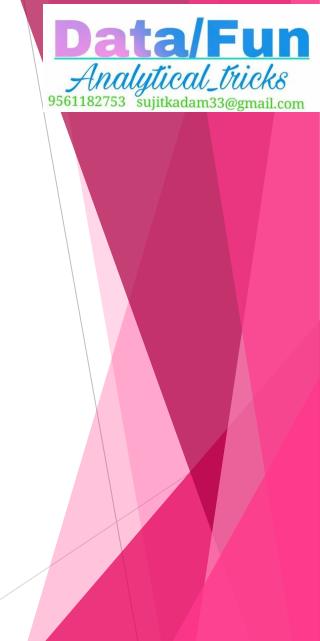
#### **Python Basics**

Lists, Functions, Packages, Dictionaries



# Data Types

• float : real numbers

• int : interger numbers

• str : text

• bool : True, False





### **Arithmetic Operations**

- Python console can be used like a calculator (just like R)
- Apart from routine arithmetic calculations, Python allows us to perform some operations on strings also using some arithmetic operators

```
a = "Data"
b = "Science"
print(a+b)

DataScience

a = "Data"
print(a * 3)

DataDataData

print('Data' 'Science')

DataScience
```

# Lists

Creating & Managing

# Data/Fun Analytical\_trick8 9561182753 sujitkadam33@gmail.com

#### List

- Convenient for storing many values
- Stores values with different data types in a single object



# Indexing of Lists

- Lists in Python have zero based indexing. i.e. index 0 corresponds to 1<sup>st</sup> element, index 1 corresponds to 2<sup>nd</sup> element etc.
- Similarly, reversely indexing is negative.

```
In [5]: custList[3]
Out[5]: 32
In [6]: custList[-7]
Out[6]: 32
```



## List Slicing

• For sequentially sub-setting the list we need to specify the indices as [start : end], where start index is inclusive and end index is exclusive

```
        Index
        Index
        Index

        0
        1
        2
        3
        4

        | Element 0 | Element 1 | Element 2 | Element 3 |
        |
        |
        |
        |
        |

        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
        |
```

```
In [7]: custList[3:7]
Out[7]: [32, 'Rohit', 41, 'Janhavi']
In [8]: custList[-5:-2]
Out[8]: [41, 'Janhavi', 42]
In [9]: custList[:6]
Out[9]: ['Suvarna', 42, 'Amit', 32, 'Rohit', 41]
In [10]: custList[6:]
Out[10]: ['Janhavi', 42, 'Deepa', 49]
```





Increments can be specified by writing third index

```
In [3]: custList = ["Suvarna",42,"Amit",32,"Rohit",41,"Janhavi",42,"Deepa",49]
In [4]: custList[0:5:2]
Out[4]: ['Suvarna', 'Amit', 'Rohit']
In [5]: custList[::2]
Out[5]: ['Suvarna', 'Amit', 'Rohit', 'Janhavi', 'Deepa']
```



#### Changing the list elements

- The list elements can be changed by specifying the corresponding indices
- The elements can be added with "+" operator or by calling append method and can be removed using del()

```
In [13]: custList=custList+["Girija",13]
In [24]: del(custList[6:8])
    ...: custList
Out[24]: ['Suvarna', 42, 'Amit', 32, 'Rohit', 41, 'Deepa', 49]
In [25]: custList.append(["Deepika",32])
    ...: print(custList)
['Suvarna', 42, 'Amit', 32, 'Rohit', 41, 'Deepa', 49, ['Deepika', 32]]
```

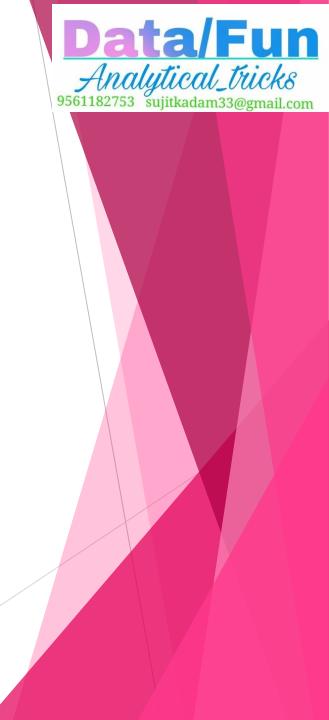




The lists are stored by reference

```
In [17]: Customers=custList
In [18]: Customers[1]=36
In [19]: Customers
Out[19]: ['Sumedha', 36, 'Amit', 34, 'Rohit', 41, 'Deepa', 49, 'Girija', 13]
In [20]: Customers.index("Rohit")
Out[20]: 4
In [21]: custList
Out[21]: ['Sumedha', 36, 'Amit', 34, 'Rohit', 41, 'Deepa', 49, 'Girija', 13]
```

#### **Packages**



# Why Packages?



- Many Functions
- Lots of diversity
- Maintaining the compartmental divisions in the code can be done with packages
- Package examples
  - Numpy
  - Matplotlib
  - Scikit-learn

# **Using Packages**

- For using any package, you need to
  - Install that package
  - Import that package
- Installing Package
  - Go to <a href="http://pip.readthedocs.org/en/stable/installing/">http://pip.readthedocs.org/en/stable/installing/</a>
  - Download get-pip.py
  - Or at Terminal type:
  - python3 get-pip.py
  - pip3 install numpy (for package numpy)







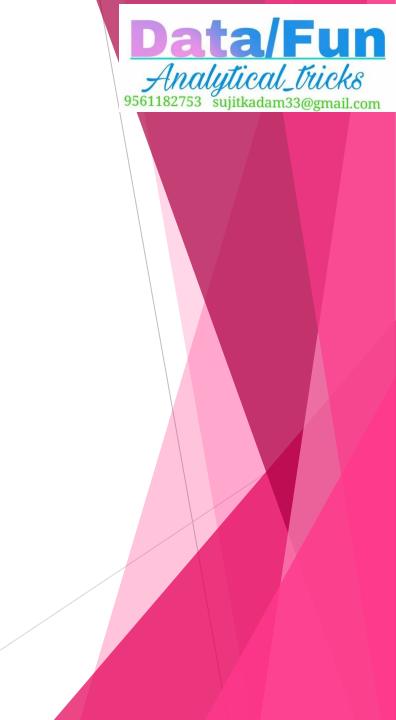
We can import package with keyword import as

Abbreviation / alias for a package name can also be used

• Or just a function can be imported from a package as

```
In [70]: from numpy import absolute
    ...: absolute(-8)
Out[70]: 8
```

Package numpy







Arithmetic calculations with lists cannot be performed

```
In [3]: length = [23,34.5,6.7,90.4,45.3]
In [4]: breadth = [21,23,45,65,12.3]
In [5]: area = length * breadth
Traceback (most recent call last):
   File "<ipython-input-5-ea519f670ae6>", line 1, in <module> area = length * breadth

TypeError: can't multiply sequence by non-int of type 'list'
In [6]: length + breadth
Out[6]: [23, 34.5, 6.7, 90.4, 45.3, 21, 23, 45, 65, 12.3]
```



#### Package numpy

- Package numpy allows us to perform mathematical calculations on lists
- We need to import numpy and then create a numpy array out of the list

```
In [7]: import numpy
                                                                In [13]: import numpy as np
In [8]: lg = numpy.array(length)
                                                                In [14]: lg = np.array(length)
In [9]: bd = numpy.array(breadth)
                                                                In [15]: bd = np.array(breadth)
In [10]: area = lg * bd
                                                                In [16]: area = lg * bd
In [11]: area
                                                                In [17]: area
                                                                Out[17]: array([ 483. , 793.5 , 301.5 , 5876. , 557.19])
Out[11]: array([ 483. , 793.5 , 301.5 , 5876. , 557.19])
                                                                In [18]: lg + bd
In [12]: lg + bd
                                                                Out[18]: array([ 44. , 57.5, 51.7, 155.4, 57.6])
Out[12]: array([ 44. , 57.5, 51.7, 155.4, 57.6])
```





```
In [28]: lg
Out[28]: array([ 23. , 34.5, 6.7, 90.4, 45.3])
In [29]: lg>30
Out[29]: array([False, True, False, True, True], dtype=bool)
In [30]: lg[lg>30]
Out[30]: array([ 34.5, 90.4, 45.3])
```



### 2D Arrays

```
In [37]: al = np.array([[12,23,29,34],[89,92,82,56]])
In [38]: al
Out[38]:
array([[12, 23, 29, 34],
       [89, 92, 82, 56]])
In [39]: al.shape
Out[39]: (2, 4)
In [40]: type(al)
Out[40]: numpy.ndarray
In [411]: a1[0]
Out[41]: array([12, 23, 29, 34])
In [42]: a1[0][2]
Out[42]: 29
In [43]: a1[0,2]
Out[43]: 29
In [44]: al[:,1:3]
Out[44]:
array([[23, 29],
       [92, 82]])
```



# Invoking Functions and Calling

#### **Attributes**

• Functions and attributes can be called using "" period operator on the object of the respective class e.g. object of class numpy has functions mean(), median(), sort()

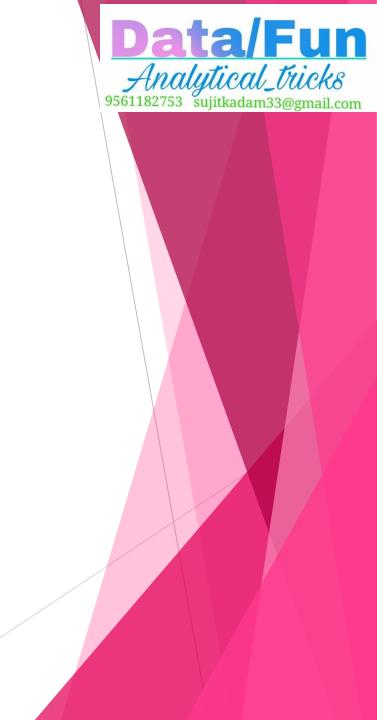
```
In [59]: lq
Out[59]: array([ 23. , 34.5, 6.7, 90.4, 45.3])
In [60]: type(np)
Out[60]: module
In [61]: np.mean(lq)
Out[61]: 39.9800000000000004
In [62]: np.median(lg)
Out[62]: 34.5
In [63]: np.sort(lq)
Out[63]: array([ 6.7, 23., 34.5, 45.3, 90.4])
In [63]:
In [64]: lq
Out[64]: array([ 23. , 34.5, 6.7, 90.4, 45.3])
In [65]: type(np)
Out[65]: module
In [66]: lg.mean()
Out[66]: 39.9800000000000004
In [67]: lq.size
Out[67]: 5
In [68]: lg.sort()
In [69]: lq
Out[69]: array([ 6.7, 23., 34.5, 45.3, 90.4])
```



## Importing files with numpy

- Function numpy.loadtxt() can be used to load the data into 2D numpy array
- Only numbers allowed

#### **Dictionaries**





#### Indexed Values

- Lists are indexed values
- If want to retrieve data from list by index, it can be done calling index() on list object
- Suppose that you have two different lists countries and pop

```
countries=["India","China","US","Indonesia"]
pop=[1339180127,1409517397,324459463,263991379]
#### Calling by Index
ch_ind=countries.index("China")
pop_ch=pop[ch_ind]
pop_ch
```



#### **Dictionaries**

- Key-Value pairs can be easily stored and retrieved in Python using dictionary
- You need to specify the key-value pairs in curly brackets and separate each key with value by ":"



### Adding and Removing

You can add element in the dictionary by just specifying it in the assignment

```
In [15]: populations['Brazil']=209288278
In [16]: populations
Out[16]:
{'Brazil': 209288278,
 'China': 1409517397,
 'India': 1339180127,
 'Indonesia': 263991379,
 'US': 324459463}
```

You can remove element from the dictionary calling the function del()

```
In [17]: del(populations['Brazil'])
In [18]: populations
Out[18]:
{'China': 1409517397,
  'India': 1339180127,
  'Indonesia': 263991379,
  'US': 324459463}
```



#### Dictionary of Dictionaries

Dictionary can be nested inside a dictionary

```
In [26]: demog = { 'India': { 'capital':'Delhi', 'population':1339180127 },
                      'China': { 'capital': 'Beijing', 'population':1409517397 }, 'US': { 'capital': 'Washington', 'population':324459463 },
                      'Indonesia': { 'capital':'Jakarta', 'population':263991379 } }
In [27]: demog['US']['capital']
Out[27]: 'Washington'
In [28]: data={
              'capital':'Brasilia',
              'population':209288278
In [29]: demog['Brazil']=data
    ...: demog
Out[29]:
{'Brazil': {'capital': 'Brasilia', 'population': 209288278},
 'China': {'capital': 'Beijing', 'population': 1409517397},
 'India': {'capital': 'Delhi', 'population': 1339180127},
 'Indonesia': {'capital': 'Jakarta', 'population': 263991379},
 'US': {'capital': 'Washington', 'population': 324459463}}
```

# Difference Between Lists and Dictionaries



#### List

- Stored values with index of numbers
- You can use list when order of values matter

#### **Dictionaries**

- Stored values with keys, no necessarily numbers
- You can use dictionaries when you want to retrieve values by a "look-up" table logic





Thank You .....!