In [30]:
```
import math
```

Data Types in Python

The following data types can be used in base python:

- **boolean**
- **integer**
- **float**
- **string**
- **list**
- **None**
- complex
- object
- set
- dictionary

We will only focus on the **bolded** ones

Let's connect these data types to the the variable types we learned from the Variable Types video (https://www.coursera.org/learn/understanding-visualization-data/lecture/iDodZ/variable-types).

Numerical or Quantitative (taking the mean makes sense)

- Discrete
  - Integer (int) #Stored exactly
- Continuous
  - Float (float) #Stored similarly to scientific notation. Allows for decimal places but loses precision.

```
In [31]: type(4)
```

```
Out[31]: int
```

```
In [32]: type(0)
```

```
Out[32]: int
```

```
In [33]: type(-3)
```

```
Out[33]: int
```

In [34]:
```
#try taking the mean
numbers = [2, 3, 4, 5]
print(sum(numbers)/len(numbers))
type(sum(numbers)/len(numbers)) #In Python 3 returns float, but in Python 2 would retur
```

```
3.5
```

Out[34]:
```
float
```

Floats

In [35]:
```
3/5
```

Out[35]:
```
0.6
```

In [36]:
```
6*10**(-1)
```

Out[36]:
```
0.6000000000000001
```

In [37]:
```
type(3/5)
```

Out[37]:
```
float
```

In [38]:
```
type(math.pi)
```

Out[38]:
```
float
```

In [39]:
```
type(4.0)
```

Out[39]:
```
float
```

In [40]:
```
# Try taking the mean
numbers = [math.pi, 3/5, 4.1]
print(sum(numbers)/len(numbers))
type(sum(numbers)/len(numbers))
```

```
2.6138642178632643
```

Out[40]:
```
float
```

Categorical or Qualitative

- Nominal
  - Boolean (bool)
  - String (str)
  - None (NoneType)
- Ordinal
  - Only defined by how you use the data
  - Often important when creating visuals
  - Lists can hold ordinal information because they have indices

Boolean

In [41]:
```
# Boolean
type(True)
```

Out[41]:
```
bool
```

In [42]:
```
# Boolean
if 6 < 5:
    print("Yes!")
```

In [43]:
```
myList = [True, 6<5, 1==3, None is None]
for element in myList:
    print(type(element))
```

```
<class 'bool'>
<class 'bool'>
<class 'bool'>
<class 'bool'>
```

In [44]:
```
print(sum(myList)/len(myList))
type(sum(myList)/len(myList))
```

```
0.5
```

Out[44]:
```
float
```

String

In [45]:
```
type("This sentence makes sense")
```

Out[45]:
```
str
```

In [46]:
```
type("Makes sentense this sense")
```

Out[46]:
```
str
```

In [47]:
```
type("math.pi")
```

Out[47]:
```
str
```

In [48]:
```
strList = ['dog', 'koala', 'goose']
sum(strList)/len(strList)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-48-b0bd059010c7> in <module>()
      1 strList = ['dog', 'koala', 'goose']
----> 2 sum(strList)/len(strList)

TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Nonetype

In [49]:
```
# None
type(None)
```

Out[49]:
```
NoneType
```

In [50]:
```
# None
x = None
type(x)
```

Out[50]:
```
NoneType
```

In [51]:
```
noneList = [None]*5
sum(noneList)/len(noneList)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-51-7df95c7db77e> in <module>()
      1 noneList = [None]*5
----> 2 sum(noneList)/len(noneList)

TypeError: unsupported operand type(s) for +: 'int' and 'NoneType'
```

Lists

A list can hold many types and can also be used to store ordinal information.

In [52]:
```
# List
myList = [1, 1.1, "This is a sentence", None]
for element in myList:
    print(type(element))
```

```
<class 'int'>
<class 'float'>
<class 'str'>
<class 'NoneType'>
```

In [53]:
```
sum(myList)/len(myList)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-53-01620fe6b2d4> in <module>()
----> 1 sum(myList)/len(myList)

TypeError: unsupported operand type(s) for +: 'float' and 'str'
```

In [54]:
```
# List
myList = [1, 2, 3]
for element in myList:
    print(type(element))
sum(myList)/len(myList) # note that this outputs a float
```

```
<class 'int'>
<class 'int'>
<class 'int'>
```

Out[54]:
```
2.0
```

In [55]:
```
myList = ['third', 'first', 'medium', 'small', 'large']
myList[0]
```

Out[55]:
```
'third'
```

In [56]:
```
myList.sort()
myList
```

Out[56]:
```
['first', 'large', 'medium', 'small', 'third']
```

There are more datatypes available when using different libraries such as Pandas and Numpy, which we will introduce to you as we use them.