

google colaboratory

Week 1:

- Introduction to Python programming language
 - overview of data analysis
- Anaconda: In this lesson, you will be getting a quick glimpse at the Anaconda environment - one of the most popular environments for doing data analysis in Python.
- Jupyter Notebooks: Jupyter Notebooks are a great tool for getting started with writing python code. Though in production you often will write code in scripts, notebooks are wonderful for sharing insights and data viz! The Data Analysis Process Learn about the data analysis process and practice investigating different datasets using Python and its powerful packages for data analysis.
- Variables, data types, and basic operations
- Working with strings and string manipulation

In []:

```
1
```

Variables and Datatypes

In []:

```
1 print("hello world!")
```

In [1]:

```
1 number = 2
2 type(number)
3
4 name = 'caleb'
5 print(name)
```

caleb

In [2]:

```
1 print(type(number))
```

<class 'int'>

In []:

```
1 first_number = int(input("enter the value: "))
2 second_number = int(input("enter the second value: "))
```

In []:

```
1 first_number, second_number
```

In []:

```
1 first_number * second_number
```

In []:

```
1 result = first_number/second_number
2 print("the result is:", result)
```

In []:

```
1 type(number)num = int(input("enter the number : "))
```

In [4]:

```
1 print("W \bord")
```

Word

2, 2.0, wee

In []:

```
1 x =2.0
2 type(x)
```

Python Escape Character Sequences

- ' ' Single Quote
- \ Backslash
- \n New Line
- \t Tab
- \b Backspace

Python Operators

- Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- Identity operators
- Membership operators
- Bitwise operators

https://www.w3schools.com/python/python_operators.asp

[. \(https://www.w3schools.com/python/python_operators.asp\)](https://www.w3schools.com/python/python_operators.asp)

== != > >= < <= is is not in not in : membership operators

In []:

```
1
```

division of integer

In []:

```
1 ##### first_number=8
2 second_number=int(input('the dividend is'))
3 result=first_number/second_number
4 print('the quotient:',result)
```

addition of floats

In []:

```
1 first_number=45.8
2 second_number=int(input('the missing number is'))
3 result=first_number+second_number
4 print('the sum is:',result)
5
```

subtraction of float from integer

In []:

```
1 first_number=56
2 second_number=float ( input('the second number is'))
3 result=first_number-second_number
4 print('the difference is:',result)
5
```

area of a cone($\pi r^2 \times \text{slant height}$)

In []:

```
1 pi=3.14
2 radius=int(input('the radius is'))
3 slant_height=5.6
4 result=pi*radius*slant_height
5 print('the area of cone is:',result)
6
```

area of a rectangle

In []:

```
1 pi=3.14
2 radius=3**3
3 slant_height=int(input ('the height is'))
4 result=pi*radius*slant_height
5 print('the area of cone is:',result)
6
```

In []:

```
1 pi=3.14
2 radius=3**3
3 slant_height= float(input('the height is'))
4 result=pi*radius*slant_height
5 print('the area of cone is:',result)
6
```

list

In [14]:

```
1 my_list.insert(1,5)
```

In [15]:

```
1 print(my_list)
```

```
[1, 5, 2, 3, 4]
```

In []:

```
1 #creating a list
2 list_1 =['b','r','i','g','h','t']
```

In []:

```
1 text = 'emmanuel'
2 l2 = list(text)
3 print(l2)
4 #converting string (ie numbers )to integers
5 num_text = '12345'
6 num_list = [int(x) for x in list(num_text)]
7 print(num_list)
```

In []:

```
1 a= 'bright'
2 list_2 =[a]
3 #printing out the values of your
4 print(list_1)
```

list operations

- indexing,slicing operation:a way to refer the individual terms within an iterable by its position
- insert(): add items to a list in a specific position
- remove():removes the first occurrence of the element with the specified value
- append(); it is a pre-defined method used to add a single item to certain collection types
- sort():arranging data in a particular order
- index():refer to a position within an ordered list
- min(), max()
- pop()
- clear()

In [28]:

```
1 score = [2,3,5,7,1,7,9]
2 min(score)
```

Out[28]:

1

In [29]:

```
1 max(score)
```

Out[29]:

9

In [16]:

```
1 my_list = [1,2,3,4]
2 print(my_list)
```

[1, 2, 3, 4]

In [18]:

```
1 my_list.append(6)
2 print(my_list)
```

[1, 2, 3, 4, 6]

In [22]:

```
1 #remove
2 my_list.remove(2)
```

In [23]:

```
1 print(my_list)
```

[1, 3, 4, 6]

In [25]:

```
1 #Creating a list
2 list_2 = [1,4,2,5,7,4,0,9,8]
3 list_2.sort()
```

In [26]:

```
1 print(list_2)
```

[0, 1, 2, 4, 4, 5, 7, 8, 9]

In [30]:

```
1 min_score = min(list_2)
2 print(min_score)
```

0

In [31]:

```
1 max_score = max(list_2)
2 print(max_score)
```

9

In [20]:

```
1 #q2a##len is used in getting the exact length of the above given list
2 len(list_2)
```

Out[20]:

9

In []:

```
1 #####list indexing/slicing operations
```

In []:

```
1 #1st element  
2 list_2[0]
```

In []:

```
1 list_2[1]
```

In []:

```
1 list_2[-1]
```

In [35]:

```
1 list_2
```

Out[35]:

```
[0, 1, 2, 4, 5, 7, 8, 9]
```

In [40]:

```
1 #slicing operations  
2 list_2[2:5]
```

Out[40]:

```
[2, 4, 5]
```

In []:

```
1 list_2[2:4]
```

In []:

```
1 list_2[2:5]
```

insert is used to add an element to the list

In []:

```
1 #in this case insert is used to add 2 and 6  
2 list_2.insert(-1,12)  
3 print(list_2)
```

In []:

```
1 str_list = ["mango", 'banana', 'orange']
```

In []:

```
1 #remove is used to abolish 4 from the list  
2 list_2.remove(4)
```

In []:

```
1 #this is calling list 2 to find out if the recent activity has been performed  
2 list_2
```

In []:

```
1 #append is used to add a single element to a list  
2 list_2.append(25)
```

In []:

```
1 list_2
```

In []:

```
1 #sort is used to arrange data in a particular order  
2 list_2.sort()
```

In []:

```
1 list_2
```

In []:

```
1 #it is used to refer to a position in a list  
2 list_2.index(4)
```

In []:

```
1 #it is used to identify the lowest value  
2 min(list_2)
```

In []:

```
1 #it is used to identify the highest value  
2 max(list_2)
```

In [32]:

```
1 #it removes an item from a specific index  
2  
3 list_2.pop(3)
```

Out[32]:

4

In []:

```
1 list_2
```

In []:

```
1 #it is used to remove all items on a list
2 list_2.clear()
```

In []:

```
1
2 list_2
```

In []:

```
1 #creating a new list
2 list_3=[1,2,3,4,5,6,7,8]
3 list_4 = ['2','4']
```

In []:

```
1 #extend can be used to join two lists together
2 list_3.extend(list_4)
```

In []:

```
1 #printing the outcome
2 print('list after extend():',list_3)
```

In []:

```
1 l_4 = [1,2,3,4,5,6,7,8]
```

In []:

```
1 l_4[:3]
```

TUPLE

Tuple is a collection which is ordered and unchangeable. it is used to store multiple items in a single variable.

In [42]:

```
1 num = '2'
2 int(num)*5
```

Out[42]:

10

In []:

```
1 score = (1,2,3)
2
```

In [46]:

```
1 #creating a tuple
2 tuple_1=('emem','lionel','blessing')
3 print(tuple_1)
4 print(type(tuple_1))
```

```
('emem', 'lionel', 'blessing')
<class 'tuple'>
```

In [47]:

```
1 #switching a tuple to a list
2 list_8=list(tuple_1)
3 print(list_8)
4 print(type(list_8))
```

```
['emem', 'lionel', 'blessing']
<class 'list'>
```

In []:

```
1 type(list_8)
```

In []:

```
1 #switching a list to a tuple
2 tuple_1=tuple(list_8)
3 print(tuple_1)
```

In []:

```
1
```

In []:

```
1 ####Dealing with set2
2 fruits_1={'orange','mango','watermelon',"salt",'pear','avocado'}
3 fruits_2={'strawberry','blueberry','pear','mango','raspberries', "salt"}
4 fruits_2.discard("pear")
5 print(fruits_2)
```

In []:

```
1 #Using the update set operation
2 # fruits_2=list(fruits_2)
3 # print(fruits_2)
4 fruits_1.update(("fruits_2", "grains",))
5 print(fruits_1)
```

In []:

```
1 #Using the remove set operation
2 fruits_1.remove("mango")
```

In []:

```
1 #calling fruits_1
2 fruits_1
```

In []:

```
1 #using the union set operation
2 fruits_1.union(fruits_2)
```

In []:

```
1 #Using the union sign to write a code
2 fruits_1|fruits_2
```

In []:

```
1 #using the intersection set operations
2 fruits_1.intersection(fruits_2)
```

In []:

```
1 #using the intersection sign to write a code
2 fruits_1 & fruits_2
```

In []:

```
1 #using the difference sign to write the code
2 print(fruits_1, '\n\n', fruits_2)
3 fruits_1 - fruits_2
```

In []:

```
1 #Using the difference set operation
2 fruits_1.difference(fruits_2)
```

In []:

```
1 #USING THE SYMMETRIC DIFFERENCE  
2 fruits_1 ^ fruits_2
```

In []:

```
1 fruits_1.symmetric_difference(fruits_2)
```

In []:

```
1 a={2,3}  
2 b={9,8,7,0,2,3,4}
```

In []:

```
1 a.issubset(b)
```

In []:

```
1 b.issuperset(a)
```

In []:

```
1 a.isdisjoint(b)
```