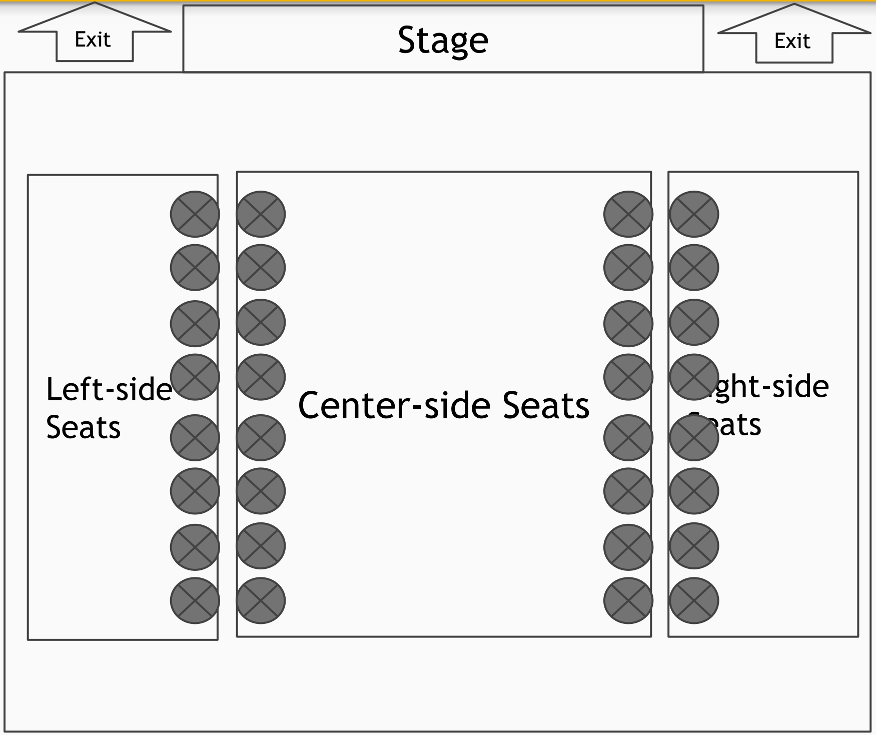
Yasumoto International Academic Park - YIA LT6

Limited Mobile Signal. Please use on-campus wifi.
32 Sockets. But please bring your own charger.



⊗ = sockets

Python Basics - Programming 201

Exception Handling, Functions, Packages

CUHK MSc Data Science & Biz Stat. Program
STAT5106 - Programming Techniques for Data Science
Week 2 @ 19 Sept 2024

Week 6 lesson to be shifted. 17 Oct → 24 Oct

Hello everyone,

I would like to take leave on **October 17** and need your approval.
The lesson will be rescheduled to **October 24**.

Best regards, Alan 🥹 🙇🏽‍♂️

The following will be executed if this is approved.

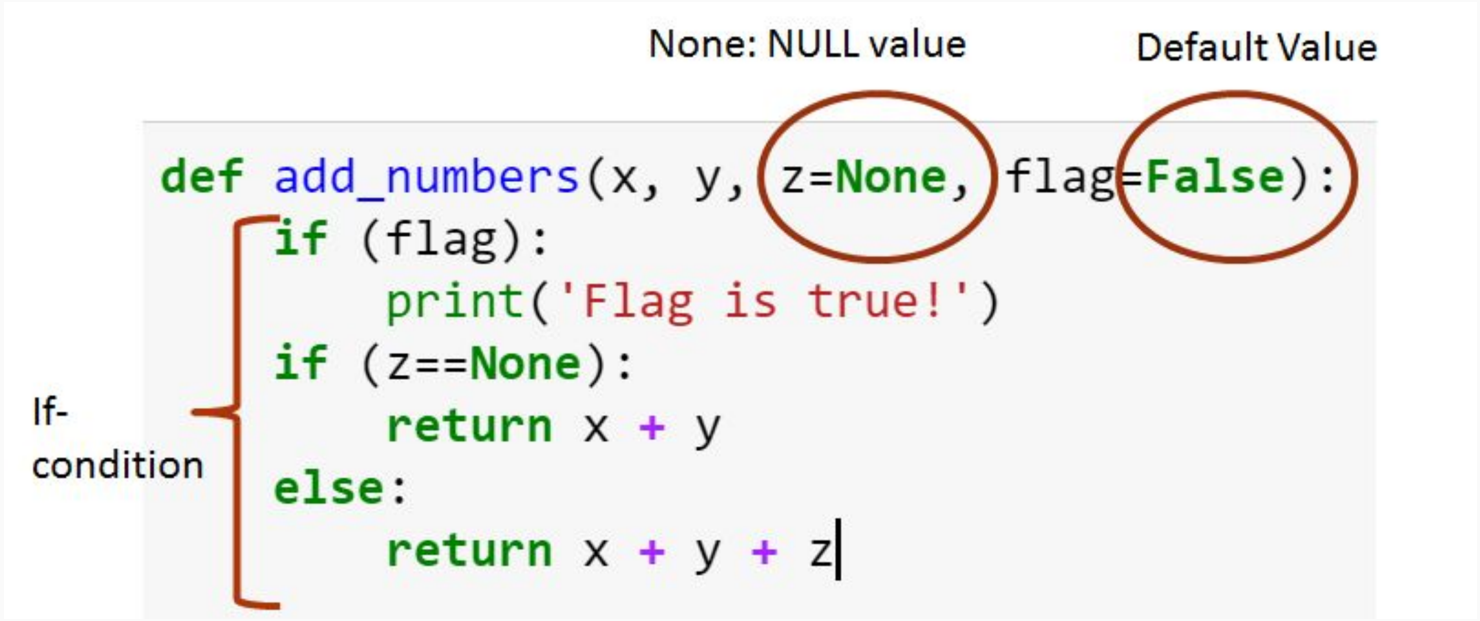
Date	Topic
17 Oct	(No class)
Week 6 - ~~17 Oct~~ 24 Oct	APIs, with More Example on Open Data
Anyday between 24-31 Oct	Mid-term Take-Home Exam (most likely 48 hours from **25 (Fri) - 27 (Sun)** Oct evenings)

Start Coding…

Please access into the

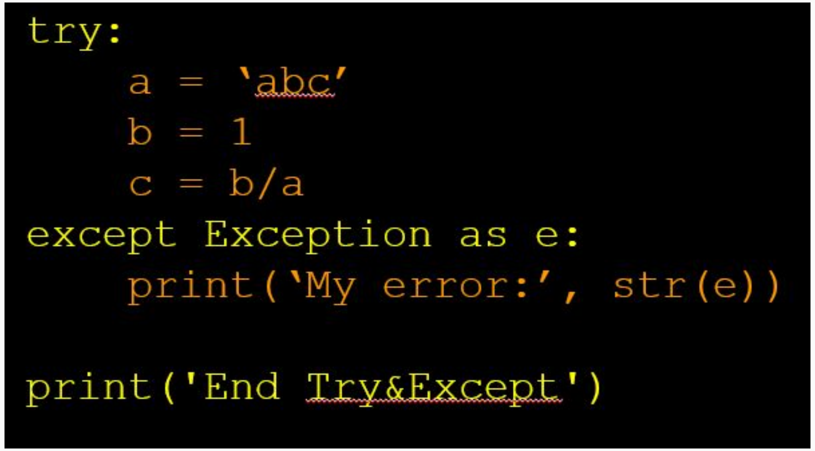
- Week 1 colab - Programming 101 …
  (We still have not yet done the part of File I/O)
- Week 2 colab - Programming 201 …

Functions

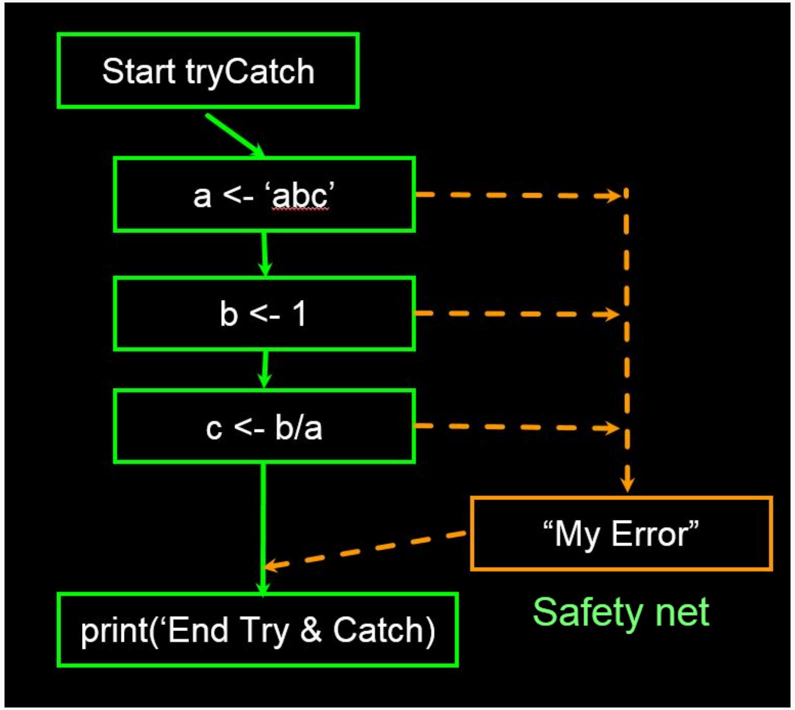


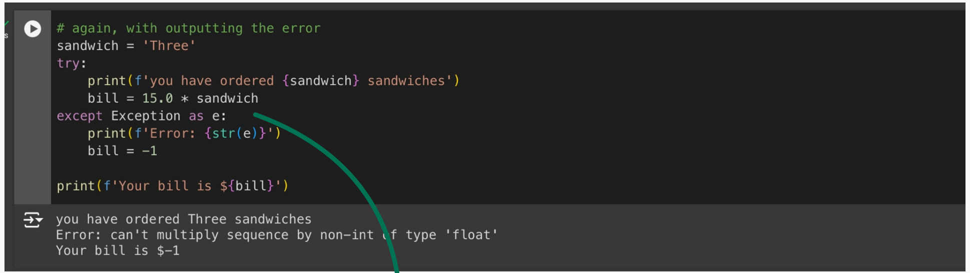
```python
def add_numbers(x, y, z=None, flag=False):
    if (flag):
        print('Flag is true!')
    if (z==None):
        return x + y
    else:
        return x + y + z
```

Try-Catch Exception Handling



```
try:
     a = 'abc'
     b = 1
     c = b/a
except Exception as e:
     print('My error:', str(e))

print('End Try&Except')
```





```
# again, with outputting the error
sandwich = 'Three'
try:
    print(f'you have ordered {sandwich} sandwiches')
    bill = 15.0 * sandwich
except Exception as e:
    print(f'Error: {str(e)}')
    bill = -1

print(f'Your bill is ${bill}')
```

```
you have ordered Three sandwiches
Error: can't multiply sequence by non-int of type 'float'
Your bill is $-1
```

save error msg as e

Python Packages

Python Package Index (PyPI)

- Up to 13 Sept 2024, package no. 568,789 - but tiny of them are useful
- os, datetime, pyodbc, re

pandas NumPy matplotlib seaborn statsmodels

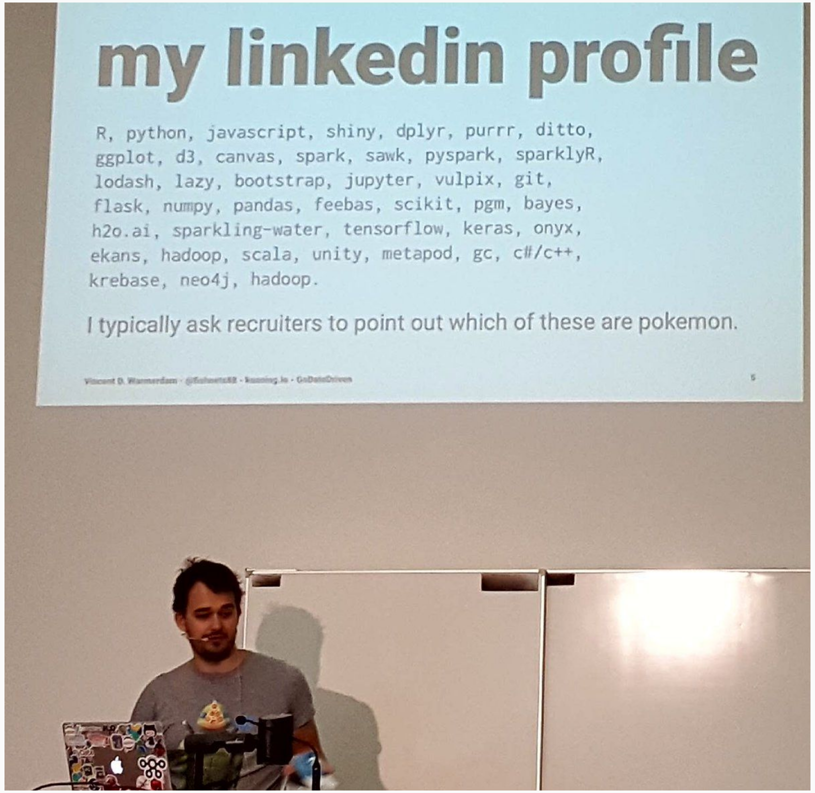
scikit learn scikit-image image processing in python spaCy TensorFlow PyTorch

How to install: `pip install "SomeProject"`
How to update: `pip install --upgrade SomeProject`

How to import:

```
import os
import pandas as pd
from urllib import request
from functions import *
```

Python Packages

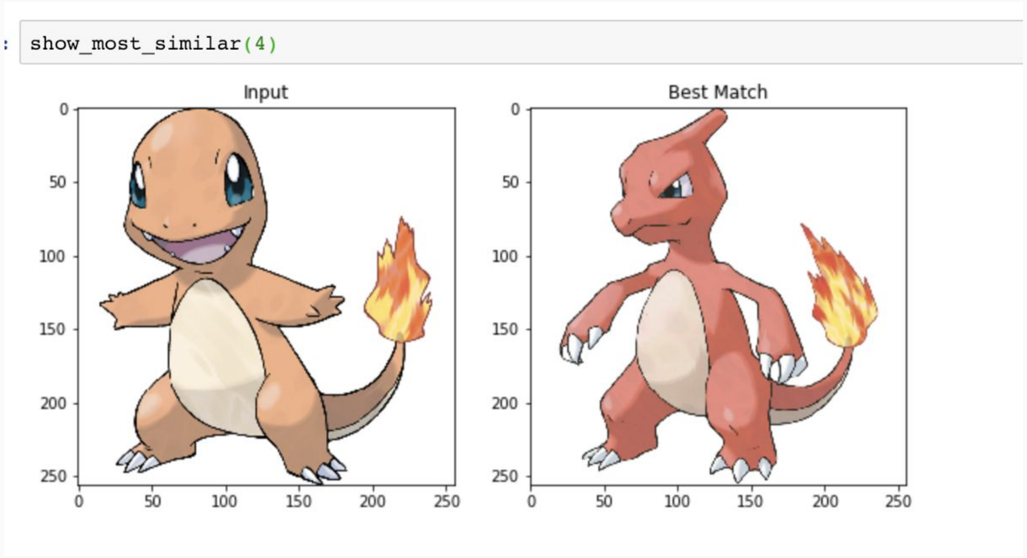


my linkedin profile

R, python, javascript, shiny, dplyr, purrr, ditto, ggplot, d3, canvas, spark, sawk, pyspark, sparklyR, lodash, lazy, bootstrap, jupyter, vulpix, git, flask, numpy, pandas, feebas, scikit, pgm, bayes, h2o.ai, sparkling-water, tensorflow, keras, onyx, ekans, hadoop, scala, unity, metapod, gc, c#/c++, krebase, neo4j, hadoop.

I typically ask recruiters to point out which of these are pokemon.

Vincent D. Warmerdam · @fishnets88 · koaning.io · GoDataDriven

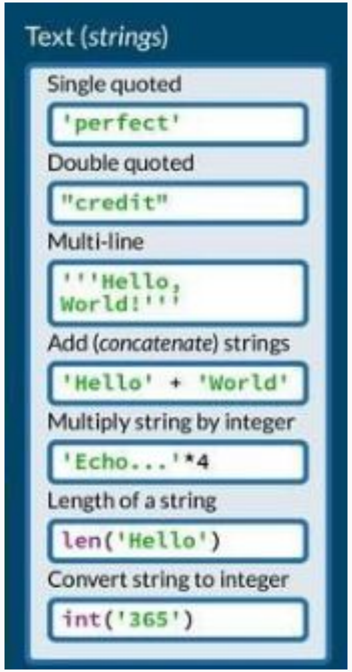


```
show_most_similar(4)
```

pyPokedex

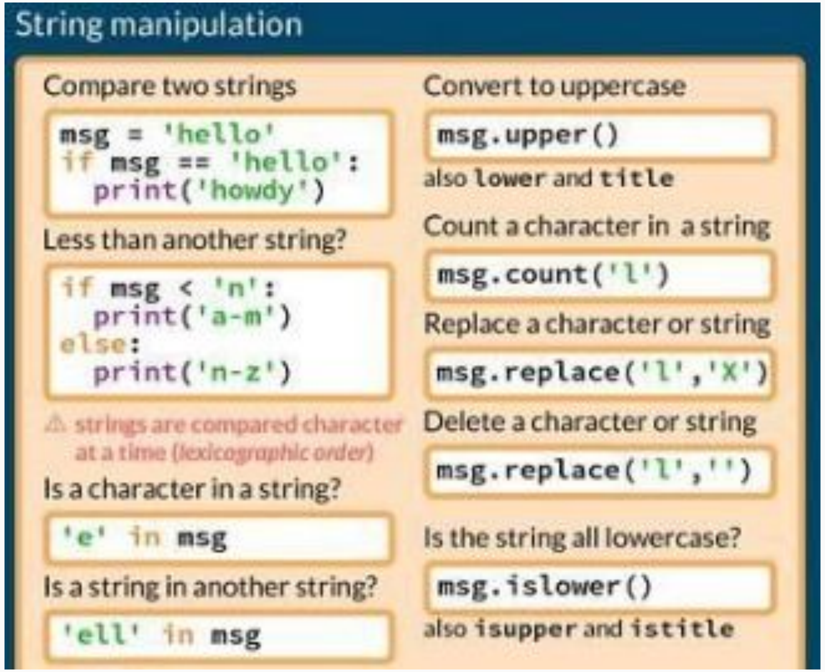
Welcome to check more funny Python Packages…

String



Text (*strings*)

Single quoted
```
'perfect'
```
Double quoted
```
"credit"
```
Multi-line
```
'''Hello,
World!'''
```
Add (*concatenate*) strings
```
'Hello' + 'World'
```
Multiply string by integer
```
'Echo...'*4
```
Length of a string
```
len('Hello')
```
Convert string to integer
```
int('365')
```



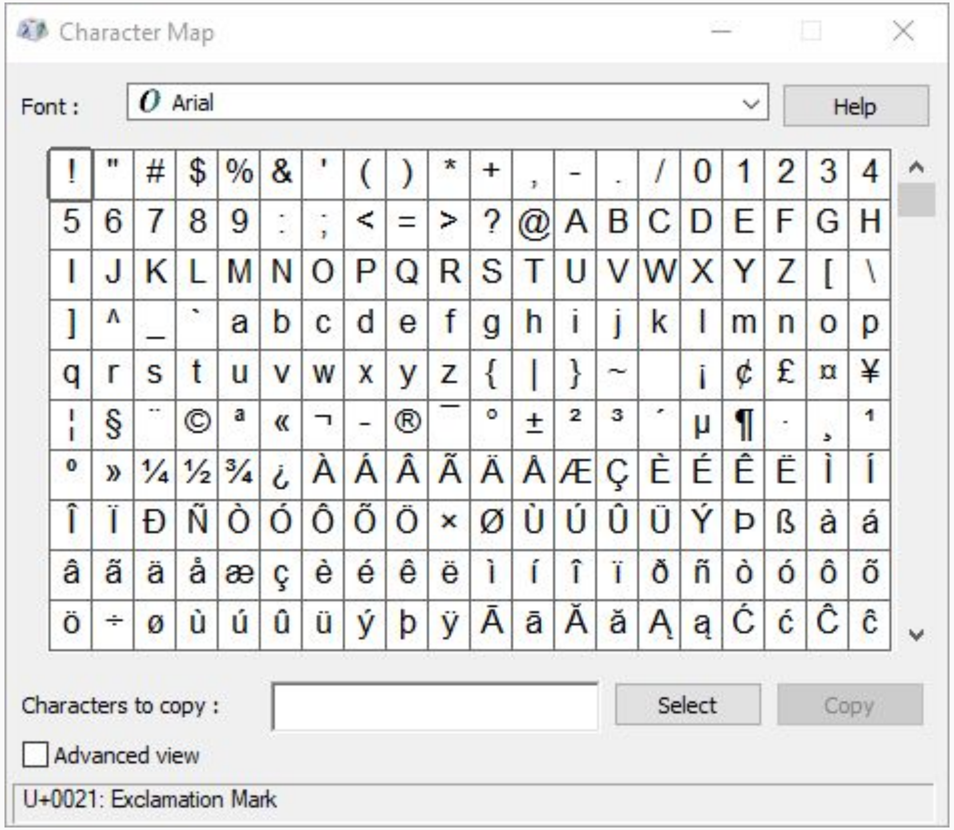
String manipulation

Compare two strings
```
msg = 'hello'
if msg == 'hello':
  print('howdy')
```
Less than another string?
```
if msg < 'n':
  print('a-m')
else:
  print('n-z')
```
⚠ strings are compared character at a time (*lexicographic order*)

Is a character in a string?
```
'e' in msg
```
Is a string in another string?
```
'ell' in msg
```
Convert to uppercase
```
msg.upper()
```
also `lower` and `title`

Count a character in a string
```
msg.count('l')
```
Replace a character or string
```
msg.replace('l','X')
```
Delete a character or string
```
msg.replace('l','')
```
Is the string all lowercase?
```
msg.islower()
```
also `isupper` and `istitle`

Unicode Character



Character Map

Font : Arial

Help

!	"	#	$	%	&	'	(	)	*	+	,	-	.	/	0	1	2	3	4
5	6	7	8	9	:	;	<	=	>	?	@	A	B	C	D	E	F	G	H
I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[	\
]	^	_	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
q	r	s	t	u	v	w	x	y	z	{	|	}	~		¡	¢	£	¤	¥
¦	§	¨	©	ª	«	¬	-	®	¯	°	±	²	³	´	µ	¶	·	¸	¹
º	»	¼	½	¾	¿	À	Á	Â	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í
Î	Ï	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß	à	á
â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï	ð	ñ	ò	ó	ô	õ
ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ	Ā	ā	Ă	ă	Ą	ą	Ć	ć	Ĉ	ĉ

Characters to copy :

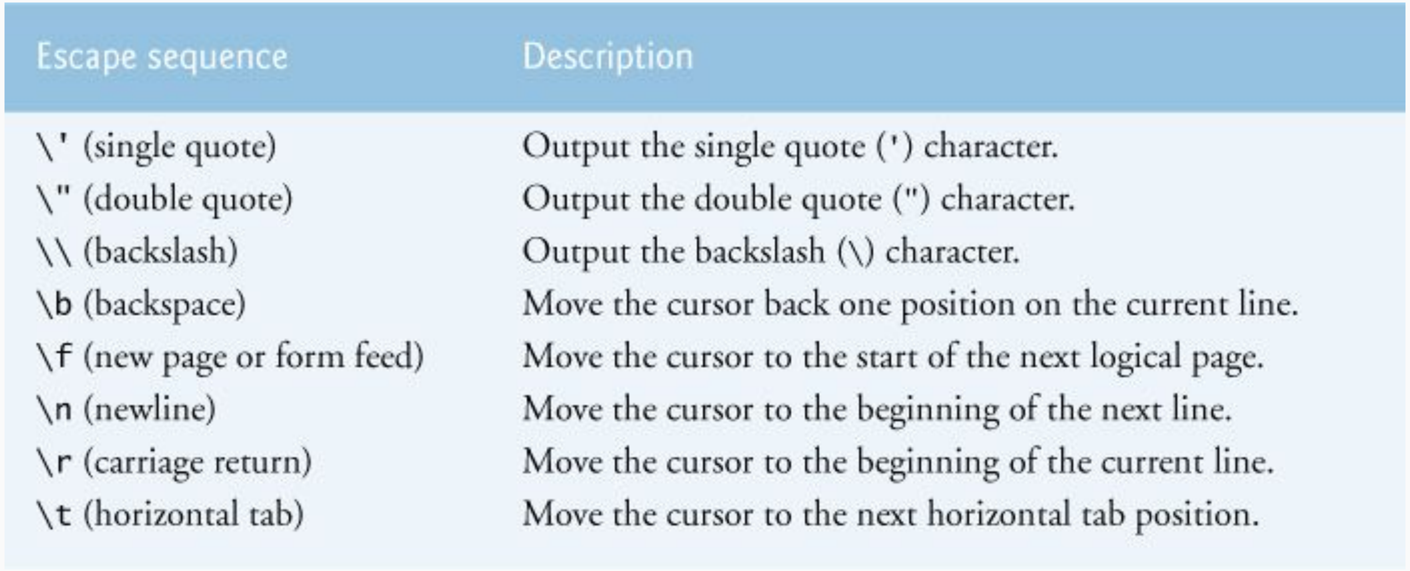
Select

Copy

☐ Advanced view

U+0021: Exclamation Mark

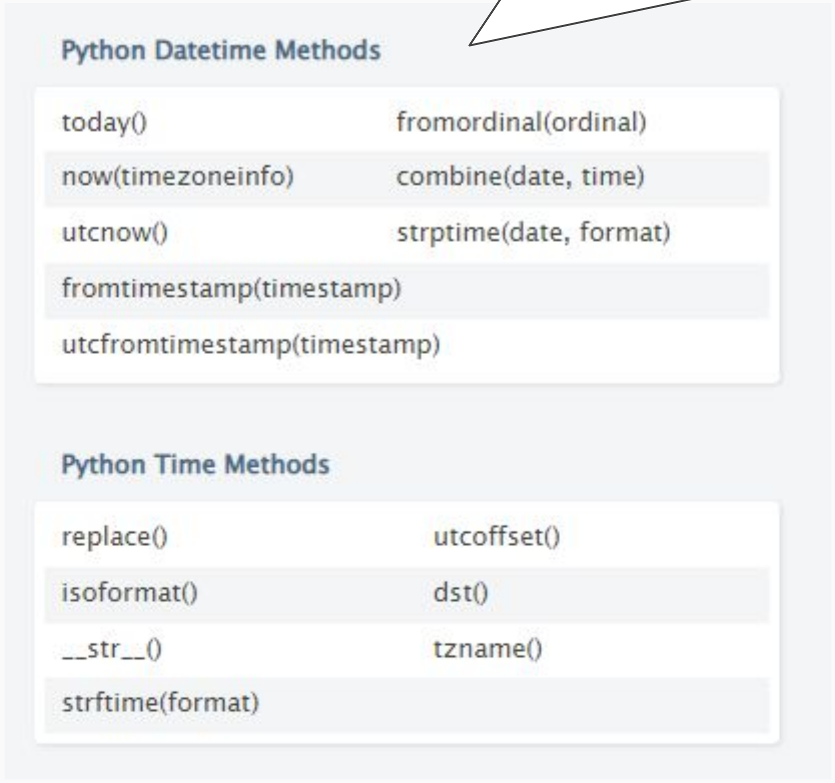
Escape Character



Escape sequence	Description
\' (single quote)	Output the single quote (') character.
\" (double quote)	Output the double quote (") character.
\\ (backslash)	Output the backslash (\) character.
\b (backspace)	Move the cursor back one position on the current line.
\f (new page or form feed)	Move the cursor to the start of the next logical page.
\n (newline)	Move the cursor to the beginning of the next line.
\r (carriage return)	Move the cursor to the beginning of the current line.
\t (horizontal tab)	Move the cursor to the next horizontal tab position.

Datetime

```
from datetime import datetime as dt
import time
```

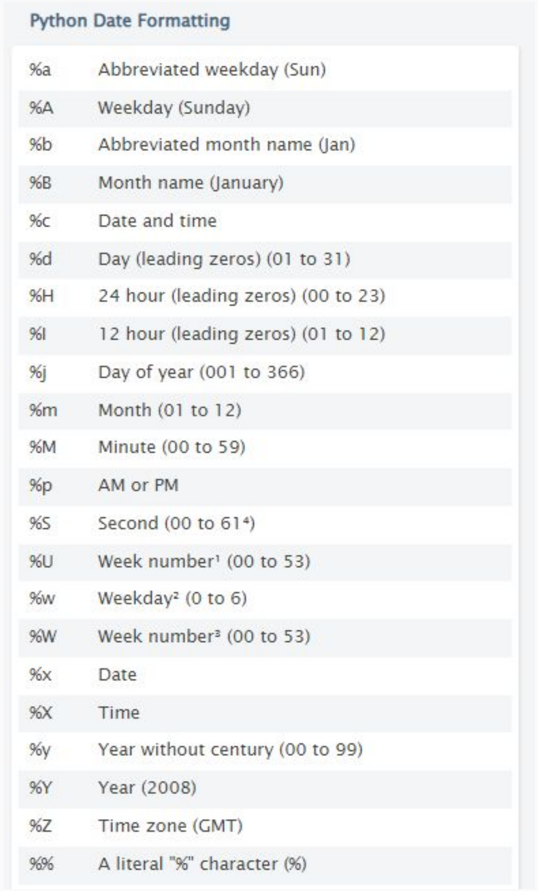


Python Datetime Methods

today()	fromordinal(ordinal)
now(timezoneinfo)	combine(date, time)
utcnow()	strptime(date, format)
fromtimestamp(timestamp)	
utcfromtimestamp(timestamp)	

Python Time Methods

replace()	utcoffset()
isoformat()	dst()
__str__()	tzname()
strftime(format)	



Python Date Formatting

%a	Abbreviated weekday (Sun)
%A	Weekday (Sunday)
%b	Abbreviated month name (Jan)
%B	Month name (January)
%c	Date and time
%d	Day (leading zeros) (01 to 31)
%H	24 hour (leading zeros) (00 to 23)
%I	12 hour (leading zeros) (01 to 12)
%j	Day of year (001 to 366)
%m	Month (01 to 12)
%M	Minute (00 to 59)
%p	AM or PM
%S	Second (00 to 61[4])
%U	Week number[1] (00 to 53)
%w	Weekday[2] (0 to 6)
%W	Week number[3] (00 to 53)
%x	Date
%X	Time
%y	Year without century (00 to 99)
%Y	Year (2008)
%Z	Time zone (GMT)
%%	A literal "%" character (%)

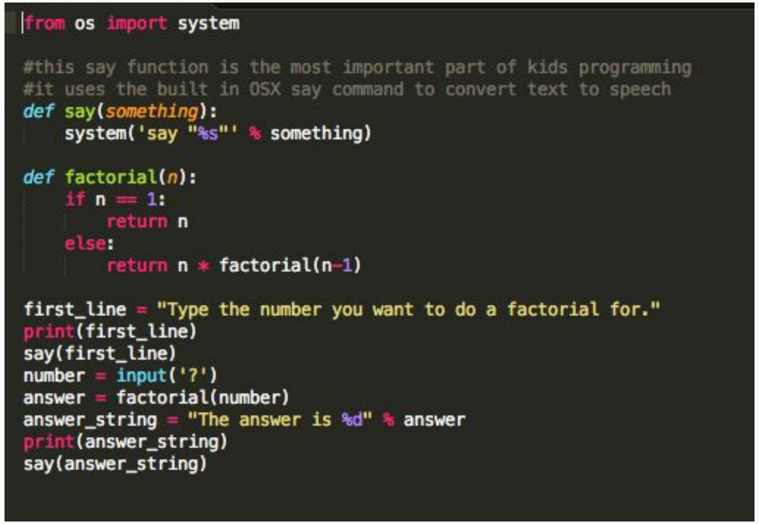
Programming Techniques

1. Know your vocab and grammar - Python Language
2. “Tell a Story” - with building block of programs
3. Create variables with meaningful names
4. Knowing the error type well
   a. Syntax Error
   b. Logic Error
   c. Semantic Error
   (Hint: Copy the whole error message to Google)
5. Debugging
   Print variables and understand the logic
6. Using the try-catch exception well

Programming vs Scripting

Program

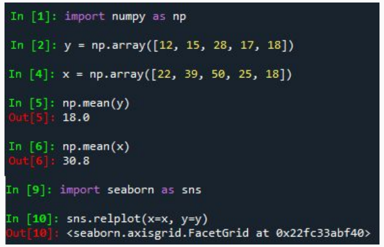
- can be complied (so you can run faster)
- like an acticle with different blocks
  - input, control, loop, exception, output



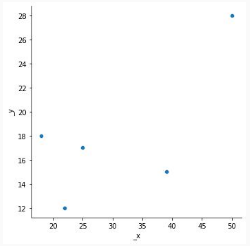
```
from os import system

#this say function is the most important part of kids programming
#it uses the built in OSX say command to convert text to speech
def say(something):
    system('say "%s"' % something)

def factorial(n):
    if n == 1:
        return n
    else:
        return n * factorial(n-1)

first_line = "Type the number you want to do a factorial for."
print(first_line)
say(first_line)
number = input('?')
answer = factorial(number)
answer_string = "The answer is %d" % answer
print(answer_string)
say(answer_string)
```

Script

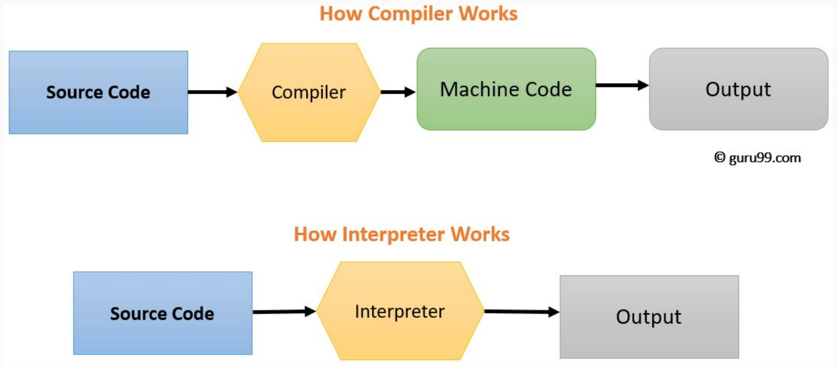
- one sentence for one task
- can be interpreted (press Enter and execute)



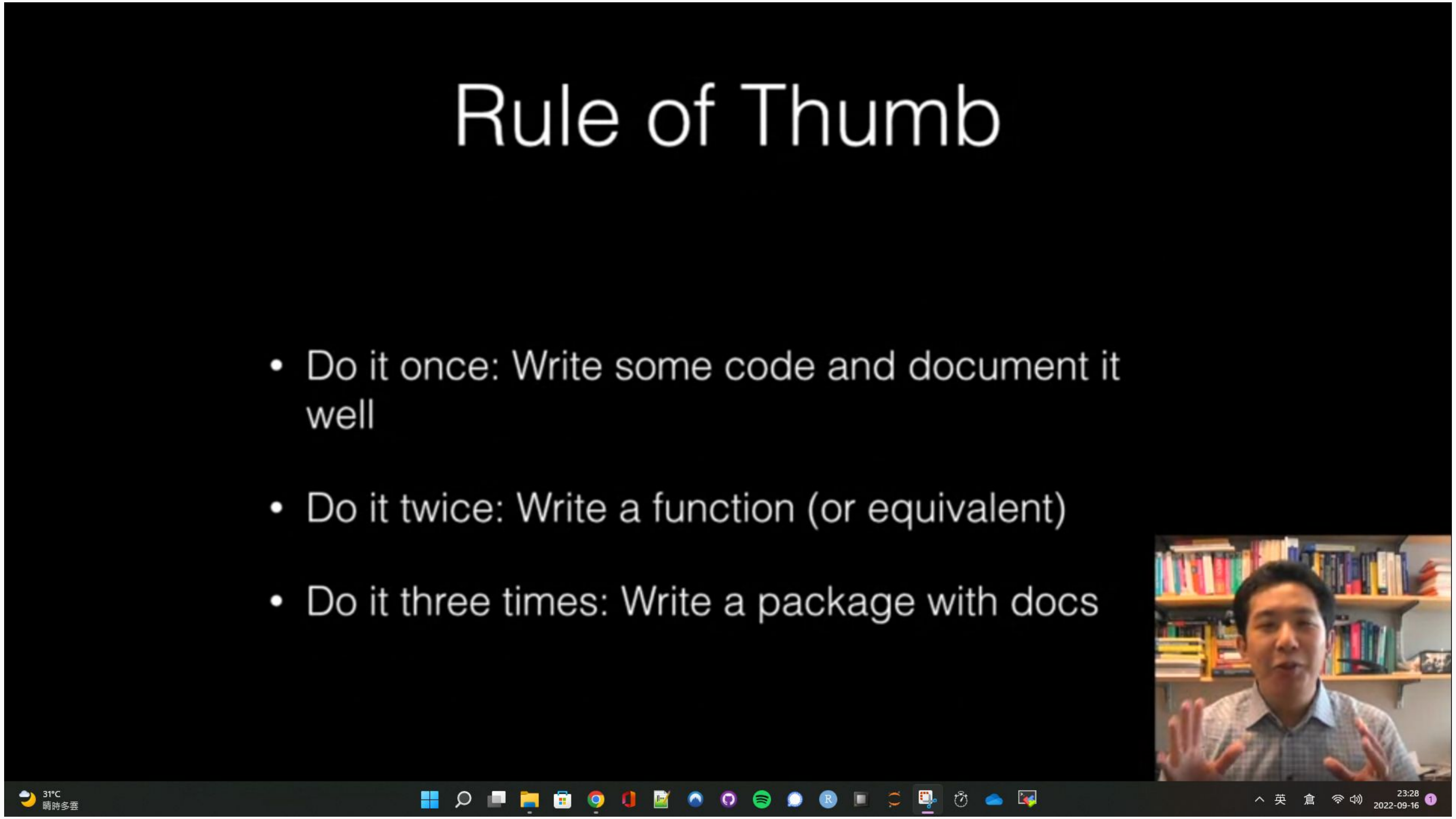
```
In [1]: import numpy as np

In [2]: y = np.array([12, 15, 28, 17, 18])

In [4]: x = np.array([22, 39, 50, 25, 18])

In [5]: np.mean(y)
Out[5]: 18.0

In [6]: np.mean(x)
Out[6]: 30.8

In [9]: import seaborn as sns

In [10]: sns.relplot(x=x, y=y)
Out[10]: <seaborn.axisgrid.FacetGrid at 0x22fc33abf40>
```



FYI: [Complier vs Interpreter]



When should we write a function / package ?



Rule of Thumb

- Do it once: Write some code and document it well
- Do it twice: Write a function (or equivalent)
- Do it three times: Write a package with docs

To be continue…