ARTIFICIAL INTELLIGENCE – SUMMER OF CODE - NAVTTC

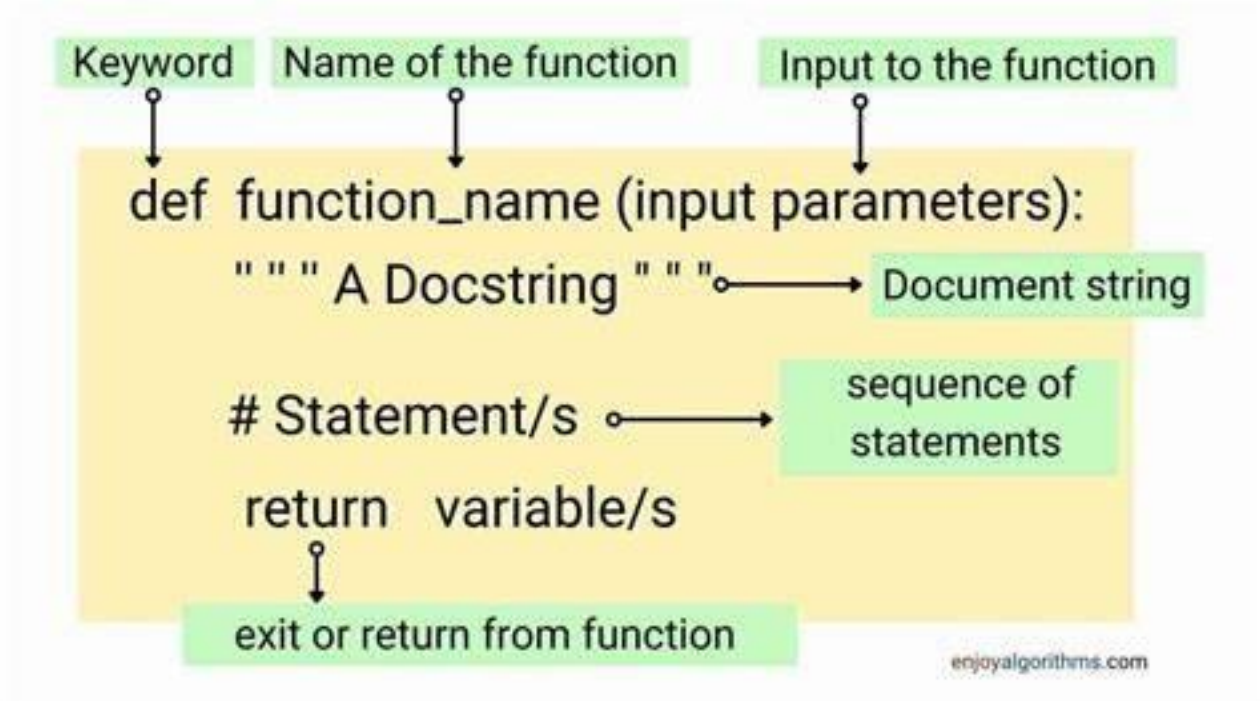
[Course]

Week 2

[See examples / code in GitHub code repository]

It is not about Theory, it is 20% Theory and 80% Practical – Technical/Development/Programming [Mostly Python based]

Functions



- Types of Python Functions:
- (Built-in functions , Functions defined in built-in modules, User-defined funct
  Pass by Reference vs Value

References:

https://www.w3schools.com/python/python_functions.asp
https://www.geeksforgeeks.org/python-functions/
https://www.tutorialspoint.com/python/python_functions.htm

File Operations

Python File Handling

1. Create Files
2. Read Files
3. Write to Files

1. List Files From Directory
2. Copy, Rename, Delete Files from Directory
3. Copy, Delete Directories

```
# Create and Write
with open('test.txt', 'w') as fp:
    fp.write('new line')
# Read
with open('test.txt', 'r') as fp:
    fp.read()
```

```
os.rename('old_file_name', 'new_file_name')
os.remove('file_path')

shutil.copy('src_file_path', 'new_path')
shutil.move('src_file_path', 'new_path')

os.listdir('dir_path') # Get all files
shutil.rmtree('path')  # Remove directory
shutil.copytree('src_path', 'dst_path') # Copy dir
```

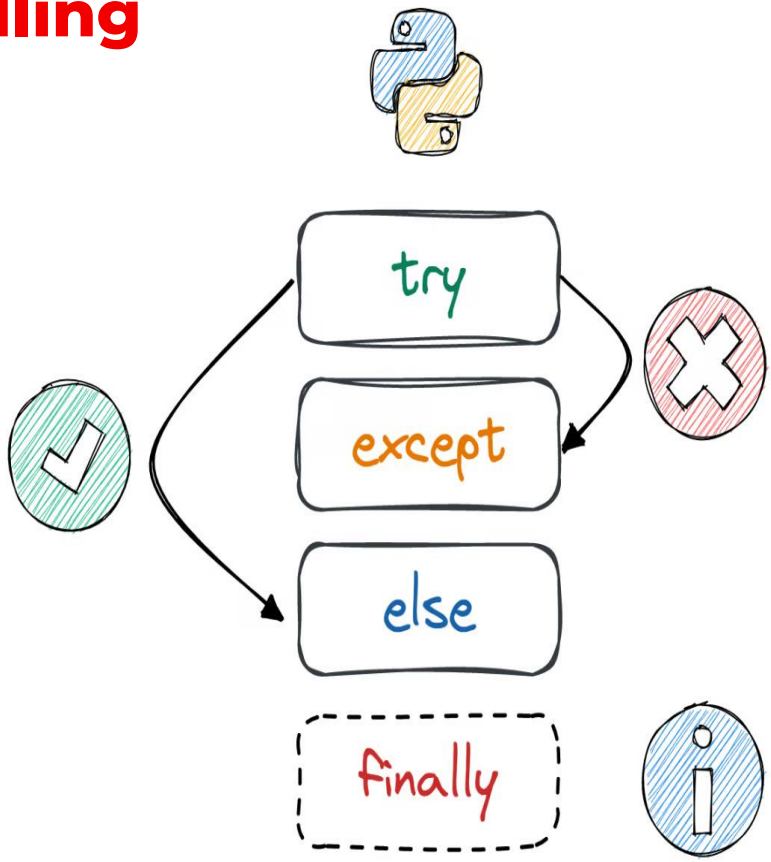
PYnative.com

References:

https://www.geeksforgeeks.org/file-handling-python/
https://www.w3schools.com/python/python_file_handling.asp
https://www.includehelp.com/python/file-handling-programs.aspx

Exercises

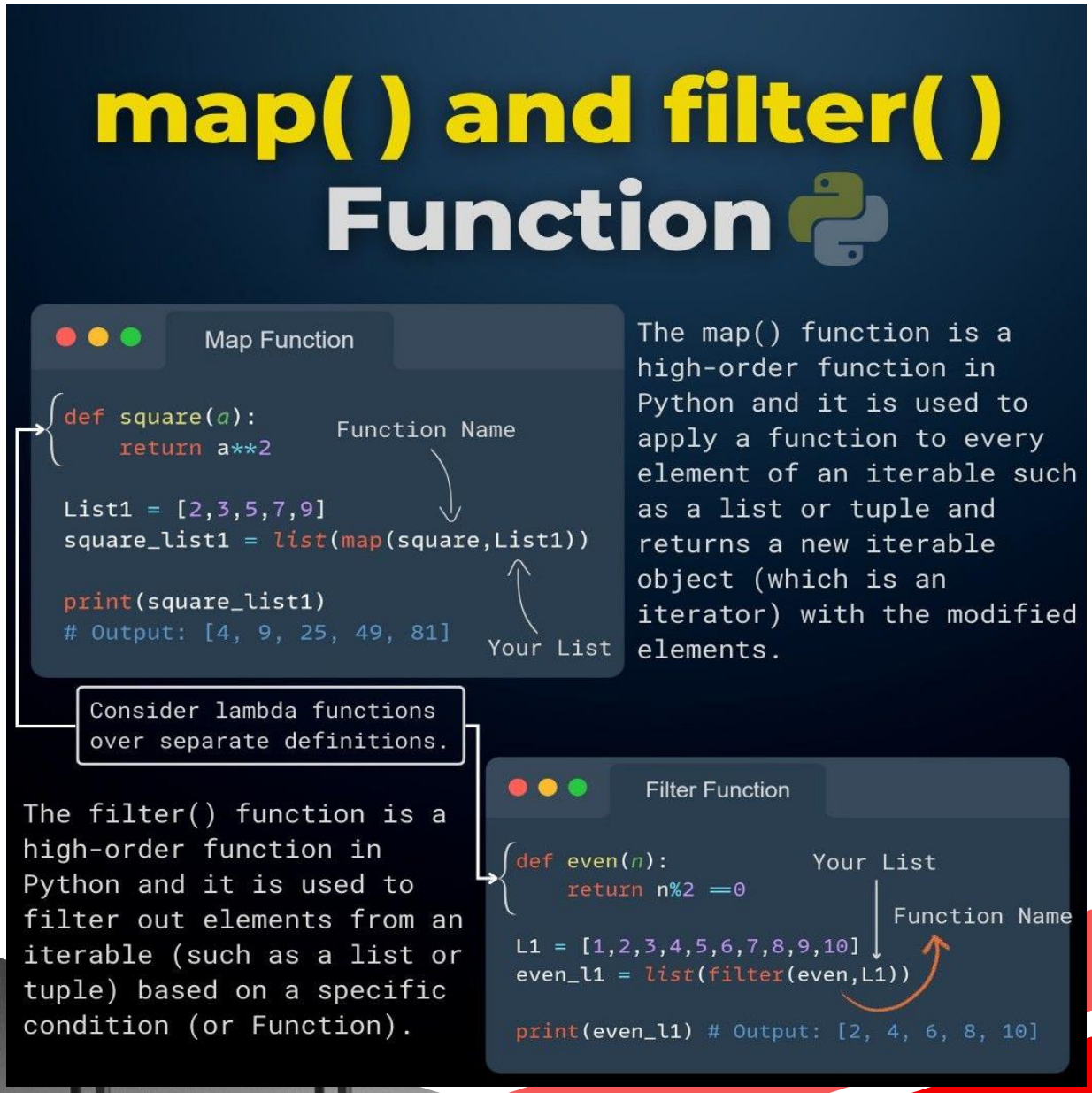
Exception Handling



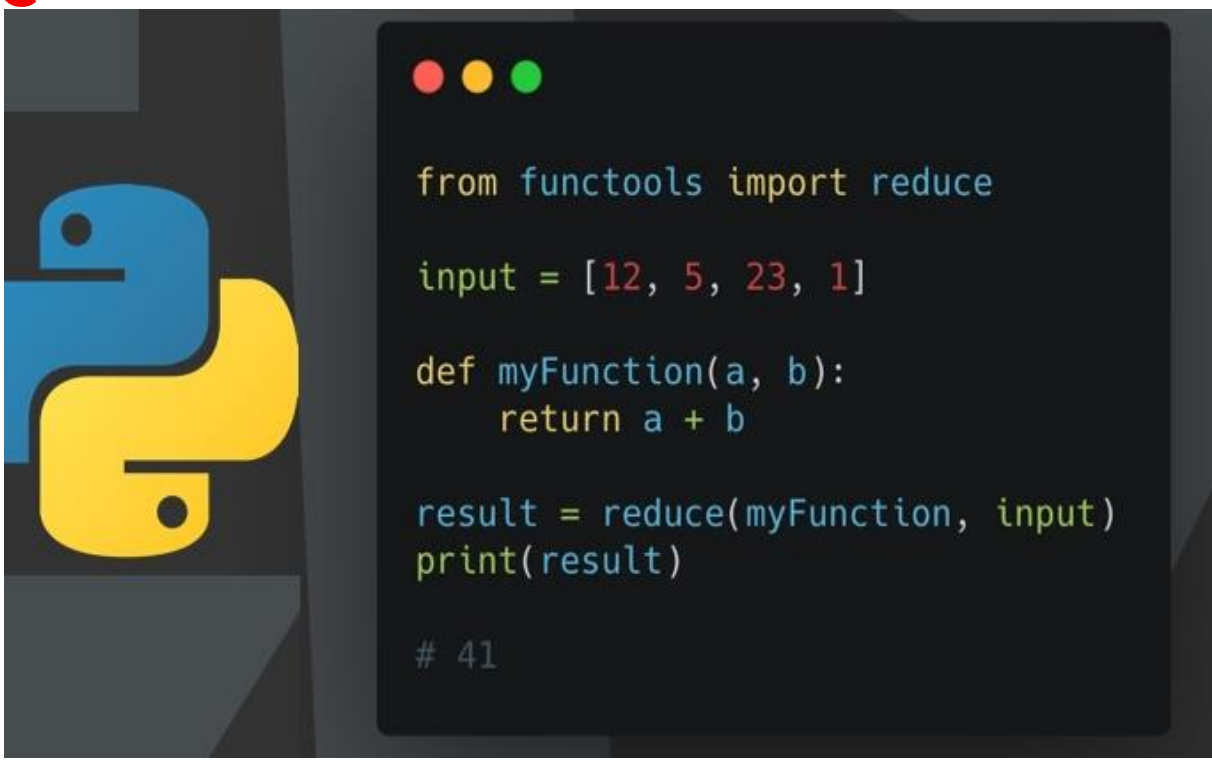
References:

https://www.geeksforgeeks.org/python-exception-handling/
https://www.programiz.com/python-programming/exception-handling
https://python.land/deep-dives/python-try-except

Map and Filter



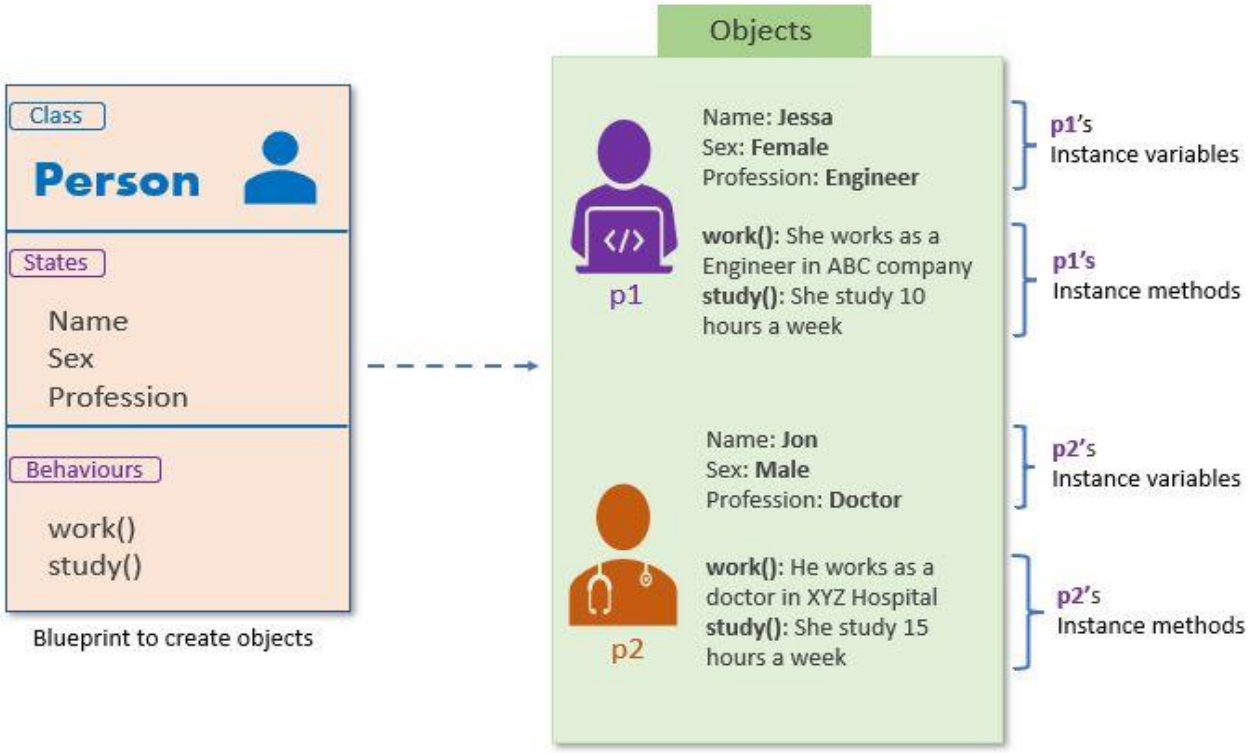
Reduce



References:

https://www.geeksforgeeks.org/map-reduce-and-filter-operations-in-python/
https://stackabuse.com/map-filter-and-reduce-in-python-with-examples/
https://www.learnpython.org/en/Map,_Filter,_Reduce
https://www.askpython.com/python/built-in-methods/map-vs-filter-function-python

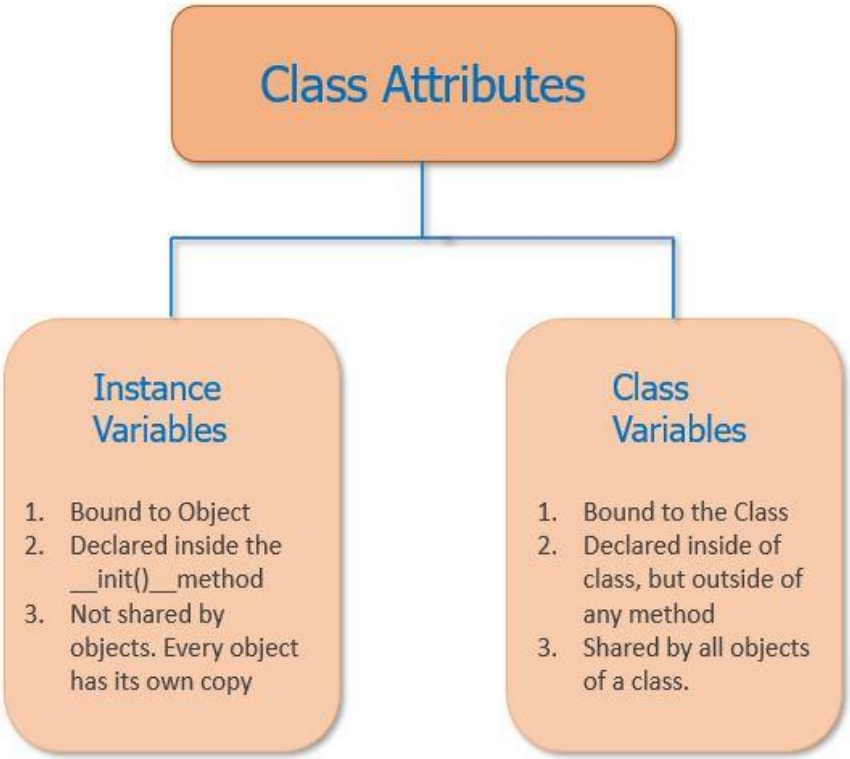
Class and Object



References:

https://www.w3schools.com/python/python_classes.asp
https://www.programiz.com/python-programming/class
https://www.tutorialspoint.com/python/python_classes_objects.htm
https://pynative.com/python-classes-and-objects/

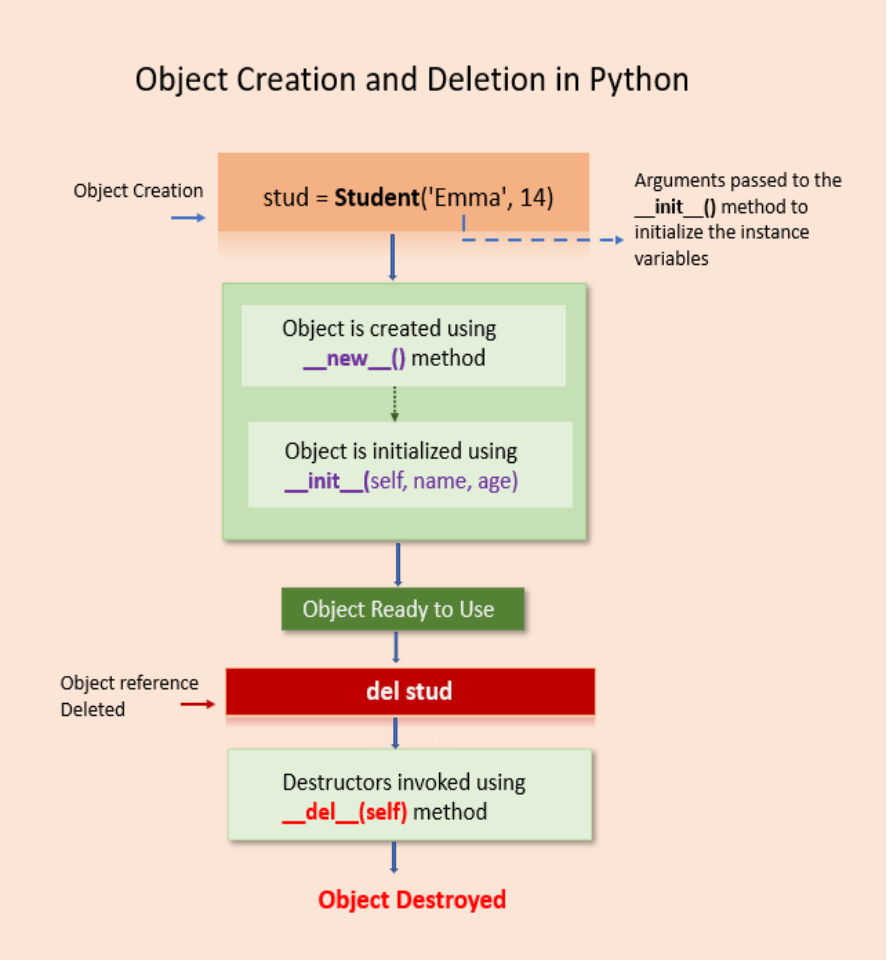
Class and Object



References:

https://www.w3schools.com/python/python_classes.asp
https://www.programiz.com/python-programming/class
https://www.tutorialspoint.com/python/python_classes_objects.htm
https://pynative.com/python-classes-and-objects/

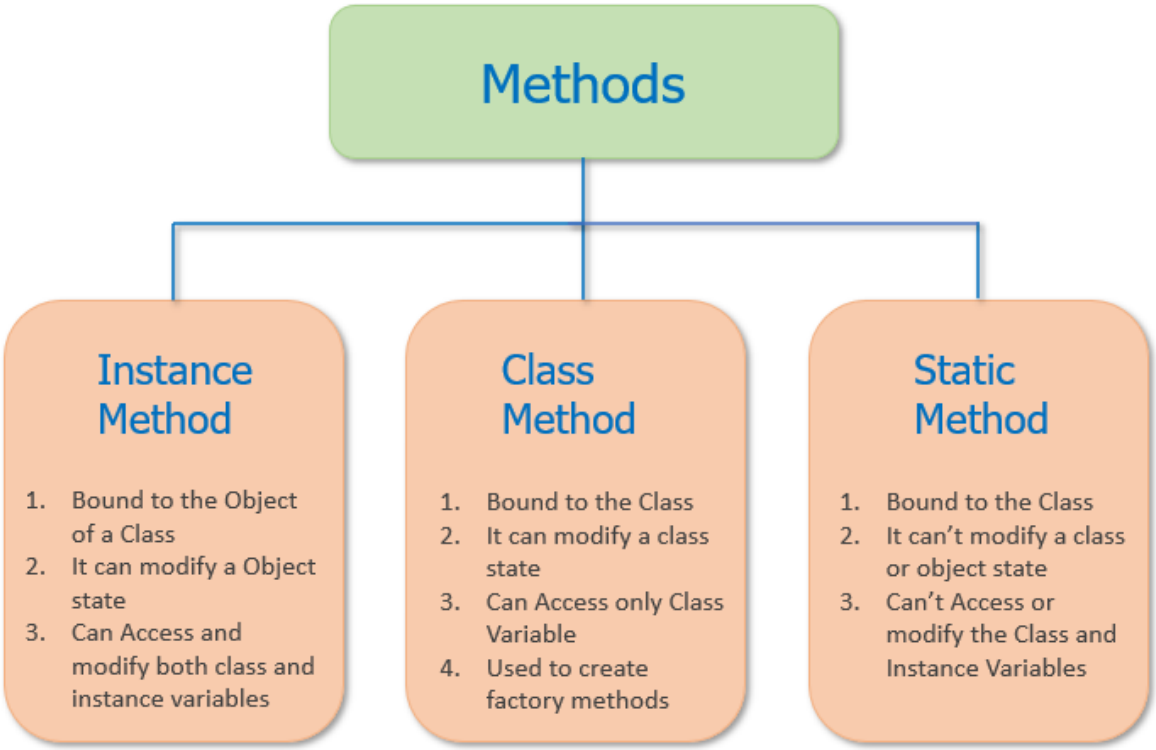
Constructors and Destructors



References:

https://pynative.com/python-destructor/
https://www.analyticsvidhya.com/blog/2024/02/destructor-in-python/
https://codedamn.com/news/python/destructors-in-python
https://medium.com/@abhishekjainindore24/demystifying-constructors-and-destructors-in-python-a-beg
b5bd6988f4bb

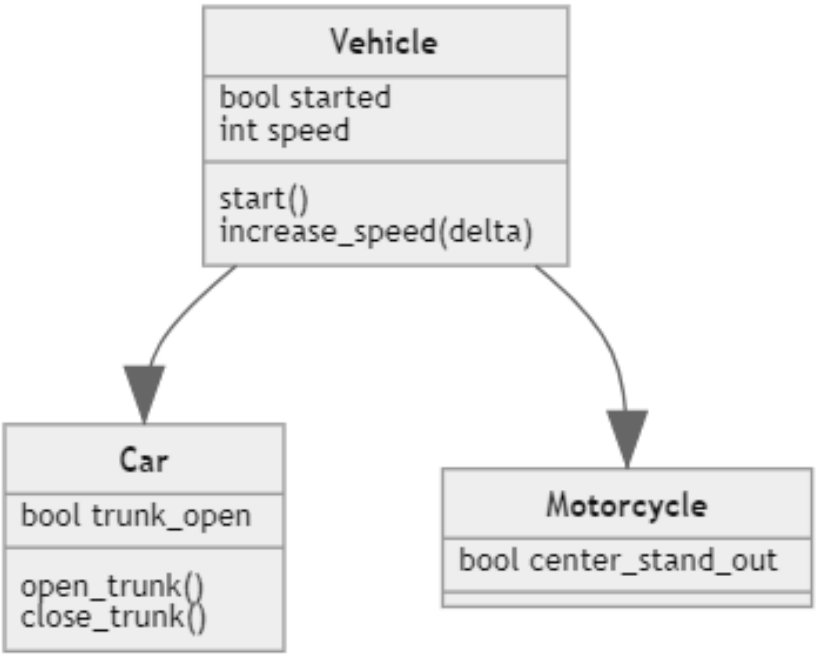
Types of Method



References:

https://pynative.com/python-class-method-vs-static-method-vs-instance-method/
https://www.linkedin.com/pulse/static-method-vs-class-instance-python-3-ryan-parsa-kvgdc/
https://medium.com/codex/python-class-methods-class-vs-instance-vs-static-methods-96d075d27c68
https://realpython.com/instance-class-and-static-methods-demystified/

Inheritance



References:

https://www.programiz.com/python-programming/polymorphism
https://www.toppr.com/guides/python-guide/tutorials/python-oops/polymorphism-in-python-with-examples/
https://www.w3schools.com/python/python_polymorphism.asp
https://www.almabetter.com/bytes/tutorials/python/python-inheritance-and-polymorphism

Types of Inheritance

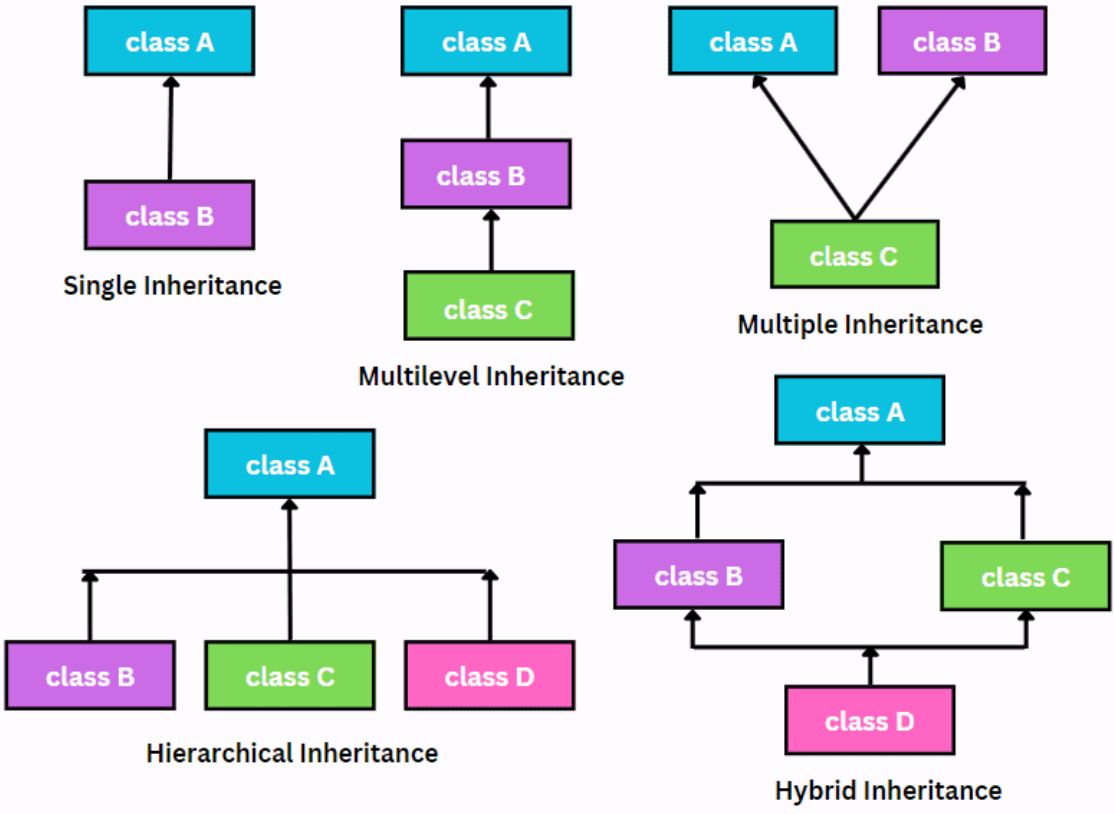


Fig: Types of inheritance in Python

References:

https://www.scientecheasy.com/2023/09/types-of-inheritance-in-python.html/
http://www.btechsmartclass.com/python/Python_Tutorial_Python_Inheritance.html
http://www.btechsmartclass.com/python/Python_Tutorial_Python_Inheritance.html
https://innovationyourself.com/types-of-inheritance-in-python/

Exercises

Access Specifiers: Private, Public, Protected

Access Modifiers	Same Class	Same Package	Sub Class	Other Packages
Public	*Y*	*Y*	*Y*	*Y*
Protected	*Y*	*Y*	*Y*	*N*
Private	*Y*	*N*	*N*	*N*

References:

https://www.scaler.com/topics/access-modifiers-in-python/
https://www.tutorialspoint.com/access-modifiers-in-python-public-private-and-protected
https://www.studytonight.com/python/access-modifier-python
https://www.tutorialsteacher.com/python/public-private-protected-modifiers

Exercises

Polymorphism : Compile Time Polymorphism/Overloading

Compile-Time Polymorphism (Method Overloading)
Method overloading occurs when a class contains many methods with the same name. The types and amount of arguments passed by these overloaded methods vary. Python does not support method overloading or compile-time polymorphism. If there are multiple methods with the same name in a class or Python script, the method specified in the latter one will override the earlier one.

Python does not use function arguments in method signatures, hence method overloading is not supported.

References:

https://www.toppr.com/guides/python-guide/tutorials/python-oops/polymorphism-in-python-with-examples/

Polymorphism : Run Time Polymorphism/Overriding

Like in other programming languages, the child classes in Python also inherit methods and attributes from the parent class. We can redefine certain methods and attributes specifically to fit the child class, which is known as **Method Overriding**.

Polymorphism is supported in Python via method overriding and operator overloading. However, Python does not support method overloading in the classic sense.

References:

https://algodaily.com/lessons/association-aggregation-composition-casting/python
https://faun.pub/association-aggregation-composition-python-ec9947832cbd
https://www.geeksforgeeks.org/python-oops-aggregation-and-composition/

Exercises

Magic Functions/Dunder Functions

Class Instantiation	
`__init__(self, ...args)`	`ClassName()`
`__del__(self)`	`del instance`

Property Lookups	
`__getattr__(self, key)`	`instance.prop` (when `prop` not present)
`__getattribute__(self, key)`	`instance.prop` (regardless of `prop` present)
`__dir__(self)`	`dir(instance)`
`__setattr__(self, key, val)`	`instance.prop = newVal`
`__delattr__(self, key)`	`del instance.prop`
`__getitem__(self, key)`	`instance[prop]`
`__setitem__(self, key, val)`	`instance[prop] = newVal`
`__delitem__(self, key)`	`del instance[prop]`

List Iteration	
`__iter__(self)`	`[x for x in instance]`
`__contains__(self, item)`	`if x in instance`

Operator Overloads	
`__add__(self, other)`	`instance + other`
`__sub__(self, other)`	`instance - other`
`__mul__(self, other)`	`instance * other`
`__eq__(self, other)`	`instance == other`
`__ne__(self, other)`	`instance ≠ other`
`__lt__(self, other)`	`instance < other`
`__gt__(self, other)`	`instance > other`
`__le__(self, other)`	`instance ⩽ other`
`__ge__(self, other)`	`instance ⩾ other`

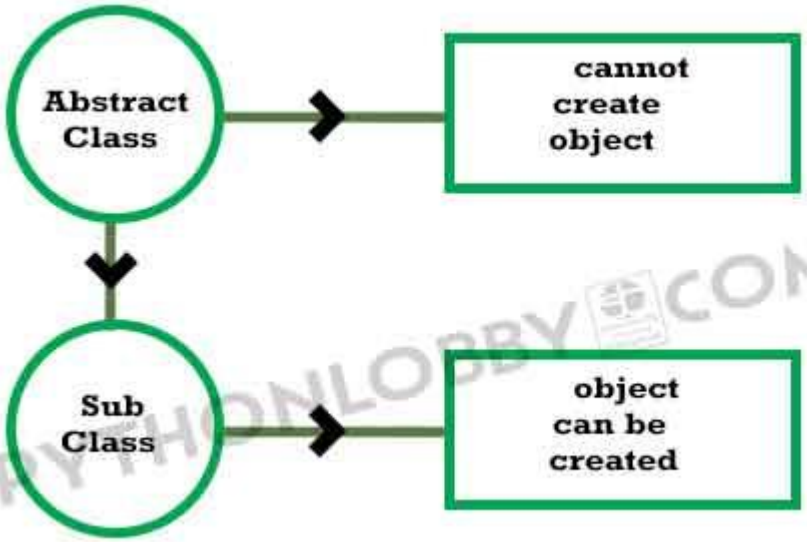
Type Casting	
`__bool__(self)`	`bool(instance)`
`__int__(self)`	`int(instance)`
`__str__(self)`	`str(instance)`

References:

https://realpython.com/python-magic-methods/
https://www.tutorialsteacher.com/python/magic-methods-in-python
https://builtin.com/data-science/dunder-methods-python

Exercises

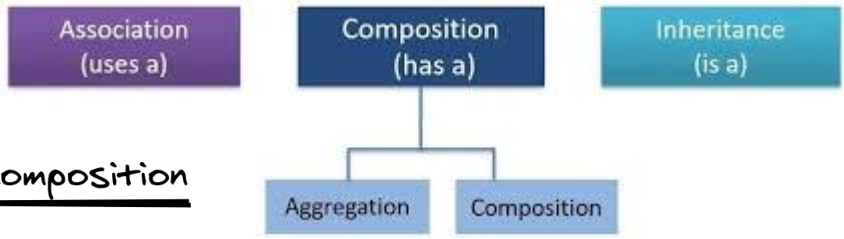
Abstract Method and Class, Empty Class, Data Class

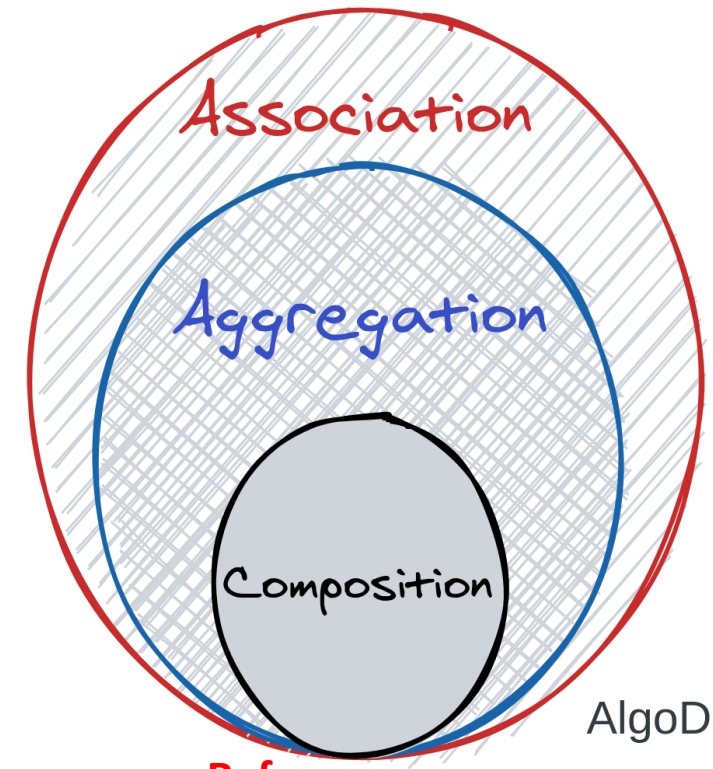


References:

https://www.scaler.com/topics/abstract-class-in-python/
https://pythonlobby.com/abstract-class-in-object-oriented-programming-oops-in-python-programming/#goo
https://www.datacamp.com/tutorial/python-abstract-classes
https://www.datacamp.com/tutorial/python-data-classes
https://realpython.com/python-data-classes/
https://www.dataquest.io/blog/how-to-use-python-data-classes/

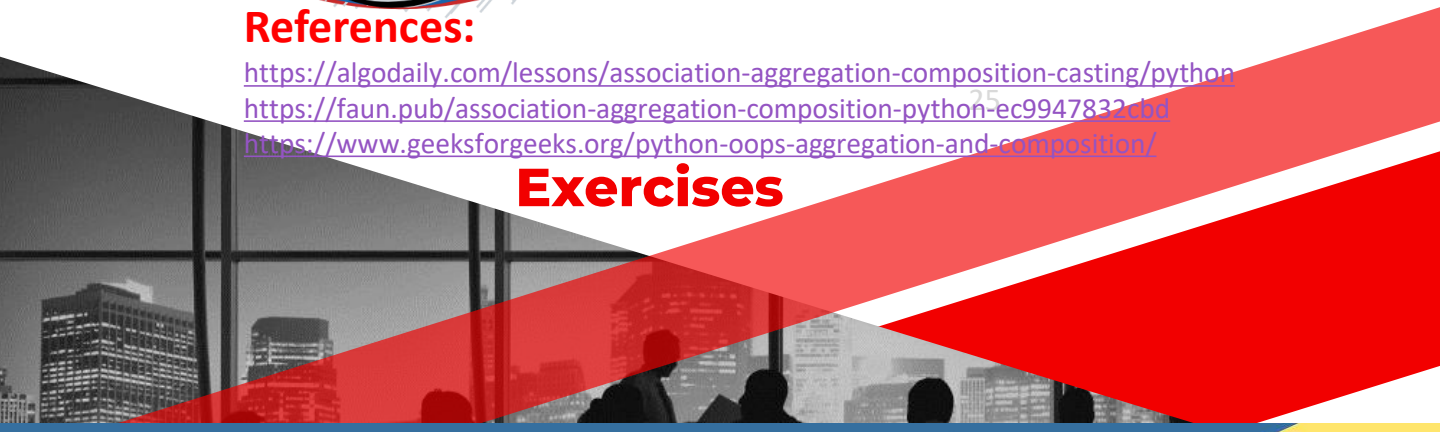
Inner/Nested Class Association, Aggregation, Composition





References:

https://algodaily.com/lessons/association-aggregation-composition-casting/python
https://faun.pub/association-aggregation-composition-python-ec9947832cbd
https://www.geeksforgeeks.org/python-oops-aggregation-and-composition/



Measures of Dispersion

Measures of dispersion are non-negative real numbers that help to gauge the spread of data about a central value.

Quartiles

Quartiles are numbers that separate the data into quarters.

first quartile is at position (n+1)/4, second quartile (i.e. the median) is at position 2(n+1)/4, and the third quartile is at position 3(n+1)/4.

Percentiles

Percentiles provide a way to assess and compare the distribution of values and the position of a specific data point in relation to the entire dataset by indicating the percentage of data points that fall below it.

$$\text{Percentile} = \frac{\text{number of data values below the measurement}}{\text{total number of data values}} \times 100\% = \frac{n}{N} \times 100\%$$

z-score

The z-score is a measure of the position of an entry in a dataset that makes use of the mean and standard deviation of the data.

$$z = \frac{x - \mu}{\sigma}$$

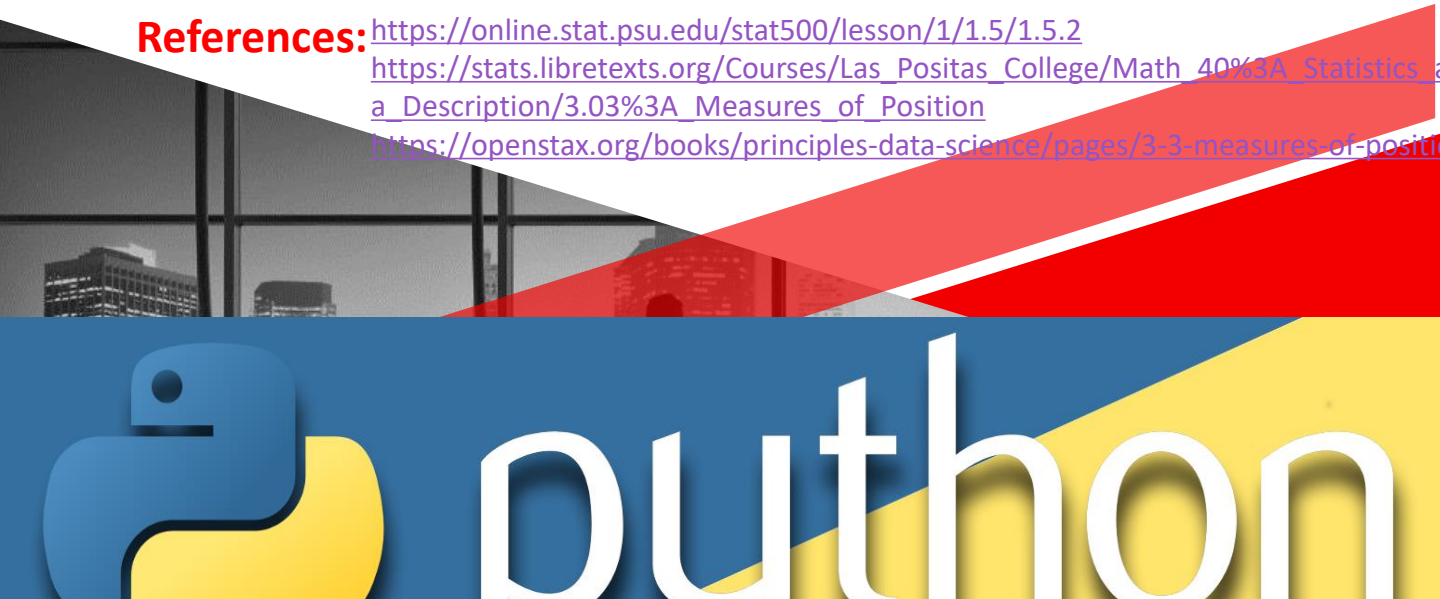
Where:

x is the measurement

μ is the mean

σ is the standard deviation

References: https://online.stat.psu.edu/stat500/lesson/1/1.5/1.5.2
https://stats.libretexts.org/Courses/Las_Positas_College/Math_40%3A_Statistics_a_Description/3.03%3A_Measures_of_Position
https://openstax.org/books/principles-data-science/pages/3-3-measures-of-positi



Thank you - for listening and participating

- ❑**Questions / Queries**
- ❑**Suggestions/Recommendation**
- ❑**Ideas…..?**

Shahzad Sarwar
Cognitive Convergence
https://cognitiveconvergence.com
shahzad@cognitiveconvergence.com
voice: +1 4242530744 (USA) +92-3004762901 (Pak)