# Lecture 2 – Working with files

Fixing the homework:

* Beginning level:

Ex1: spaces matter before and after the function

Math is a built-in library, you need only import the math function

Down-tab to show suggestion of the functions in the library edit box.

## Data type

W3schools.com

Numeric: complex is longer

Ex5:

Ex6: better to assign value to a variable so you can do something with that variable.

Result = int(4.9)

Print(result)

Ex7: int returns automatically value to the closest value.

Ex8:

Print(\*names, sep=’\n’) -> print value in each row

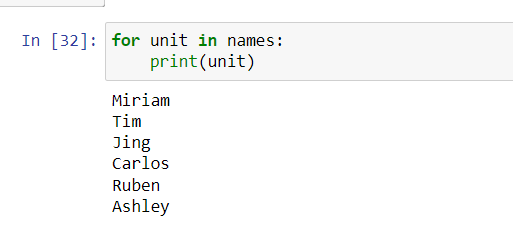
/\* to tell what is inside in the list as separate values

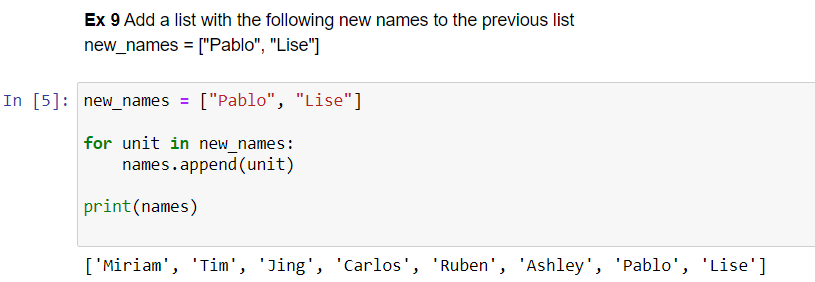
Print(names, sep=’\n’) -> print a list directly

Better not to use the same variable as iterate count in different loops.

For loop doesn’t have condition to end the loop

Print directly the value of variable from the list, doesn’t need to based on the index. Change unit variable to name so it is easier to recognize.

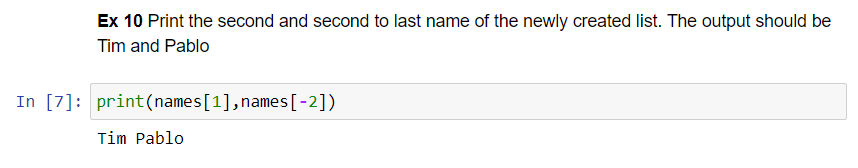




Another way: extend() , names list will take each item from new\_names to put in names’ list

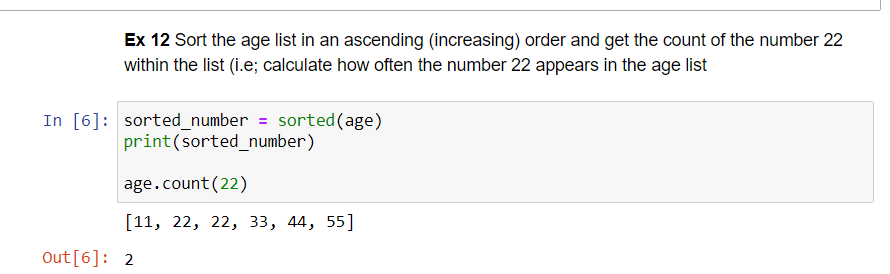
Names.extend(new\_names)

With extend() function, you don’t need a loop



If you print a value outside the list, error -> outside the range.

Mean() function to calculate the average number.



Sort method:

Sorted(age) will return the value to the left

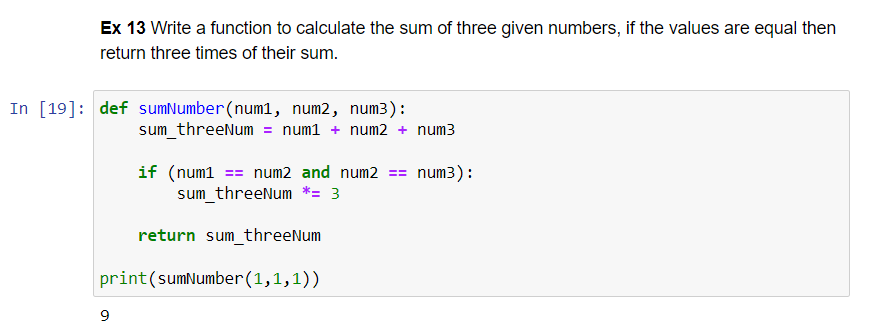
Result = sorted(age)

Age is a list, so there will be a built-in method:

Method is preceeded by a dot in front, function does very specific thing and takes parameter

Sorted(iterable,

Age.sort() -> do the sorting on age



Analyse the question: ‘’Write a function’’ then start googling ‘’write a function’’

After defining a function, you need to call the function to make it run/ activate it.

Step by step:

Step 1: define the name

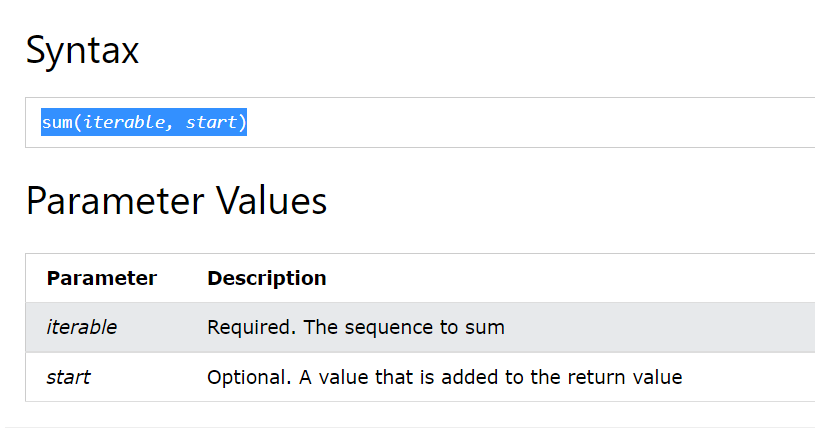
Def sum\_three():

Step 2: put argument

* Convention of the variables: use underscore to separate variable name, f.x. sum\_three\_numbers

Def sum\_three(num\_1,num\_2,num\_3):

Step 3: look for sum function



Def sum\_three(num\_1,num\_2,num\_3):

Sum([num\_1, num\_2, num\_3])

When you use function, the value returns to the right side, there is nothing to the left side, we need to assign a variable to the left side, which is Result\_sum in this case. It doesn’t return value unless you use ‘’**return**’’ to get the value.

3 numbers has to be an iterable -> 3 numbers has to be converted into a list

Def sum\_three(num\_1,num\_2,num\_3):

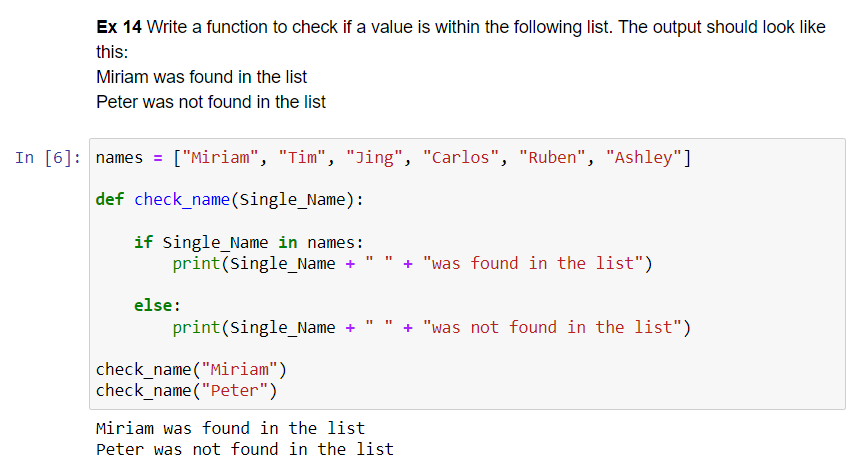
Sum([num\_1, num\_2, num\_3])

#Or return num\_1 + num\_2 + num\_3

Result\_sum = sum\_three(3, 4, 5)

Step 3:

When the code hits **return**, the function will stop and give value with whatever stay to the right side of **return**, doesn’t matter whatelse it happens.



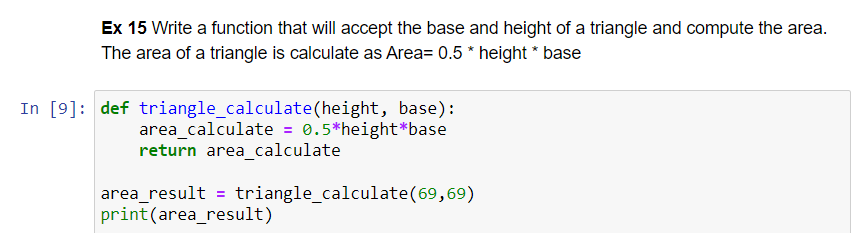
You can also return value and print it at the end.

Function will take a value as an argument.

Convention of an argument: either underscore with all lowercase, or without underscore but have cap in first letter.

Google: check if item is in list python

* Fastest way: argument **in** name of list to check



Step 1: define a function

Def calculate\_triangle\_are():

#we need 2 argument base and height

Def calculate\_triangle\_are(base, height):

Step 2: define a calculation

Def calculate\_triangle\_are(base, height):

0.5\*height\*base

#Would it work? You define a function, but you don’t call it yet.

If you try to print print(calculate\_triangle\_are(10, 10)

* **None** cuz there is no value to return.

#you need to return value to show.

Def calculate\_triangle\_are(base, height):

**Return** 0.5\*height\*base

Print(calculate\_triangle\_are(10, 20))

* 100

Practice: Write a function that takes a name and if the name is longer than 5 letters, it will print name + “, your name is too long” otherwise print “Hello + name + “!”

Step 1: define function

Def name\_analyser(name):

If len(name) > 5:

Print(name + “, your name is too long”)

Else:

Print(“Hello + name + “!”)

name\_analyser(“Anh”)

# Lecture 2