1st practical work
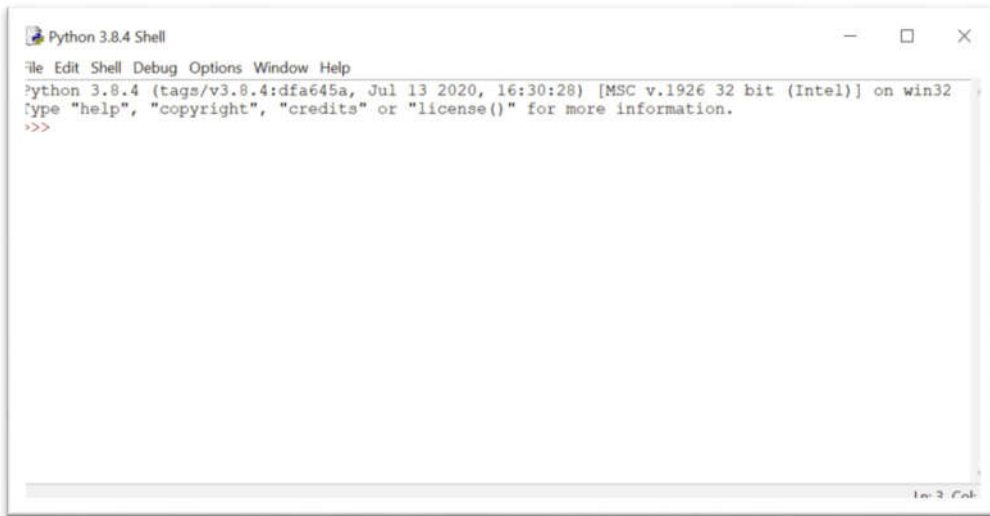
All students are invited to do the following:

- Install the latest version of Python 3 on your computer: https://www.python.org/downloads/windows/
  Run the Installer

- Open **IDLE**, Python's built-in **I**ntegrated **D**evelopment and **L**earning **E**nvironment.
  1-Click the Start menu and locate the Python folder.
  2- Open the folder and select IDLE

Write a Python Program

IDLE's interactive window contains a **Python shell**, which is a textual user interface used to interact with the Python language. You can type a bit of Python code into the interactive window and press Enter to immediately see the results.

The >>> symbol in the last line is called the **prompt**. This is where you'll write your code

Note : for a program (a set of instructions) in python Idle : click: *file--New*. if you want to run your program click : **Run -- Run Module** from the menu in editor window.

1- Comments:

Comments are used to explain the code and are ignored by the Python interpreter. You can use the **#** symbol for single-line comments.

```
>>>  #  Nothing to do , it's just a comment                    # ...............................
```

2- Shell as a calculator

a) Type and explain the result of these operations?

```
>>> 5 + 2                # ..................................................
>>> 10 - 2               # ..................................................
>>> 6 * 2                 # ..................................................
>>> 8 / 3                # ..................................................
>>> 10 // 3              # ..................................................
>>> 5 % 2                # ..................................................
>>> 5 ** 2               # ..................................................
>>> 9 ** 0.5             # ..................................................
>>> 5  and  0            # ..................................................
>>> 1  or  1             # ..................................................
>>> (5 + 1 * 3) +2       # ..................................................
>>> 5 + 3 ** 2 * 2       # ..................................................
>>> 5 + 3 +*2 * 2        # ..................................................
>>> 5 + 3 ** 2 / 0       # ..................................................
```

b) Command Print.

To print strings to console or echo some data to console output, use Python inbuilt print() function. *Type and explain the result ?*

```
>>> print('hello world')                          # ..................................................
>>> print ('5 + 3 ** 2 ',  5 + 3 ** 2)            # ..................................................
>>> print ('5 + 3 ** 2 ',  5 + 3 ** 2, sep = '=')  # ..................................................
>>> print('hello world', end='.')                  # ..................................................
>>> print('hello', end='\n' ); print('world', end='.')   # ..................................................
```

what is the role of semi colon(;)?

3- built-in Functions:

some bultin functions in python. *Type the following and explain each result ?*

```
>>> abs(-1.5)                # ..................................................
>>> bin(24)                  # ..................................................
>>> chr(65) ;ord('m')        # ..................................................
>>> chr(ord('m')+1)          # ..................................................
>>> chr(ord('B')-1)          # ..................................................
>>> str(65)                  # ..................................................
>>> min([-1.5,2,3.1,10,12])  # ..................................................
>>> max([-1.5,2,3.1,10,12])  # ..................................................
>>> sum([-1.5,2,3.1,10,12])  # ..................................................
>>> format(0.000152,'E')     # ..................................................
>>> format(0. 152,'%')       # ..................................................
>>> int(3.41)                # ..................................................
```

3- Using the editor :   Type, save and run this program. What does this program do?

```python
import time
seconds = 10
while seconds > 0:
        print(f"Time remaining: {seconds} seconds")
        time.sleep(1)   # Delay for 1 second
        seconds -= 1

print("Time's up!")
```

Primitives Variables

In Python, **variables** are names that can be assigned and store (in memory) a value and then used to refer to that value throughout your code.

Python variables do not need explicit declaration to reserve memory space. The declaration happens automatically when you assign a value to a variable. The equal sign (=) is used to assign values to variables.

1- Create variables : *Give the answer of the following :*

- Create a variable named **Var1** and assign the value **"Hello"** to it

  ....

- Create a variable named **X1** and assign the **Real value 2** to it

  ....

- explain the purpose of *type* function using this command :   *type(2.5)*

  ....

- x = 5 ; print (x)

  ....

- x = 5 ; print(x , type (x) , sep = ' is type of ')

  ....

- z , x = 5 , 0 ; print(z ,'   ',x)

  ....

- x = int (20) ; print (type (x))

  ....

- y = 20.5 ; print (y , type (y), sep = ' is type of ')

  ....

- y = float (20.5) ; print (y , type (y), sep = ' is type of ')

  ....

- S= "Hello, World" ; print (S , type (S), sep = ' is type of ')

  ....

- L= True ; print (L , type (L), sep = ' is type of ')

  ....

- L = str ("True") ; print (type (L))

  ....

- a=10 ; b= a + 2; c = a + b ; print (a , b, c, sep='   '))

  ....

- a=10 ; a= a + 1; print (a)

  ....

- a=5; b=10 ; c=3; b+=1 ; c**=2; a-=1;print (a, b, c, sep =(' '))

  ....

- a=5; b=10 ; c=a; a =b ; b = c ; print ('a become ', a ,' and b = ', b)

  ....

- a = 8.5; b =1.2 ; a = a + b; b = a-b ; a = a - b; print ('a become ', a ,' and b = ', b)

  ....

- a = 6; b =(a+3)*2 ; a = a + b+4; b = (b+2)*3 ; print ('a = ', a ,' and b = ', b)

  ....

Python Basic Input and Output

In Python, we use the **print()** function to output data to the screen. Sometimes we might want to take input from the user. We can do so by using the **input()** function.

Note : Python takes all the input as a string input by default.

1- Give the outputs of the following instructions:

*   print(1, 1, 2024, sep='/', end='.')

    ....

*   x = 3 ; y = 12; mul = x * y ; print('The value of x is {} and y is {}'.format(x, y))

    ....

*   print('{2} is the multiplication of {0} and {1}'.format(x, y, mul))

    ....

*   add = x + y ; print("The new number after addition is %d" % add)

    ....

*   sub = x - y ; print(f' The new number after subtraction is {sub} ')

    ....

*   name = input("Enter your first name: ") ;

    ....

*   age = **int** (input ("Enter your age: ")); n = age + 1; print (name + " your age will be next year!   " , n)

    ....

*   S1,S2 = input ("Enter two values separated by space character").split(); print(" S1=   " , S1, " S2= ",S2)

    ....

Writing scripts:

Use the editor provided by (you can use any third party python IDE) to write the following scripts:

1. Write a python script that takes two variables as input then exchanges their values. Example: when a = 2 and b = 5, the script will give a = 5 and b = 2.

2. Write a script that receives two **integers**, then writes their sum and their difference

3. Write the Cylinder program, which calculates and displays the volume of a cylinder after entering its radius R and its height H.

4. Write a program that calculates and displays the absolute value of a real number x entered by the user (use the built-in function abs(x)).

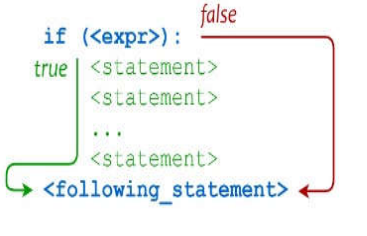## Python Conditions and If .. elif ... else statements

Condition: Python supports the usual logical conditions from mathematics (== ,   != ,   >,   <= ,   >=, <) .
Note: we can create more complex logical conditions. We can use the logical "AND" (**and**), logical "OR" (**or**), logical negation (**not**) and brackets (**( )**).

Construct the truth table for the following expressions:

- a = 2; b = 6; print (((a == b) or (a != b)) and (b >= a))   ....
- print (not ((a != b) and (a == b) or (b >= a)))   ....

If , elif , else statements: here is the complete syntax with examples

	```if Condition: STATEMENT1 else: STATEMENT2```	```if Condition1: STATEMENT1 elif: Condition2: STATEMENT2 else: STATEMENT3```

Type and conclude?		
```a=float(input('a= ')) if a > 0: print('a is positive')```	```a=float(input('a= ')) if a > 0: print('a is positive') else: print('a is <= 0 ')```	```a=float(input('a= ')) if a > 0: print('a is positive') elif a=0: print('a is null ') else print('a is Negative ')```

Writing scripts:

5. Write a script that asks the user for two real numbers and then informs them if their product is negative, positive or null. you should not calculate the product of the two numbers

6. Write a Python program that determines whether a given number (accepted from the user) is even or odd, and prints an appropriate message to the user

7. rewrite the script in -4- without using the built-in function

8. Write a program that makes it possible to discern a mention for a student based on the average mark of their grades:
    - "Very good" for an average mark between 16 and 20 (16<= average <=20).
    - "Good" for an average mark between 14 and 16 (14<= average <16).
    - "Fairly good" for an average mark between 12 and 14 (12<= average <14).
    - "Fair" for an average mark between 10 and 12 (10<= average <12)

9. Write a script that input two real numbers and calculates their sum if they are positives or their product if they are negatives.

10. Residents of a given city pay tax according to the following rules:
    - Men over 20 pay tax
    - Women pay tax if they are between 18 and 35 years old
    - The others do not pay tax

    Write a script that asks for the age and gender ('M' or 'W') of a resident and displays whether they are taxable.

11. Write a script called CALCULATOR, which reads in this order:
    1- The first real.
    2- Arithmetic operator (+, -, *, /).
    3- The second real.

    Each valid operator corresponds to a calculation that should be performed and display result or an error message, if not applicable

Python iterative statements (Loops)

Loops: as all programming language, Python supports the usual Loops to fulfil the looping needs: ***while*** and ***for*** loops.

The while loop:	

while **(condition):**
    Statements
	while **(condition):**
	    Statements
	else:
	    Statements

What does the following programs output?		

counter = 0
while counter < 10:
    counter = counter + 1
    print("Python Loops")
print("iteration count ", Counter)
	counter = 0
	while counter != 10:
	    counter = counter + 1
	    print(counter)
		Nbr = int (input("Entre a negative number:"))
		
		while nbr >= 0 :
		
		    nbr = int (input (" a negative number !!?: "))
		
		print (" you have entered a negative number ")

counter = 0
while counter < 10:
    print("Python Loops")
    counter = counter + 1
print("iteration count ", Counter)
	counter = 0
	while counter == 10:
	    counter = counter + 1
	else:
	    print("Python Loops")
	print("iteration count ", Counter)
		i = 1
		while i < 6:
		    print(i)
		    if i == 3:
		        break # exit from the loop
		    i += 1

The For loop:		

General Syntax:
 	
for looping variable in sequence:
    code bloc
	for variable **in** range(n, m, p) :
	        statements
	
	n: Start value
	m: end value
	p: step value
		for variable **in** range(n, m, p) :
		        statements
		else :
		        statements

What does the following programs output?				

for i in **range**(-1, 5, 2):
    print(i, end=", ")
	for i in **range**(1 , 10):
	    print("Iteration", i)
		for i in **range**(10):
		    print("Iteration", i)
			for i in **range**(10,1):
			    print("Iteration", i)
			else :
			    print("code else")
				for **c** in **"hello world"**:
				    print(c)

Writing scripts (use the both while and for in each script):

12. Write a program that displays the even numbers between 0 and 100

13. Write a program that calculates the factorial of a user-entered integer without using the factorial built-in function

14. Considering the Series defined by: *S = 1 + 2 + 3 + ... + n*. Write a program that calculates S (***n*** is a user-entered integer value).

15. Same question in 14 with $S = 1 + \frac{x}{1} + \frac{x^2}{2} + \frac{x^3}{3} + \cdots + \frac{x^n}{n}$ (x is a user-entered real value)

16. Calculate the Nth term UN of the FIBONACCI sequence which is given by the recurrence relation: U1=1, U2=1, UN = UN -1 + UN -2 (for N > 2). N is a user-entered integer value.

17. Create a simple calculator for additions! Repeatedly ask the user to enter numbers. Each time that the user provides input, this input is first converted to ***float*** and then added to a running total (which should start from 0). When the user enters **'add'**, the loop stops and the running total is printed out.

Functions and modules in Python

A function is a block of code, which only runs when it is called. You can pass data, known as parameters, into a function. A function can return data as a result using the keyword "return".

Fig: Function definition
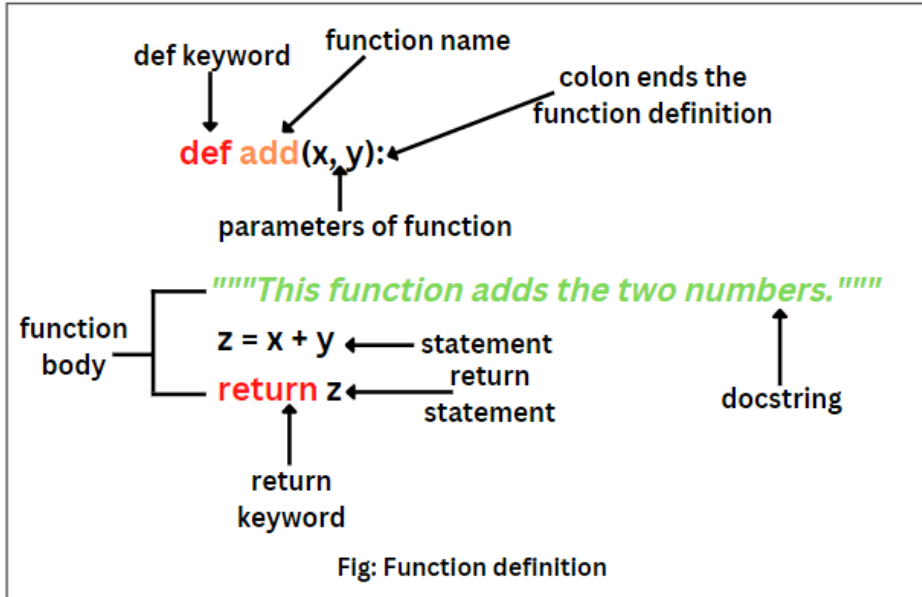
If you have experience in C/C++ or Java then you must be thinking about the *return type* of the function and *data type* of arguments. That is possible in Python as well. Type and run this example

18. Write a function in Python to check whether the number passed as an argument to the function is even or odd

19. Use the function in 18 to display all even number that are in range(n,m) (n and m are two natural numbers given by user)

20. Write a Python function to calculate the factorial of a number (a non-negative integer). The function accepts the number as an argument.

21. Write a Python function to find the maximum of two numbers.

22. Write a Python function to find the maximum of four numbers by using the previous function

23. Write a Python program to reverse a string. *Sample String* : "hello" *Output* String : "olleh"

24. Write a Python function that checks whether a passed string is a palindrome or not.
    Note: A palindrome is a word, phrase, or sequence that reads the same backward as forward, e.g., madam or nursesrun.

Lists and arrays in Python

Arrays and lists are used to store multiple values in a single variable, instead of declaring separate variables for each value. In Python, a list is denoted by square brackets [ ]. It can hold elements of different data types, such as numbers, strings, or even other lists. Lists are mutable, which means you can modify them by adding, removing, or changing elements.

1. Consider the list 'fruits' and 'numbers' below

fruits = ["apple", "banana", "orange", "grape"]
numbers = [20, 10, 0, -5]
Give the output of:
- >>> print(fruits)

  ……………………………………………………………………………………….

- >>> print(numbers) , print(type(numbers))

  …………………………………………………………………..…………………………….

- Output the 3rd element from the given lists to the console.
  >>> …………………………………………………………………………………..

- Update the 3rd element in the numbers list with 5
  >>> …………………………………………………………………………………..

- Create a string array for the following values "Monday", "Tuesday", and "Wednesday
  >>> …………………………………………………………………………………..

2. What command line should be entered to transition from the left list to the right list?

Left		Right
`L8=[10,20,30,40]`	➡ ➡	`L8=[10,20,30,50]`
`L9=[0.1,0.3,0.1,0.2,0.2]`	➡ ➡	`L9=[0.1,0.3,0.1,0.2,0.2,0.4]`
`L10=[1,1,0]`	➡ ➡	`L10=[1,5,1,0]`
`L11=[92,95,94,97,93,92,98]`	➡ ➡	`L11=[92,92,93,94,95,97,98]`
`L12=[-1,17,23,48,5]`	➡ ➡	`L12=[-1,23,48,5]`
`L12=[-1,17,23,48,5]`	➡ ➡	`L12=[-1,48,5]`

3. Give the output of each instruction?
   >>> *L1 = [3*n +1   for n in range(11)]*

   ………………………………………………………………………………………

   >>> *L2 = [2*n   for n in range(50)]*

   ………………………………………………………………………………………

   >>> …………………………………………………………..?
     L3 = [1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31,33,34,35]

   >>> *L4 = [[a for a in range(3)] for n in range(4)]*

     ………………………………………………………………………………

4. Give the python Code for generate the matrix below ?
   L5= [[0, 2, 4, 6, 8], [0, 2, 4, 6, 8], [0, 2, 4, 6, 8], [0, 2, 4, 6, 8], [0, 2, 4, 6, 8]]
   - calculate the sum of all the numbers
   - Make diagonal as 0
   - Calculate the max and min of this matrix without using the built-in ***min*** and ***max***