**Scripts vs Modules:** A plain text file containing Python code that is intended to be directly executed by the user is usually called **script**. On the other hand, a plain text file, which contains Python code that is designed to be imported and used from another Python file, is called **module**. So, the main difference between a module and a script is that **modules are meant to be imported**, while **scripts are made to be directly executed**.

**Scipy.io:** SciPy in Python is an open-source library, which is a collection of many mathematical algorithms, used **for solving mathematical, scientific, engineering, and technical problems**. It allows users to manipulate the data and visualize the data using a wide range of high-level Python commands. SciPy is built on the Python NumPy extension.

**Loadmat (Load data from MAT-file):** The function loadmat **loads all variables stored in the MAT-file into a simple Python data structure, using only Python's dict and list objects**.

**Savemat:** It exports the data into **MATLAB** file format specially **NumPy** arrays to **“.** **mat”** file.

**Import and from statement:** The **import statement** allows you to import all the functions from a module into your code. Often, though, you'll only want to import a few functions, or just one. If this is the case, you can use the **from statement**. Python code uses the import statement to load and make code accessible. MATLAB automatically loads Python when you type py. in front of the module name and function name.

**dit:** Python package for information theory.

**Time:** The time () function returns the number of seconds passed since epoch. For Unix system, January 1, 1970, 00:00:00 at **UTC** is epoch (the point where time begins).

import time

seconds = time.time()

print("Seconds since epoch =", seconds)

**Comment:** Python provides three kinds of comments including block comment, inline comment, and documentation string.

To create a block comment, you start with a single hash sign (#) followed by a single space and a text string. For example:

*# increase price by 5%*

price = price \* 1.05

Similar to a block comment, an inline comment begins with a single hash sign (#) and is followed by a space and a text string.

The following example illustrates an inline comment:

salary = salary \* 1.02 *# increase salary by 2%*

Documentation strings is called docstrings. Python provides two kinds of docstrings: one-line docstrings and multi-line docstrings.

### **1) One-line docstrings**

As its name implies, a one-line docstring fits one line. A one-line docstring begins with triple quotes (""") and also ends with triple quotes ("""). Also, there won’t be any blank line either before or after the one-line docstring.

The following example illustrates a one-line docstring in the quicksort() function:

def quicksort():

""" sort the list using quicksort algorithm """

...

Code language: Python (python)

### **2) Multi-line docstrings**

Unlike a one-line docstring, a multi-line docstring can span multiple lines. A multi-line docstring also starts with triple quotes (""") and ends with triple quotes (""").

The following example shows you how to use multi-line docstrings:

def increase(salary, percentage, rating):

""" increase salary base on rating and percentage

rating 1 - 2 no increase

rating 3 - 4 increase 5%

rating 4 - 6 increase 10%

"""

**Sys (System-specific parameters and functions)**: This module provides access to some variables used or maintained by the interpreter and to functions that interact strongly with the interpreter. [**Command-line arguments**](https://www.geeksforgeeks.org/command-line-arguments-in-python/)are those which are passed during the calling of the program along with the calling statement. To achieve this using the **sys** module, the **sys** module provides a variable called **[sys.argv](https://www.geeksforgeeks.org/how-to-use-sys-argv-in-python/).**It’s main purposes are:

* It is a list of command-line arguments.
* len(sys.argv) provides the number of command-line arguments.
* sys.argv[0] is the name of the current Python script.

**Dictionaries(dict):** Dictionaries are **used to store data values in key:value pairs**. A dictionary is a collection which is ordered, changeable and do not allow duplicates. For example,

dict = {key1:value1, key2:value2, ... }.

**List:** A list is an ordered data structure with elements separated by a comma and enclosed within square brackets. It is basically a data storage or a memory space. The element of a list represents item or type of data. A list also may contain some lists as its element.

**Example: 1.** list1= [1,2,3,4,5,6]

**2.** list2= [‘python’, ‘is’, ‘awesome’]

**3.** list3= [‘python’, 1, ‘awesome’, 2]

4. list4= [‘python’, 1, ‘list1’, ‘list2’] etc.

**Sub-setting of List:** The index of element starts from 0 in list. So, for the

list3= [‘python’, ‘hello’, ‘awesome’, 2, ‘student’]

the indexes of the elements are 0,1,2,3 respectively.

We can select a specific element from the list in the following way:

List3[0] gives us 1st element python.

List3[1] gives us 2nd element hello.

List3[3] gives us 4th element 2.

List3[1:5] gives us 2nd – 4th elements (*i.e.,* hello, awesome, 2)

List3[-1] gives us last (5th) element student.

List3[-2] gives us the 2nd element from the last side (4th) element 2.

**Adding and Deleting List element:** A single element can be added with a list using append function. More than one element can be added using extend function. Append can also add a list with another list. Just look at the following ways:

If the list is,

list= [‘python’, 3, ‘awesome’]

then list.append(4) implies list= [‘python’, ‘3’, ‘awesome’,4]

list.extend(2,5) implies list= [‘python’, ‘3’, ‘awesome’,2,5]

list.append([6,7]) implies list= [‘python’, ‘3’, ‘awesome’,[6,7]]

on the other hand, we can delete element from a list in two ways. According to 1st way, use remove function and write that element in remove function which you want to remove from the list. According to 2nd way, use del keyword or pop() and write the index of that element which you want to delete.

list.remove(3) implies list= [‘python’, ‘awesome’]

del list[2] implies list= [‘python’, 3]

list.pop(2) implies list= [‘python’, 3]

**Looping over a list:** To print each element of a list, we can write for loop as follows:

If list= [‘python’, 3, ‘awesome’] then

for i in list:

print(i)

The outcomes are, python, 3, awesome.

**Vstack () & hstack () function:** The vstack() function is used to stack arrays in sequence vertically and the hstack() function is used to stack arrays in sequence horizontally.

Example: If a=np.array([1,2,3,4]) , b=np.array([5,6,7,8]) and c=np.array([9,12,10,11]) then

d= =np.vstack([a,b,d]) gives

[[ 1 2 3 4]

[ 5 6 7 8]

[ 9 12 10 11]]

And

d= =np.hstack([a,b,d]) gives [ 1 2 3 4 5 6 7 8 9 12 10 11].

**Encryption and Decryption**: **Encryption** is the process by which a readable message is converted to an unreadable form to prevent unauthorized parties from reading it. **Decryption** is the process of converting an encrypted message back to its original (readable) format. The original message is called the plaintext message.

**Secret Key Agreement:** A secret key (or “private key”) is a piece of information or a framework that is used to decrypt and encrypt messages. Key agreement refers to one form of key exchange in which two or more users execute a protocol to securely share a resultant key value. As for example, assume that **Alice**, **Bob**, and their opponent **Eve** know random variables X, Y,and Z, respectively, with joint probability distribution . **Alice** and **Bob** share no secret key initially, and they only communicate with each other in a public communication channel that is completely accessible to **Eve**. If **Alice** and **Bob** agree to converse publicly and generate bits which is completely independent of everything **Eve** has access to, then *S* is known as a secret key.