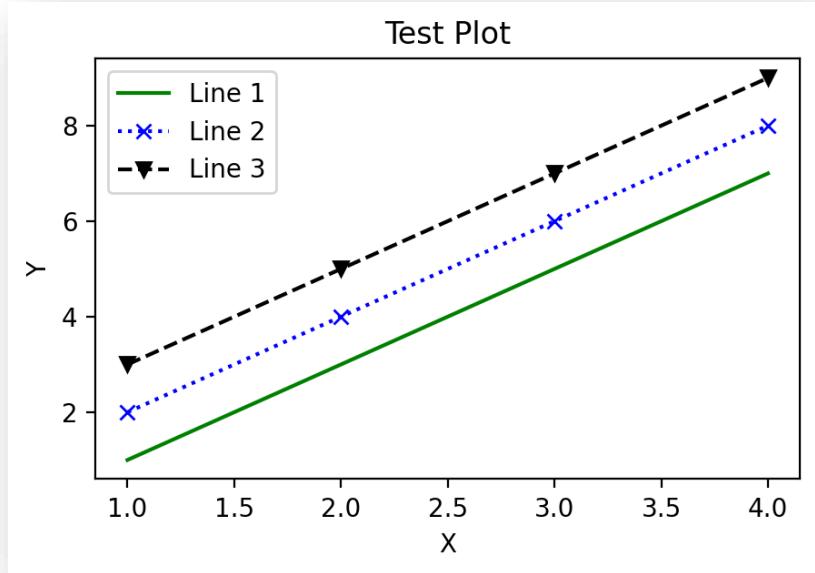
plt.show()
```

1. *matplotlib.pyplot.plot - Matplotlib 3.8.0 documentation.*  
[https://matplotlib.org/stable/api/\\_as\\_gen/matplotlib.pyplot.plot.html](https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.plot.html)
2. *List of named colors – Matplotlib 3.8.0 documentation.*  
[https://matplotlib.org/stable/gallery/color/named\\_colors.html](https://matplotlib.org/stable/gallery/color/named_colors.html)

# Business Process Automation with Python

## Matplotlib - Plot

- Labels
  - `plot(label='abc')`
    - Need `plt.legend()`
  - `plt.title()`
  - `plt.xlabel()`
  - `plt.ylabel()`



```
# Labels
plt.figure(figsize=(5.0, 3.0), dpi=200, facecolor='white')

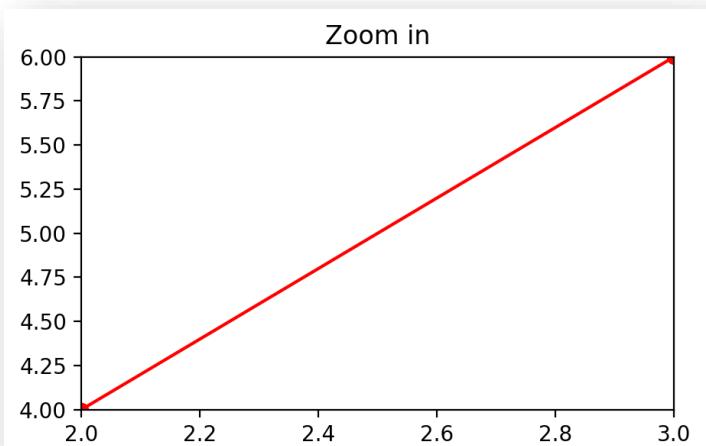
plt.title('Test Plot')
plt.plot([1, 2, 3, 4], [1, 3, 5, 7], '-g', label='Line 1')
plt.plot([1, 2, 3, 4], [2, 4, 6, 8], ':xb', label='Line 2')
plt.plot([1, 2, 3, 4], [3, 5, 7, 9], '--vk', label='Line 3')
plt.xlabel('X')
plt.ylabel('Y')
plt.legend(loc='best') # loc: 'upper right' / 'center left' / 'lower center'

plt.show()
```

# Business Process Automation with Python

## Matplotlib - Plot

- Zoom
  - `plt.xlim((x_min, x_max))`
  - `plt.ylim((y_min, y_max))`



```
▶ # Zoom
    plt.figure(figsize=(5.0, 3.0), dpi=200, facecolor='white')

    plt.title('Zoom in')
    plt.plot([1, 2, 3, 4], [2, 4, 6, 8], '-or')
    plt.xlim((2, 3))
    plt.ylim((4, 6))

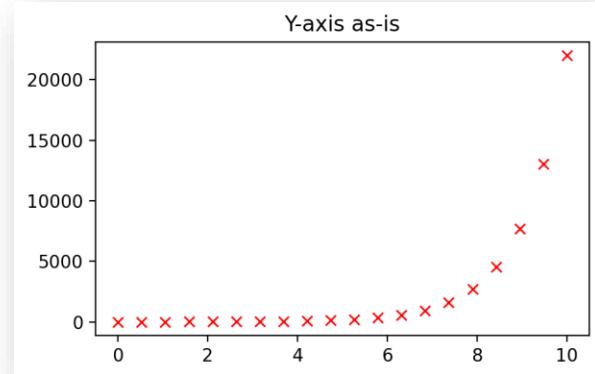
    plt.show()
```

# Business Process Automation with Python

## Matplotlib - Plot

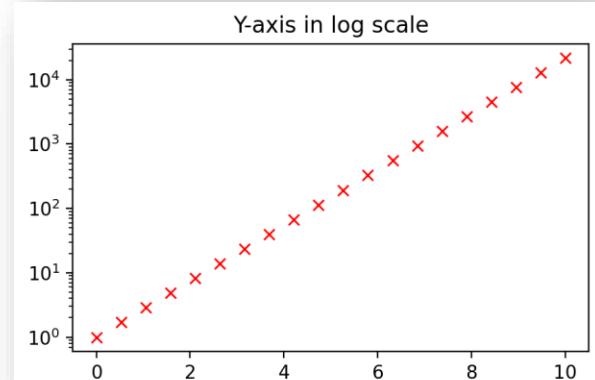
- Scale
  - plt.xscale() / plt.yscale() → linear / log / logit

```
import numpy as np
x_axis_linear = np.linspace(0, 10, num=20)
plt.figure(figsize=(5.0, 3.0), dpi=200, facecolor='white')
plt.title('Y-axis as-is')
plt.plot(x_axis_linear, np.exp(x_axis_linear), 'xr')
plt.show()
```



▶ # Scaling

```
# Scaling
x_axis_linear = np.linspace(0, 10, num=20)
plt.figure(figsize=(5.0, 3.0), dpi=200, facecolor='white')
plt.title('Y-axis in log scale')
plt.yscale('log')
plt.plot(x_axis_linear, np.exp(x_axis_linear), 'xr')
plt.show()
```



# Business Process Automation with Python

## Matplotlib – Multiple Figures

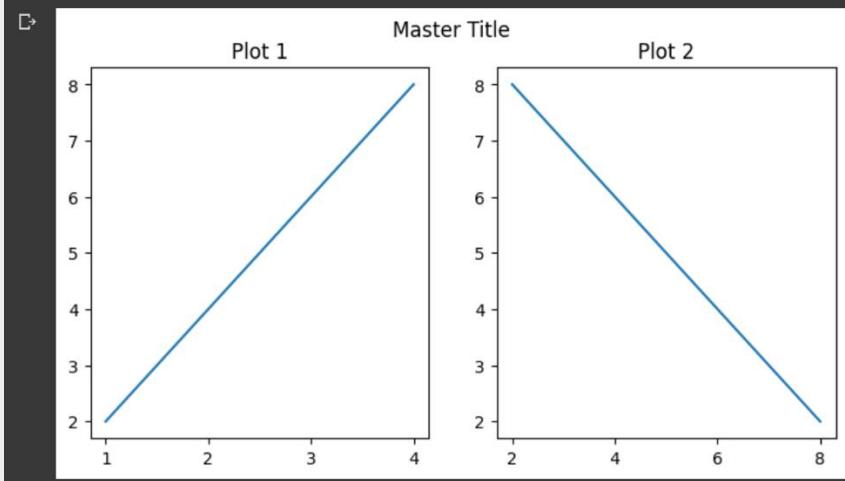
- **plt.subplots(rows, columns)**
  - → fig (figure)  
→ ax (Axes → plot)
  - Figure-level title
    - **fig.suptitle()**
  - Plot-level properties
    - Add “set\_”
    - **plt.title()**  
→ **ax[i].set\_title()**

```
▶ fig, ax = plt.subplots(1, 2, figsize=(8.0, 4.0))

# Super title
fig.suptitle('Master Title')

# Plot 1
ax[0].set_title('Plot 1')
ax[0].plot([1, 2, 3, 4], [2, 4, 6, 8])
# Plot 2
ax[1].set_title('Plot 2')
ax[1].plot([2, 4, 6, 8], [8, 6, 4, 2])

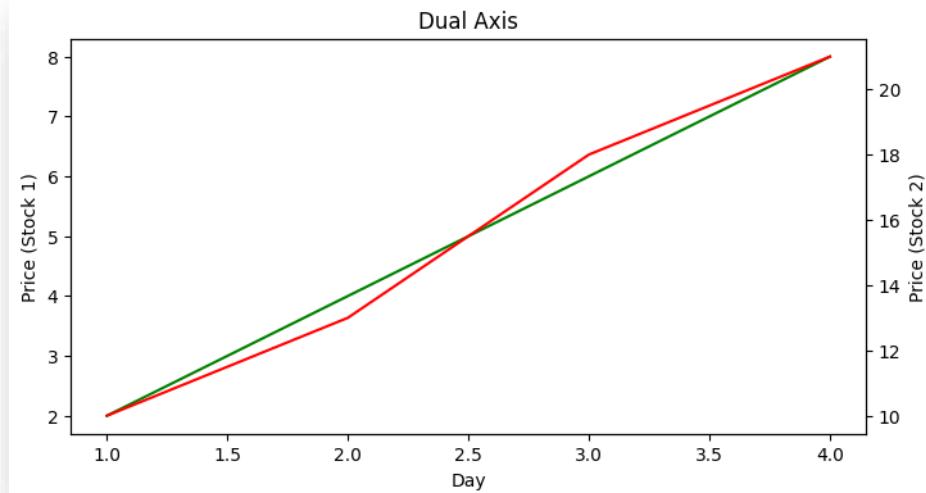
fig.show()
```



# Business Process Automation with Python

## Matplotlib – Multiple Axes

- **ax2 = ax.twinx()**
- As if there is a subplot



```
▶ fig, ax1 = plt.subplots(figsize=(8.0, 4.0))

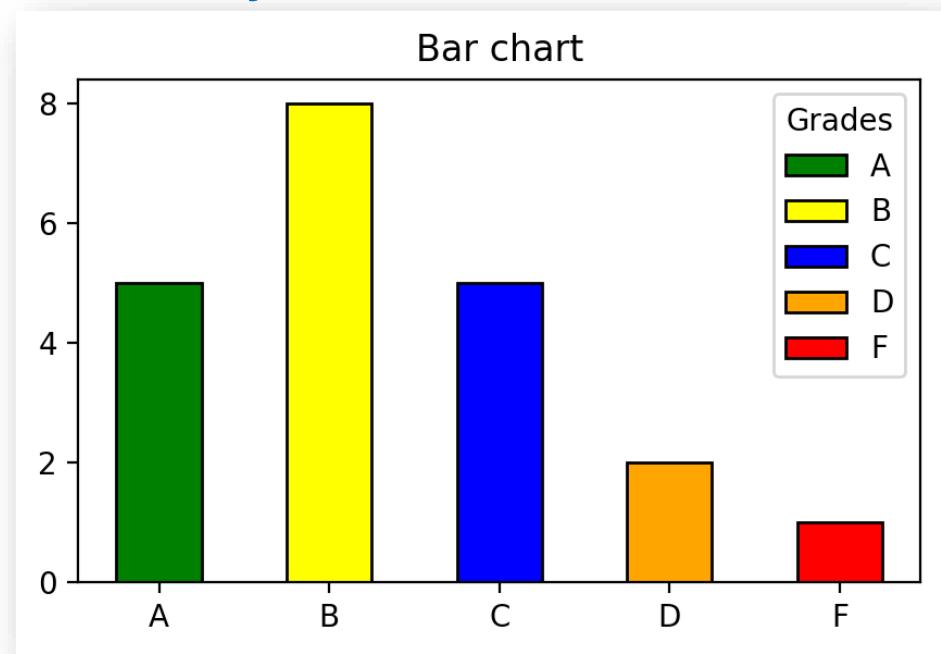
    # Plot 1
    ax1.set_title('Dual Axis')
    ax1.plot([1, 2, 3, 4], [2, 4, 6, 8], '-g', label='Stock 1')
    ax1.set_xlabel('Day')
    ax1.set_ylabel('Price (Stock 1)')

    # Plot 2
    ax2 = ax1.twinx()
    ax2.plot([1, 2, 3, 4], [10, 13, 18, 21], '-r', label='Stock 2')
    ax2.set_ylabel('Price (Stock 2)')
fig.show()
```

# Business Process Automation with Python

## Matplotlib – Bar Chart

- **plt.plot() → plt.bar()**
- Great for category counts



```
▶ plt.figure(figsize=(5.0, 3.0), dpi=200, facecolor='white')

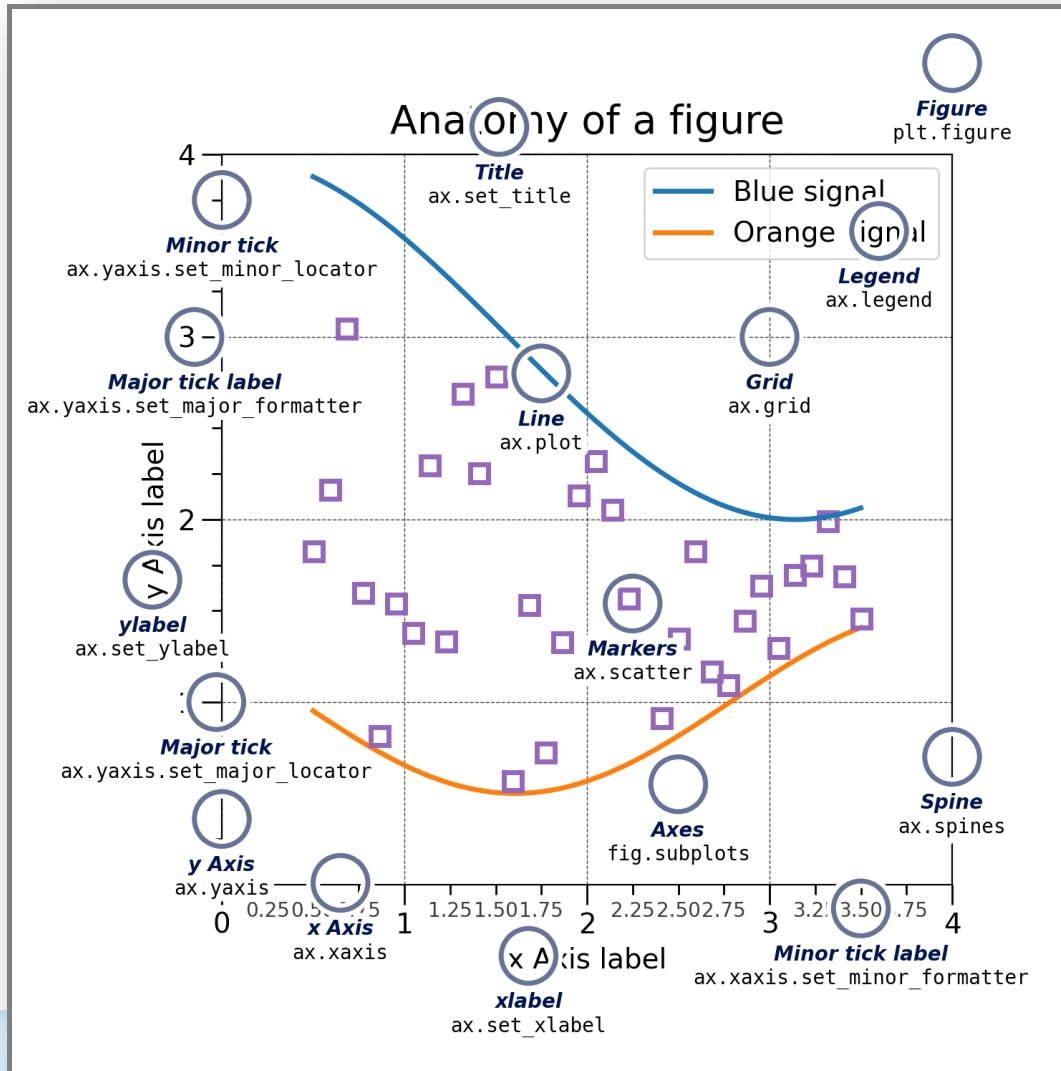
    plt.title('Bar chart')
    categories = ['A', 'B', 'C', 'D', 'F']
    values = [5, 8, 5, 2, 1]
    bar_colors = ['green', 'yellow', 'blue', 'orange', 'red']

    plt.bar(categories, values, width=0.5,
            label=categories, color=bar_colors, edgecolor="black")
    plt.legend(title='Grades', loc='best')
    plt.show()
```



# Business Process Automation with Python

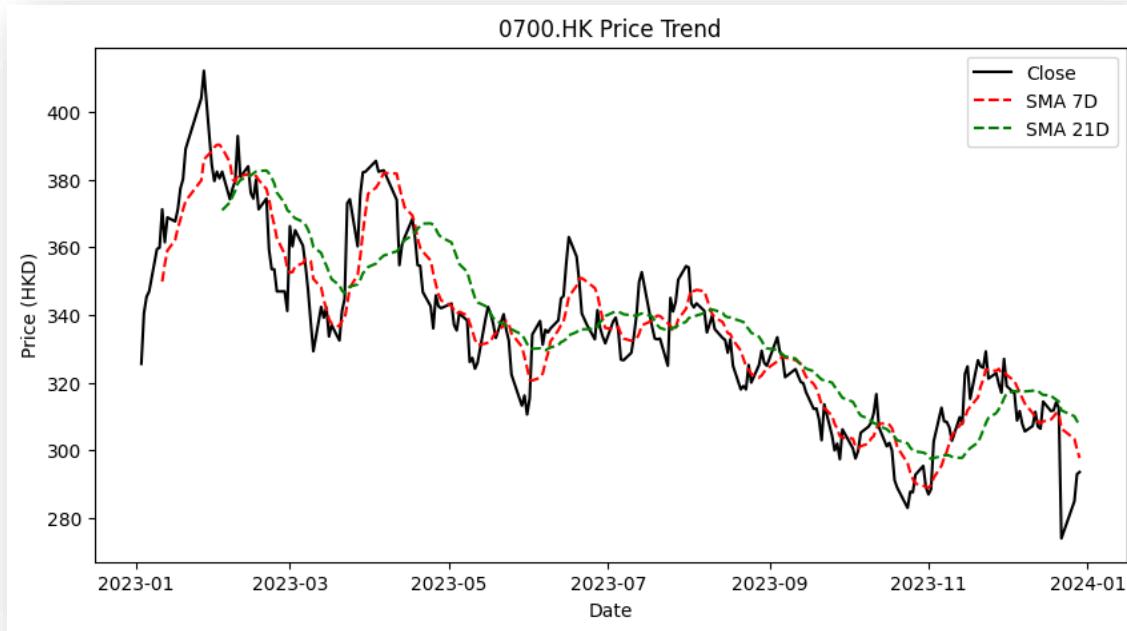
## Matplotlib – Summary



# Business Process Automation with Python

## Practice 7 – Visualizing Stock Price Trends

- Try out what you have learnt using real stock data!



- Starting notebook  
[https://drive.google.com/file/d/1rBE8g-BcQNIQ\\_8GZj9P3Z966MmwMuHJt/view?usp=sharing](https://drive.google.com/file/d/1rBE8g-BcQNIQ_8GZj9P3Z966MmwMuHJt/view?usp=sharing)



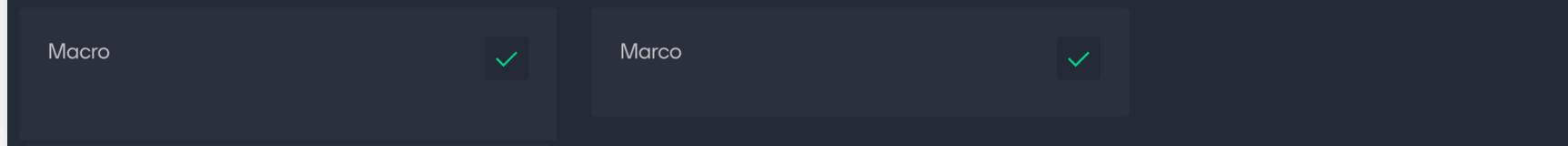
# Fuzzy Matching

# Business Process Automation with Python

## Fuzzy Match

- Motivation
  - Real-world inputs may be different from expected

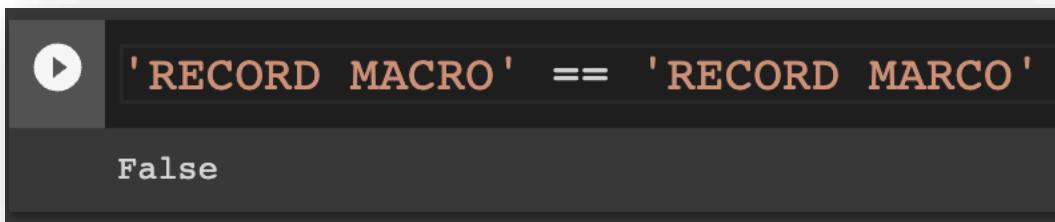
An easy way to draft VBA is to record {what} so we can replay them later...



# Business Process Automation with Python

## Fuzzy Match

- Exact matching
  - May not be best



A screenshot of a code editor window. On the left is a dark grey sidebar with a white play button icon. The main area has a dark background with light-colored text. It displays the following code and output:

```
'RECORD MACRO' == 'RECORD MARCO'  
False
```

# Business Process Automation with Python

## Fuzzy Match

- Edit Distance (Going from X to Y)
  - What is the unit of difference?
    - In real life → inch / meter
    - String comparison → ???



# Business Process Automation with Python

## Fuzzy Match

- Clue from childhood
  - Spot the difference → Number of replacement?



# Business Process Automation with Python

## Fuzzy Match

- Minimum number of replacement
  - C → R, R → C
  - Distance = 2
  - This is called Hamming distance

```
X = 'RECORD MACRO'  
Y = 'RECORD MARCO'
```



### Hamming Distance

1  1  
1  1 → Hamming Distance = 2

0  1 → Hamming Distance = 3

## Fuzzy Match

- Minimum number of replacement
  - Q: How to measure it in Python?

```
X = 'RECORD MACRO'  
Y = 'RECORD MARCO'
```



# Business Process Automation with Python

## Fuzzy Match

- Hamming distance
  - Option 1

```
X = 'RECORD MACRO'  
Y = 'RECORD MARCO'
```

```
▶ X = 'RECORD MACRO'  
    Y = 'RECORD MARCO'  
  
    replacement_count = 0  
    for i in range(len(X)):  
        if X[i] != Y[i]:  
            replacement_count = replacement_count + 1  
  
    print(f'{replacement_count} letters need to be replaced')  
⇒ 2 letters need to be replaced
```

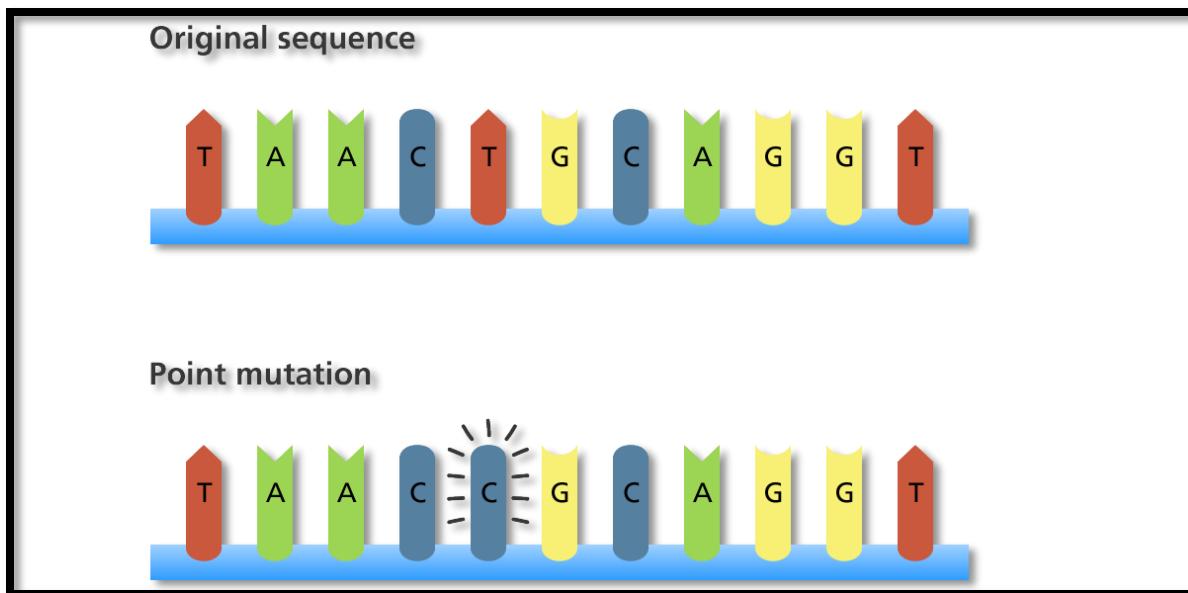
- Option 2

```
▶ sum([1 for i in range(len(X)) if X[i]!=Y[i]])  
⇒ 2
```

# Business Process Automation with Python

## Fuzzy Match

- Hamming distance
  - Can be used in error detection
    - Computer memory / Digital signals / DNA / etc

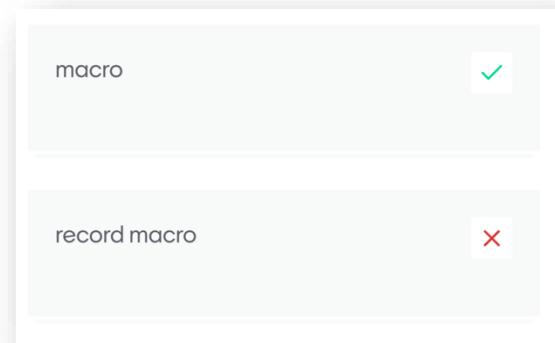


1. Mohammadi-Kambs, M., Hölz, K., Somoza, M. M., & Ott, A. (2017, April 5). *Hamming Distance as a Concept in DNA Molecular Recognition*. PubMed Central (PMC). <https://doi.org/10.1021/acsomega.7b00053>

# Business Process Automation with Python

## Fuzzy Match

- Hamming distance
  - What if length is different?



```
▶ X = 'RECORD MACRO'  
    Y = 'MACRO'  
    sum([1 for i in range(len(X)) if X[i]!=Y[i]])  
  
[  
    -----  
    IndexError                                         Traceback (most recent call last  
    <ipython-input-12-edb8fe2cb8f1> in <cell line: 3>()  
        1 X = 'RECORD MACRO'  
        2 Y = 'MACRO'  
----> 3 sum([1 for i in range(len(X)) if X[i]!=Y[i]])  
  
    <ipython-input-12-edb8fe2cb8f1> in <listcomp>(.0)  
        1 X = 'RECORD MACRO'  
        2 Y = 'MACRO'  
----> 3 sum([1 for i in range(len(X)) if X[i]!=Y[i]])  
  
IndexError: string index out of range
```

## Fuzzy Match

- Hamming distance
  - New considerations → Addition & Deletion



## Fuzzy Match

- Levenshtein distance
  - New considerations → Addition & Deletion
    - From X: Delete “RECORD”
    - From Y: Add “RECORD”

```
X = 'RECORD MACRO'  
Y = 'MACRO'
```

- Each add/delete → +1 distance

# Business Process Automation with Python

## Fuzzy Match

- Q: How to find the Levenshtein distance?
  - X = Sunday
  - Y = Saturday
- Replace / Add / Delete

S	u	n	d	a	y		
S	a	t	u	r	d	a	y

# Business Process Automation with Python

## Fuzzy Match

- Levenshtein Algorithm
  - Initialize from 0 to maximum length

		Y								
		s	a	t	u	r	d	a	y	
		0	1	2	3	4	5	6	7	8
X		s	1	0	1	2	3	4	5	6
		u	2	1	1	2	2	3	4	5
		n	3	2	2	2	3	3	5	5
		d	4	3	3	3	3	4	3	4
		a	5	4	3	4	4	4	3	4
		y	6	5	4	4	5	5	4	3

# Business Process Automation with Python

## Fuzzy Match

- Levenshtein Algorithm
  - $X=S$  vs  $Y=S \rightarrow$  distance is 0

		Y								
		S	a	t	u	r	d	a	y	
X		0	1	2	3	4	5	6	7	8
S	1	0	1	2	3	4	5	6	7	8
u	2	1	1	2	2	3	4	5	6	6
n	3	2	2	2	3	3	5	5	6	
d	4	3	3	3	3	4	3	4	5	
a	5	4	3	4	4	4	4	3	4	
y	6	5	4	4	5	5	5	4	3	3

# Business Process Automation with Python

## Fuzzy Match

- Levenshtein Algorithm
  - X=S vs Y=SA → why distance is 1?

		Y								
		S	a	t	u	r	d	a	y	
		0	1	2	3	4	5	6	7	8
X		S	0	1	2	3	4	5	6	7
s	u	1	0	1	2	3	4	5	6	7
u	n	2	1	1	2	2	3	4	5	6
n	d	3	2	2	2	3	3	5	5	6
d	a	4	3	3	3	3	4	3	4	5
a	y	5	4	3	4	4	4	4	3	4
y		6	5	4	4	5	5	5	4	3

# Business Process Automation with Python

## Fuzzy Match

- Levenshtein Algorithm
  - Each step: minimum of 3 directions + add 1 if mismatch

	s	a	t	u	r	d	a	y	
s	0	1	2	3	4	5	6	7	8
u	1	0	1	2	3	4	5	6	7
n	2	1	1	2	2	3	5	5	6
d	3	2	2	2	3	3	5	5	6
a	4	3	3	3	3	4	3	4	5
a	5	4	3	4	4	4	4	3	4
y	6	5	4	4	5	5	5	4	3

# Business Process Automation with Python

## Fuzzy Match

- Levenshtein Algorithm – More example
  - $\min(1, 0, 1) = 0$
  - Add 1 because u is not a
  - $0 + 1 \rightarrow 1$

	s	a	t	u	r	d	a	y
0	1	2	3	4	5	6	7	8
s	0	1	2	3	4	5	6	7
u	1	0	1	2	3	4	5	6
n	2	1	2	2	3	4	5	6
d	3	2	2	3	3	5	5	6
a	4	3	3	3	4	3	4	5
y	5	4	4	4	4	4	3	4
	6	5	4	4	5	5	4	3

# Business Process Automation with Python

## Fuzzy Match

- Levenshtein Algorithm – More example
  - $\min(4, 3, 5) \rightarrow 3$
  - No need to add 1 because d vs d
  - Answer: 3

	s	a	t	u	r	d	a	y
0	1	2	3	4	5	6	7	8
s	1	0	1	2	3	4	5	6
u	2	1	1	2	2	3	4	5
n	3	2	2	2	3	3	5	6
d	4	3	3	3	3	4	4	5
a	5	4	3	4	4	4	3	4
y	6	5	4	4	5	5	4	3

# Business Process Automation with Python

## Fuzzy Match

- Levenshtein Algorithm
  - $\min(4, 3, 4) \rightarrow 3$
  - No need to add 1 because y vs y
- Answer: 3

S	u	n	d	a	y		
S	a	t	u	r	d	a	y

A 7x8 grid representing the Levenshtein distance between 'sunday' and 'saturday'. The columns are labeled 0 through 8, and the rows are labeled with the letters of 'sunday' (s, u, n, d, a, y). The cell at (0,0) contains 0. The cell at (6,7) contains 3, highlighted with a yellow box and arrows pointing to it from the bottom right.

	s	a	t	u	r	d	a	y
0	1	2	3	4	5	6	7	8
s	1	0	1	2	3	4	5	6
u	2	1	1	2	2	3	4	5
n	3	2	2	2	3	3	5	5
d	4	3	3	3	3	4	3	4
a	5	4	3	4	4	4	3	4
y	6	5	4	4	5	5	4	3

# Business Process Automation with Python

## Fuzzy Match

- Levenshtein Algorithm
  - Q: How to write it in Python?

S	u	n	d	a	y		
S	a	t	u	r	d	a	y

S			u	n	d	a	y
S	a	t	u	r	d	a	y

# Business Process Automation with Python

## Fuzzy Match

- Levenshtein Algorithm
  - Interview question (not for this course 😊)
  - Explainer from NeetCode if you are interested: <https://youtu.be/XYi2-LPrwm4>

```
class Solution:  
    def minDistance(self, word1: str, word2: str) -> int:  
        cache = [[float("inf")]] * (len(word2) + 1) for i in range(len(word1) + 1)]  
  
        for j in range(len(word2) + 1):  
            cache[len(word1)][j] = len(word2) - j  
        for i in range(len(word1) + 1):  
            cache[i][len(word2)] = len(word1) - i  
  
        for i in range(len(word1) - 1, -1, -1):  
            for j in range(len(word2) - 1, -1, -1):  
                if word1[i] == word2[j]:  
                    cache[i][j] = cache[i + 1][j + 1]  
                else:  
                    cache[i][j] = 1 + min(cache[i + 1][j], cache[i][j + 1], cache[i+1][j+1])  
  
        return cache[0][0]
```

# Business Process Automation with Python

## Fuzzy Match

- Levenshtein Algorithm
  - <https://pypi.org/project/python-Levenshtein/>
  - pip install python-Levenshtein

**python-Levenshtein 0.21.0**

pip install python-Levenshtein 

- Import Levenshtein

```
▶ import Levenshtein
```

# Business Process Automation with Python

## Fuzzy Match

- Levenshtein Algorithm
  - `Levenshtein.distance(X, Y)`
  - Result is symmetric

```
▶ X = 'SUNDAY'  
Y = 'SATURDAY'  
Levenshtein.distance(X, Y)
```

```
⇨ 3
```

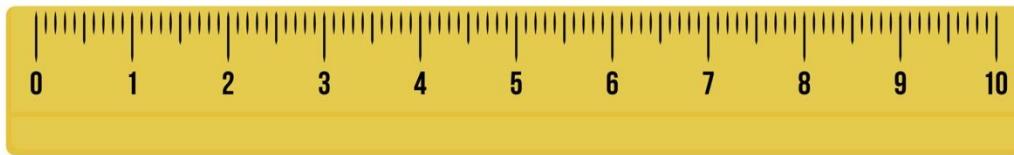
```
▶ Levenshtein.distance(Y, X)
```

```
⇨ 3
```

# Business Process Automation with Python

## Fuzzy Match

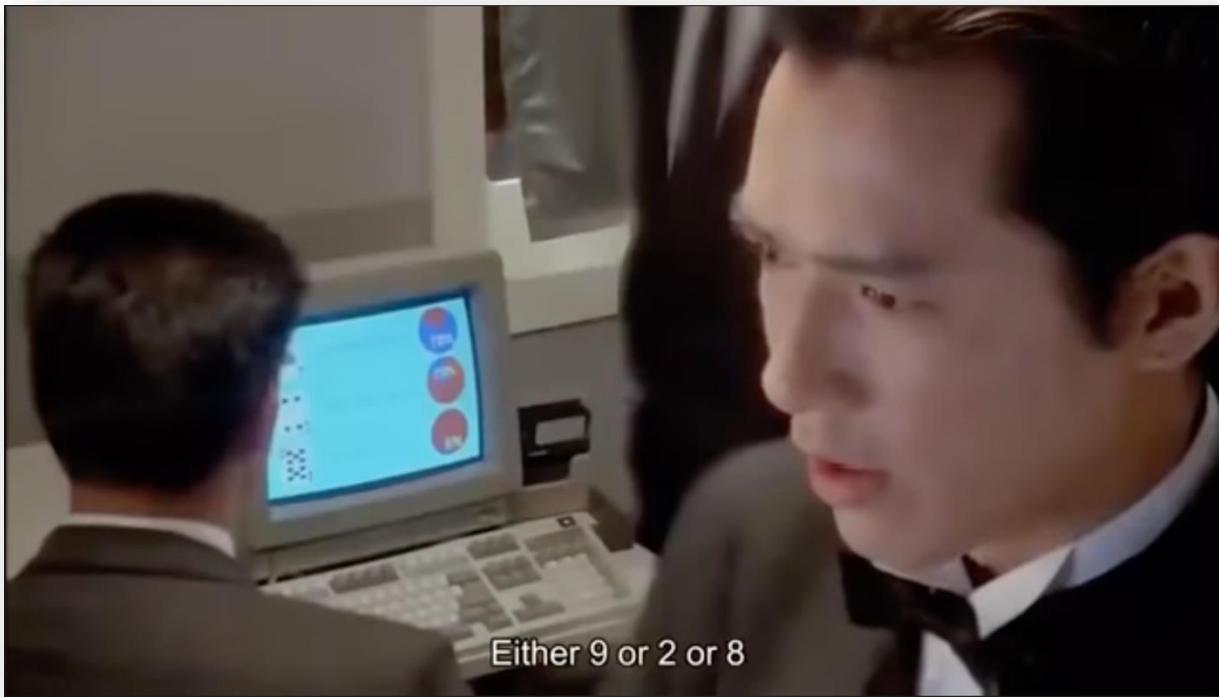
- **Q: Wait, what were we trying to do?**
  - Measure similarity of 2 string



# Business Process Automation with Python

## Fuzzy Match

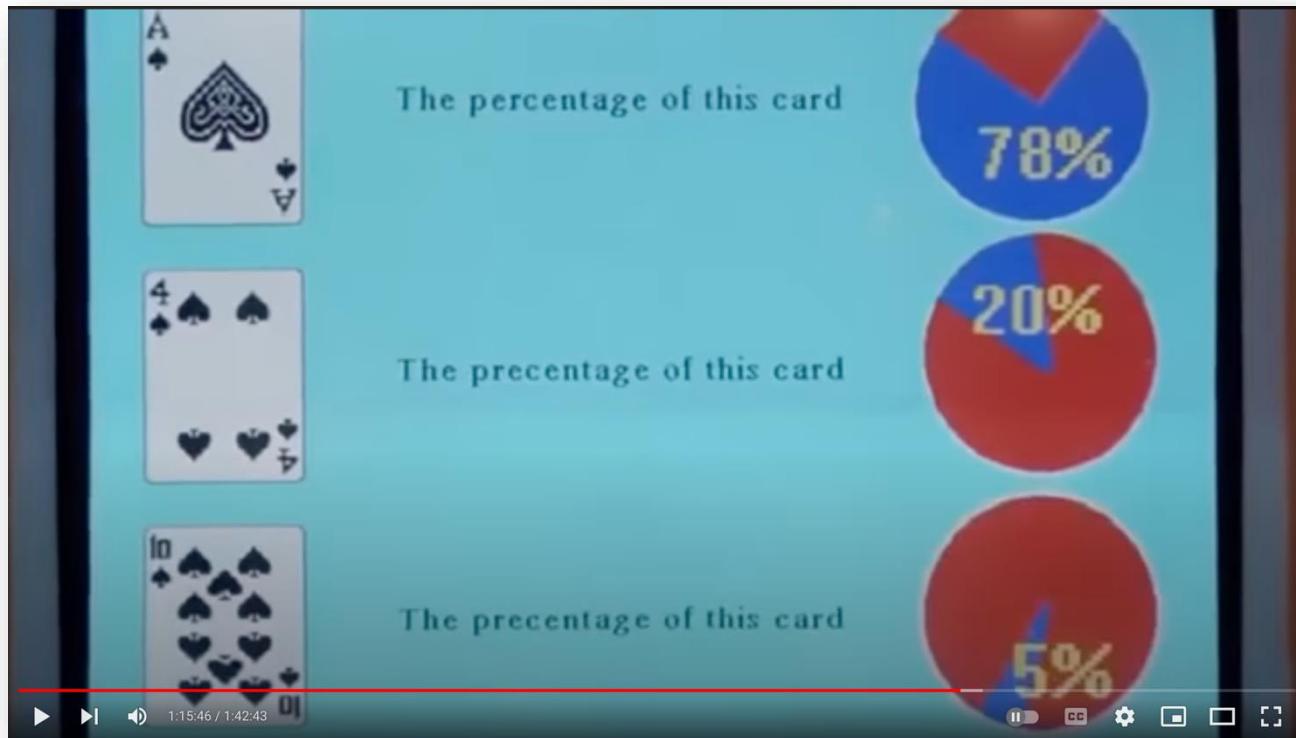
- Clues from God of Gamblers 2 (1990)
  - <https://youtu.be/Nl8PkOifBw?si=cujCSkENbXfvITfC&t=235>



# Business Process Automation with Python

## Fuzzy Match

- Distance → Similarity scores
  - Convert the distance into a number from 0 to 100%



^ sum of possibilities may not be 100%

# Business Process Automation with Python

## Fuzzy Match

- FuzzyWuzzy
  - <https://pypi.org/project/fuzzywuzzy/>

FuzzyWuzzy



Fuzzy string matching like a boss. It uses [Levenshtein Distance](#) to calculate the differences between sequences in a simple-to-use package.

- pip install fuzzywuzzy

fuzzywuzzy 0.18.0

pip install fuzzywuzzy 

- Import before use

```
▶ from fuzzywuzzy import fuzz, process
```

# Business Process Automation with Python

## Fuzzy Match

- `process.extract()` → Use partial ratio by default
  - Useful → matching substring (e.g., start/end)

```
▶ process.extract('hku', choices=['cuhk', 'hkust', 'hkuspace'])  
⇒ [('hkust', 90), ('hkuspace', 90), ('cuhk', 57)]
```

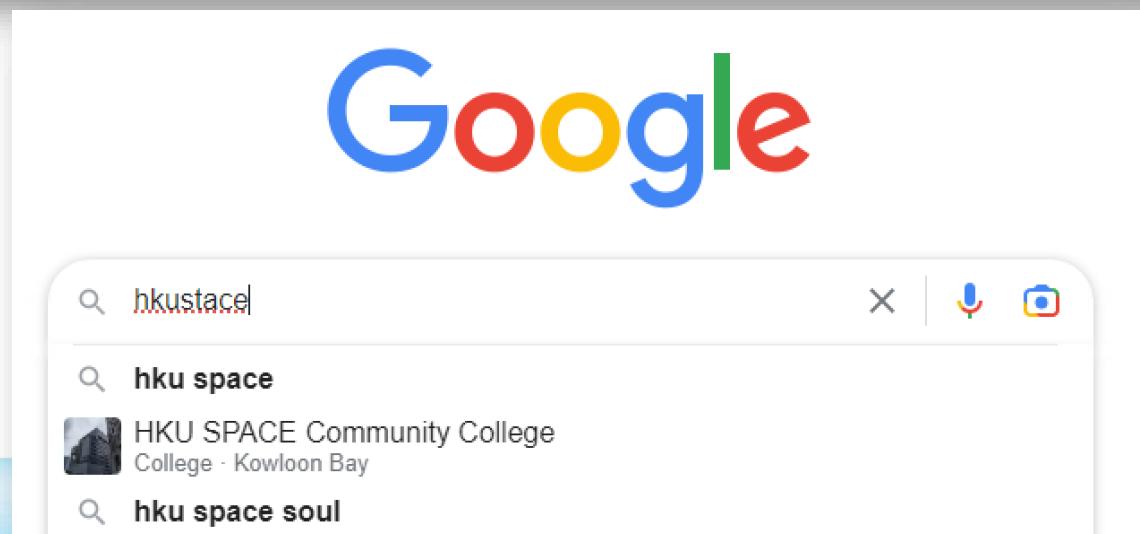


# Business Process Automation with Python

## Fuzzy Match

- Similarity ratio
  - Length factor: hku → hkust > hkuspace
  - Block factor (continuous): hku → hkust > cuhk
  - Useful → Comparing words of similar length

```
▶ # Similarity Ratio
    process.extract('hku', choices=['cuhk', 'hkust', 'hkuspace'], scorer=fuzz.ratio)
⇒ [('hkust', 75), ('cuhk', 57), ('hkuspace', 55)]
```



# Business Process Automation with Python

## Fuzzy Match

- Partial token sort ratio
  - Sort into a word list to compare
  - Useful → compare names (first/last name order)

```
▶ process.extract('Hello Hong Kong',
                  choices=['Kong Hello Hong', 'Hello King Kong'],
                  scorer=fuzz.partial_token_sort_ratio)

⇒ [('Kong Hello Hong', 100), ('Hello King Kong', 87)]
```



# Business Process Automation with Python

## Fuzzy Match

- Partial token set ratio
  - Considers unique tokens with loose comparison
  - Useful → Check against plagiarism

```
▶ process.extract('Ding Ding',
                  choices=['Ding', 'Dong'],
                  scorer=fuzz.partial_token_set_ratio)

⇒ [('Ding', 100), ('Dong', 75)]
```

```
▶ process.extract('Nanae Aoyama',
                  choices=['Aoyama Nanae', 'Aoyama Nanako'],
                  scorer=fuzz.partial_token_set_ratio)

⇒ [('Aoyama Nanae', 100), ('Aoyama Nanako', 100)]
```

```
[35] process.extract('Nanae Aoyama',
                     choices=['Aoyama Nanae', 'Aoyama Nanako'],
                     scorer=fuzz.partial_token_sort_ratio)

⇒ [('Aoyama Nanae', 100), ('Aoyama Nanako', 92)]
```

# Business Process Automation with Python

## Fuzzy Match

- More practical application
  - Clean up data (e.g., Survey response)
    1. Input → Hong Kong
    2. Expected → Hong Kong SAR
    3. New Column → Fuzzy match

<u>Input</u>	<u>Fuzzy Matched</u>
Hong Kong	Hong Kong SAR

FirstName	Surname	CompanyName	Address1	Town
Lee	Leighton	Alter Image	131 Sundown Close	
L	Leighton	Alter Image Ltd	131 Sundown Rd	Leicester
Emma	Wrighte	Write Way	280 Bath road	Birmingham
E	Wright	The Write Way	280 Bath rd	
Emma	Wright	The Write Way Ltd	280 Bath road	Birmingham
William	Giles	Social & Military Club	78 Ford St	Bolton
Bill	Giles	S & M Club	78, Ford Street	Bolton
Alan	Baker	Bakery Baker Ltd	7 main road	Lowton
Al	Baker	Bakery Baker	7 main Rd	Lowton



**HKU SPACE**  
香港大學專業進修學院  
HKU School of Professional and Continuing Education

# THANK YOU

