Python:

A list of frequently asked Python interview questions with answers for freshers and experienced are given below.

### 1) What is Python?

Python was created by **Guido van Rossum**, and released in **1991**.

It is a general-purpose computer programming language. It is a high-level, object-oriented language which can run equally on different platforms such as Windows, Linux, UNIX, and Macintosh. It is widely used in data science, machine learning and artificial intelligence domain.

It is easy to learn and require less code to develop the applications.

It is widely used for:

* Web development (server-side).
* Software development.
* Mathematics.
* System scripting.

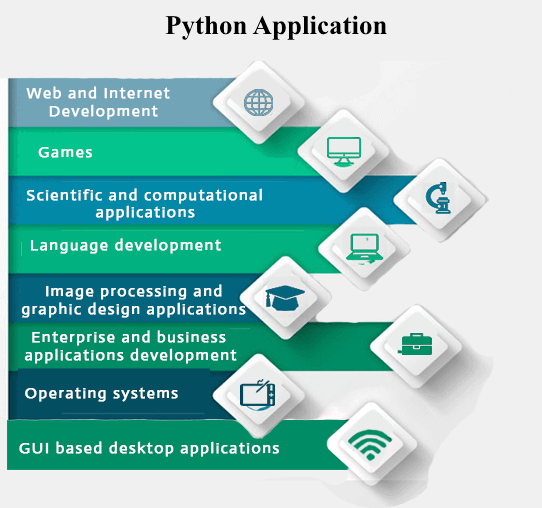
**Why Python?**

* 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. It helps to provide a prototype very quickly.
* 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 some application areas are given below.

* Web and Internet Development
* Games
* Scientific and computational applications
* Language development
* Image processing and graphic design applications
* Enterprise and business applications development
* Operating systems
* GUI based desktop applications



Python provides various web frameworks to develop web applications. The popular python web frameworks are **Django, Pyramid, Flask**.

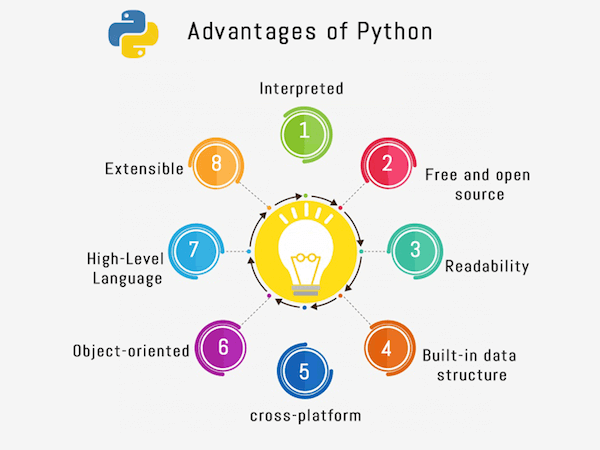
Python's standard library supports for E-mail processing, FTP, IMAP, and other Internet protocols.

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

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

### 3. What are the advantages of Python?

* Interpreted
* Free and open source
* Extensible
* Object-oriented
* Built-in data structure
* 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.
* Extensible: It is very flexible and extensible with any module.
* Object-oriented: Python allows to implement the Object Oriented concepts to build application solution.
* Built-in data structure: Tuple, List, and Dictionary are useful integrated data structures provided by the language.



### 4. 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 was written by Guido van Rossum, Barry Warsaw and Nick Coghlan in 2001.

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?

Literals can be defined as a data which is given in a variable or constant. Python supports the following literals:

**String Literals**

String literals are formed by enclosing text in the single or double quotes. For example, string literals are string values.

**E.g.:**

"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. A function is a block of self-contained statements which has a valid name, parameters list, and body. Functions make programming more functional and modular to perform modular tasks. Python provides several built-in functions to complete tasks and also allows a user to create new functions as well.

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.

**Example**: A general syntax of user defined function is given below.

1. **def** function\_name(parameters list):
2. #--- statements---
3. **return** a\_value

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

Python **zip()** function returns a zip object, which maps a similar index of multiple containers. It takes an iterable, convert into iterator and aggregates the elements based on iterables passed. It returns an iterator of tuples.

**Signature**

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

**Parameters**

**iterator1, iterator2, iterator3:** These are iterator objects that are joined together.

**Return**

It returns an iterator from two or more iterators.

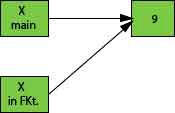
#### Note: If the given lists are of different lengths, zip stops generating tuples when the first list ends. It means two lists are having 3, and 5 lengths will create a 3-tuple.

### 8. What is Python's parameter passing mechanism?

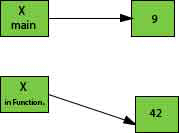
There are two parameters passing mechanism in Python:

* Pass by references
* Pass by value

By default, all the parameters (arguments) are passed "by reference" to the functions. Thus, if you change the value of the parameter within a function, the change is reflected in the calling function as well. It indicates the original variable. For example, if a variable is declared as a = 10, and passed to a function where it?s value is modified to a = 20. Both the variables denote to the same value.



The pass by value is that whenever we pass the arguments to the function only values pass to the function, no reference passes to the function. It makes it immutable that means not changeable. Both variables hold the different values, and original value persists even after modifying in the function.



Python has a default argument concept which helps to call a method using an arbitrary number of arguments.

### 9. How to overload constructors or methods in Python?

Python's constructor: \_init\_\_ () is the first method of a class. Whenever we try to instantiate an object \_\_init\_\_() is automatically invoked by python to initialize members of an object. We can't overload constructors or methods in Python. It shows an error if we try to overload.

1. **class** student:
2. **def** \_\_init\_\_(self,name):
3. self.name = name
4. **def** \_\_init\_\_(self, name, email):
5. self.name = name
6. self.email = email
8. # This line will generate an error
9. #st = student("rahul")
11. # This line will call the second constructor
12. st = student("rahul", "rahul@gmail.com")
13. **print**(st.name)
14. Output:
15. rahul

### 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**.

#### Note: You don't need to import any extra module to use these functions for removing an element from the list.

We cannot use these methods with a tuple because the tuple is different from the list.

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

It is a string's function which converts all uppercase characters into lowercase and vice versa. It is used to alter the existing case of the string. This method creates a copy of the string which contains all the characters in the swap case. If the string is in lowercase, it generates a small case string and vice versa. It automatically ignores all the non-alphabetic characters. See an example below.

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?

To remove leading characters from a string, we can use lstrip() function. It is Python string function which takes an optional char type parameter. If a parameter is provided, it removes the character. Otherwise, it removes all the leading spaces from the string.

1. string = "  javatpoint "
2. string2 = "    javatpoint        "
3. **print**(string)
4. **print**(string2)
5. **print**("After stripping all leading whitespaces:")
6. **print**(string.lstrip())
7. **print**(string2.lstrip())

javatpoint

javatpoint

After stripping all leading whitespaces:

javatpoint

javatpoint

Python Interview Questions

After stripping, all the whitespaces are removed, and now the string looks like the below:

Python Interview Questions

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

The join() is defined as a string method which returns a string value. It is concatenated with the elements of an iterable. It provides a flexible way to concatenate the strings. See an example below.

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?

This method shuffles the given string or an array. It randomizes the items in the array. This method is present in the random module. So, we need to import it and then we can call the function. It shuffles elements each time when the function calls and produces different output.

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

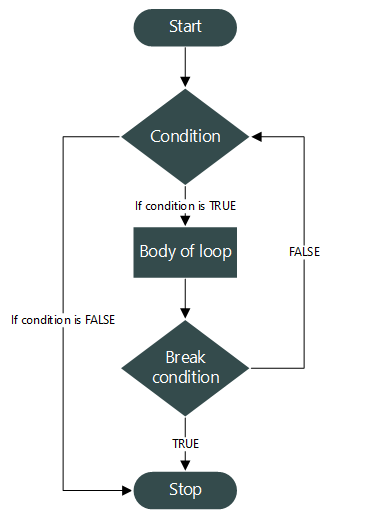
4

6

8

10

odd value found 11



Python Break statement flowchart.

### 17. What is tuple in Python?

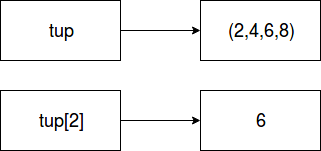
A tuple is a built-in data collection type. It allows us to store values in a sequence. It is immutable, so no change is reflected in the original data. It uses () brackets rather than [] square brackets to create a tuple. We cannot remove any element but can find in the tuple. We can use indexing to get elements. It also allows traversing elements in reverse order by using negative indexing. Tuple supports various methods like max(), sum(), sorted(), Len() etc.

To create a tuple, we can declare it as below.

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



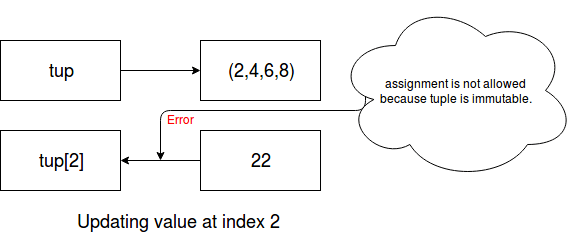
It is immutable. So updating tuple will lead to an error.

1. # Declaring tuple
2. tup = (2,4,6,8)
3. # Displaying value
4. **print**(tup)
6. # Displaying Single value
7. **print**(tup[2])
9. # Updating by assigning new value
10. tup[2]=22
11. # Displaying Single value
12. **print**(tup[2])

tup[2]=22

TypeError: 'tuple' object does not support item assignment

(2, 4, 6, 8)



### 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.

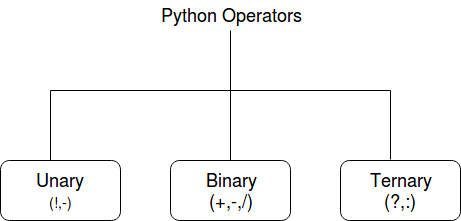
### 19. What are the different file processing modes supported by Python?

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. An operator works on operands. Operands are numeric literals or variables which hold some values. Operators can be unary, binary or ternary. An operator which require a single operand known as a unary operator, which require two operands known as a binary operator and which require three operands is called ternary operator.



**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)

### 21. What are the different types of operators in Python?

Python uses a rich set of operators to perform a variety of operations. Some individual operators like membership and identity operators are not so familiar but allow to perform operations.

* Arithmetic OperatorsRelational Operators
* Assignment Operators
* Logical Operators
* Membership Operators
* Identity Operators
* Bitwise Operators



**Arithmetic operators** perform basic arithmetic operations. For example "+" is used to add and "?" is used for subtraction.

1. # Adding two values
2. **print**(12+23)
3. # Subtracting two values
4. **print**(12-23)
5. # Multiplying two values
6. **print**(12\*23)
7. # Dividing two values
8. **print**(12/23)

**Relational Operators** are used to comparing the values. These operators test the conditions and then returns a boolean value either True or False.

# Examples of Relational Operators

1. a, b = 10, 12
2. **print**(a==b) # False
3. **print**(a<b) # True
4. **print**(a<=b) # True
5. **print**(a!=b) # True

**Assignment operators** are used to assigning values to the variables. See the examples below.

1. # Examples of Assignment operators
2. a=12
3. **print**(a) # 12
4. a += 2
5. **print**(a) # 14
6. a -= 2
7. **print**(a) # 12
8. a \*=2
9. **print**(a) # 24
10. a \*\*=2
11. **print**(a) # 576

**Logical operators** are used to performing logical operations like And, Or, and Not. See the example below.

1. # Logical operator examples
2. a = True
3. b = False
4. **print**(a **and** b) # False
5. **print**(a **or** b) # True
6. **print**(**not** b) # True

**Membership operators** are used to checking whether an element is a member of the sequence (list, dictionary, tuples) or not. Python uses two membership operators in and not in operators to check element presence. See an example.

1. # Membership operators examples
2. list = [2,4,6,7,3,4]
3. **print**(5 **in** list) # False
4. cities = ("india","delhi")
5. **print**("tokyo" **not** **in** cities) #True

Identity Operators (is and is not) both are used to check two values or variable which are located on the same part of the memory. Two variables that are equal does not imply that they are identical. See the following examples.

1. # Identity operator example
2. a = 10
3. b = 12
4. **print**(a **is** b) # False
5. **print**(a **is** **not** b) # True

**Bitwise Operators** are used to performing operations over the bits. The binary operators (&, |, OR) work on bits. See the example below.

1. # Identity operator example
2. a = 10
3. b = 12
4. **print**(a & b) # 8
5. **print**(a | b) # 14
6. **print**(a ^ b) # 6
7. **print**(~a) # -11

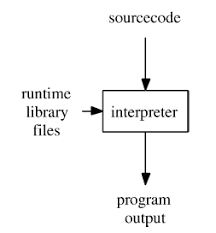
### 22) How to create a Unicode string in Python?

In Python 3, the old Unicode type has replaced by "str" type, and the string is treated as Unicode by default. We can make a string in Unicode by using art.title.encode("utf-8") function.

### 23) 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.



### 24) 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.
* We can easily allocate heap space for Python objects by the Python memory manager. The core API gives access of some tools to the programmer for coding purpose.
* 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?

Decorators are very powerful and a useful tool in Python that allows the programmers to modify the behaviour of any class or function. It allows us to wrap another function to extend the behaviour of the wrapped function, without permanently modifying it.

1. # Decorator example
2. **def** decoratorfun():
3. **return** another\_fun

**Functions vs. Decorators**

A function is a block of code that performs a specific task whereas a decorator is a function that modifies other functions.

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

In Python, variables that are only referenced inside a function are called implicitly global. If a variable is assigned a new value anywhere within the function's body, it's assumed to be a local. If a variable is ever assigned a new value inside the function, the variable is implicitly local, and we need to declare it as 'global' explicitly. To make a variable globally, we need to declare it by using global keyword. Local variables are accessible within local body only. Global variables are accessible anywhere in the program, and any function can access and modify its value.

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

The namespace is a fundamental idea to structure and organize the code that is more useful in large projects. However, it could be a bit difficult concept to grasp if you're new to programming. Hence, we tried to make namespaces just a little easier to understand.

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.

### 28) 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).

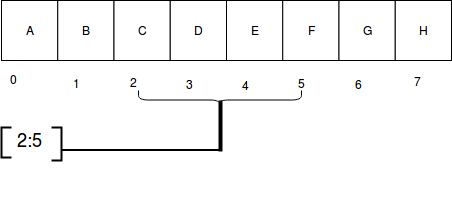
### 29) What is a generator in Python?

In Python, the generator is a way that specifies how to implement iterators. It is a normal function except that it yields expression in the function. It does not implements \_\_itr\_\_ and next() method and reduce other overheads as well.

If a function contains at least a yield statement, it becomes a generator. The yield keyword pauses the current execution by saving its states and then resume from the same when required.

### 30) What is slicing in Python?

Slicing is a mechanism used to select a range of items from sequence type like list, tuple, and string. It is beneficial and easy to get elements from a range by using slice way. It requires a : (colon) which separates the start and end index of the field. All the data collection types List or tuple allows us to use slicing to fetch elements. Although we can get elements by specifying an index, we get only single element whereas using slicing we can get a group of elements.



### 31) What is a dictionary in Python?

The Python dictionary is a built-in data type. It defines a one-to-one relationship between keys and values. Dictionaries contain a pair of keys and their corresponding values. It stores elements in key and value pairs. The keys are unique whereas values can be duplicate. The key accesses the dictionary elements.

Keys index dictionaries.

**Let's take an example.**

The following example contains some keys Country Hero & Cartoon. Their corresponding values are India, Modi, and Rahul respectively.

1. >>> dict = {'Country': 'India', 'Hero': 'Modi', 'Cartoon': 'Rahul'}
2. >>>**print** dict[Country]
3. India
4. >>>**print** dict[Hero]
5. Modi
6. >>>**print** dict[Cartoon]
7. Rahul

### 32) What is Pass in Python?

Pass specifies a Python statement without operations. It is a placeholder in a compound statement. If we want to create an empty class or functions, this pass keyword helps to pass the control without error.

1. # For Example
2. **class** Student:
3. **pass** # Passing class
4. **class** Student:
5. **def** info():
6. **pass** # Passing function

### 33) Explain docstring in Python?

The Python docstring is a string literal that occurs as the first statement in a module, function, class, or method definition. It provides a convenient way to associate the documentation.

String literals occurring immediately after a simple assignment at the top are called "attribute docstrings".

String literals occurring immediately after another docstring are called "additional docstrings".

Python uses triple quotes to create docstrings even though the string fits on one line.

Docstring phrase ends with a period (.) and can be multiple lines. It may consist of spaces and other special chars.

**Example**

1. # One-line docstrings
2. **def** hello():
3. """A function to greet."""
4. **return** "hello"

### 34) What is a negative index in Python?

Python sequences are accessible using an index in positive and negative numbers. For example, 0 is the first positive index, 1 is the second positive index and so on. For negative indexes -1 is the last negative index, -2 is the second last negative index and so on.

Index traverses from left to right and increases by one until end of the list.

Negative index traverse from right to left and iterate one by one till the start of the list. A negative index is used to traverse the elements into reverse order.

### 35) What is pickling and unpickling in Python?

The Python pickle is defined as a module which accepts any Python object and converts it into a string representation. It dumps the Python object into a file using the dump function; this process is called **pickling**.

The process of retrieving the original Python objects from the stored string representation is called as Unpickling.

### 36) Which programming language is a good choice between Java and Python?

Java and Python both are object-oriented programming languages. Let's compare both on some criteria given below:

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Java** | **Python** |
| Ease of use | Good | Very Good |
| Coding Speed | Average | Excellent |
| Data types | Static type | Dynamic type |
| Data Science and Machine learning application | Average | Very Good |

### 37) What is the usage of help() and dir() function in Python?

Help() and dir() both functions are accessible from the Python interpreter and used for viewing a consolidated dump of built-in functions.

**Help() function**: The help() function is used to display the documentation string and also facilitates us to see the help related to modules, keywords, and attributes.

**Dir() function**: The dir() function is used to display the defined symbols.

### 38) How can we make forms in Python?

You have to import CGI module to access form fields using FieldStorage class.

**Attributes of class FieldStorage for the form:**

**form.name**: The name of the field, if specified.

**form.filename**: If an FTP transaction, the client-side filename.

**form.value**: The value of the field as a string.

**form.file**: file object from which data read.

**form.type**: The content type, if applicable.

**form.type\_options**: The options of the 'content-type' line of the HTTP request, returned as a dictionary.

**form.disposition**: The field 'content-disposition'; None, if unspecified.

**form.disposition\_options**: The options for 'content-disposition'.

**form.headers**: All of the HTTP headers returned as a dictionary.

1. **import** cgi
2. form = cgi.FieldStorage()
3. **if** **not** (form.has\_key("name") **and** form.has\_key("age")):
4. **print** "<H1>Name & Age not Entered</H1>"
5. **print** "Fill the Name & Age accurately."
6. **return**
7. **print** "<p>name:", form["name"].value
8. **print** "<p>Age:", form["age"].value

### 40) What are the differences between Python 2.x and Python 3.x?

Python 2.x is an older version of Python. Python 3.x is newer and latest version. Python 2.x is legacy now. Python 3.x is the present and future of this language.

The most visible difference between Python2 and Python3 is in print statement (function). In Python 2, it looks like print "Hello", and in Python 3, it is print ("Hello").

String in Python2 is ASCII implicitly, and in Python3 it is Unicode.

The xrange() method has removed from Python 3 version. A new keyword as is introduced in Error handling.

### 41) How can you organize your code to make it easier to change the base class?

You have to define an alias for the base class, assign the real base class to it before your class definition, and use the alias throughout your class. You can also use this method if you want to decide dynamically (e.g., depending on availability of resources) which base class to use.

**Example**

1. BaseAlias = <real base **class**>
2. **class** Derived(BaseAlias):
3. **def** meth(self):
4. BaseAlias.meth(self)

### 43) How Python does Compile-time and Run-time code checking?

In Python, some amount of coding is done at compile time, but most of the checking such as type, name, etc. are postponed until code execution. Consequently, if the Python code references a user-defined function that does not exist, the code will compile successfully. The Python code will fail only with an exception when the code execution path does not exist.

### 44) What is the shortest method to open a text file and display its content?

The shortest way to open a text file is by using "with" command in the following manner:

1. with open("file-name", "r") as fp:
2. fileData = fp.read()
3. #to print the contents of the file
4. **print**(fileData)

### 45) What is the usage of enumerate () function in Python?

The enumerate() function is used to iterate through the sequence and retrieve the index position and its corresponding value at the same time.

1. For i,v **in** enumerate(['Python','Java','C++']):
2. **print**(i,v)
3. 0 Python
4. 1 Java
5. 2 C++
6. # enumerate using an index sequence
7. **for** count, item **in** enumerate(['Python','Java','C++'], 10):

### 46) Give the output of this example: A[3] if A=[1,4,6,7,9,66,4,94].

Since indexing starts from zero, an element present at 3rd index is 7. So, the output is 7.

### 47) What will be the output of ['!!Welcome!!']\*2?

The output will be ['!!Welcome!! ', '!!Welcome!!']

### 48) What will be the output of data[-2] from the list data = [1,5,8,6,9,3,4]?

In the list, an element present at the 2nd index from the right is 3. So, the output will be 3.

### 49. How to send an email in Python Language?

To send an email, Python provides smtplib and email modules. Import these modules into the created mail script and send mail by authenticating a user.

It has a method SMTP(smtp-server, port). It requires two parameters to establish SMTP connection.

A simple example to send an email is given below.

1. **import** smtplib
2. # Calling SMTP
3. s = smtplib.SMTP('smtp.gmail.com', 587)
4. # TLS for network security
5. s.starttls()
6. # User email Authentication
7. s.login("sender\_email\_id", "sender\_email\_id\_password")
8. # message to be sent
9. message = "Message\_you\_need\_to\_send"
10. # sending the mail
11. s.sendmail("sender\_email\_id", "receiver\_email\_id", message)

It will send an email to the receiver after authenticating sender username and password.

### 50) What is the difference between list and tuple?

The difference between list and tuple is that a list is mutable while tuple is not.

### 51) What is lambda function in Python?

The anonymous function in python is a function that is defined without a name. The normal functions are defined using a keyword "def", whereas, the anonymous functions are defined using the lambda function. **The anonymous functions are also called as lambda functions**.

### 52) Why do lambda forms in Python not have the statements?

Because it is used to make the new function object and return them in runtime.

### 53) How can you convert a number to string?

We can use the inbuilt function **str()** to convert a number into a string. If you want an octal or hexadecimal representation, we can use the oct() or hex() inbuilt function.

### 54) Mention the rules for local and global variables in Python?

**Local variables:** If a new value is assigned by a variable within the function's body, it is assumed to be local.

**Global variables:** These are the variables that are only referenced inside a function are implicitly global.