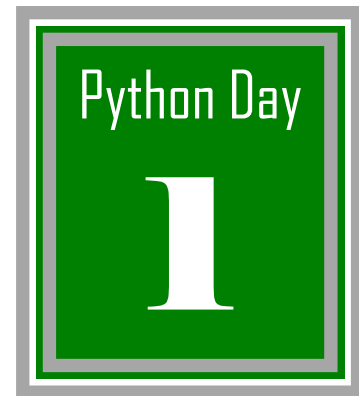




King Khalid University
College of Computer Science



Programming in Python **(Module No.1)**

INTRODUCTION TO PYTHON

Prepared by

Eng. Abdelmoty Mahmoud Mohamed Ahmed

Lecturer in Computer engineering department, College of Computer Science, King Khalid University

M.Sc. in Computers and systems engineering department

Faculty of Engineering (boys) in Cairo

Al-Azhar University

E-mail: amoate@kku.edu.sa

abd2005moty@yahoo.com

Mobile: 00966504538402

2018-2019

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

إِنْ أَرِيدُ إِلَّا الْإِصْلَاحَ مَا اسْتَطَعْتُ وَمَا تَوْفِيقِي إِلَّا بِاللَّهِ عَلَيْهِ تَوَكَّلْتُ وَإِلَيْهِ أُنِيبُ

صدق الله العظيم

Diploma in Machine Learning with python(150 hours)

Goals

1. Learning Python Programming
2. Using Python's API's (Tensorflow, Keras, Numpy, Scipy,....etc)
3. Build classification Algorithms
4. Build regression Algorithms

Vision

- Replace traditional Learning to modern learning (Self, E-learning, Intelligent Learning ,Deep Learning,.....)



Module No.1 (Introduction to Python)-20Hours

Day No.1

- Introduction to Python
- Install python development environment.
- The first program with python
- Install Anaconda Framework and Jupiter
- Python Basic Syntax

Day No.2

- Print statement
- Variables and Datatypes
- Operators
- Python Input, Output and Import

Day No.3

- Decision Making
- Loops in python

Day No.4

- Python Lists
- Python Tuples
- Python Dictionary

Day No.5

- Python Functions
- Python Modules

Module No.1- Specially to KKU student's

Module No.2 (Python's APIs)- 24Hours**Contents:-**

- Day No.1:-** Object Oriented
- Day No.2:-** Exceptions Handling
- Day No.3:-** Numpy Library
- Day No.4:-** Scipy Library
- Day No.5:-** Matplotlib Library
- Day No.6:-** pandas Library
- Day No.7:-** Data Analysis by Python
- Day No.8:-** Applications and simple Projects

Module No.3 (Python's Digital Image Processing "DIP")- 30Hours***Python 2.7 +32bit.***

- Day No.1:-** Introduction to Python -Variables and Datatypes
- Day No.2:-** Decision Making -Loops in python
- Day No.3:-** Python Lists - Tuples – Dictionary using in DIP
- Day No.4:-** Python Functions - Modules
- Day No.5:-** Classes that using in DIP
- Day No.6:-** GUI
- Day No.7:-** DIP libraries
- Day No.8:-** Open Computer Vision Library (Open CV) and Python Imaging Library (PIL)
- Day No.9:-** Core operation in open CV
- Day No.10:-** Core operation in PILLOW
- Day No.11:-** Video Analysis in open CV
- Day No.12:-** Machine Learning in DIP
- Day No.13:-** Object Detection in open CV
- Day No.14:-** Applications and simple Projects
- Day No.15:-** Applications and simple Projects

Module No.4 (Python's Data Treatment)-24Hours

- Day No.1:-** Data Machine Learning.
- Day No.2:-** How to load data machine learning.
- Day No.3:-** understand data with descriptive statistics
- Day No.4:-** understand data with visualization.
- Day No.5:-** prepare data for machine learning.
- Day No.6:-** Applications and simple Projects

Module No.5(Machine Learning 1) -28Hours

- Day No.1:-**
 - Introduction to Machine Learning
 - Features Selection for Machine Learning.
- Day No.2:-**
 - Evaluate the performance of machine learning.
 - Algorithms with resampling in machine learning.
- Day No.3:-**
 - Machine learning algorithms performance matrices.
- Day No.4:-**
 - Classifications algorithms
- Day No.5:-**
 - Regression algorithms
- Day No.6:-**
 - Applications and simple Projects

Day No.7:-

- Applications and simple Projects

Module No.6 (Machine Learning 2) -24Hours

Day No.1:-

- Compare between Classifications algorithms
- Compare between Regression algorithms

Day No.2:-

- Automate machine learning workflow (Keras pipelines).
- Algorithms with resampling in machine learning.

Day No.3:-

- Save and loading Machine learning Models.

Day No.4:-

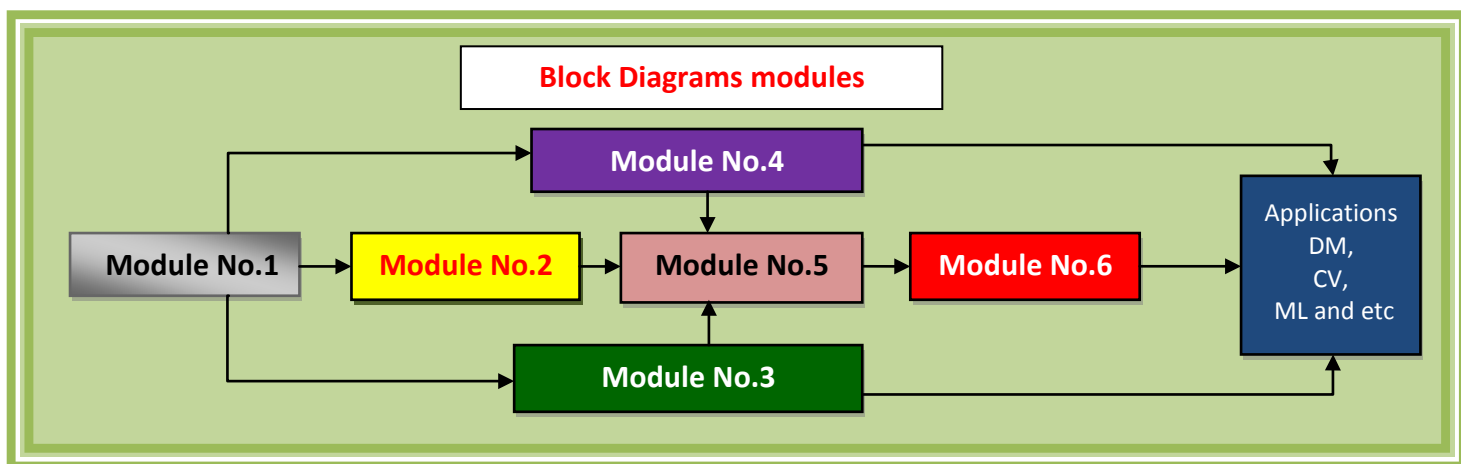
- Classifications Projects

Day No.5:-

- Regression Projects

Day No.6:-

- Applications and simple Projects



Module No.1 (Introduction to Python)-28Hours

The first part in Python Day No.1

- Introduction in Python
- Why Python?
- Python editions.
- Install python development environment.
- Check Python version
- Two different modes of programming in Python
- The First Program with Python

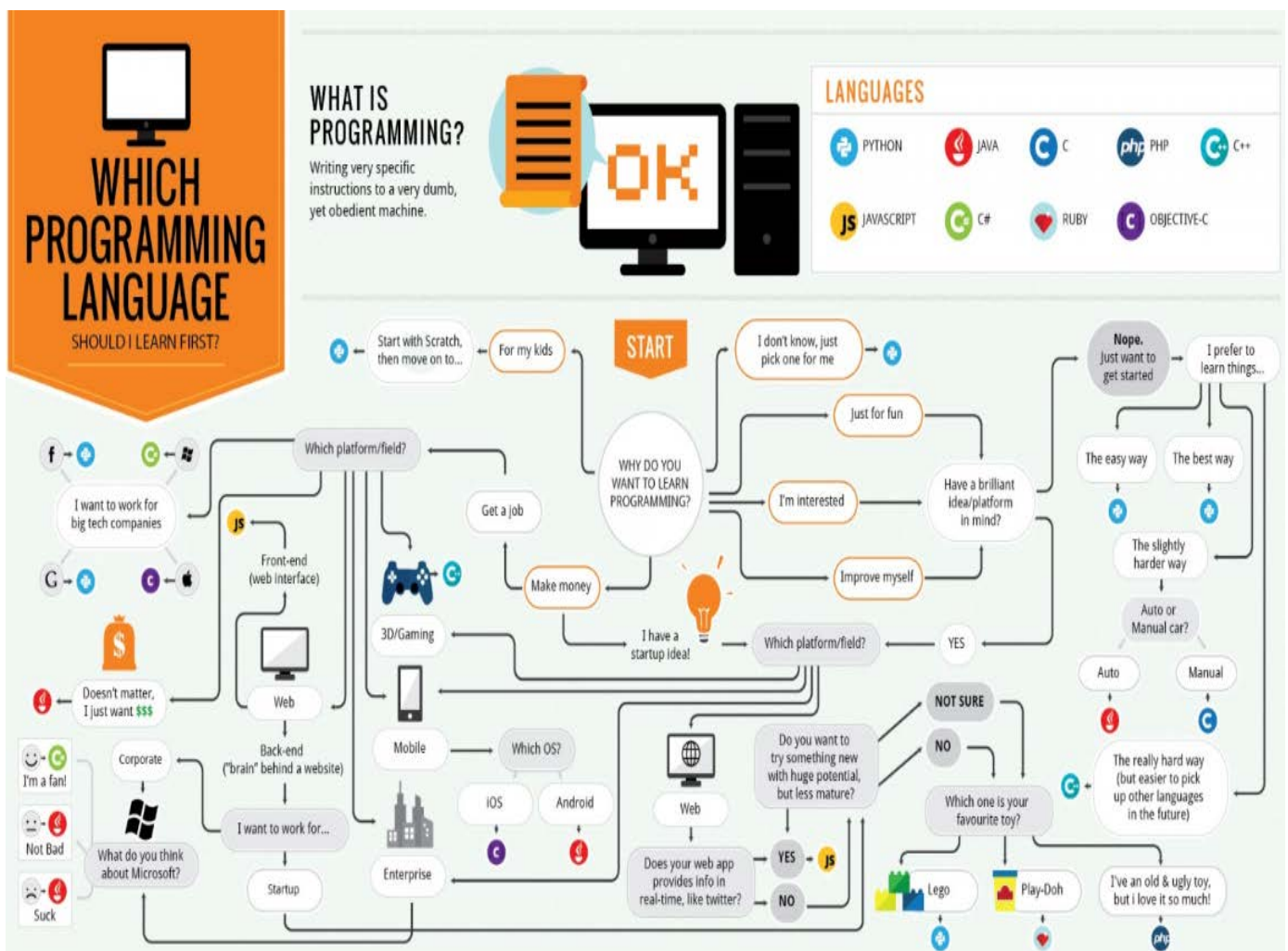







Introduction in Python?

Python is a widely used high-level programming language for general-purpose programming, created by Guido van Rossum and first released in 1991. Python is very simple, yet incredibly powerful programming language. You can use it for writing web and desktop applications, do scientific computations, create scripts, and more. You can use Python for almost anything. In this course, you will learn the basics of Python syntax, functions and creating console apps.



Why Python?



THE LORD OF THE RINGS ANALOGY TO PROGRAMMING LANGUAGES									
Python	Java	C	C++	JavaScript	C#	Ruby	PHP	Objective-C	
									
Python The Ent	Java Gandalf	C One Ring	C++ Saruman	JavaScript Hobbit	C# Elf	Ruby Man (Middle Earth)	PHP Orc	Objective-C Smog	
Help the Hobbits Beginning to understand programming concepts Help Wizards (computer scientists) to conduct researches Widely regarded as the best programming language for beginners Easiest to learn Widely used in scientific, technical & academic field, i.e. Artificial Intelligence You can build website using Django a popular Python web framework	Plots peace & works with everyone possible Very popular on all platforms, OS, and devices due to its portability One of the most in demand & highest paying programming languages Slogan: write once, work everywhere	The power of C is known to them all Everyone wants to get its Power Lingua franca of programming language One of the oldest and most widely used language in the world Popular language for system and hardware programming A subset of C++ except the little details	Everyone thinks that he is the good guy But once you get to know him, you will realize he wants the power, not good deeds Complex version of C with a lot more features Widely used for developing games, industrial and performance-critical applications Learning C++ is like learning how to manufacture, assemble, and drive a car Recommended only if you have a mentor to guide you	Frequently underestimated (powerful) Well-known for the time, gentle (if of the Sire web browser) Java and JavaScript are similar like Car and Carpet are similar - drag weight Most popular client-side web scripting language A must learn for front-end web developer (HTML and CSS as well) One of the hottest programming language now due to its increasing popularity as server-side language (node.js)	Beautiful creature language, but stays in their mind, Rwanda (Africa) A popular choice for enterprise to create websites and Windows application using .NET framework Can be used to build website with ASP.NET, a web framework from Microsoft Similar to Java in basic syntax and some features Learn C# instead of Java if you are targeting to work on Windows platform only	Very emotional creature They some Ruby developers, few they are superior & need to rule the Middle Earth Mostly known for its popular web framework, Ruby on Rails Focuses on getting things done Designed for fun and productive coding Best for fun and personal projects, startups, and rapid development	Ugly guy (language) and doesn't respect the rules (inconsistent and unpredictable) Big headache to those developers to manage them (code) Not 100% dominates the Middle-earth (most popular web scripting language) Suitable for building small and simple sites within a short time frame Supported by almost every web hosting services with lower price	Lives in Mordor (Apple ecosystem) Lonely and over-god Primary language used by Apple for Mac OS X & iOS Choose this if you want to focus on developing iOS or OS X apps only Consider to learn Swift (newly introduced by Apple in 2014) as your next language	
POPULARITY ★★★★★	POPULARITY ★★★★★	POPULARITY ★★★★★	POPULARITY ★★★★★	POPULARITY ★★★★★	POPULARITY ★★★★★	POPULARITY ★★★★★	POPULARITY ★★★★★	POPULARITY ★★★★★	POPULARITY ★★★★★
USED TO BUILD YouTube, Instagram, Spotify	USED TO BUILD Gmail, Microsoft, Most Android Apps, Enterprise applications	USED TO BUILD Operating systems and hardware	USED TO BUILD Operating systems, hardware, and browsers	USED TO BUILD Paypal, front-end of majority websites	USED TO BUILD Enterprise and Windows applications	USED TO BUILD Hulu, Groupon, SlideShare	USED TO BUILD WordPress, Wikipedia, Flickr	USED TO BUILD Most iOS Apps and part of Mac OS X	
AVG. SALARY \$107,000	AVG. SALARY \$102,000	AVG. SALARY \$102,000	AVG. SALARY \$104,000	AVG. SALARY \$99,000	AVG. SALARY \$94,000	AVG. SALARY \$107,000	AVG. SALARY \$89,000	AVG. SALARY \$107,000	



The whole process

Python code is executed using an interpreter, which is a program that reads one statement from the source code, translates it to the machine code or virtual machine code, and then executes it right away.

Python is interpreted language, which means that Python code is translated and executed by an interpreter, one statement at a time. Interpreter is a program that converts the high-level program we write into low-level program that the computer understands

At first, we write our python source file, a file that has .py extension. Then we use the python interpreter to interpret and run the program.



Python Editions

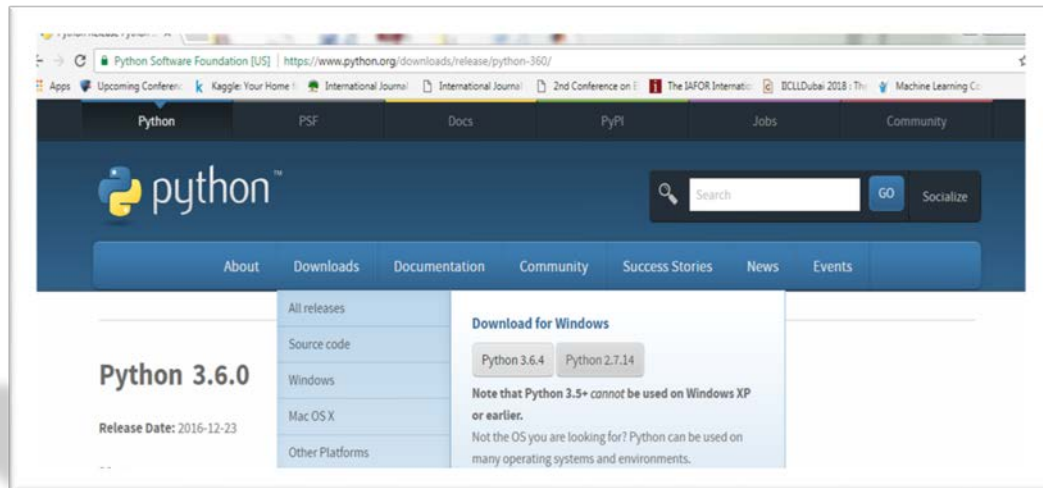
Python is now being developed and maintained by a large team of volunteers and is available for free from the Python Software Foundation. Two versions of Python are currently coexistent: Python 2.7 and Python 3.6. The programs written in Python 3.6 will not run in Python 2. Python 3.6 is a newer version, but it is not backward-compatible with Python 2. This means that if you write a program using the Python 2 syntax, it may not work with a Python 3 interpreter. Python provides a tool that automatically converts code written in Python 2 into syntax Python 3 can use. In this course, we will learn Python 3.6.

The power of Python programming is not in the use of commands related to it but in the use of its own libraries.

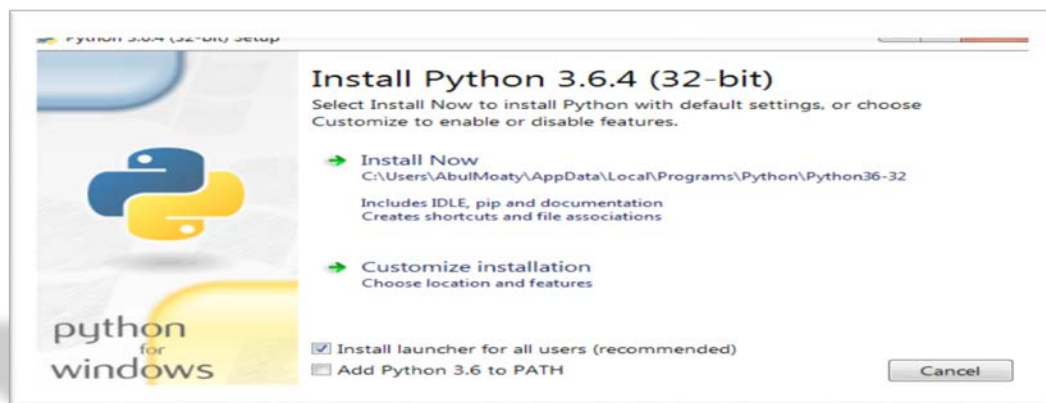


Install python development environment

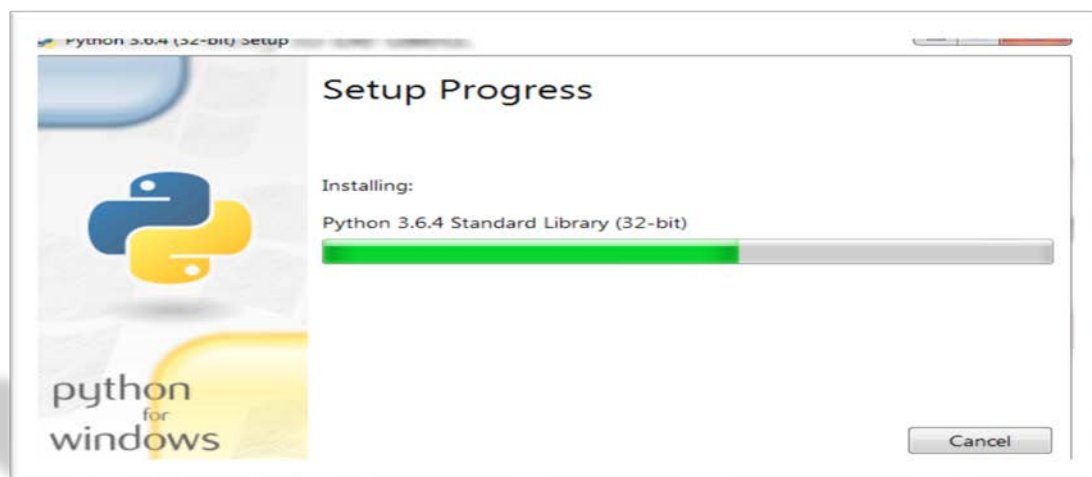
1. Go to Python's downloads page. <https://www.python.org/downloads/>
2. From the page, choose to download the second edition as illustrated in the figure below.



3. Run the installation wizard that you have just downloaded.



4. The installation process will start. At the end, you'll have Python installed on your machine and ready to be used.





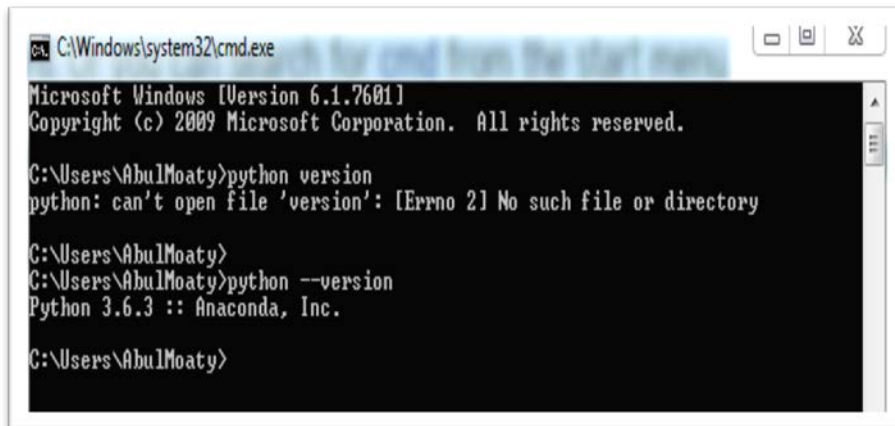
Check Python version

To check Python version installed on your machine follow the following steps.

1. To run the command prompt, press Win + R at the same time, then type cmd in the wizard. Or you can search for cmd from the start menu.
2. In the command prompt print the following command to test python's version.

```
python --version
```

3. Press Enter. The following result will be displayed on the screen.



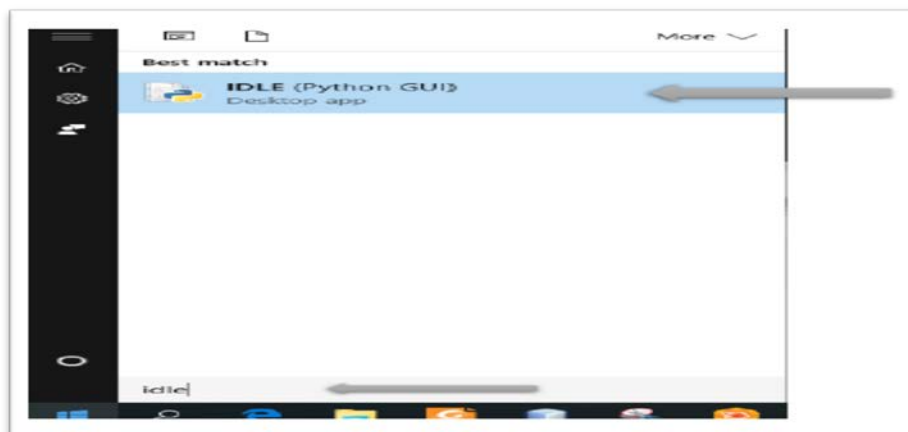
```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\AbulMoaty>python version
python: can't open file 'version': [Errno 2] No such file or directory

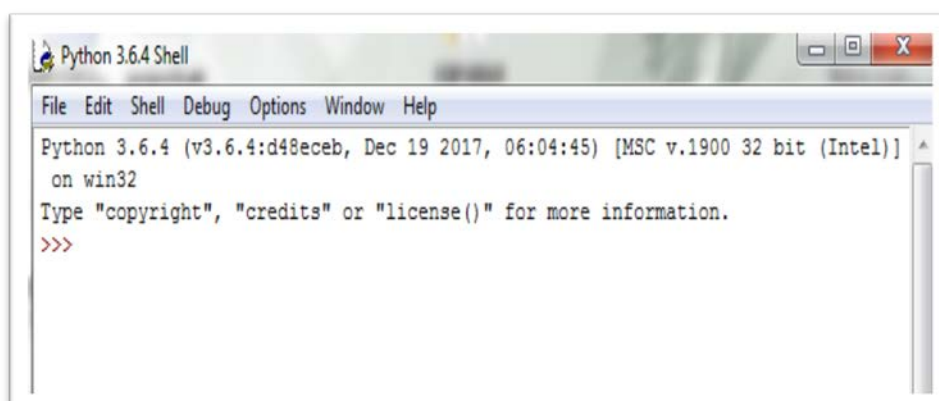
C:\Users\AbulMoaty>
C:\Users\AbulMoaty>python --version
Python 3.6.3 :: Anaconda, Inc.

C:\Users\AbulMoaty>
```

The following icon appears after type search window from now on we simply double-click this icon to open the Python IDE, it is a simple IDE.



The command window, shell, appears:



Two different modes of programming in Python

1. Interactive Mode Programming:

Invoking the interpreter without passing a script file as a parameter.

We can see the prompt `>>>`, where we can write and execute Python code in interactive mode. At the prompt, enter your first command.

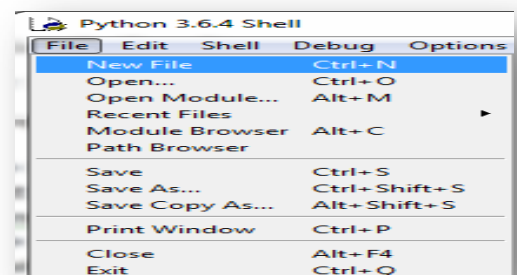
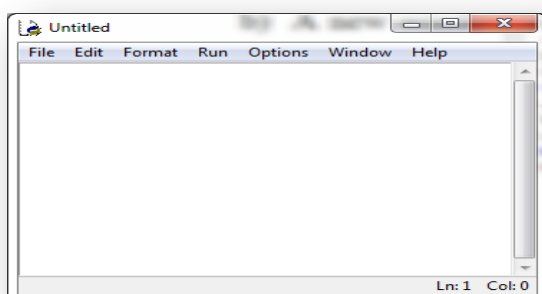
2. Script Mode Programming (batch mode):

Invoking the interpreter with a script parameter begins process of writing and executing the lines of Python code one by one until the script is finished. When the script is finished, the interpreter is no longer active.

This becomes impractical for larger programs, so an alternative exists. In batch mode, we first write all the lines in a separate file, and then run them all at once.

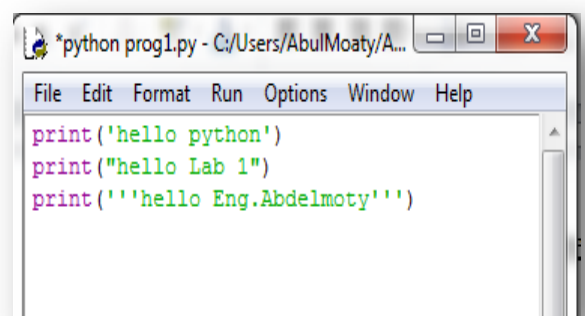
Let us write a simple Python program in a script. Python files have extension `.py` type the following source code in a `hello.py` file.

- a) In the shell window, click File → New File



- b) A new window appears, without the Python prompt `>>>`

- c) Where we can enter one or more Python commands before executing them all at once.

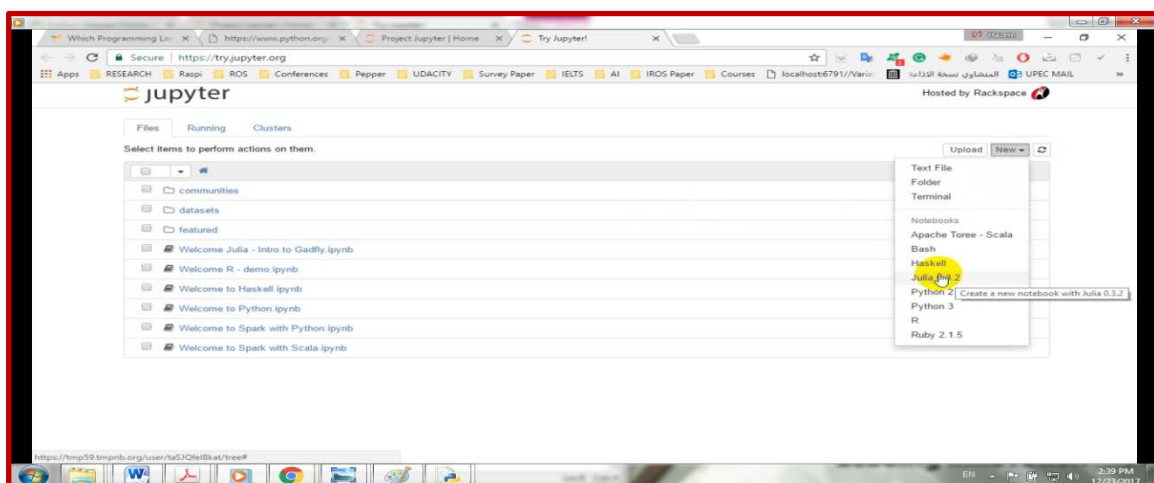
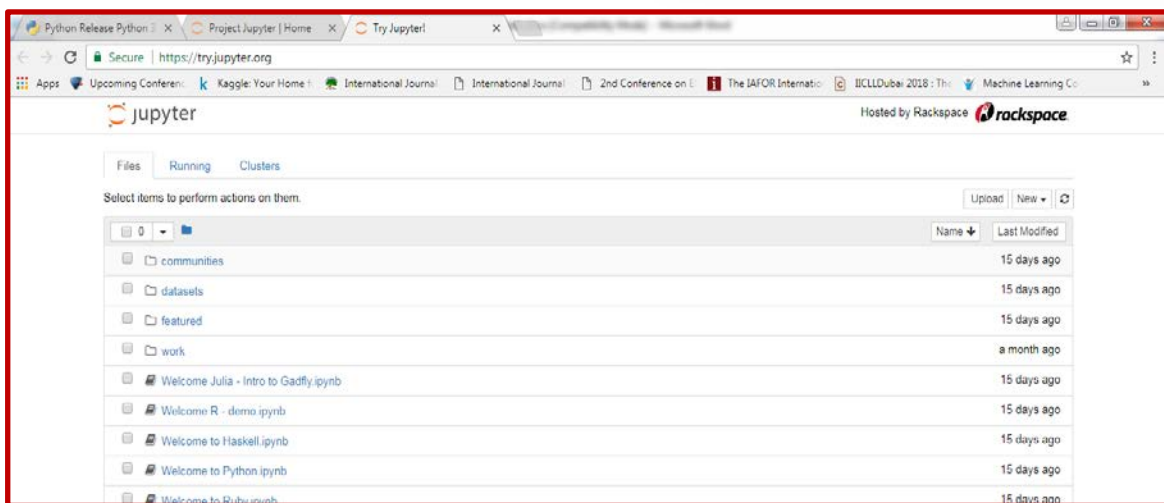
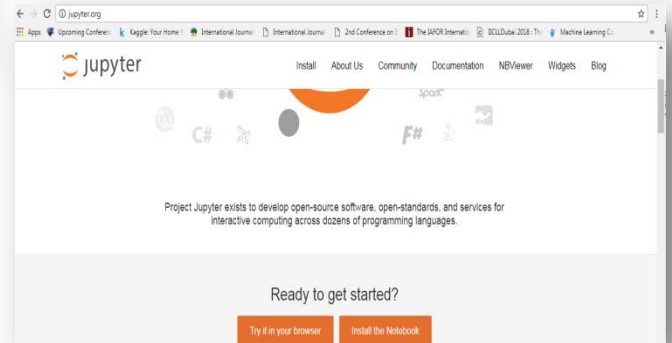


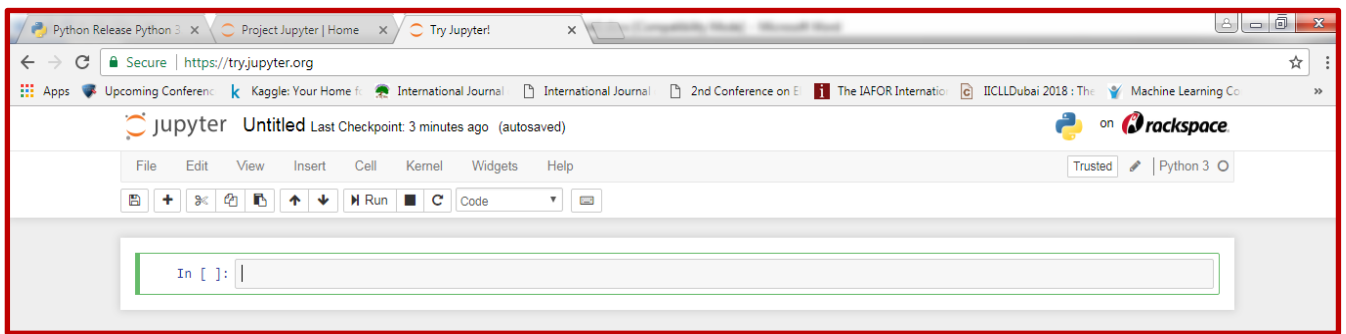
```
Python 3.6.4 Shell
File Edit Shell Debug Options Window Help
Python 3.6.4 (v3.6.4:d48ebeb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>>
RESTART: C:/Users/AbulMoaty/AppData/Local/Programs/Python/Python36-32/python pr
og1.py
hello python
hello Lab 1
hello Eng.Abdelmoty
>>>
```

d) When you're ready, click Run → Run Module (or just press the function key F5). IDLE will first ask you to save the file, then it will execute it, with the results shown back in the shell window.

e) In our lab we will use another IDE (Integrated development environment)

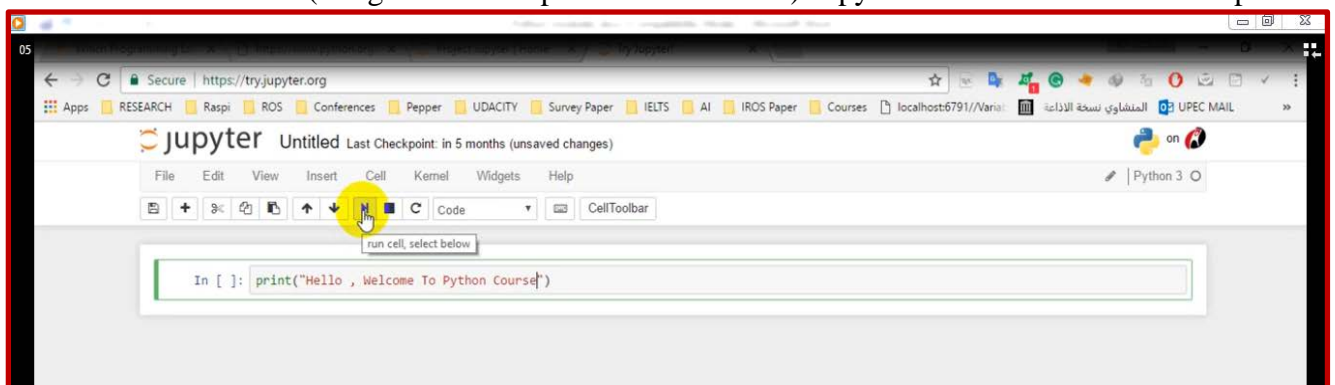
Jupyter .<https://www.Jupyter.org/>



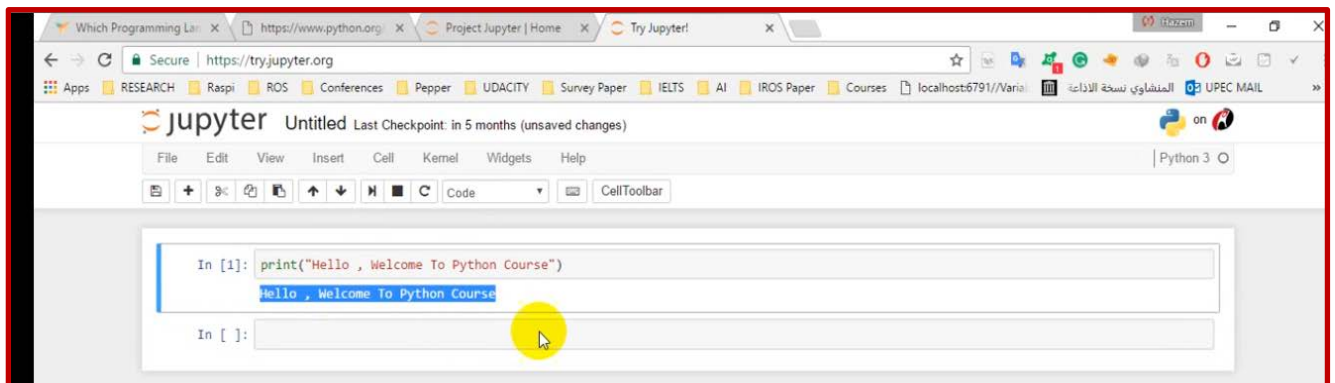


The First Program with Python

1. In our IDE (Integrated development environment) Jupyter write the first command print



2. Click run and press enter



For collect these steps and install one package to use python programming we will download Anaconda Python package in the second part of Python Day No.1

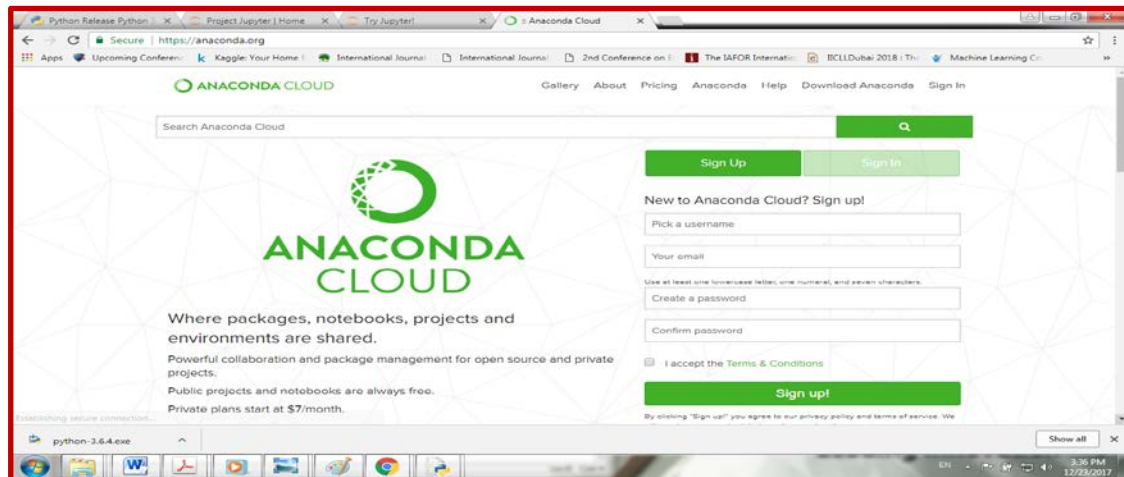


The Second part in Python Day No.1

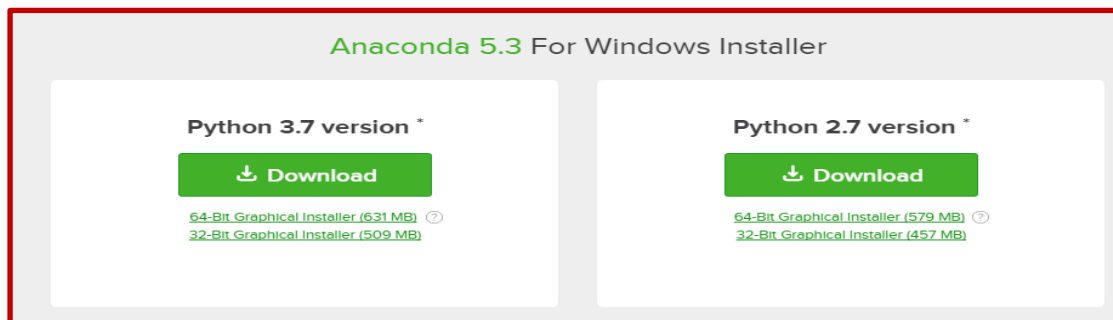
- Install Anaconda Framework
- Python examples

Install Anaconda Framework

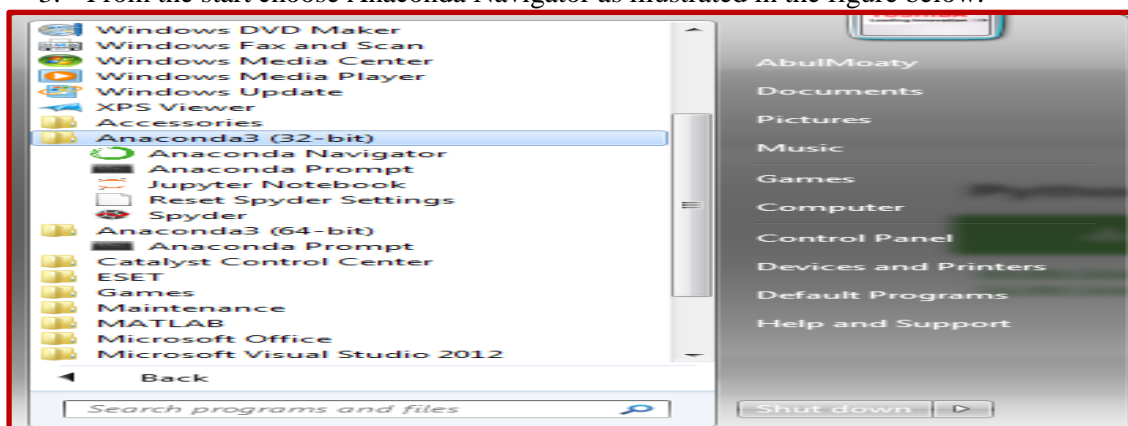
1. Go to ANACONDA downloads page. <https://www.anaconda.com/download/>



2. From the page <https://www.anaconda.com/download/>, choose to download the python 3.7 as illustrated in the figure below.

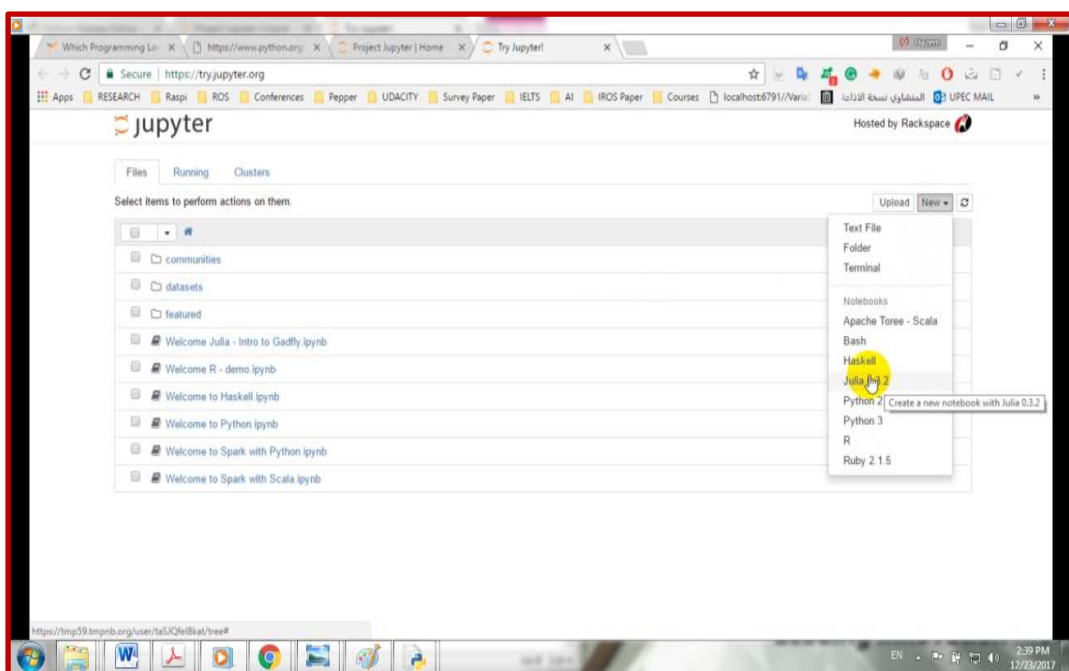
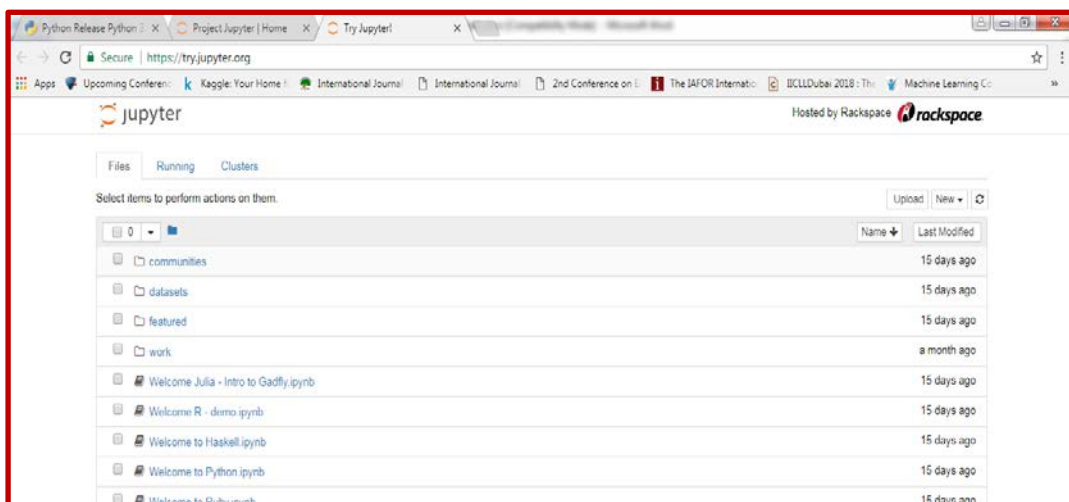
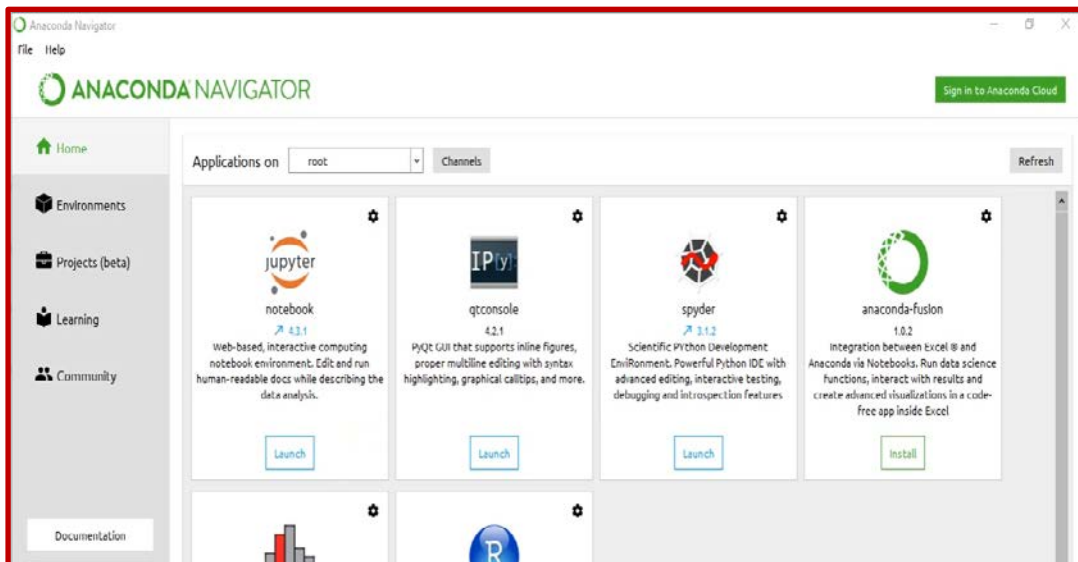


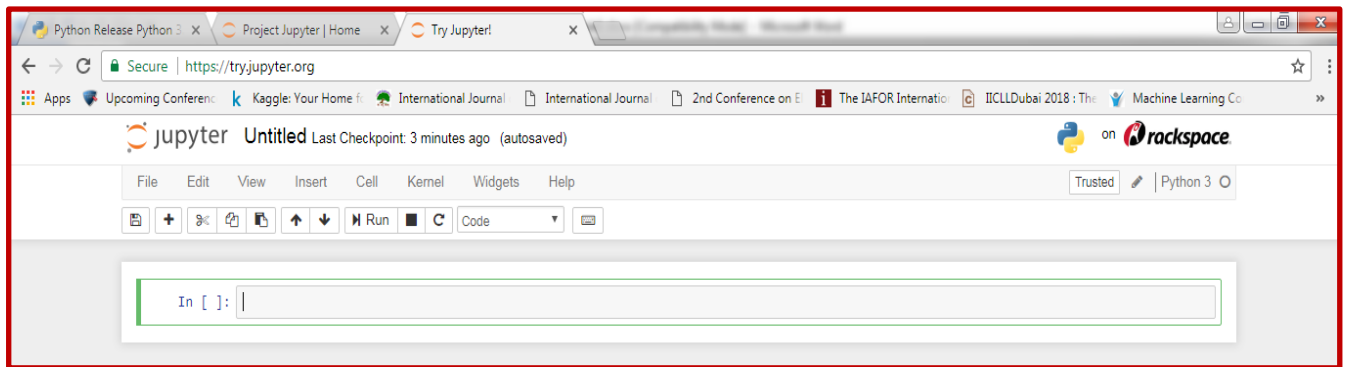
3. From the start choose Anaconda Navigator as illustrated in the figure below.



Anaconda is set of tools (Python, Jupyter IDE, Spyder,etc)

4. From the Anaconda Navigator page choose jupyter notebook as illustrated in the figures below





Python Examples

This is a collection of various statements, features, etc. of IPython and the Python language. Much of this content is taken from other notebooks so I can't take credit for it, I just extracted the highlights I felt were most useful.

Code cells are run by pressing shift-enter or using the play button in the toolbar.

```
In [1]: a = 10

In [2]: print(a)

10

In [3]: import math

In [4]: x = math.cos(2 * math.pi)
print(x)

1.0
```

Import the whole module into the current namespace instead.

```
In [5]: from math import *
x = cos(2 * pi)
print(x)

1.0
```

Write a program that displays the arrow pattern? (Display a pattern).

Some examples for print statement

```
In [19]: print(1)
1

In [20]: print(1+2)
3

In [21]: print('1+2')
1+2

In [22]: print("abdelmty's course")
abdelmty's course

In [24]: print('eng.tamer tetch "python course"')
eng.tamer tetch "python course"

In [28]: print(''hello'')
hello

In [32]: print("""Abdelmoty course""")
"Abdelmoty course"
```

Reading input from user

We can use the input function to read values from user. The input function prompt the user to enter a value and returns the value to be stored in a variable. The following example illustrates the input function:

```
In [14]: name = input("Enter Your Name: ")
print("Your name is: ", name)
print("Type of variable 'name' is: ", type(name))

Enter Your Name: Eng.Abdelmoty M. Ahmed
Your name is: Eng.Abdelmoty M. Ahmed
Type of variable 'name' is: <class 'str'>
```

Python Day 1 Home Work

1. Write a program that displays “Welcome to Engineering”, “Welcome to Python”, and “Programming is fun” (Display three messages).
2. Write a program that displays the following pattern? (Display a pattern).

```

      *
     ***
    *****
   *********
  **********
 *          *
**        **
***      ***
****    ****
*****  *****
*       *
**      **
***     ***
****    ****
*****  *****
*       *
**      **
***     ***
****    ****
*****  *****

```

3. Write a program that displays the result of $1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9$
4. Write a program that displays the result of $\frac{9.5 * 4.5 - 2.5 * 3}{45.5 - 3.5}$
5. Write a program that displays the area and perimeter of a rectangle with the width of 4.9 and height of 7.5 using the following formulas:
 - $area = width * height$
 - $perimeter = 2 * width + 2 * height$

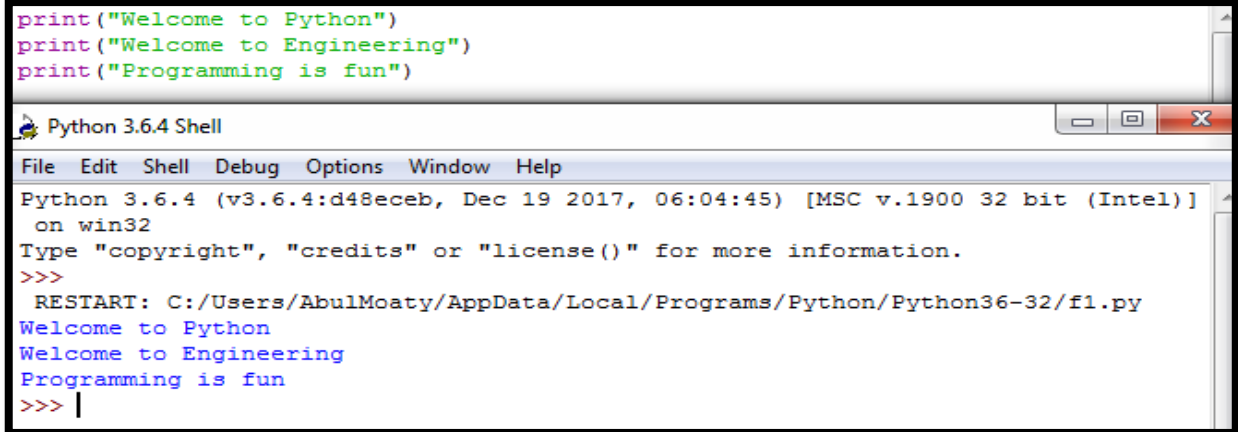
Python Day No.1 Notes

This image shows a full page of white paper with horizontal dotted lines, typical of primary school writing paper. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

[illegible]

Python Day No.1 Solution Home Work

1. Write a program that displays *Welcome to Python*, *Welcome to Engineering* and *Programming is fun* on the screen. Include some comments.



```
print("Welcome to Python")
print("Welcome to Engineering")
print("Programming is fun")

Python 3.6.4 Shell
File Edit Shell Debug Options Window Help
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>>
RESTART: C:/Users/AbulMoaty/AppData/Local/Programs/Python/Python36-32/f1.py
Welcome to Python
Welcome to Engineering
Programming is fun
>>> |
```

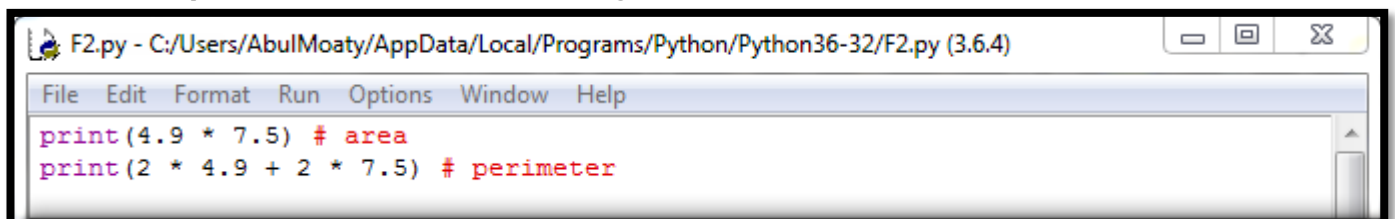
2. Write a program that displays the result of $1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9$

```
>>> print(1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9)
45
>>>
```

3. Write a program that displays the result of $\frac{9.5 * 4.5 - 2.5 * 3}{45.5 - 3.5}$

```
>>> print((9.5 * 4.5 - 2.5 * 3) / (45.5 - 3.5))
0.8392857142857143
```

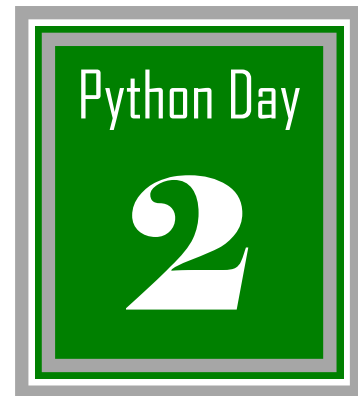
4. Write a program that displays the area and perimeter of a rectangle with the width of 4.9 and height of 7.5 using the following formulas:
 - $area = width * height$
 - $perimeter = 2 * width + 2 * height$



```
F2.py - C:/Users/AbulMoaty/AppData/Local/Programs/Python/Python36-32/F2.py (3.6.4)
File Edit Format Run Options Window Help
print(4.9 * 7.5) # area
print(2 * 4.9 + 2 * 7.5) # perimeter
```



King Khalid University
College of Computer Science



Programming in Python (Module No.1)

VARIABLES AND DATA TYPE

Prepared by

Eng. Abdelmoty Mahmoud Mohamed Ahmed

Lecturer in Computer engineering department, College of Computer Science, King Khalid University

M.Sc. in Computers and systems engineering department

Faculty of Engineering in Cairo

Al-Azhar University

2018-2019

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

إِنْ أَرِيدُ إِلَّا الْإِصْلَاحَ مَا اسْتَطَعْتُ وَمَا تَوْفِيقِي إِلَّا بِاللَّهِ عَلَيْهِ تَوَكَّلْتُ وَإِلَيْهِ أَنِيبُ

صَدَقَ اللَّهُ الْعَظِيمُ

Python Day2

- Print statement
- Variables and Datatypes
- Operators
- Python Input, Output and Import

The first part in Python Day No.2



Print Statement

- How to print Number, word, string on screen

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

- Print Number

```
In [1]: print(2)
```

2

```
In [2]: print(5.5)
```

5.5

- Print math operation

```
In [3]: print(1+2)
```

3

- Print word or string by single quotation

```
In [4]: print('Abdelmoty')
```

Abdelmoty

```
In [5]: print('1+2')
```

1+2

- Error in print word by single quotation

```
In [2]: print('abdelmoty's course')
```

```
File "<ipython-input-2-def112331b40>", line 1
print('abdelmoty's course')
      ^
```

SyntaxError: invalid syntax

- Print word or string by double quotation

```
In [3]: print("abdelmoty's course")
```

abdelmoty's course

```
In [4]: print('abdelmoty said"Python is excellent"')
         abdelmoty said"Python is excellent"
```

➤ **Print word or string by triple quotation**

```
In [8]: print(' abdelmoty's course is good')
```

abdelmoty's course is good

```
In [9]: print("""abdelmoty's "course" is good """)
```

abdelmoty's "course" is good

Notes on print statement's

[illegible]



Variables

Variables are the names that reference values stored in memory based on the data type of a variable. They are called “variables” because they may reference different values. The value referenced by a variable may vary, that’s why we call them “variables”. (vary: تتغير) the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals, or characters in these variables.

The statement for assigning a value to a variable is called an assignment statement. In Python, the equal sign (=) is used as the assignment operator. The syntax for assignment statements is as follows:

```
variable = expression
```

Here are some examples of assignment statements:

```
x = 1
```

```
height = 5.5
```

```
y = 4 * (5 - 3) + 2 / 4
```

```
z = x + y
```

```
squareArea = height * height
```

As we see, an expression represents a computation involving values, variables, and operators that evaluate to a value.

You can use a variable in an expression. A variable can also be used in both sides of the operator. For example:

```
x = 1
```

```
x = x + 1
```

- In this assignment statement, the result of $x + 1$ is assigned to x . If x is 1 before the statement is executed, then it becomes 2 after the statement is executed.
- In mathematics, $x = 2 * x + 1$ denotes an equation. However, in Python, $x = 2 * x + 1$ is an assignment statement that evaluates the expression $2 * x + 1$ and assigns the result to x .
- To assign a value to a variable, you must place the variable name to the left of the assignment operator. Thus, the following statement is wrong:

```
1 = x # Wrong
```

If a value is assigned to multiple variables, you can use a syntax called multiple assignment:

```
i = j = k = 1
```

which is equivalent to:

```
k = 1
```

```
j = k
```

```
i = j
```

A variable must be declared and assigned a value before it can be used. For example, the following code is wrong:

```
j = i + 1
```

If we tried to execute the previous example we will get the following error:

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

To fix it, we may write the code like this:

```
i = 0
j = i + 1
```



Assigning Values to Variables

Python variables do not need explicit declaration to reserve memory space. The declaration happens automatically when you assign a value to a variable.



Multiple Assignment:

Python allows you to assign a single value to several variables simultaneously. For example:

Example 1:

```
a = b = c = 1
```

Here, an integer object is created with the value 1, and all three variables are assigned to the same memory location.

```
In [11]: print(id(a))
1543248832
```

```
In [12]: print(id(b))
1543248832
```

```
In [13]: print(id(c))
1543248832
```

You can also assign multiple values to multiple variables. For example:

Example 2:

```
a, b, c = 1, 2, "IUG"
```

Identifiers are the names that identify the elements such as classes, methods, and variables in a program, all identifiers must obey the following rules:

- Identifiers must contain at least one character.
- The first character must be an alphabetic letter (upper or lower case) or the underscore (_)
- The remaining characters (if any) may be alphabetic characters (upper or lower case), the underscore, or a digit
- No other characters (including spaces) are permitted in identifiers.
- A reserved word (Python Keywords) cannot be used as an identifier.

and	del	from	None	try
as	elif	global	nonlocal	True
assert	else	if	not	while
break	except	import	or	with
class	False	in	pass	yield
continue	finally	is	raise	
def	for	lambda	return	

Notes on Variables

[illegible]



Standard Data Types:

Python has various standard data types that are used to define the operations possible on them and the storage method for each of them. Python has 6 standard data types:

1. Python Numbers

2. Python List
3. Python Tuple
- 4. Python Strings**
5. Python Set
6. Python Dictionary

4. Python Strings

➤ Conversion between data types

*In this lab we will take just about Numbers and String



Python Numbers:

Number data types store numeric values. Number objects are created when you assign a value to them.

For example:

Example 3

```
var1 = 10
var2 = 1.5
var3 = 2.5-1j
```

Python supports four different numerical types:

- **int (signed integers)**
- **long (long integers) (not used)**
- **float (floating point real values)**
- **complex (complex numbers)**

int	long	float	complex
10	51924361L	15.20	3.14j
-786	0xDEFA BCECBDAECBFBAEI	32.3+e18	4.53e-7j

- ✓ A complex number consists of an ordered pair of real floating-point numbers denoted by $x + yj$, where x is the real part and b is the imaginary part of the complex number.

[illegible][illegible]


```
In [17]: x=10  
  
In [18]: type(x)  
Out[18]: int
```

```
In [19]: y=4  
  
In [20]: z=5.6  
  
In [21]: w=5+7j  
  
In [22]: type(x)  
Out[22]: int  
  
In [25]: type(y)  
Out[25]: int  
  
In [26]: type(z)  
Out[26]: float  
  
In [27]: type(w)  
Out[27]: complex
```

Multiple Assignments in Python

In python more than one variable can be defined for different types of data in the same order at the same time.

```
In [28]: x,y,z=1.2,5+4j,3  
  
In [29]: type(x)  
Out[29]: float  
  
In [30]: type(y)  
Out[30]: complex  
  
In [31]: type(z)  
Out[31]: int
```

Python String

- Strings in Python are identified as a contiguous set of characters represented in the quotation marks either pairs of single or double quotes.

```
In [32]: w='Abdelmoty :)'  
  
In [33]: type(w)  
Out[33]: str
```

```
In [34]: H="Abdelmoty Abdeen"  
  
In [35]: print(H)  
Abdelmoty Abdeen  
  
In [36]: type(H)  
Out[36]: str
```

- Subsets of strings can be taken using the slice operator ([] and [:]) with indexes starting at 0 in the beginning of the string.

```
In [1]: a="Eng.Abdelmoty Abdeen"
In [2]: print(a)
Eng.Abdelmoty Abdeen
In [3]: type(a)
Out[3]: str
In [4]: print(a[0])
E
In [5]: print(a[2-6])
d
In [7]: print(a[2:5])
g.A
In [8]: print(a[2:])
g.Abdelmoty Abdeen
```

- The plus (+) sign is the string concatenation operator.

```
In [9]: print(a+a)
Eng.Abdelmoty AbdeenEng.Abdelmoty Abdeen
In [11]: print(a+' is Lecturer in KKU')
Eng.Abdelmoty Abdeen is Lecturer in KKU
```

- The asterisk (*) is the repetition operator.

```
In [13]: print(a*2)
Eng.Abdelmoty AbdeenEng.Abdelmoty Abdeen
In [15]: print(a*4)
Eng.Abdelmoty AbdeenEng.Abdelmoty AbdeenEng.Abdelmoty AbdeenEng.Abdelmoty Abdeen
```

- The command of len (str) is the return the value length of string.

```
In [19]: print(len(a))
20
```

Conversion functions:

- int () : convert string containing an integer number to an integer .

```
In [20]: a="34"
In [21]: type(a)
Out[21]: str
In [22]: x=int(a)
In [23]: type(x)
Out[23]: int
```

- float () :convert string containing a floating point number to a floating point number .

```
In [28]: a="2.5"
In [29]: type(a)
Out[29]: str
In [30]: x=float(a)
In [31]: type(x)
Out[31]: float
```

- `str()` :convert int or float number to a string point.

```
In [32]: x=5
In [33]: type(x)
Out[33]: int
In [34]: a=str(x)
In [35]: type(a)
Out[35]: str
```

Notes on Data Types

[illegible]

The Second part in Python Day No.2



What are operators in python?

Operators are special symbols in Python that carry out arithmetic or logical computation. The value that the operator operates on is called the operand.

```
In [17]: 2+3
Out[17]: 5
```

Here, `+` is the operator that performs addition. `2` and `3` are the operands and `5` is the output of the operation.

Python Arithmetic Operators:

Arithmetic operators are used to perform mathematical operations like addition, subtraction, multiplication etc - Assume variable `a` holds 2 and variable `b` holds 5, then:

Arithmetic operators in Python		
Operator	Meaning	Example
<code>+</code>	Add two operands or unary plus	<code>x + y +2</code>
<code>-</code>	Subtract right operand from the left or unary minus	<code>x - y -2</code>
<code>*</code>	Multiply two operands	<code>x * y</code>
<code>/</code>	Divide left operand by the right one (always results into float)	<code>x / y</code>
<code>%</code>	Modulus - remainder of the division of left operand by the right	<code>x % y</code> (remainder of <code>x/y</code>)
<code>//</code>	Floor division - division that results into whole number adjusted to the left in the number line	<code>x // y</code>
<code>**</code>	Exponent - left operand raised to the power of right	<code>x**y</code> (<code>x</code> to the power <code>y</code>)

```
In [1]: 5+2
Out[1]: 7

In [2]: 4-8
Out[2]: -4

In [3]: 3*9
Out[3]: 27

In [4]: 5/5
Out[4]: 1.0

In [5]: 8/9
Out[5]: 0.8888888888888888
```

```
In [9]: 8%2
Out[9]: 0

In [10]: 9%2
Out[10]: 1
```

```
In [6]: 2**3
Out[6]: 8

In [7]: 4**0.5
Out[7]: 2.0
```

```
In [11]: 9/2
Out[11]: 4.5

In [12]: 9//2
Out[12]: 4
```

Python Mathematical Operations

```
In [13]: a,b,c=5,2.5,4+5j
```

```
In [14]: a+b+c
```

```
Out[14]: (11.5+5j)
```

```
In [15]: a-b+c
```

```
Out[15]: (6.5+5j)
```

```
In [16]: d=a+b+c
```

```
print(d)
```

```
(11.5+5j)
```

Comparison operators

Comparison operators are used to compare values. It either returns `True` or `False` according to the condition.

Comparison operators in Python		
Operator	Meaning	Example
>	Greater than - True if left operand is greater than the right	<code>x > y</code>
<	Less than - True if left operand is less than the right	<code>x < y</code>
==	Equal to - True if both operands are equal	<code>x == y</code>
!=	Not equal to - True if operands are not equal	<code>x != y</code>
>=	Greater than or equal to - True if left operand is greater than or equal to the right	<code>x >= y</code>
<=	Less than or equal to - True if left operand is less than or equal to the right	<code>x <= y</code>

```
In [18]: x = 10
         y = 12

# Output: x > y is False
print('x > y is',x>y)

# Output: x < y is True
print('x < y is',x<y)

# Output: x == y is False
print('x == y is',x==y)

# Output: x != y is True
print('x != y is',x!=y)

# Output: x >= y is False
print('x >= y is',x>=y)

# Output: x <= y is True
print('x <= y is',x<=y)

x > y is False
x < y is True
x == y is False
x != y is True
x >= y is False
x <= y is True
```

Logical operators

Logical operators are the and, or, not operators.

Logical operators in Python		
Operator	Meaning	Example
and	True if both the operands are true	<code>x and y</code>
or	True if either of the operands is true	<code>x or y</code>
not	True if operand is false (complements the operand)	<code>not x</code>

```
In [19]: x = True
y = False

# Output: x and y is False
print('x and y is',x and y)

# Output: x or y is True
print('x or y is',x or y)

# Output: not x is False
print('not x is',not x)

x and y is False
x or y is True
not x is False
```

Bitwise operators

Bitwise operators act on operands as if they were string of binary digits. It operates bit by bit, hence the name.

For example, 2 is 10 in binary and 7 is 111.

In the table below: Let x = 10 (0000 1010 in binary) and y = 4 (0000 0100 in binary)

Bitwise operators in Python		
Operator	Meaning	Example
&	Bitwise AND	x & y = 0 (0000 0000)
	Bitwise OR	x y = 14 (0000 1110)
~	Bitwise NOT	~x = -11 (1111 0101)
^	Bitwise XOR	x ^ y = 14 (0000 1110)
>>	Bitwise right shift	x >> 2 = 2 (0000 0010)
<<	Bitwise left shift	x << 2 = 40 (0010 1000)

Python Assignment Operators

Assignment operators are used in Python to assign values to variables.

a = 5 is a simple assignment operator that assigns the value 5 on the right to the variable a on the left.

There are various compound operators in Python like a += 5 that adds to the variable and later assigns the same. It is equivalent to a = a + 5.

Assignment operators in Python		
Operator	Example	Equivalent to
=	x = 5	x = 5
+=	x += 5	x = x + 5
-=	x -= 5	x = x - 5
*=	x *= 5	x = x * 5
/=	x /= 5	x = x / 5
%=	x %= 5	x = x % 5
//=	x //= 5	x = x // 5
**=	x **= 5	x = x ** 5
&=	x &= 5	x = x & 5
=	x = 5	x = x 5
^=	x ^= 5	x = x ^ 5
>>=	x >>= 5	x = x >> 5
<<=	x <<= 5	x = x << 5

Special operators

Python language offers some special type of operators like the identity operator or the membership operator. They are described below with examples.

Identity operators

is and is not are the identity operators in Python. They are used to check if two values (or variables) are located on the same part of the memory. Two variables that are equal does not imply that they are identical.

Identity operators in Python		
Operator	Meaning	Example
is	True if the operands are identical (refer to the same object)	x is True
is not	True if the operands are not identical (do not refer to the same object)	x is not True

```
In [20]: x1 = 5
          y1 = 5
          x2 = 'Hello'
          y2 = 'Hello'
          x3 = [1,2,3]
          y3 = [1,2,3]

          # Output: False
          print(x1 is not y1)

          # Output: True
          print(x2 is y2)

          # Output: False
          print(x3 is y3)

False
True
False
```

Membership operators

in and not in are the membership operators in Python. They are used to test whether a value or variable is found in a sequence (string, list, tuple, set and dictionary).

In a dictionary we can only test for presence of key, not the value.

Operator	Meaning	Example
in	True if value/variable is found in the sequence	5 in x
not in	True if value/variable is not found in the sequence	5 not in x

```
In [21]: x = 'Hello world'
          y = {1:'a',2:'b'}

          # Output: True
          print('H' in x)

          # Output: True
          print('hello' not in x)

          # Output: True
          print(1 in y)

          # Output: False
          print('a' in y)

True
True
True
False
```

Python Output Using print() function

This part focuses on two built-in functions print() and input() to perform I/O task in Python. Also, you will learn to import modules and use them in your program.

Python Output (print statement)

Python Input

Python Import

Python Input

We can use the input function to read values from user. The input function prompts the user to enter a value and returns the value to be stored in a variable. The following example illustrates the input function:

```
In [23]: num = input('Enter a number: ')
Enter a number: 85
```

```
In [24]: int('2+3')

-----
ValueError                                Traceback (most recent call last)
<ipython-input-24-506ddc2101a0> in <module>()
----> 1 int('2+3')

ValueError: invalid literal for int() with base 10: '2+3'
```

```
In [25]: eval('2+3')
Out[25]: 5
```

```
In [26]: num1=input('Enter the value of num1=')
num2=input('Enter the value of num2=')

Enter the value of num1=5
Enter the value of num2=6.5

In [28]: num1=int(num1)
num2=float(num2)
res=num1+num2
print('the result of nim1+num2',res)

the result of nim1+num2 11.5
```

```
In [29]: 5+2*3-4/2
Out[29]: 9.0

In [30]: (5+2)*(3-4)/2
Out[30]: -3.5
```

Python Import

When our program grows bigger, it is a good idea to break it into different modules.

A module is a file containing Python definitions and statements. Python modules have a filename and end with the extension .py.

Definitions inside a module can be imported to another module or the interactive interpreter in Python. We use the import keyword to do this. For example, we can import the math module by typing in import math.

```
In [39]: import math
print(math.pi)

3.141592653589793
```

Now all the definitions inside `math` module are available in our scope. We can also import some specific attributes and functions only, using the `from` keyword. For example:

```
In [40]: from math import pi
In [41]: pi
Out[41]: 3.141592653589793
```

```
In [43]: import sys
print(sys.path)

['', 'C:\\Users\\AbulMoaty\\Anaconda3\\python36.zip', 'C:\\Users\\AbulMoaty\\Anaconda3\\DLLs', 'C:\\Users\\AbulMoaty\\Anaconda3\\lib', 'C:\\Users\\AbulMoaty\\Anaconda3', 'C:\\Users\\AbulMoaty\\Anaconda3\\lib\\site-packages', 'C:\\Users\\AbulMoaty\\Anaconda3\\lib\\site-packages\\Babel-2.5.0-py3.6.egg', 'C:\\Users\\AbulMoaty\\Anaconda3\\lib\\site-packages\\win32', 'C:\\Users\\AbulMoaty\\Anaconda3\\lib\\site-packages\\win32\\lib', 'C:\\Users\\AbulMoaty\\Anaconda3\\lib\\site-packages\\Pythonwin', 'C:\\Users\\AbulMoaty\\Anaconda3\\lib\\site-packages\\IPython\\extensions', 'C:\\Users\\AbulMoaty\\.ipython']
```

Notes on Operators

Python Day 2 Home Work

1. Write a program that reads a Celsius degree from the console and converts it to Fahrenheit and displays the result. The formula for the conversion is as follows:

$$\text{fahrenheit} = (9 / 5) * \text{celsius} + 32$$

2. Write a program that converts miles into kilometers. Use the following algorithm:
 - Use a variable named **miles** that read value from keyboard.
 - Multiply miles by 1.609 and assign it to a variable named kilometers.
 - Print the value of kilometers.
3. Assume a = 22 , and b = 4 .. write python program that finds the value of:
 - a+b
 - a-b
 - a*b
 - a/b
 - a%b
 - a**b
 - a//b
4. Assume str = your own name , then print :
 - All the string .
 - The last character in the first name .
 - The first name only .
 - The last name only .
 - The name concatenated with " hello" .
 - The name repeated 3 times .
 - The length of str .
 - The type of str .
5. Write a program that let the user enter two numbers , then calculate their addition and subtraction and print them .
6. Write a program that reads in the radius and length of a cylinder and computes the area and volume. Here is a sample output:

```
Enter the radius and length of a cylinder: 7.5, 15
Area is 176.7144
Volume is 2650.7166
```

7. Classify the following identifiers as either valid or invalid Python identifier:

basem	Sa	if	2k	my app
Test	\$5	x	+9	_a
x+y	#44	width	my_app	_
A#	myApp	If	my-app	_5

Python Day No.2 Notes

[illegible]

[illegible]