1) What is Python?

Python is an interpreted, high-level and general-purpose programming language.

General purpose means - it can be **used for** other types of programming and software development. That includes back end development, software development, data science and writing system scripts.

Python is widely used for – web development, software development, Games

**Why Python?**

* Python is compatible with different platforms like **Windows, Mac, Linux, Raspberry Pi,** etc.
* Python has a simple syntax as compared to other languages.
* Python allows a developer to write programs with fewer lines than some other programming languages.
* python runs on an interpreter system, means that the code can be executed as soon as it is written.
* Python can be described as a procedural way, an object-orientated way or a functional way.

### 2. What are the applications of Python?

Python is used in various software domains

* Web and Internet Development
* Games
* Scientific and computational applications
* Enterprise and business applications development
* GUI based desktop applications

Python's **SciPy** and **NumPy** helps in scientific and computational application development.

Python's **Tkinter** library supports to create a desktop based GUI applications.

### What are the advantages of Python?

* Interpreted
* Free and open source
* Object-oriented
* Readability
* High-Level Language
* Cross-platform  
  Interpreted: Python is an interpreted language. It does not require prior compilation of code and executes instructions directly.

Free and open source: It is an open source project which is publicly available to reuse. It can be downloaded free of cost.

Portable: Python programs can run on cross platforms without affecting its performance.

* Object-oriented: Python allows to implement the Object Oriented concepts to build application solution.

### What is PEP 8?

PEP 8 is defined as a document that helps us to provide the guidelines on how to write the Python code.

It stands for Python Enhancement Proposal, and its major task is to improve the readability and consistency of Python code.

### 5. What do you mean by Python literals?

a **literal** is a notation for representing a fixed value in source code.

**String Literals** : A string literal is a sequence of characters surrounded by quotes. We can use both single, double quotes for a string. Ex: "Aman", '12345'.

**Numeric Literals:** Python supports three types of numeric literals integer, float and complex. See the examples.

1. # Integer literal
2. a = 10
3. #Float Literal
4. b = 12.3
5. #Complex Literal
6. x = 3.14j

**Boolean Literals:** Boolean literals are used to denote boolean values. It contains either True or False.

1. # Boolean literal
2. isboolean = True

### 6. Explain Python Functions?

A function is a section of the program or a block of code that is written once and can be executed whenever required in the program.

There are two types of functions:

* Built-In Functions: copy(), len(), count() are the some built-in functions.
* User-defined Functions: Functions which are defined by a user known as user-defined functions.
* **def** function\_name(parameters list):
* #--- statements---
* **return** a\_value

### 7. What is zip() function in Python?

The **zip() function** takes iterables (can be zero or more), aggregates them in a tuple, and return it.

**Python's zip()** function is defined as **zip**(\*iterables) . The function takes in iterables as arguments and returns an iterator. This iterator generates a series of tuples containing elements from each iterable. **zip()** can accept any type of iterable, such as files, lists, tuples, dictionaries, sets, and so on.

1. zip(iterator1, iterator2, iterator3 ...)

test = zip() # referring a zip class

print('The type of an empty zip : ', type(test))

list1 = ['Alpha', 'Beta', 'Gamma', 'Sigma']

list2 = ['one', 'two', 'three', 'six']

test = zip(list1, list2) # zip the values

print('\nPrinting the values of zip')

for values in test:

print(values) # print each tuples

Output:

The type of an empty zip : <class 'zip'> Print the values of zip ('Alpha', 'one') ('Beta', 'two') ('Gamma', 'three') ('Sigma', 'six')

### 10. What is the difference between remove() function and del statement?

You can use the remove() function to delete a specific object in the list.

If you want to delete an object at a specific location (index) in the list, you can either use **del** or **pop**.

### 11. What is swapcase() function in the Python?

It is a string's function which converts all uppercase characters into lowercase and vice versa.

1. string = "IT IS IN LOWERCASE."
2. **print**(string.swapcase())
4. string = "it is in uppercase."
5. **print**(string.swapcase())

it is in lowercase.

IT IS IN UPPERCASE.

### 12. How to remove whitespaces from a string in Python?

To remove the whitespaces and trailing spaces from the string, Python providies strip([str]) built-in function. This function returns a copy of the string after removing whitespaces if present. Otherwise returns original string.

1. string = "  javatpoint "
2. string2 = "    javatpoint        "
3. string3 = "       javatpoint"
4. **print**(string)
5. **print**(string2)
6. **print**(string3)
7. **print**("After stripping all have placed in a sequence:")
8. **print**(string.strip())
9. **print**(string2.strip())
10. **print**(string3.strip())

javatpoint

javatpoint

javatpoint

After stripping all have placed in a sequence:

javatpoint

javatpoint

javatpoint

### 13. How to remove leading whitespaces from a string in the Python?

1. string = "  javatpoint "
2. **print**(string.lstrip())

o/p: javatpoint

### 14. Why do we use join() function in Python?

 It is concatenated with the elements of an iterable. It provides a flexible way to concatenate the strings.

1. str = "Rohan"
2. str2 = "ab"
3. # Calling function
4. str2 = str.join(str2)
5. # Displaying result
6. **print**(str2)

**Output:**

aRohanb

### 15) Give an example of shuffle() method?

Given an Array, the **method** returns an array with elements in random order.

1. **import** random
3. list = [12,25,15,65,58,14,5,];
4. **print**(list)
5. random.shuffle(list)
6. **print** ("Reshuffled list : \n",  list)

[12, 25, 15, 65, 58, 14, 5]

Reshuffled list :

[58, 15, 5, 65, 12, 14, 25]

### 16) What is the use of break statement?

It is used to terminate the execution of the current loop. Break always breaks the current execution and transfer control to outside the current block. If the block is in a loop, it exits from the loop, and if the break is in a nested loop, it exits from the innermost loop.

1. even = [2,4,6,8,10,11,12,14]
2. odd = 0
3. **for** val **in** even:
4. **if** val%2!=0:
5. odd = val
6. **break**
7. **print**(val)
8. **print**("odd value found",odd)

2

4

6

8

10

odd value found 11

### 17. What is tuple in Python?

A tuple is a built-in data collection type. It allows us to store values in a sequence.

**Tuple**. **Tuples** are used to store multiple items in a single variable.

It is immutable, so no change is reflected in the original data. It uses () brackets

1. # Declaring tuple
2. tup = (2,4,6,8)
3. # Displaying value
4. **print**(tup)
6. # Displaying Single value
7. **print**(tup[2])

(2, 4, 6, 8)

6

### 18. Which are the file related libraries/modules in Python?

The Python provides libraries/modules that enable you to manipulate text files and binary files on the file system. It helps to create files, update their contents, copy, and delete files. The libraries are os, os.path, and shutil.

Here, os and os.path - modules include a function for accessing the filesystem

while shutil - module enables you to copy and delete the files.

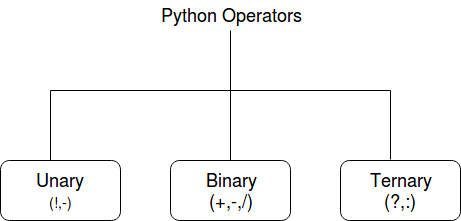
19. What are the different file processing modes supported by Python?

Python provides three modes to open files. The read-only, write-only, read-write and append mode. 'r' is used to open a file in read-only mode, 'w' is used to open a file in write-only mode, 'rw' is used to open in reading and write mode, 'a' is used to open a file in append mode. If the mode is not specified, by default file opens in read-only mode.

* Read-only mode : Open a file for reading. It is the default mode.
* Write-only mode: Open a file for writing. If the file contains data, data would be lost. Other a new file is created.
* Read-Write mode: Open a file for reading, write mode. It means updating mode.
* Append mode: Open for writing, append to the end of the file, if the file exists.

### 20. What is an operator in Python?

An operator is a particular symbol which is used on some values and produces an output as a result.



**For example:**

1. -a # Unary
2. 3 + 2 = 5 # Binary
3. Here, "+" **and** "=" are operators.
4. a, b = 2,5
5. # Assign minimum value using ternary operator
6. min = a **if** a < b **else** b
7. **print**(min)

### 3) is Python interpreted language?

Python is an interpreted language. The Python language program runs directly from the source code. It converts the source code into an intermediate language code, which is again translated into machine language that has to be executed.

Unlike Java or C, Python does not require compilation before execution.

) How is memory managed in Python?

Memory is managed in Python by the following way:

* The Python memory is managed by a Python private heap space. All the objects and data structures are located in a private heap. The programmer does not have permission to access this private heap.
* Python also has an inbuilt garbage collector, which recycle all the unused memory and frees the memory for the heap space.

### 25) What is the Python decorator?

A decorator is a design pattern in **Python** that allows a user to add new functionality to an existing object without modifying its structure.

This is also called **metaprogramming** because a part of the program tries to modify another part of the program at compile time.

def first(msg):

print(msg)

first("Hello")

second = first

second("Hello")

**Output**

Hello

Hello

def inc(x):

return x + 1

def dec(x):

return x - 1

def operate(func, x):

result = func(x)

return result

We invoke the function as follows.

>>> operate(inc,3)

4

>>> operate(dec,3)

2

### 26) What are the rules for a local and global variable in Python?

There are two types of **variables**: **global variables** and **local variables**. A **global variable** can be reached anywhere in the code, a **local** only in the scope. A **global variable** (x) can be reached and modified anywhere in the code, **local variable** (z) exists only in block 3

### 27) What is the namespace in Python?

A namespace is defined as a simple system to control the names in a program. It ensures that names are unique and won't lead to any conflict.

Also, Python implements namespaces in the form of dictionaries and maintains name-to-object mapping where names act as keys and the objects as values.

### 8) What are iterators in Python?

In Python, iterators are used to iterate a group of elements, containers like a list. Iterators are the collection of items, and it can be a list, tuple, or a dictionary. Python iterator implements \_\_itr\_\_ and next() method to iterate the stored elements. In Python, we generally use loops to iterate over the collections (list, tuple).

**yield** is a keyword in **Python** that is used to return from a function without destroying the states of its local variable and when the function is called, the execution starts from the last **yield** statement. Any function that contains a **yield** keyword is termed as generator. Hence, **yield** is what makes a generator