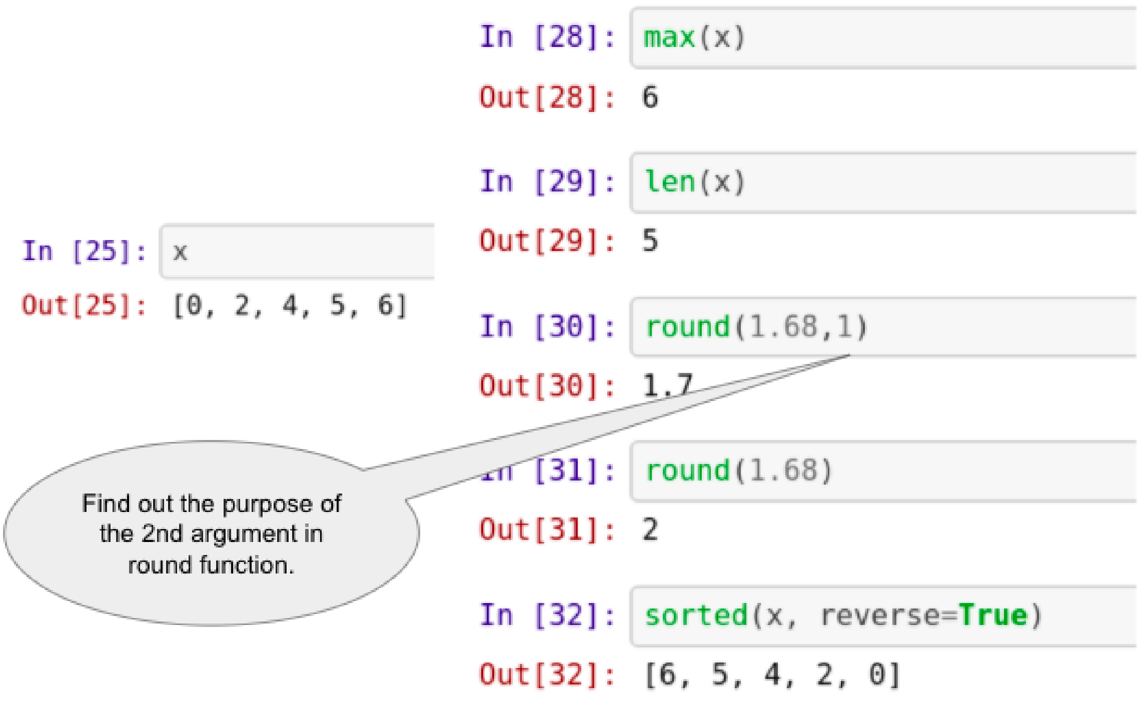
# Functions

- This something we have been indirectly using since the first module.
- print( ) and input( ) are some functions that you've already used. And these are inbuilt functions. `print ("hello world") input ("enter your age")`
- If you want, you can even create your own function to get some desired output.
- Let's learn some in-built functions in the video mentioned below.

**Examples of in-built functions**

**Youtube Link - [https://youtu.be/7rjJrQy9gi4](https://youtu.be/7rjJrQy9gi4)**



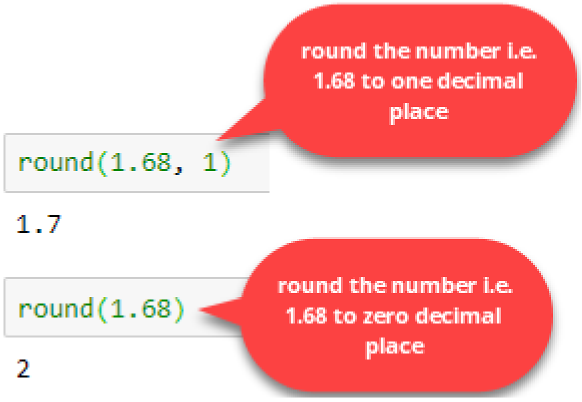
```
In [25]: x
Out[25]: [0, 2, 4, 5, 6]

In [28]: max(x)
Out[28]: 6

In [29]: len(x)
Out[29]: 5

In [30]: round(1.68,1)
Out[30]: 1.7

In [31]: round(1.68)
Out[31]: 2

In [32]: sorted(x, reverse=True)
Out[32]: [6, 5, 4, 2, 0]
```

Find out the purpose of the 2nd argument in round function.

# sum() and round() functions

**round()**

- Round function in python rounds off the decimal value of a number to its nearest integer. For example: round(65.66) = 66
- In addition, one can give an input of the specified number of decimals the number should be rounded off to, For example: round(65.66, 1) = 65.7
- It's syntax is: `round(number, ndigits)`, where; **number** - the number to be rounded, **ndigits** - number up to which the given number is rounded; defaults to 0



```
round(1.68, 1)
```

round the number i.e. 1.68 to one decimal place

```
1.7
```

```
round(1.68)
```

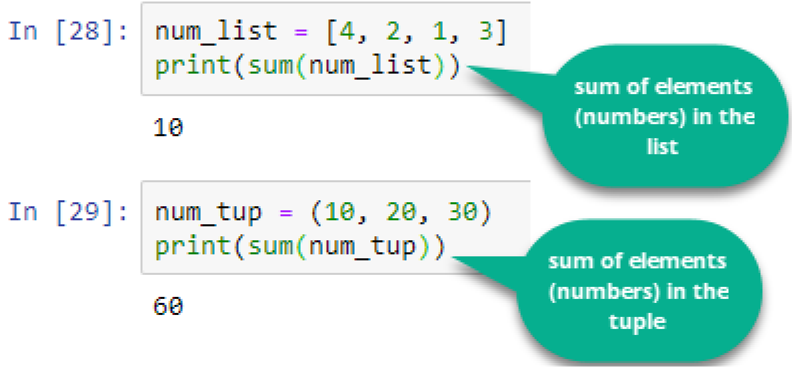
round the number i.e. 1.68 to zero decimal place

```
2
```

round()

**sum()**

You can perform the addition of a collection of numbers using the sum() function in Python.



```
In [28]: num_list = [4, 2, 1, 3]
         print(sum(num_list))

         10
```

sum of elements (numbers) in the list

```
In [29]: num_tup = (10, 20, 30)
         print(sum(num_tup))

         60
```

sum of elements (numbers) in the tuple

sum function

## Define Functions

- In Python a function is defined using the **def keyword.**
- In the next tutorial, we will be learning:
  - How to write a function
  - How to call a function - Of course, if we have created a function, we need to use it somewhere by calling it right? ;)
  - We will also learn how to write a function that would convert bitcoin into USD :P
- Note: The tutor used some other user-friendly local python ide (similar to Colab or jupyter notebook), so don’t press the panic button looking at the new coding interface. You can comfortably run the same code on collab/jupyter notebook (or any python environment) and it will work.

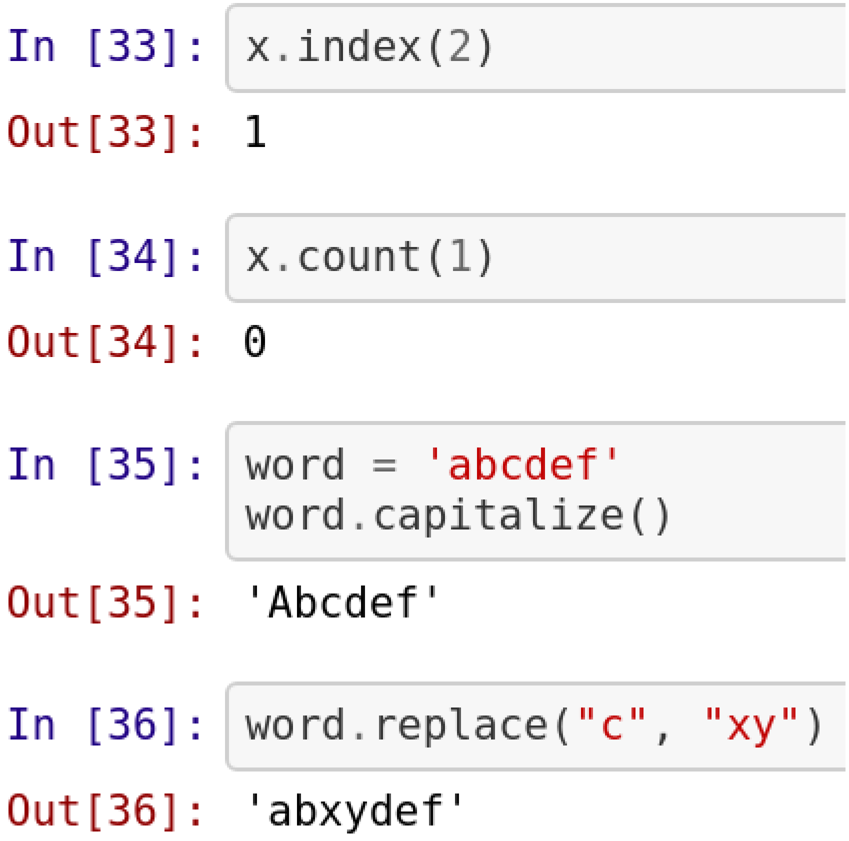
**Youtube Link -** [https://youtu.be/j2xhtI0WTew](https://youtu.be/j2xhtI0WTew)

## Methods

- In Python, everything is an object (be it a list, string etc).

- A method in Python is somewhat similar to function the only difference is that a method is dependent on the object while a function is independent of the object.
- Can you recall we had used a function len() in the earlier modules? Well, this is a function. len() can be used on list, tuple, strings, dictionaries, etc. to get the number of elements in them. While append() is a method that is associated with list only. You cannot use append() method on any of tuple, strings or dictionaries. This append() is associated with list data type only.

- Objects have various methods associated with it (depending on type of the object)



```
In [33]: x.index(2)

Out[33]: 1

In [34]: x.count(1)

Out[34]: 0

In [35]: word = 'abcdef'
         word.capitalize()

Out[35]: 'Abcdef'

In [36]: word.replace("c", "xy")

Out[36]: 'abxydef'
```

## Packages in Python

- In simple terms Python packages are collection of multiple Python files

- And these Python files are known as modules in Python
- If we keep all of our code in the same file, it will result in:
  - Huge code base: Ends up messy
  - Lots of code you won't use

- Maintenance problem

- If we take an example of ourselves, we don't usually store all of our files on our computer in the same location. We use a well-organized hierarchy of directories/folders for easier access.
- Similar files are kept in the same directory, for example, we may keep all the songs in the "music" directory. Analogous to this, Python has packages for directories and modules for files.
- As our application program grows larger in size with a lot of modules, we place similar modules in one package and different modules in different packages. This makes a project (program) easy to manage and conceptually clear.
- Similarly, as a directory can contain subdirectories and files, a Python package can have sub-packages and modules.
- Directory/Folder of Python Scripts.
- Where each script is a module that performs a specific function.
- We can specify functions, methods, types in a script.
- Thousands of packages are available in python.

- For data science, the commonly used packages are:
  - Numpy: Working with arrays
  - Matplotlib: Data Visualisation
  - Scikit-learn: ML

**Importing packages/modules**

- A Python package can have sub-packages in it. Further these sub-packages have some modules (i.e. Python files).

- Each module consists of some Python functions.
- To make use of these functions we need to load the module in our working environment.
- To load any package or module we use the term import followed by module name or the package name.

- In the video, tutor has loaded 'math' module of Python and used functions like sqrt (to calculate square root of a number), pow (to calculate power of a number), etc.

**Youtube Links - https://youtu.be/DdGVBZv46PI**

**https://youtu.be/V27FQ6UBTPY**