In [7]:
```
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

Quantitative Variables:

- Numerical,measurable quantities in which aritmetic operations often make sense

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 [2]:
```
#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 return int
```

```
3.5
```

Out[2]:
```
float
```

Floats

In [4]:
```
3/5
```

Out[4]:
```
0.6
```

In [5]:
```
6*10**(-1)
```

Out[5]:
```
0.6000000000000001
```

In [6]:
```
type(3/5)
```

Out[6]:
```
float
```

In [8]:
```
type(math.pi)
```

Out[8]:
```
float
```

In [9]:
```
math.pi
```

Out[9]:
```
3.141592653589793
```

In [10]:
```
type(4.0)
```

Out[10]:
```
float
```

In [11]:
```
# Try taking the mean
numbers = [math.pi, 3/5, 4.1]
type(sum(numbers)/len(numbers))
```

Out[11]:
```
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

Nominal Data

Nominal scal classifies data into distinct categories in which no ranking is implied.

Example:

- Gender, Marital status

0 represent male

1 represent female

This 0,1 don't have any effect. These are marked for convenience.

Boolean

In [13]:
```
# Boolean
type(True)
```

Out[13]:
```
bool
```

In [17]:
```
# Boolean
if 6 < 5:
    print("Yes!")
```

In [16]:
```
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 [22]:
```
myList = [True, 6<5, 1==3, None is None]
for element in myList:
    if element == True:
        print(element,"1")
    else:
        print(element,"0")
```

```
True 1
False 0
False 0
True 1
```

In [19]:
```
sum(myList)
```

Out[19]:
```
2
```

In [23]:
```
len(myList)
```

Out[23]:
```
4
```

In [18]:
```
print(sum(myList)/len(myList))
type(sum(myList)/len(myList))
```

```
0.5
```

Out[18]:
```
float
```

String

In [24]:
```
type("This sentence makes sense")
```

Out[24]:
```
str
```

In [25]:
```
type("Makes sentense this sense")
```

Out[25]:
```
str
```

In [26]:
```
type("math.pi")
```

Out[26]:
```
str
```

Nonetype

In [27]:
```
# None
type(None)
```

Out[27]:
```
NoneType
```

In [28]:
```
# None
x = None
type(x)
```

Out[28]:
```
NoneType
```

In [31]:
```
noneList = [None]*5
sum(noneList)/len(noneList)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-31-7df95c7db77e> 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 [35]:
```
# 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 [36]:
```
sum(myList)/len(myList)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-36-01620fe6b2d4> in <module>
----> 1 sum(myList)/len(myList)

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

In [37]:
```
# 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[37]:
```
2.0
```

In [38]:
```
myList = ['third', 'first', 'medium', 'small', 'large']
myList[0]
```

Out[38]:
```
'third'
```

In [39]:
```
myList.sort()
myList
```

Out[39]:
```
['first', 'large', 'medium', 'small', 'third']
```

In [ ]:
```

```