edureka!

Python Programming Certification Training

Interview Preparation

Python Interview Preparation

Python is the most sought-after skill in the programming domain. In this **Python Interview Preparation** document, you will be introduced to the most frequently asked questions in Python interviews.

We have compiled a list of top Python interview questions, which are classified into 7 sections, namely:

1. Basic Interview Questions
2. OOPS Interview Questions
3. Basic Python Programs
4. Python Libraries Interview Questions
5. Web Scraping Interview Questions
6. Data Analysis Interview Questions
7. Multiple Choice Questions (MCQ)

Let us begin with some Basic Python Interview Questions.

Q1. What is the difference between lists and tuples in Python?

Ans.

Lists	Tuples
Lists are mutable, that is, they can be edited.	Tuples are immutable (tuples are lists that can’t be edited).
Lists are slower than tuples.	Tuples are faster than lists.
Syntax: `list_1 = [10, ‘Chelsea’, 20]`	**Syntax**: `tup_1 = (10, ‘Chelsea’ , 20)`

Q2. What are the key features of Python?

- Python is an **interpreted** language. That means that, unlike languages like C and its variants, Python does not need to be compiled before it is run. Other interpreted languages include PHP and Ruby.
- Python is **dynamically typed**. This means that you don’t need to state the types of variables when you declare them or anything like that. You can do things like x = 111 and then x = "I'm a string" without error.
- Python is well suited to **object-orientated programming** in that it allows the definition of classes along with composition and inheritance. Python does not have access specifiers (like C++’s public or private).
- In Python, **functions** are **first-class objects**. This means that they can be assigned to variables, returned from other functions, and passed into functions. Classes are also first-class objects.

- **Writing Python code is quick,** but running it is often slower than compiled languages. Fortunately, Python allows the inclusion of C-based extensions, so bottlenecks can be optimized away and often are. The **numpy** package is a good example of this; it's really quite quick because a lot of the number-crunching it does isn't actually done by Python
- Python finds **use in many spheres—**web applications, automation, scientific modeling, big data applications, and many more. It's also often used as "glue" code to get other languages and components to play nice.

Q3. What type of language is Python? Programming or scripting?

Ans: Python is capable of scripting, but in a general sense, it is considered as a general-purpose programming language.

Q4. Python is an interpreted language. Explain.

Ans: An interpreted language is any programming language that is not in machine-level code before runtime. Therefore, Python is an interpreted language.

Q5. What is pep 8?

Ans: PEP stands for **Python Enhancement Proposal.** It is a set of rules that specify how to format Python code for maximum readability.

Q6. What are the benefits of using Python?

Ans: The benefits of using Python are**—**

1. **Easy to use**: Python is a high-level programming language that is easy to use, read, write, and learn.
2. **Interpreted language**: Since Python is an interpreted language, it executes the code line-by-line and stops if an error occurs in any line.
3. **Dynamically typed**: The developer does not assign data types to variables at the time of coding. It is automatically assigned during execution.
4. **Free and open source**: Python is free to use and distribute and it is an open source.
5. **Extensive support for libraries**: Python has vast libraries that contain almost any function needed. It also further provides the facility to import other packages using the Python Package Manager (pip).
6. **Portable**: Python programs can run on any platform without requiring any changes.
7. **User-friendly**: The data structures used in python are user-friendly.
8. **Enhanced functionality**: It provides more functionality with less coding.

Q7. What are Python namespaces?

Ans: A namespace in Python refers to the name which is assigned to each object in Python. The objects are variables and functions. As each object is created, its name, along with space (the address of the outer function in which the object is), gets created. The namespaces are maintained in Python like a dictionary where the key is the namespace and the value is the address of the object. There are 4 types of namespace in python.

1. **Built-in namespace**: These namespaces contain all the built-in objects in Python and are available whenever Python is running.
2. **Global namespace**: These are namespaces for all the objects created at the level of the main program.
3. **Enclosing namespace**: These namespaces are at the higher level or outer function.
4. **Local namespace**: These namespaces are at the local or inner function.

Q8. What are decorators in Python?

Ans: Decorators are used to add some design patterns to a function without changing its structure. Decorators generally are defined before the function they are enhancing. To apply a decorator, we first define the decorator function. Then we write the function it is applied to and simply add the decorator function above the function it has to be applied to. For this, we use the @ symbol before the decorator.

Q9. What are Dict and List comprehensions?

Ans: Dictionary and list comprehensions are just another concise way to define dictionaries and lists.

An example of list comprehension

```
x= [i for i in range(5)]
```

The above code creates a list as below:

```
[0,1,2,3,4]
```

An example of dictionary comprehension

```
x= [i : i+2 for i in range(5)]
```

The above code creates a list as below:

```
[0: 2, 1: 3, 2: 4, 3: 5, 4: 6]
```

Q10.What are the common built-in data types in Python?

Ans: The common built-in data types in python:

Numbers: They include integers, floating-point numbers, and complex numbers.

Example: 1, 7.9,3+4i

List: An ordered sequence of items is called a list. The elements of a list may belong to different data types.

Example: [5,'market',2.4]

Tuple: It is also an ordered sequence of elements. Unlike lists, tuples are immutable, which means they can't be changed.

Example: (3,'tool',1)

String: A sequence of characters is called a string. They are declared within single or double quotes.

Example: "Sana"; 'She is going to the market'

Set: Sets are a collection of unique items that are not in order.

Example: {7,6,8}

Dictionary: A dictionary stores values in key and value pairs where each value can be accessed through its key. The order of items is not important.

Example: {1:'apple',2:'mango}

Boolean: There are 2 boolean values—**True** and **False**.

Q11.What is the difference between .py and .pyc files?

Ans: The .py files are the Python source code files. While the .pyc files contain the bytecode of the Python files. The .pyc files are created when the code is imported from some other source. The interpreter converts the source .py files to .pyc files which helps by saving time.

Q12.What is slicing in Python?

Ans: Slicing is used to access parts of sequences like lists, tuples, and strings. The syntax of slicing is [start:end:step]. The step can be omitted as well. When we write [start:end] this returns all the elements of the sequence from the start (inclusive) till the end-1 element. If the start or end element is negative i, it means the ith element from the end. The step indicates the jump or how many elements have to be skipped. For example, if there is a list- [1,2,3,4,5,6,7,8]. Then [-1:2:2] will return elements starting from the last element till the third element by printing every second element. That is, [8,6,4].

Q13.What are Keywords in Python?

Ans: Keywords in Python are reserved words that have a special meaning. They are generally used to define the type of variables. Keywords cannot be used for variable or function names. There are the following 33 keywords in Python:

- And
- Or
- Not
- If
- Elif
- Else
- For
- While
- Break
- As
- Def
- Lambda
- Pass
- Return
- True
- False
- Try
- With
- Assert
- Class
- Continue
- Del
- Except
- Finally
- From
- Global
- Import
- In
- Is

- None
- Nonlocal
- Raise
- Yield

Q14. What are Literals in Python and explain about different Literals?

Ans: A literal in Python source code represents a fixed value for primitive data types. There are 5 types of literals in Python.

1. **String literals**: A string literal is created by assigning some text enclosed in single or double quotes to a variable. To create multiline literals, assign the multiline text enclosed in triple quotes. Example: name=”Tanya”
2. **Character literal**: It is created by assigning a single character enclosed in double quotes. Example: a=’t’
3. **Numeric literals:** It includes numeric values that can be either an integer, floating point value, or complex number. Example: a=50
4. **Boolean literals**: These can be 2 values—either True or False.
5. **Literal Collections**: These are of 4 types—

   a) List collections - Example: a=[1,2,3,’Amit’]

   b) Tuple literals - Example: a=(5,6,7,8)

   c) Dictionary literals - Example: dict={1: ’apple’, 2: ’mango, 3: ’banana`’}

   d) Set literals - Example: {“Tanya”, “Rohit”, “Mohan”}

6. **Special literal:** Python has 1 special literal None which is used to return a null variable.

Q15.How to combine dataframes in pandas?

Ans: The dataframes in Python can be combined in the following ways:

1. Concatenating them by stacking the 2 dataframes vertically.
2. Concatenating them by stacking the 2 dataframes horizontally.
3. Combining them on a common column. This is referred to as joining.

The concat() function is used to concatenate two dataframes. Its syntax is pd.concat([dataframe1, dataframe2]).

Dataframes are joined together on a common column called a key. When we combine all the rows in a dataframe, it is a union, and the join used is the outer join. When we combine the common rows or intersections, the join used is the inner join. Its syntax is pd.concat([dataframe1, dataframe2], axis=’axis’, join=’type_of_join’)

Q16.What are the new features added in Python 3.9.0.0 version?

Ans: The new features in Python 3.9.0.0 version are

- New Dictionary functions Merge(|) and Update(|=)
- New String Methods to Remove Prefixes and Suffixes
- **Type Hinting Generics in Standard Collections**
- New Parser based on PEG rather than LL1
- New modules like zoneinfo and graphlib.
- **Improved Modules like ast and asyncio.**
- **Optimizations such as optimized idiom for assignment, signal handling, and optimized python built-ins.**
- Deprecated functions and commands such as deprecated parser and symbol modules, and deprecated functions.
- Removal of erroneous methods and functions.

Q17. How is memory managed in Python?

Ans: Memory is managed in Python in the following ways:

1. Memory management in Python is managed by the **Python private heap space**. All Python objects and data structures are located in a private heap. The programmer does not have access to this private heap. The Python interpreter takes care of this instead.
2. The allocation of heap space for Python objects is done by Python's memory manager. The core API gives access to some tools for the programmer to code.
3. Python also has an inbuilt garbage collector, which recycles all the unused memory so that it can be made available to the heap space.

Q18. What is a namespace in Python?

Ans: A namespace is a naming system used to make sure that names are unique to avoid naming conflicts.

Q19. What is PYTHONPATH?

Ans: It is an environment variable which is used when a module is imported. Whenever a module is imported, PYTHONPATH is also looked up to check for the presence of the imported modules in various directories. The interpreter uses it to determine which module to load.

Q20. What are the Python modules? Name some commonly used built-in modules in Python.

Ans: Python modules are files containing Python code. This code can either be functions, classes, or variables. A Python module is a .py file containing executable code.

Some of the commonly used built-in modules are:

- os
- sys
- math
- random
- data time
- JSON

Q21.What are local variables and global variables in Python?

Global Variables

Variables declared outside a function or in a global space are called global variables. These variables can be accessed by any function in the program.

Local Variables

Any variable declared inside a function is known as a local variable. This variable is present in the local space and not in the global space.

Example

```
a=2
def add():
b=3
c=a+b
print(c)
add()
```

Output: 5

When you try to access the local variable outside the function `add()`, it will throw an error.

Q22. Is Python case-sensitive?

Ans: Yes. Python is a case-sensitive language.

Q23.What is type conversion in Python?

Ans: Type conversion refers to the conversion of one data type into another.

`int()` – converts any data type into integer type.

`float()` – converts any data type into float type.

`ord()` – converts characters into integer.

`hex()` – converts integers to hexadecimal.

`oct()` – converts an integer to octal.

`tuple()` – used to convert to a tuple.

`set()` – returns the type after converting to set.

`list()` – convert any data type to a list type.

`dict()` – convert a tuple of order (key, value) into a dictionary.

`str()` – convert an integer into a string.

`complex(real,imag)` – converts real numbers to complex(real,imag) number.

Q24. How to install Python on Windows and set path variables?

Ans: To install Python on Windows, follow the below steps:

- Install Python from this link: https://www.python.org/downloads/
- After this, install it on your PC. Look for the location where Python has been installed on your PC using the following command on your command prompt: cmd python.
- Then go to advanced system settings and add a new variable and name it as PYTHON_NAME and paste the copied path.
- Look for the path variable, select its value, and select 'edit'.
- Add a semicolon towards the end of the value if it's not present, and then type %PYTHON_HOME%.

Q25. Is indentation required in Python?

Ans: Indentation is necessary for Python. It specifies a block of code. All code within loops, classes, and functions is specified within an indented block. It is usually done using four space characters. If your code is not indented necessarily, it will not execute accurately and will throw errors as well.

Q26. What is the difference between Python Arrays and Lists?

Ans: Arrays and lists in Python, have the same way of storing data. But arrays can hold only a single data type element, whereas lists can hold any data type element.

Example

```
import array as arr
My_Array=arr.array('i',[1,2,3,4])
My_list=[1,'abc',1.20]
print(My_Array)
print(My_list)
```

Output

```
array('i', [1, 2, 3, 4]) [1, 'abc', 1.2]
```

Q27. What are functions in Python?

Ans: A function is a block of code that is executed only when it is called. To define a Python function, the **def** keyword is used.

Example

```
def Newfunc():
print("Hi, Welcome to Edureka")
Newfunc(); #calling the function
```

Output

```
Hi, Welcome to Edureka
```

Q28.What is `__init__`?

Ans: `__init__` is a method or constructor in Python. This method is automatically called to allocate memory when a new object/instance of a class is created. All classes have the `__init__` method.

Example

```
class Employee:
def __init__(self, name, age,salary):
self.name = name
self.age = age
self.salary = 20000
E1 = Employee("XYZ", 23, 20000)
# E1 is the instance of class Employee.
#__init__ allocates memory for E1.
print(E1.name)
print(E1.age)
print(E1.salary
```

Output

```
XYZ

23

20000
```

Q29.What is a lambda function?

Ans: An anonymous function is known as a lambda function. This function can have any number of parameters but can have just one statement.

Example

```
a = lambda x,y : x+y
print(a(5, 6))
```

Output

```
11
```

Q30. What is self in Python?

Ans: Self is an instance or an object of a class. In Python, this is explicitly included as the first parameter. However, this is not the case in Java, where it's optional. It helps to differentiate between the methods and attributes of a class with local variables.

The self-variable in the `init` method refers to the newly created object, while in other methods it refers to the object whose method was called.

Q31. How does break, continue, and pass work?

Break	Allows loop termination when some condition is met, and control is transferred to the next statement.
Continue	Allows skipping some part of a loop when some specific condition is met, and the control is transferred to the beginning of the loop.
Pass	Used when you need some block of code syntactically but you want to skip its execution. This is basically a null operation. Nothing happens when this is executed.

Q32. What does `[::-1]` do?

Ans: `[::-1]` is used to reverse the order of an array or a sequence.

Example

```
import array as arr
My_Array=arr.array('i',[1,2,3,4,5])
My_Array[::-1]
```

Output

```
 array(‘i’, [5, 4, 3, 2, 1])
```

`[::-1]` reprints a reversed copy of ordered data structures such as an array or a list. The original array or list remains unchanged.

Q33. How can you randomize the items of a list in place in Python?

Ans: Consider the example shown below.

```
from random import shuffle
x = ['Keep', 'The', 'Blue', 'Flag', 'Flying', 'High']
shuffle(x)
print(x)
```

The output of the following code is as below.

```
['Flying', 'Keep', 'Blue', 'High', 'The', 'Flag']
```

Q34. What are Python iterators?

Ans: Iterators are objects which can be traversed through or iterated upon.

Q35. How can you generate random numbers in Python?

Ans: Random module is the standard module that is used to generate a random number. The method is defined as:

```
import random
random.random
```

The statement `random.random()` method return the floating-point number that is in the range of [0, 1). The function generates random float numbers. The methods that are used with the random class are the bound methods of the hidden instances. The instances of the Random can be done to show the multi-threading programs that create different instances of individual threads. The other random generators that are used in this are:

1. **randrange(a, b)**: It chooses an integer and defines the range in-between [a, b). It returns the elements by selecting it randomly from the range that is specified. It doesn’t build a range object.
2. **uniform(a, b)**: It chooses a floating point number that is defined in the range of [a,b).Iyt returns the floating point number.
3. **normalvariate(mean, sdev)**: It is used for the normal distribution where the mu is a mean and the sdev is a sigma that is used for standard deviation.
4. **The Random class:** It is used and instantiated creates independent multiple random number generators.

Q36. What is the difference between range and xrange?

Ans: For the most part, xrange and range are the exact same in terms of functionality. They both provide a way to generate a list of integers for you to use, however you please. The only difference is that range returns a Python list object and x range returns an xrange object.

This means that xrange doesn't actually generate a static list at run-time as the range does. It creates the values as you need them with a special technique called yielding. This technique is used with a type of object known as a generator. That means that if you have a really gigantic range, and you'd like to generate a list for, say, one billion, xrange is the function to use.

This is especially true if you have a really memory-sensitive system, such as a cell phone that you are working with, as the range will use as much memory as it can to create your array of integers, which can result in a memory error and crash your program. It's a memory-hungry beast.

Q37. How do you write comments in Python?

Ans: Comments in Python start with a # character. However, alternatively, at times, commenting is done using docstrings (strings enclosed within triple quotes).

Example

```
<span data-mce-type="bookmark" style="display: inline-block;
width: 0px; overflow: hidden; line-height: 0;"
class="mce_SELRES_end"></span>
<pre><span>#Comments in Python start like this
print("Comments in Python start with a #")
```

Output

```
Comments in Python start with a #
```

Q38. What is pickling and unpickling?

Ans: Pickle module accepts any Python object and converts it into a string representation and dumps it into a file by using the dump function, this process is called pickling. The process of retrieving original Python objects from the stored string representation is called unpickling.

Q39. What are the generators in Python?

Ans: Functions that return an iterable set of items are called generators.

Q40. How will you capitalize the first letter of string?

Ans: In Python, the capitalize() method capitalizes the first letter of a string. If the string already consists of a capital letter at the beginning, then, it returns the original string.

Q41. How will you convert a string to all lowercase?

Ans: To convert a string to lowercase, the lower() function can be used.

Example:

```
stg='ABCD'
print(stg.lower())
```

Output:

```
abcd
```

Q42. How to comment multiple lines in Python?

Ans: Multi-line comments appear in more than one line. All the lines to be commented are to be prefixed by a #. You can also a very good **shortcut method to comment multiple lines**. All you need to do is hold the Ctrl key and **left-click** in every place wherever you want to include a # character and type a # just once. This will comment all the lines where you introduced your cursor.

Q43.What are docstrings in Python?

Ans: Docstrings are not actually comments, but **documentation strings**. These docstrings are within triple quotes. They are not assigned to any variable and therefore, at times, serve the purpose of comments as well.

Example

```
"""
Using docstring as a comment.
This code divides 2 numbers
"""
x=8
y=4
z=x/y
print(z)
```

Output

```
2.0
```

Q44. What is the purpose of `'is'`, `'not'` and `'in'` operators?

Ans: Operators are special functions. They take one or more values and produce a corresponding result.

is: Returns true when 2 operands are true (Example: “a” is ‘a’)

not: Returns the inverse of the boolean value

in: Checks if some element is present in some sequence

Q45. What is the usage of `help()` and `dir()` functions in Python?

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

1. **Help() function**: The help() function is used to display the documentation string and also facilitates you to see the help related to modules, keywords, attributes, and so on.
2. **Dir() function**: The dir() function is used to display the defined symbols.

Q46. Whenever Python exits, why isn’t all the memory de-allocated?
Ans:

1. Whenever Python exits, especially those Python modules which have circular references to other objects or the objects that are referenced from the global namespaces, they are not always de-allocated or freed.
2. It is impossible to de-allocate those portions of memory that are reserved by the C library.
3. On exit, because of having its own efficient clean-up mechanism, Python would try to de-allocate/destroy every other object.

Q47. What is a dictionary in Python?

Ans: The built-in datatypes in Python are called a dictionary. It defines the one-to-one relationship between keys and values. Dictionaries contain pair of keys and their corresponding values. Dictionaries are indexed by keys.

Let’s take an example:

The following example contains some keys. Country, Capital, and PM. Their corresponding values are India, Delhi, and Modi respectively.

```
Dict={‘Country’:’India’,’Capital’:’Delhi’,’PM':'Modi'}

print dict[Country]
```

Output
```
India
```

```
print dict[Capital]
```

Output

```
Delhi
```

```
print dict[PM]
```

Output

```
Modi
```

Q48. How can the ternary operators be used in Python?

Ans: The ternary operator is the operator that is used to show conditional statements. This consists of the true or false values with a statement that has to be evaluated for it.

Syntax

The ternary operator will be given as:
[on_true] if [expression] else [on_false]x, y = 25, 50big = x if x < y else y

Example

The expression gets evaluated like if x < y else y, in this case, if x < y is true, then the value is returned as big = x and if it is incorrect, then big = y will be sent as a result.

Q49. What does this mean: `*arg` and `**kwargs`? And why would we use it?

Ans: We use *args when we aren't sure how many arguments are going to be passed to a function, or if we want to pass a stored list or tuple of arguments to a function. **kwargs is used when we don't know how many keyword arguments will be passed to a function, or it can be used to pass the values of a dictionary as keyword arguments. The identifiers args and kwargs are a convention, you could also use *bob and **billy but that would not be wise.

Q50. What does `len()` do?

Ans: It is used to determine the length of a string, list, array, and so on.

Example

```
stg='ABCD'
len(stg)
```

Output:

```
4
```

Q51. Explain `split()`, `sub()`, and `subn()` methods of "re" module in Python.

Ans: To modify the strings, Python's "re" module is providing 3 methods. They are:

- **split()** – Uses a regex pattern to "split" a given string into a list.
- **sub()** – Finds all substrings where the regex pattern matches and then replace them with a different string.
- **subn()** – It is similar to sub() and also returns the new string along with the number of replacements.

Q52. What are negative indexes and why are they used?

Ans: The sequences in Python are indexed and consist of positive as well as negative numbers. The numbers that are positive use '0' that is used as the first index and '1' as the second index and the process goes on like that.

The index for the negative number starts from '-1' which represents the last index in the sequence and '-2' as the penultimate index and the sequence carries forward like the positive number.

The negative index is used to remove any new-line spaces from the string and allows the string to except the last character that is given as S[:-1]. The negative index is also used to show the index to represent the string in the correct order.

Q53. What are Python packages?

Ans: Python packages are namespaces containing multiple modules.

Q54. How can files be deleted in Python?

Ans: To delete a file "xyz.txt" in Python, you need to import the OS Module. After that, you need to use the `os.remove()` function.

Example

```
import os
os.remove("xyz.txt")
```

Q55. What are the built-in types of Python?

Ans: Built-in types in Python are as follows:

- Integers
- Floating-point
- Complex numbers

- Strings
- Boolean
- Built-in functions

Q56. What advantages do NumPy arrays offer over (nested) Python lists?
Ans:

1. Python's lists are efficient general-purpose containers. They support (fairly) efficient insertion, deletion, appending, and concatenation, and Python's list comprehensions make them easy to construct and manipulate.
2. They have certain limitations: they don't support "vectorized" operations like elementwise addition and multiplication, and the fact that they can contain objects of differing types means that Python must store type information for every element and must execute type-dispatching code when operating on each element.
3. NumPy is not just more efficient; it is also more convenient. You get a lot of vector and matrix operations for free, which sometimes allows one to avoid unnecessary work. And they are also efficiently implemented.
4. NumPy array is faster and you get a lot built-in with NumPy, FFTs, convolutions, fast searching, basic statistics, linear algebra, histograms, and so on.

Q57. How to add values to a Python array?

Ans: Elements can be added to an array using **`append()`**, **`extend()`**, and the **`insert(i,x)`** functions.

Example

```
a=arr.array('d', [1.1 , 2.1 ,3.1] )
a.append(3.4)
print(a)
a.extend([4.5,6.3,6.8])
print(a)
a.insert(2,3.8)
print(a)
```

Output

```
array('d', [1.1, 2.1, 3.1, 3.4])

array('d', [1.1, 2.1, 3.1, 3.4, 4.5, 6.3, 6.8])

array('d', [1.1, 2.1, 3.8, 3.1, 3.4, 4.5, 6.3, 6.8])
```

Q58. How to remove values to a Python array?

Ans: Array elements can be removed using **`pop()`** or **`remove()`** method. The difference between these two functions is that the former returns the deleted value whereas the latter does not.

Example

```
a=arr.array('d', [1.1, 2.2, 3.8, 3.1, 3.7, 1.2, 4.6])
print(a.pop())
print(a.pop(3))
a.remove(1.1)
print(a)
```

Output

```
4.6

3.1

array('d', [2.2, 3.8, 3.7, 1.2])
```

Q59. Does Python have OOPS concepts?

Ans: Python is an object-oriented programming language. This means that any program can be solved in python by creating an object model. However, Python can be treated as a procedural as well as a structural language.

Q60. What is the difference between deep and shallow copy?

Ans: Shallow copy is used when a new instance type gets created and it keeps the values that are copied in the new instance. Shallow copy is used to copy the reference pointers just like it copies the values. These references point to the original objects, and the changes made to any member of the class will also affect the original copy of it. Shallow copy allows faster execution of the program, and it depends on the size of the data that is used.

Deep copy is used to store the values that are already copied. Deep copy doesn't copy the reference pointers to the objects. It makes a reference to an object, and the new object that is pointed by some other object gets stored. The changes made in the original copy won't affect any other copy that uses the object. Deep copy makes execution of the program slower due to making certain copies for each object that is been called.

Q61. How is multithreading achieved in Python?
Ans:

1. Python has a multithreading package, but if you want to multi-thread to speed your code up, then it's usually not a good idea to use it.

2. Python has a construct called the Global Interpreter Lock (GIL). The GIL makes sure that only one of your 'threads' can be executed at any one time. A thread acquires the GIL, does a little work, then passes the GIL onto the next thread.
3. This happens very quickly, so to the human eye, it may seem like your threads are executing in parallel, but they are really just taking turns using the same CPU core.
4. All this GIL passing adds overhead to execution. This means that if you want to make your code run faster, then using the threading package often isn't a good idea.

Q62. What is the process of compilation and linking in Python?

Ans: The compiling and linking allow the new extensions to be compiled properly without any error, and the linking can be done only when it passes the compiled procedure. If dynamic loading is used, then it depends on the style that is being provided with the system. The Python interpreter can be used to provide dynamic loading of the configuration setup files and will rebuild the interpreter.

The steps that are required in this as:

1. Create a file with any name and in any language that is supported by the compiler of your system. For example, file.c or file.cpp.
2. Place this file in the Modules/directory of the distribution which is being used.
3. Add a line in the file Setup.local that is present in the Modules/directory.
4. Run the file using spam file.o.
5. After a successful run of this, rebuild the interpreter by using the make command on the top-level directory.
6. If the file is changed, then run rebuildMakefile by using the command as 'make Makefile'.

Q63. What are Python libraries? Name a few of them.

Python libraries are a collection of Python packages. Some of the majorly used python libraries are Numpy, Pandas, Matplotlib, Scikit-learn, and many more.

Q64. What is split used for?

The `split()` method is used to separate a given String in Python.

Example

```
a="edureka python"
print(a.split())
```

Output

```
['edureka', 'python']
```

Q65. How to import modules in Python?

Modules can be imported using the **import** keyword. You can import modules in three ways:

Example

```
import array              #importing using the original module
                          #name
import array as arr       # importing using an alias name
from array import *       #imports everything present in the
                          #array module
```

OOPS Python Interview Questions

Q66. Explain Inheritance in Python with an example.

Ans: Inheritance allows one class to gain all the members (say attributes and methods) of another class. Inheritance provides code reusability and makes it easier to create and maintain an application. The class from which we are inheriting is called a super-class, and the class that is inherited is called a derived/child class.
There are different types of inheritance supported by Python:

1. **Single Inheritance** – a derived class acquires the members of a single superclass.
2. **Multi-level inheritance** – a derived class d1 is inherited from base class base1, and d2 is inherited from base2.
3. **Hierarchical inheritance** – from one base class you can inherit any number of child classes.
4. **Multiple inheritance** – a derived class is inherited from more than one base class.

Q67. How are classes created in Python?

Ans: Class in Python is created using the **class** keyword.

Example

```
class Employee:
def __init__(self, name):
self.name = name
E1=Employee("abc")
print(E1.name)
```

Q68. What is monkey patching in Python?

Ans: In Python, the term monkey patching only refers to dynamic modifications of a class or module at run-time.

Consider the below example:

```
# m.py
class MyClass:
```

```
def f(self):
print "f()"
```

We can then run the monkey-patch testing like this:

```
import m
def monkey_f(self):
print "monkey_f()"

m.MyClass.f = monkey_f
obj = m.MyClass()
obj.f()
```

The output will be as below:

```
monkey_f()
```

As we can see, we did make some changes in the behavior of *f()* in *MyClass* using the function we defined, *monkey_f()*, outside of the module *m*.

Q69. Does Python support multiple inheritance?

Ans: Multiple inheritance means that a class can be derived from more than one parent class. Python does support multiple inheritance, unlike Java.

Q70. What is Polymorphism in Python?

Ans: Polymorphism means the ability to take multiple forms. So, for instance, if the parent class has a method named ABC, then the child class can also have a method with the same name ABC, having its parameters and variables. Python allows polymorphism.

Q71. Define encapsulation in Python.

Ans: Encapsulation means binding the code and the data together. A Python class is an example of encapsulation.

Q72. How do you do data abstraction in Python?

Ans: Data Abstraction provides only the required details and hides the implementation from the world. It can be achieved in Python by using interfaces and abstract classes.

Q73. Does python make use of access specifiers?

Ans: Python does not deprive access to an instance variable or function. Python lays down the concept of prefixing the name of the variable, function, or method with a single or double underscore to imitate the behavior of protected and private access specifiers.

Q74. How to create an empty class in Python?

Ans: An empty class is a class that does not have any code defined within its block. It can be created using the pass keyword. However, you can create objects of this class outside the class itself.

In Python, the pass command does nothing when it's executed. It's a null statement.

Example

```
class a:
  pass
obj=a()
obj.name="xyz"
print("Name = ",obj.name)
```

Output

```
Name = xyz
```

Q75. What does an `object()` do?

Ans: It returns a featureless object that is a base for all classes. Also, it does not take any parameters.
Next, let us have a look at some basic Python programs in these Python Interview Questions.

Basic Python Programs (Python Interview Questions)

Q76. Write a program in Python to execute the Bubble sort algorithm.

Ans:

```
def bs(a):
# a = name of list
   b=len(a)-1nbsp;
# minus 1 because we always compare 2 adjacent values
   for x in range(b):
        for y in range(b-x):
              a[y]=a[y+1]


   a=[32,5,3,6,7,54,87]
   bs(a)
```

Output

```
 [3, 5, 6, 7, 32, 54, 87]
```

Q77. Write a program in Python to produce a Star triangle.

Ans:

```
def pyfunc(r):
    for x in range(r):
        print(' '*(r-x-1)+'*'*(2*x+1))
pyfunc(9)
```

Output

```
        *
       ***
      *****
     *******
    *********
   ***********
  *************
 ***************
*****************
```

Q78. Write a program to produce the Fibonacci series in Python.

Ans:

```
# Enter number of termsneedednbsp;#0,1,1,2,3,5....
a=int(input("Enter the terms"))
f=0;#first element of series
s=1#second element of series
if a=0:
   print("The requested series is",f)
else:
  print(f,s,end=" ")
   for x in range(2,a):
         print(next,end=" ")
         f=s
         s=next
```

Output

Enter the terms 5 0 1 1 2 3

Q79. Write a program in Python to check if a number is prime.

Ans:

```
a=int(input("enter number"))
if a=1:
   for x in range(2,a):
         if(a%x)==0:
          print("not prime")
   break
   else:
      print("Prime")
else:
    print("not prime")
```

Output
enter number 3
Prime

Q80. Write a program in Python to check if the sequence is a Palindrome.

Ans:
```
a=input("enter sequence")
b=a[::-1]
if  a==b:
  print("palindrome")
else:
  print("Not a Palindrome")
```

Output
enter sequence 323 palindrome

Q81. Write a one-liner that will count the number of capital letters in a file. Your code should work even if the file is too big to fit in memory.

Ans: Let us first write a multiple-line solution and then convert it to one-liner code.

```
with open(SOME_LARGE_FILE) as fh:
count = 0
text = fh.read()
for character in text:
    if character.isupper():
count += 1
```

We will now try to transform this into a single line.

```
count sum(1 for line in fh for character in line if
character.isupper())
```

Q82. Write a sorting algorithm for a numerical dataset in Python.

Ans: The following code can be used to sort a list in Python:

```
list = ["1", "4", "0", "6", "9"]
list = [int(i) for i in list]
list.sort()
print (list)
```

Q83. Looking at the below code, write down the final values of A0, A1, …An.

Ans:

```
A0 = dict(zip(('a','b','c','d','e'),(1,2,3,4,5)))
A1 = range(10)A2 = sorted([i for i in A1 if i in A0])
A3 = sorted([A0[s] for s in A0])
A4 = [i for i in A1 if i in A3]
A5 = {i:i*i for i in A1}
A6 = [[i,i*i] for i in A1]
print(A0,A1,A2,A3,A4,A5,A6)
```

The following will be the final outputs of A0, A1, … A6

A0 = {'a': 1, 'c': 3, 'b': 2, 'e': 5, 'd': 4} # the order may vary
A1 = range(0, 10)
A2 = []
A3 = [1, 2, 3, 4, 5]
A4 = [1, 2, 3, 4, 5]
A5 = {0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81}
A6 = [[0, 0], [1, 1], [2, 4], [3, 9], [4, 16], [5, 25], [6, 36], [7, 49], [8, 64], [9, 81]]

Python Libraries (Python Interview Questions)

Q84. Explain what Flask is and its benefits.

Ans: Flask is a web microframework for Python based on “Werkzeug, Jinja2 and good intentions” BSD license. Werkzeug and Jinja2 are two of their dependencies. This means it will have little to no dependencies on external libraries. It makes the framework light while there is a little dependency to update and fewer security bugs.

A session allows you to remember information from one request to another. In a flask, a session uses a signed cookie so the user can look at the session contents and modify them. The user can modify the session if only they have the secret key, Flask.secret_key.

Q85. Is Django better than Flask?

Ans: Django and Flask map the URLs or addresses typed in the web browsers to functions in Python.

Flask is much simpler compared to Django, but Flask does not do a lot for you, meaning you will need to specify the details, whereas Django does a lot for you wherein you would not need to do much work. Django consists of prewritten code, which the user will need to analyze whereas Flask gives the users to create their code, therefore, making it simpler to understand the code. Technically, both are equally good, and both have their pros and cons.

Q86. Mention the differences between Django, Pyramid, and Flask.

Ans:

- Flask is a “microframework” primarily built for a small application with simpler requirements. In a flask, you must use external libraries. Flask is ready to use.
- Pyramid is built for larger applications. It provides flexibility and lets the developer use the right tools for their project. The developer can choose the database, URL structure, templating style, and more. The pyramid is heavily configurable.
- Django can also be used for larger applications, just like Pyramid. It includes an ORM.

Q87. Discuss Django architecture.

Ans:

Django MVT Pattern:

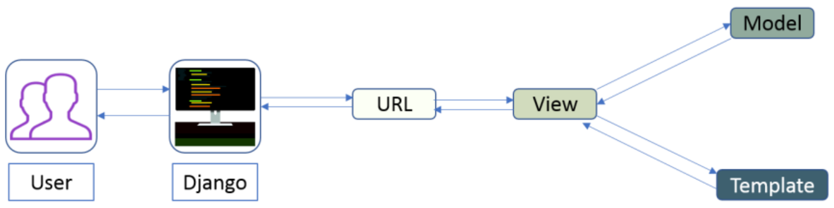

Figure: *Python Interview Questions – Django Architecture*

The developer provides the model, the view, and the template, then just maps it to a URL and Django does the magic to serve it to the user.

Q88. Explain how you can set up the Database in Django.

Ans: You can use the command edit mysite/setting.py. It is a normal Python module with module-level representing Django settings.

Django uses SQLite by default; it is easy for Django users as such it won’t require any other type of installation. In this case, your database choice is different that you have the following keys in the DATABASE ‘default’ item to match your database connection settings.

- **Engines**: You can change the database by using ‘django.db.backends.sqlite3’ , ‘django.db.backeneds.mysql’, ‘django.db.backends.postgresql_psycopg2’, ‘django.db.backends.oracle’ and so on.
- **Name**: The name of your database. In case you are using SQLite as your database, in that case, the database will be a file on your computer, Name should be a full absolute path, including the file name of that file.
- If you are not choosing SQLite as your database, then settings like password, host, user, and so on must be added.

Django uses SQLite as a default database. It stores data as a single file in the filesystem. If you do have a database server—PostgreSQL, MySQL, Oracle, MSSQL—and want to use it rather than SQLite, then use your database's administration tools to create a new database for your Django project. Either way, with your (empty) database in place, all that remains is to tell Django how to use it. This is where your project's settings.py file comes in.

We will add the following lines of code to the *setting.py* file:

```
DATABASES = {
     'default': {
          'ENGINE'  : 'django.db.backends.sqlite3',
          'NAME'  : os.path.join(BASE_DIR, 'db.sqlite3'),
     }
}
```

Q89. Give an example of how you can write a VIEW in Django.

Ans: This is how we can use write a view in Django:

```
from django.http import HttpResponse
import datetime

def Current_datetime(request):
     now = datetime.datetime.now()
     html = "It is now %s/body/html % now
     return HttpResponse(html)
```

Returns the current date and time, as an HTML document

Q90. Mention what the Django templates consist of.

Ans: The template is a simple text file. It can create any text-based format such as XML, CSV, and HTML. A template contains variables that get replaced with values when the template is evaluated and tags (% tag %) that control the logic of the template.

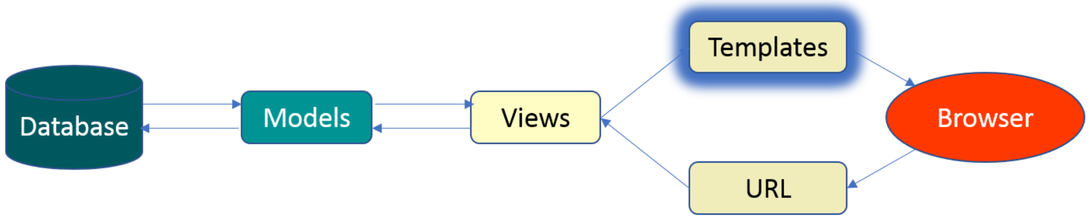

Figure: *Python Interview Questions – Django Template*

Q91. Explain the use of the session in the Django framework.

Ans: Django provides a session that lets you store and retrieve data on a per-site-visitor basis. Django abstracts the process of sending and receiving cookies by placing a session ID cookie on the client side and storing all the related data on the server side.

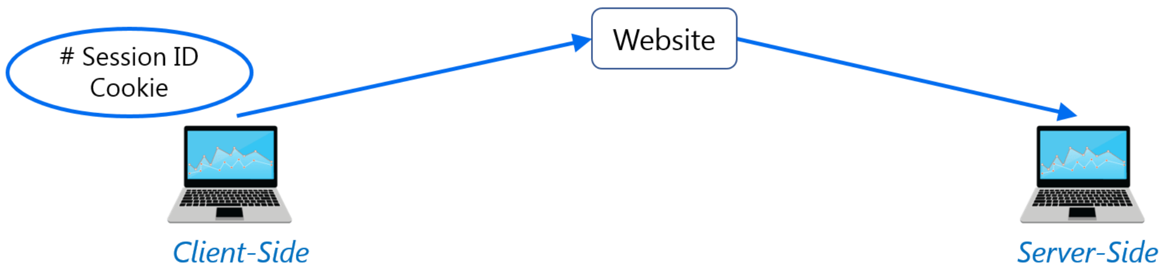

Figure: *Python Interview Questions – Django Framework*

So, the data itself is not stored on the client side. This is nice from a security perspective.

Q92. List out the inheritance styles in Django.

Ans: In Django, there are three possible inheritance styles:

1. Abstract Base Classes: This style is used when you only want the parent's class to hold information that you don't want to type out for each child model.
2. Multi-table Inheritance: This style is used if you are sub-classing an existing model and need each model to have its database table.
3. Proxy models: You can use this model if you only want to modify the Python-level behavior of the model, without changing the model's fields.

Web Scraping (Python Interview Questions)

Q93. How to save an image locally using Python whose URL address I already know?

Ans: We will use the following code to save an image locally from an URL address.

```
import urllib.request
urllib.request.urlretrieve("URL", "local-filename.jpg")
```

Q94. How can you get the Google cache age of any URL or web page?

Ans: Use the following URL format:
http://webcache.googleusercontent.com/search?q=cache:URLGOESHERE

Be sure to replace "URLGOESHERE" with the proper web address of the page or site whose cache you want to retrieve and see the time for. For example, to check the Google Web cache age of edureka.co you'd use the following URL:
http://webcache.googleusercontent.com/search?q=cache:edureka.co

Q95. You are required to scrap data from IMDb's top 250 movies page. It should

only have fields like movie name, year, and rating.

Ans: We will use the following lines of code:

```
from bs4 import BeautifulSoup

import requests
import sys

url = '<a
href="http://www.imdb.com/chart/top">http://www.imdb.com
/chart/top</a>'
response = requests.get(url)
soup = BeautifulSoup(response.text)
tr = soup.findChildren("tr")
tr = iter(tr)
next(tr)

for movie in tr:
title = movie.find('td', {'class': 'titleColumn'}
).find('a').contents[0]
year = movie.find('td', {'class': 'titleColumn'}
).find('span', {'class': 'secondaryInfo'}).contents[0]
rating = movie.find('td', {'class': 'ratingColumn
imdbRating'} ).find('strong').contents[0]
row = title + ' - ' + year + ' ' + ' ' + rating

print(row)
```

The above code will help scrape data from IMDb's top 250 list.

Data Analysis (Python Interview Questions)

Q96. What is map function in Python?

Ans: map function executes the function given as the first argument on all the elements of the iterable given as the second argument. If the function given takes more than 1 argument, then many iterables are given.

Q97. Is Python numpy better than lists?

Ans: We use Python numpy array instead of a list because of the below three reasons:

1. Less Memory
2. Fast
3. Convenient

Q98. How do get indices of N maximum values in a NumPy array?

Ans: We can get the indices of N maximum values in a NumPy array using the below code:

```
import numpy as np
arr = np.array([1, 3, 2, 4, 5])
print(arr.argsort()[-3:][::-1])
```

Output

```
[ 4 3 1 ]
```

Q99. How do you calculate percentiles with Python/ NumPy?

Ans: We can calculate percentiles with the following code.

```
import numpy as np
a = np.array([1,2,3,4,5])
p = np.percentile(a, 50) #Returns 50th percentile, e.g. median
print(p)
```

Output:3

Q100. What is the difference between NumPy and SciPy?
Ans:

NumPy	SciPy
It refers to Numerical Python.	It refers to Scientific Python.
It has fewer new scientific computing features.	Most new scientific computing features belong in SciPy.
It contains less linear algebra functions.	It has more fully featured versions of the linear algebra modules, as well as many other numerical algorithms.
NumPy has a faster processing speed.	SciPy on the other hand has a slower computational speed.

Q101. How do you make 3D plots/visualizations using NumPy/SciPy?

Ans: Like 2D plotting, 3D graphics is beyond the scope of NumPy and SciPy, but just as in the 2D case, packages exist that integrate with NumPy. Matplotlib provides basic 3D plotting in the mplot3d subpackage, whereas Mayavi provides a wide range of high-quality 3D visualization features, utilizing the powerful VTK engine.

Multiple Choice Questions (Python Interview Questions)

Q102. Which of the following statements create a dictionary? (Multiple Correct Answers are Possible)

a) d = {}
b) d = {“john”:40, “peter”:45}

c) d = {40:”john”, 45:”peter”}
d) d = (40:”john”, 45:”50”)

Answer: b, c, and d.

Dictionaries are created by specifying keys and values.

Q103. Which one of these is floor division?

a) /
b) //
c) %
d) None of the mentioned

Answer: b

When both of the operands are integer then python chops out the fraction part and gives you the round off value, to get the accurate answer use floor division. For example, 5/2 = 2.5 but both of the operands are integer so answer of this expression in python is 2. To get the 2.5 as the answer, use floor division using //. So, 5//2 = 2.5

Q104. What is the maximum possible length of an identifier?

a) 31 characters
b) 63 characters
c) 79 characters
d) None of the above

Answer: d

Identifiers can be of any length.

Q105. Why are local variable names beginning with an underscore discouraged?

a) They are used to indicate a private variable of a class.
b) They confuse the interpreter.
c) They are used to indicate global variables.
d) They slow down execution.

Answer: a

As Python has no concept of private variables, leading underscores are used to indicate variables that must not be accessed from outside the class.

Q106. Which of the following is an invalid statement?

a) abc = 1,000,000

b) a b c = 1000 2000 3000

c) a,b,c = 1000, 2000, 3000

d) a_b_c = 1,000,000

Answer: b

Spaces are not allowed in variable names.

Q107. What is the output of the following?

```
try:
    if '1' != 1:
        raise "someError"
    else:
        print("someError has not occured")
except "someError":
    print ("someError has occured")
```

a) someError has occured
b) someError has not occured
c) invalid code
d) none of the above

Answer: c

A new exception class must inherit from a BaseException. There is no such inheritance here.

Q108. Suppose list1 is [2, 33, 222, 14, 25], What is list1[-1]?

a) Error

b) None

c) 25

d) 2

Answer: c

The index -1 corresponds to the last index in the list.

Q109. To open a file c:scores.txt for writing, we use _____.

a) outfile = open(“c:scores.txt”, “r”)

b) outfile = open(“c:scores.txt”, “w”)

c) outfile = open(file = “c:scores.txt”, “r”)

d) outfile = open(file = “c:scores.txt”, “o”)

Answer: b

w is used to indicate that file is open in write mode.

Q110. What is the output of the following?

```
f = None

for i in range (5):
    with open("data.txt", "w") as f:
        if (i > 2):
            break

print f.closed
```

a) True
b) False
c) None
d) Error

Answer: a

The WITH statement when used with open file guarantees that the file object is closed when the with block exits.

Q111. When will the else part of try-except-else be executed?

a) always
b) when an exception occurs
c) when no exception occurs
d) when an exception occurs into except block

Answer: c

The else part is executed when no exception occurs.