Minia University
Faculty of Computers & information

Artificial Neural Networks and Deep Learning

Slides By:

T.A. Sarah Osama Talaat

E-mail: SarahOsama.fci@gmail.com

Slides were prepared based on set of references mentioned in the last slide

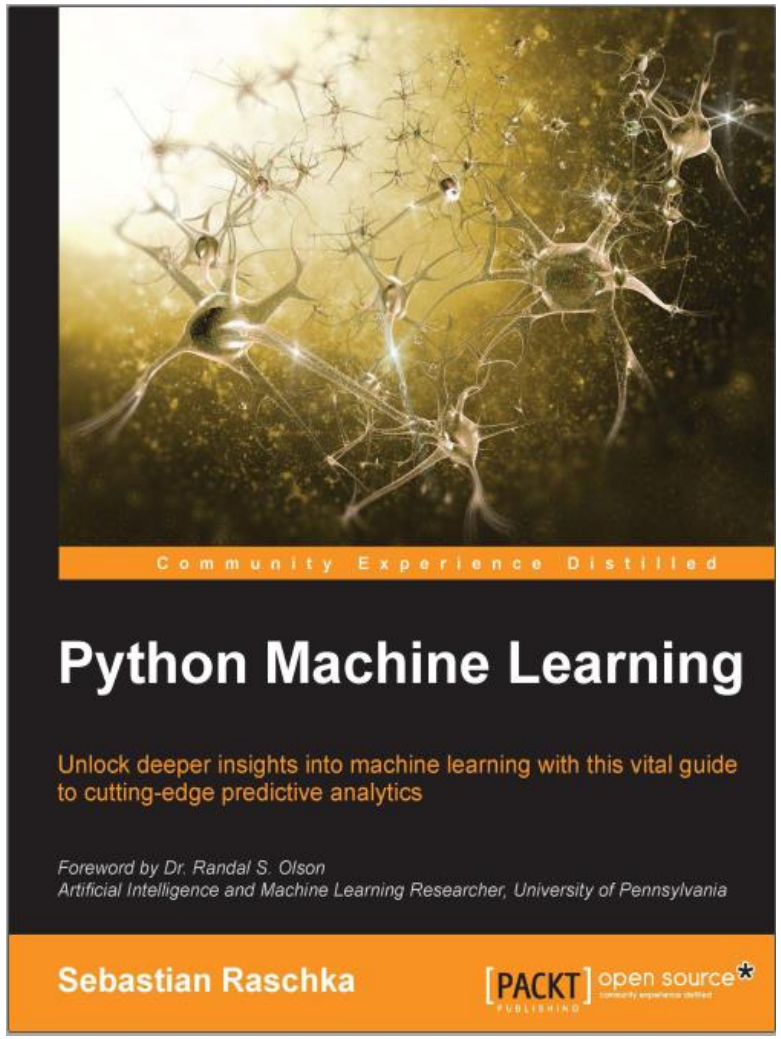
Agenda

- Logistic
- Rule and Regulations
- Introduction
- Why Python
- Environment setup
- Basic syntax.
- Data types and variables.

Logistic

❑ Textbook

Raschka, Sebastian. “Python machine learning”. Packt Publishing Ltd, 2015.



Let's Start

Logistic

❑ Online material

- You can found A collection of IPython notebooks covering various topics in 👇

https://github.com/SarahOsamaTalaat/ipython-notebooks

Rule and Regulations

❑ Evaluation

- ❑ Year Work (25 Marks)
  - Lab work
  - Project
  - Lab Exam

Rule and Regulations

❑ Assignments

- ❑ Assignments are INDIVIDUAL work.
- ❑ Never share code/solution Assignments.

Environment Setup

- Download python 3.6 (Interpreter) 👇
  - https://www.python.org/downloads/

- Download Anaconda for python 3.6 version (64/32 bit) 👇
  - https://www.anaconda.com/download/

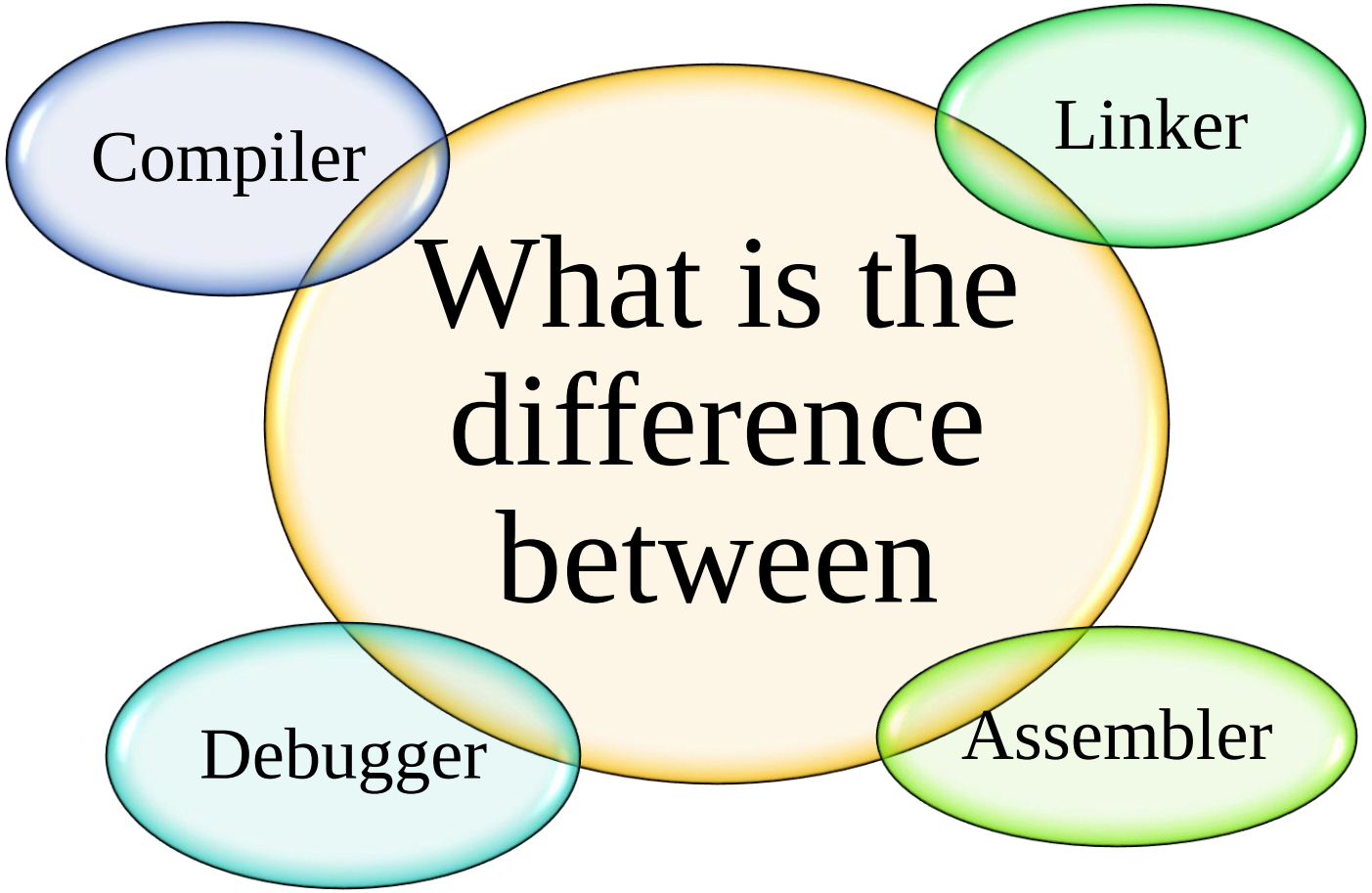
Why Python ?

Visit 👇

https://visual.ly/community/infographic/technology/which-programming-language-should-i-learn-first

Introduction

What is compiler, debugger, assembler, linker, loader?



Introduction

What is compiler, debugger, assembler, linker, loader?

- **Assembler**:

  Is a utility program that converts source code programs from assembly language into machine language

- **Linker**:

  Is a utility programs that combines individual files created by assembler into a single executable file

- **Debugger**:

  Is a utility program lets you to step through a program while it's running and examine registers and memory

Introduction

What is compiler, debugger, assembler, linker, loader?

- **Compiler:**

  Is a computer program (or set of programs) that transforms source code written in a programming language (the source language) into another computer language (the target language, often having a binary form known as object code). The most common reason for wanting to transform source code is to create an executable program

Introduction

Python Code using Anaconda Navigator

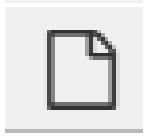
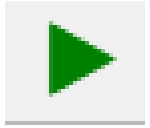
- **We have two ways to write the Python code, the steps of the first one are:**

1. Open Anaconda navigator
2. Open **Jupyter** notebook
3. Any time we want to make a new Python project, we do the following
    1. Go to File tab => new notebook => Python3
    2. To write a new statement click **Enter**
    3. To run your code click **Enter + Shift**
    4. To save your code go to  and click on it.

Introduction

Python Code using Anaconda Navigator

❑ The steps of the second way are:

1. Open Anaconda navigator
2. Open **spyder**
3. Any time we want to make a new Python project, we do the following
    1. Go to File tab => new file or click on 
    2. To run your code click on 
    3. To save your code go to file => save as or 

Basic Syntax

Procedure Name	Task
print	Prints the given text message or expression value on the console, and moves the cursor down to the next line.
input	Reads a number from user input. You can assign (store) the result of the input into a variable.

Basic Syntax

❑Write to console:

```
In [1]: print('Welcome to the first Python lab')
        Welcome to the first Python lab

In [2]: print("Welcome to the first Python lab")
        Welcome to the first Python lab

In [3]: print('1')

In [4]: print('1+2')
        1+2

In [5]: print(10+5)
        15
```

Basic Syntax

❑Read input from the user:

```
In [19]: Var1 = input("Please enter a value: ")

         Please enter a value: 100

In [20]: type(Var1)

Out[20]: str

In [21]: Var1 = input("Please enter a value: ")

         Please enter a value: Hadeer

In [22]: type(Var1)

Out[22]: str
```

Basic Syntax

❑Quotations

```
In [5]: print('Sarah's car')

  File "<ipython-input-5-a01e1f9851d3>", line 1
    print('Sarah's car')
                 ^
SyntaxError: invalid syntax
```

```
In [6]: print("Sarah's car")

Sarah's car
```

Basic Syntax

❑Quotations (cont.):

```
In [7]: print('The Course Title is "Introduction to Python"')
```

```
The Course Title is "Introduction to Python"
```

```
In [8]: print('''Sarah's car stands in "Tahrir" Street''')
```

```
Sarah's car stands in "Tahrir" Street
```

```
In [9]: print("""Sarah's car stands in "Tahrir" Street""")
```

```
Sarah's car stands in "Tahrir" Street
```

Basic Syntax

❑Comment:

- We use # symbol to insert comment in python script.

```
In [12]: #python code
```

```
In [ ]:
```

- Note that, we can use the triple double quotation (“ “ “ any thing ” ” ”) as a comment

Basic Syntax

❑Multi Line Statement:

- We use \ symbol to write multi line statement in python script.

▪ Example

```
In [14]: total = 1 + \
         2 + \
         3
```

```
In [15]: print(total)
```

```
6
```

Basic Syntax

❑Multiple Statements on a Single Line:

- We use **;** symbol to write multiple statements on a single line in python script.

▪ Example

```
In [17]: a=1;b=2;c=3

In [18]: print(a)
         1

In [19]: print(b)
         2

In [20]: print(c)
         3
```

Basic Syntax

❑Lines and Indentation:

- Python provides no braces to indicate blocks of code for class and function definitions or flow control.
- Blocks of code are denoted by line indentation.
- The number of spaces in the indentation is variable, but all statements within the block must be indented the same amount.

In [16]:
```
if True:
    print("First True")
    if True:
        print("Second True")
    else:
        print("Second False")
else:
    print("First False")
```

```
First True
Second True
```

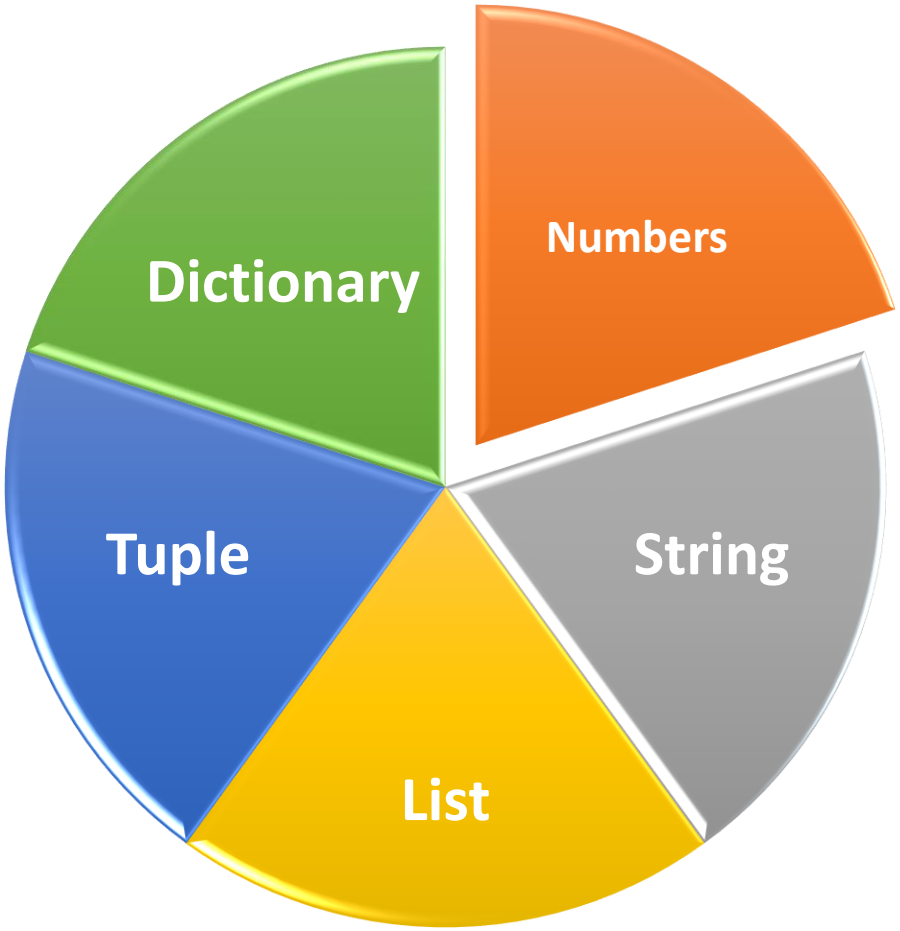
Basic Syntax

❑Multiple Assignment:

```
In [24]: a,b,c=10,11,"Khaled"

In [25]: print(a)
         10

In [26]: print(b)
         11

In [27]: print(c)
         Khaled

In [28]: a=b=c=1

In [29]: print(a,b,c)
         1 1 1
```

Data Types and Variables

Python Standard Data Types



Data Types and Variables

- In python there is no data type declaration before variable creation.
- The variable acquires the data type from the value assigned to it.

```
In [1]: string = "Python 1st lab"

In [2]: type(string)

Out[2]: str

In [3]: Integer = 5

In [4]: type(Integer)

Out[4]: int
```

Data Types and Variables

- In python there is no data type declaration before variable creation.
- The variable acquires the data type from the value assigned to it.

```
In [1]: string = "Python 1st lab"

In [2]: type(string)

Out[2]: str

In [3]: Integer = 5

In [4]: type(Integer)

Out[4]: int
```

Data Types and Variables

- In python there is no data type declaration before variable creation.
- The variable acquires the data type from the value assigned to it.

```
In [1]: string = "Python 1st lab"

In [2]: type(string)
Out[2]: str

In [3]: Integer = 5

In [4]: type(Integer)
Out[4]: int
```

Data Types and Variables

- In python there is no data type declaration before variable creation.
- The variable acquires the data type from the value assigned to it.

```
In [5]: List = [1,2,3]

In [6]: type(List)
Out[6]: list

In [7]: List = ["s","a","r","a","h"]

In [8]: type(List)
Out[8]: list
```

Data Types and Variables

- In python there is no data type declaration before variable creation.
- The variable acquires the data type from the value assigned to it.

```
In [13]: Tuble = (1,2,3)

In [14]: type(Tuble)
Out[14]: tuple

In [15]: Tuble = ("S","a","r","a","h")

In [16]: type(Tuble)
Out[16]: tuple
```

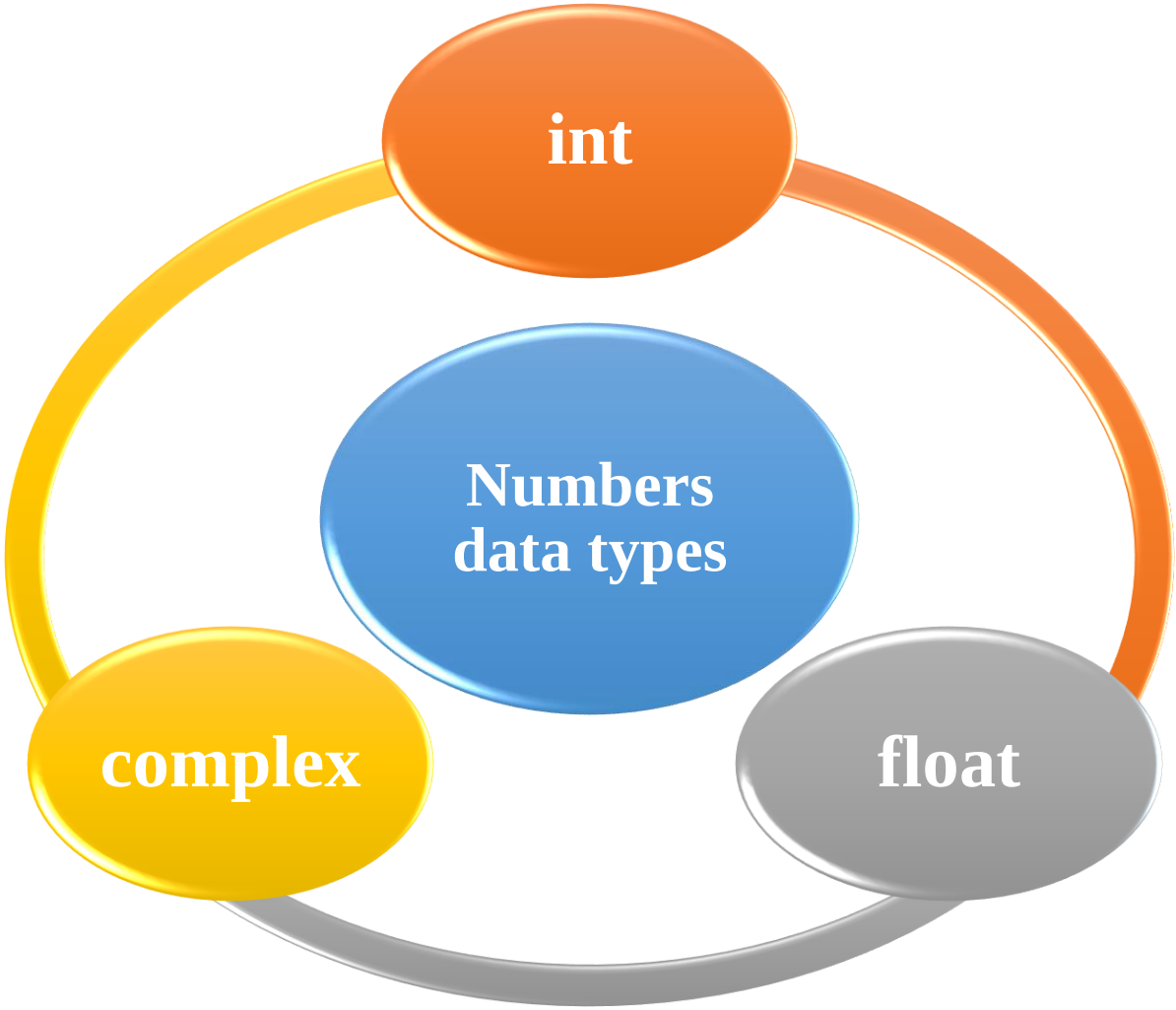
Data Types and Variables

- In python there is no data type declaration before variable creation.
- The variable acquires the data type from the value assigned to it.

```
In [19]: Dictionary = {1:'100',2:'500',3:'700'}

In [20]: type(Dictionary)
Out[20]: dict

In [21]: Dictionary = {1:'File',2:'Edit',3:'View'}

In [22]: type(Dictionary)
Out[22]: dict
```

Data Types and Variables

Numbers



Data Types and Variables

Numbers

- Python supports three different numerical types:

```
In [23]: integer = 5

In [24]: type(integer)
Out[24]: int

In [25]: fraction = 0.5

In [26]: type(fraction)
Out[26]: float

In [34]: complexNumber = 5+2j

In [35]: type(complexNumber)
Out[35]: complex
```

Data Types and Variables

String

- Strings in Python are identified as a contiguous set of characters represented in the quotation marks.

```
In [37]: String = "Sarah"+"Osama"

In [38]: type(String)
Out[38]: str

In [39]: String = "Sarah Osama"

In [40]: type(String)
Out[40]: str
```

Data Types and Variables

String

- Strings in Python are identified as a contiguous set of characters represented in the quotation marks.

```
In [50]: str="Hello world"

In [51]: print(str[0])            # Prints first character of the string
         H

In [52]: print(str[2:5])         # Prints characters starting from 3rd to 5th
         llo

In [53]: print(str[2:])          # Prints string starting from 3rd character
         llo world

In [54]: print(str[:7])          # Prints characters starting from first to 7th
         Hello w

In [55]: print(str * 2)          # Prints string two times
         Hello worldHello world

In [57]: print(str + " TEST")    # Prints concatenated string
         Hello world TEST
```

Data Types and Variables

List

- The list in Python can store various types of data types.
- The list index in Python start from 0.
- The list index in Python can be negative.
- The negative index is used to access the list from the last element to the first element.
- In Python we can write the index as a specific range of indexes by using only one statement

```
In [111]:
List = [1,"sarah",3,4.5,5+7j]
print(List[0:3])

[1, 'sarah', 3]
```

```
In [114]:
List[-2]

Out[114]: 4.5
```

```
In [115]:
print(List[2:])

[3, 4.5, (5+7j)]
```

Data Types and Variables

List

- Python contains a set of built in functions con use with list;

Function Name	Task
len	Return the length of the given list
del	Delete an given element from the list. And also, the dil function is also uses to delete a given list.
min	Return the minimum value from the give list. The value can be both number or text.
max	Return the maximum value from the give list. The value can be both number or text.

Data Types and Variables

List

- Python contains a set of built in functions con use with list;

Function Name	Task
append	Insert a new element to a list. The element will be inserted at an end of a list.
count	Count the frequency of a given element in a list
extend	Add a given list to a currant list
index	Return an index of a given value. By other word, this function is used to search in a list and return the index as a search result.
sort	Sort an elements of a give list.

Data Types and Variables

List

- Python contains a set of built in functions con use with list;

Function Name	Task
insert	Insert a new value in a specific index in a list. The insert function takes two arguments; the first one is an index and the second one is the value
pop	Return an element from a end of a list and remove it from a list. And also, this function can take an index to return and remove it.
remove	Remove an element from the list
reverse	Reverse a containt of a given list

Data Types and Variables

List

❑Slicing

- In addition to accessing list elements one at a time, Python provides concise syntax to access sublists; this is known as slicing:

```
In [21]: # range is a built-in function that creates a list of integers
         List1 = range(4)
         #Prints "range(0, 4)"
         print(List1)
         # Get a slice from index 1 to 3 (exclusive); prints "range(1, 3)"
         print(List1[1:3])
         # Get a slice from index 3 to the end; prints "range(3, 4)"
         print(List1[3:])
         # Get a slice from the start to index 3 (exclusive); prints "range(0, 3)"
         print(List1[:3])
         # Get a slice of the whole list; prints "range(0, 4)"
         print(List1[:])
         # Slice indices can be negative; prints "range(0,2)"
         print(List1[:-2])
         # Prints "range(0, 4)"
         print(List1)

         range(0, 4)
         range(1, 3)
         range(3, 4)
         range(0, 3)
         range(0, 4)
         range(0, 2)
         range(0, 4)
```

Data Types and Variables

List

In [137]:
```
List = ["sarah",25,"FCI"]
```

In [138]:
```
5 in List
```

Out[138]: False

In [140]:
```
'sarah' in List
```

Out[140]: True

Data Types and Variables

List

In [121]:
```
List1 = ["sarah","Osama","Talaat"]
List2 = [1,2,3.5]
# Add the List1 and List2
List1+List2
```

Out[121]: ['sarah', 'Osama', 'Talaat', 1, 2, 3.5]

In [126]:
```
List1 = ["sarah","Osama","Talaat"]
List2 = [1,2,3.5]
# Add the List1 and List2
List1*2
```

Out[126]: ['sarah', 'Osama', 'Talaat', 'sarah', 'Osama', 'Talaat']

Data Types and Variables

List

In [127]:

```
List1 = ["sarah","Osama","Talaat"]
List2 = [1,2,3.5]

List1/List2
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-127-2a74fdd2bc79> in <module>()
      2 List2 = [1,2,3.5]
      3 # Add the List1 and List2
----> 4 List1/List2

TypeError: unsupported operand type(s) for /: 'list' and 'list'
```

Data Types and Variables

List

In [128]:

```
List1 = ["sarah","Osama","Talaat"]
List2 = [1,2,3.5]

List1*List2
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-128-17c9ea38e885> in <module>()
      2 List2 = [1,2,3.5]
      3 # Add the List1 and List2
----> 4 List1*List2

TypeError: can't multiply sequence by non-int of type 'list'
```

Data Types and Variables

List

In [130]:
```
List1 = [5,10,8]
List2 = [1,2,3.5]
List1/List2
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-130-7bc9d7e06abb> in <module>()
      1 List1 = [5,10,8]
      2 List2 = [1,2,3.5]
----> 3 List1/List2

TypeError: unsupported operand type(s) for /: 'list' and 'list'
```

Data Types and Variables

List

In [133]:
```
List1 = [5,10,8]
List2 = [1,2,3.5]
# Get the length of List1
len(List1)
```

Out[133]: 3

In [135]:
```
# Delete List2
del(List2)
```

In [136]:
```
List2
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-136-6e11ad4207ad> in <module>()
----> 1 List2

NameError: name 'List2' is not defined
```

Data Types and Variables

List

In [148]:
```
List1 = [1,2,3,1,1]
List1.append("Sarah")
print(List1)
```

```
[1, 2, 3, 1, 1, 'Sarah']
```

In [146]:
```
List1 = [1,2,3,1,1]
List1.count(1)
```

Out[146]: 3

In [144]:
```
List1 = [1,2,3]
List2 = [10,20,30]
List2.extend(List1)
print(List2)
```

```
[10, 20, 30, 1, 2, 3]
```

Data Types and Variables

List

```
In [160]: List1 = ["Sarah","Hadeer","FCI"]

In [161]: List1.pop(1)

Out[161]: 'Hadeer'

In [162]: List1.pop()

Out[162]: 'FCI'

In [163]: print(List1)

          ['Sarah']

In [166]: List1 = [1,2,3]
          List1.remove(2)
          print(List1)

          [1, 3]
```

Data Types and Variables

List

```
In [58]: list = [ 'abcd', 786 , 2.23, 'john', 70.2 ]

In [59]: tinylist = [123, 'john']

In [61]: print(list[0])        # Prints first element of the list
         abcd

In [62]: print(list[1:3])      # Prints elements starting from 2nd till 3rd
         [786, 2.23]

In [63]: print(list[2:])       # Prints elements starting from 3rd element
         [2.23, 'john', 70.2]

In [64]: print(tinylist * 2)   # Prints list two times
         [123, 'john', 123, 'john']

In [65]: print(list + tinylist)  # Prints concatenated lists
         ['abcd', 786, 2.23, 'john', 70.2, 123, 'john']
```

Data Types and Variables

List

```
In [66]: list[2] = 2001            # List update

In [67]: print(list)
         ['abcd', 786, 2001, 'john', 70.2]

In [68]: del list[2]        # Delete list element

In [69]: print(list)
         ['abcd', 786, 'john', 70.2]

In [71]: list.append(50)    # add element in list

In [72]: print(list)
         ['abcd', 786, 'john', 70.2, 50]

In [73]: print(list[4])
         50

In [74]: print(list[-1])
         50
```

Data Types and Variables

Tuple

- A tuple is another sequence data type that is similar to the list. A tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parenthesis.
- The main difference between lists and tuples are − Lists are enclosed in square brackets [ ] and their elements and size can be changed, while tuples are enclosed in parentheses ( ) and cannot be updated. Tuples can be thought of as **read only** lists.

Data Types and Variables

Tuple

```
In [43]: Tuble = (1,2,3)

In [44]: Tuble = ("S","a",1,2)

In [45]: Tuble[1] = "Sarah"

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-45-6b37363ac0dc> in <module>()
----> 1 Tuble[1] = "Sarah"

TypeError: 'tuple' object does not support item assignment
```

Data Types and Variables

Tuple

```
In [48]: del Tuble[0] # Delete instruction
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-48-ca652cd1bd3d> in <module>()
----> 1 del Tuble[0] # Delete instruction

TypeError: 'tuple' object doesn't support item deletion
```

Data Types and Variables

Tuple

```
In [75]: tuple1 = (123, 'john')

In [77]: print (tuple1[0])        # Prints first element of the tuple

         123

In [78]: tuple1[1]='Ali'          # Update

         ---------------------------------------------------------------------------
         TypeError                                 Traceback (most recent call last)
         <ipython-input-78-5b239f77a954> in <module>()
         ----> 1 tuple1[1]='Ali'

         TypeError: 'tuple' object does not support item assignment

In [79]: del tuple1[0]        # Delete

         ---------------------------------------------------------------------------
         TypeError                                 Traceback (most recent call last)
         <ipython-input-79-d7cec9be47de> in <module>()
         ----> 1 del tuple1[0]

         TypeError: 'tuple' object doesn't support item deletion
```

Data Types and Variables

Dictionary

- Dictionary is a sequence data type, each item consists of pair of ***key*** and ***value***.
- The ***key*** has unique value.
- Each ***key*** is separated from its ***value*** by a colon (**:**), the items are separated by commas, and the whole thing is enclosed in curly braces **{}**
- The ***value*** of each item can be retrieved by using its ***key*** instead of index as list and tuple.

Data Types and Variables

Dictionary

```
In [56]: #Create a dictionary
         Dictionary = {"Name":"Sarah Osama","Age":26,"Jub":"Teaching Assistance"}

In [58]: #Retrieve a second value from a dictionary
         Dictionary["Age"]

Out[58]: 26

In [60]: #Another way to retrieve a second value from a dictionary
         Dictionary.get("Age")

Out[60]: 26
```

Data Types and Variables

Dictionary

```
In [50]: Dictionary = {"Name":"Sarah Osama","Age":26,"Jub":"Teaching Assistance"}
```

```
In [51]: Dictionary[1]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-51-824dc68a1d03> in <module>()
----> 1 Dictionary[1]

KeyError: 1
```

```
In [61]: #Insert a new item to a dictionary
         Dictionary['Faculty'] = 'FCI'
```

```
In [62]: Dictionary.get('Faculty')

Out[62]: 'FCI'
```

Data Types and Variables

Dictionary

```
In [63]: Dictionary
```

```
Out[63]: {'Age': 26,
          'Faculty': 'FCI',
           'Jub': 'Teaching Assistance',
           'Name': 'Sarah Osama'}
```

Data Types and Variables

Dictionary

```
In [64]: #Update an item value in a dictionary
         Dictionary['Name'] = 'Sarah Osama Talaat'

In [65]: Dictionary['Name']

Out[65]: 'Sarah Osama Talaat'

In [66]: #Remove the item wich called Age from a dictionary
         del Dictionary['Age']

In [67]: Dictionary

Out[67]: {'Faculty': 'FCI', 'Jub': 'Teaching Assistance', 'Name': 'Sarah Osama Tal
         aat'}
```

Data Types and Variables

Dictionary

In [72]:
```
# Remova the inserted item from the dictionary
# Make the dictionary empty
Dictionary.clear()
```

In [73]:
```
Dictionary
```

Out[73]:
```
{}
```

In [74]:
```
# Remova the dictionary
del Dictionary
```

In [75]:
```
Dictionary
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-75-430b48a0ca9c> in <module>()
----> 1 Dictionary

NameError: name 'Dictionary' is not defined
```

Data Types and Variables

Dictionary

```
In [81]: #Create a dictionary
         Dictionary1 = {"Name":"Sarah Osama","Age":26,"Jub":"Teaching Assistance"}
         #Copy the elements of the Dictionarry1 to the Dictionary2
         Dictionary2 = Dictionary1.copy()

In [82]: Dictionary2

Out[82]: {'Age': 26, 'Jub': 'Teaching Assistance', 'Name': 'Sarah Osama'}

In [86]: #Create a new list to store the keys of a new dictionary wich called NewDic
         ListOfKeys = ["Name","Age","Faculty"]
         #Create a new dictionary called NewDic which its keys stored in ListOfKeys
         NewDic = dict.fromkeys(ListOfKeys)

In [87]: NewDic

Out[87]: {'Age': None, 'Faculty': None, 'Name': None}
```

Data Types and Variables

Dictionary

```
In [93]: #Read the content of a dictionary as a list of tubles,
         #each tubles containes key and value
         Dictionary2.items()

Out[93]: dict_items([('Name', 'Sarah Osama'), ('Age', 26), ('Jub', 'Teaching Assis
         tance')])
```

```
In [98]: #Retrieve a dictionary values as a list
         Dictionary2.values()

Out[98]: dict_values(['Sarah Osama', 26, 'Teaching Assistance'])
```

```
In [99]: #Retrieve a dictionary keys as a list
         Dictionary2.keys()

Out[99]: dict_keys(['Name', 'Age', 'Jub'])
```

Any Questions!?

Thank you