Top 50 OOPs Interview Questions & Answers

Here are OOPs interview questions and answers for fresher as well experienced candidates to get their dream job.

1) What is OOPS?

OOPS is abbreviated as Object Oriented Programming system in which programs are considered as a collection of objects. Each object is nothing but an instance of a class.

2) Write basic concepts of OOPS?

Following are the concepts of OOPS:

1. Abstraction
2. Encapsulation
3. Inheritance
4. Polymorphism

3) What is a class?

A class is simply a representation of a type of object. It is the blueprint/plan/template that describes the details of an object.

---

4) What is an Object?

An object is an instance of a class. It has its own state, behavior, and identity.

---

5) What is Encapsulation?

Encapsulation is an attribute of an object, and it contains all data which is hidden. That hidden data can be restricted to the members of that class.

Levels are Public, Protected, Private, Internal, and Protected Internal.

6) What is Polymorphism?

Polymorphism is nothing but assigning behavior or value in a subclass to something that was already declared in the main class. Simply, polymorphism takes more than one form.

---

7) What is Inheritance?

Inheritance is a concept where one class shares the structure and behavior defined in another class. If Inheritance applied to one class is called Single Inheritance, and if it depends on multiple classes, then it is called multiple Inheritance.

---

8) What are manipulators?

Manipulators are the functions which can be used in conjunction with the insertion (<<) and extraction (>>) operators on an object. Examples are endl and setw.

---

9) Explain the term constructor

A constructor is a method used to initialize the state of an object, and it gets invoked at the time of object creation. Rules for constructor are:

- Constructor Name should be the same as a class name.
- A constructor must have no return type.

---

10) Define Destructor?

A destructor is a method which is automatically called when the object is made of scope or destroyed. Destructor name is also same as class name but with the tilde symbol before the name.

---

11) What is an Inline function?

An inline function is a technique used by the compilers and instructs to insert complete body of the function wherever that function is used in the program source code.

12) What is a virtual function?

A virtual function is a member function of a class, and its functionality can be overridden in its derived class. This function can be implemented by using a keyword called virtual, and it can be given during function declaration.

A virtual function can be declared using a token(virtual) in C++. It can be achieved in C/Python Language by using function pointers or pointers to function.

13) What is a friend function?

A friend function is a friend of a class that is allowed to access to Public, private, or protected data in that same class. If the function is defined outside the class cannot access such information.

A friend can be declared anywhere in the class declaration, and it cannot be affected by access control keywords like private, public, or protected.

14) What is function overloading?

Function overloading is a regular function, but it is assigned with multiple parameters. It allows the creation of several methods with the same name which differ from each other by the type of input and output of the function.

Example

```
void add(int& a, int& b);

void add(double& a, double& b);

void add(struct bob& a, struct bob& b);
```

15) What is operator overloading?

Operator overloading is a function where different operators are applied and depends on the arguments. Operator,-,* can be used to pass through the function, and it has its own precedence to execute

---

16) What is an abstract class?

An abstract class is a class which cannot be instantiated. Creation of an object is not possible with an abstract class, but it can be inherited. An abstract class can contain only an Abstract method. Java allows only abstract method in abstract class while other languages allow non-abstract method as well.

---

17) What is a ternary operator?

The ternary operator is said to be an operator which takes three arguments. Arguments and results are of different data types, and it depends on the function. The ternary operator is also called a conditional operator.

---

18) What is the use of finalize method?

Finalize method helps to perform cleanup operations on the resources which are not currently used. Finalize method is protected, and it is accessible only through this class or by a derived class.

---

19) What are the different types of arguments?

A parameter is a variable used during the declaration of the function or subroutine, and arguments are passed to the function body, and it should match with the parameter defined. There are two types of Arguments.

- Call by Value – Value passed will get modified only inside the function, and it returns the same value whatever it is passed into the function.
- Call by Reference – Value passed will get modified in both inside and outside the functions and it returns the same or different value.

20) What is the super keyword?

The super keyword is used to invoke the overridden method, which overrides one of its superclass methods. This keyword allows to access overridden methods and also to access hidden members of the superclass.

It also forwards a call from a constructor, to a constructor in the superclass.

21) What is method overriding?

Method overriding is a feature that allows a subclass to provide the implementation of a method that overrides in the main class. It will override the implementation in the superclass by providing the same method name, same parameter, and same return type.

22) What is an interface?

An interface is a collection of an abstract method. If the class implements an interface, it thereby inherits all the abstract methods of an interface.

Java uses Interface to implement multiple inheritances.

23) What is exception handling?

An exception is an event that occurs during the execution of a program. Exceptions can be of any type – Runtime exception, Error exceptions. Those exceptions are adequately handled through exception handling mechanism like try, catch, and throw keywords.

24) What are tokens?

A compiler recognizes a token, and it cannot be broken down into component elements. Keywords, identifiers, constants, string literals, and operators are examples of tokens.

Even punctuation characters are also considered as tokens. Example: Brackets, Commas, Braces, and Parentheses.

---

25) What is the main difference between overloading and overriding?

Overloading is static Binding, whereas Overriding is dynamic Binding. Overloading is nothing but the same method with different arguments, and it may or may not return the equal value in the same class itself.

Overriding is the same method names with the same arguments and return types associated with the class and its child class.

---

26) What is the main difference between a class and an object?

An object is an instance of a class. Objects hold multiple information, but classes don't have any information. Definition of properties and functions can be done in class and can be used by the object.

A class can have sub-classes, while an object doesn't have sub-objects.

---

27) What is an abstraction?

Abstraction is a useful feature of OOPS, and it shows only the necessary details to the client of an object. Meaning, it shows only required details for an object, not the inner constructors, of an object. Example – When you want to switch on the television, it is not necessary to know the inner circuitry/mechanism needed to switch on the TV. Whatever is required to switch on TV will be shown by using an abstract class.

---

28) What are the access modifiers?

Access modifiers determine the scope of the method or variables that can be accessed from other various objects or classes. There are five types of access modifiers, and they are as follows:

- Private
- Protected
- Public
- Friend
- Protected Friend

---

29) What are sealed modifiers?

Sealed modifiers are the access modifiers where the methods can not inherit it. Sealed modifiers can also be applied to properties, events, and methods. This modifier cannot be used to static members.

---

30) How can we call the base method without creating an instance?

Yes, it is possible to call the base method without creating an instance. And that method should be “Static method.”

Doing Inheritance from that class.-Use Base Keyword from a derived class.

---

31) What is the difference between new and override?

The new modifier instructs the compiler to use the new implementation instead of the base class function. Whereas, Override modifier helps to override the base class function.

---

32) What are the various types of constructors?

There are three types of constructors:

– Default Constructor – With no parameters.

– Parametric Constructor – With Parameters. Create a new instance of a class and also passing arguments simultaneously.

– Copy Constructor – Which creates a new object as a copy of an existing object.

33) What is early and late Binding?

Early binding refers to the assignment of values to variables during design time, whereas late Binding refers to the assignment of values to variables during run time.

34) What is ‘this’ pointer?

THIS pointer refers to the current object of a class. THIS keyword is used as a pointer which differentiates between the current object with the global object. It refers to the current object.

35) What is the difference between structure and a class?

The default access type of a Structure is public, but class access type is private. A structure is used for grouping data, whereas a class can be used for grouping data and methods. Structures are exclusively used for data, and it doesn’t require strict validation, but classes are used to encapsulate and inherent data, which requires strict validation.

36) What is the default access modifier in a class?

The default access modifier of a class is Internal and the default access modifier of a class member is Private.

37) What is a pure virtual function?

A pure virtual function is a function which can be overridden in the derived class but cannot be defined. A virtual function can be declared as Pure by using the operator =0.

Example –

```
Virtual void function1() // Virtual, Not pure
Virtual void function2() = 0 //Pure virtual
```

38) What are all the operators that cannot be overloaded?

Following are the operators that cannot be overloaded -.

1. Scope Resolution (::)
2. Member Selection (.)
3. Member selection through a pointer to function (.*)

39) What is dynamic or run time polymorphism?

Dynamic or Run time polymorphism is also known as method overriding in which call to an overridden function is resolved during run time, not at the compile time. It means having two or more methods with the same name, same signature but with different implementation.

40) Do we require a parameter for constructors?

No, we do not require a parameter for constructors.

41) What is a copy constructor?

This is a special constructor for creating a new object as a copy of an existing object. There will always be only one copy constructor that can be either defined by the user or the system.

42) What does the keyword virtual represented in the method definition?

It means we can override the method.

43) Whether static method can use nonstatic members?

False.

---

44) What are a base class, subclass, and superclass?

The base class is the most generalized class, and it is said to be a root class.

A Subclass is a class that inherits from one or more base classes.

The superclass is the parent class from which another class inherits.

---

45) What is static and dynamic Binding?

Binding is nothing but the association of a name with the class. Static Binding is a binding in which name can be associated with the class during compilation time, and it is also called as early Binding.

Dynamic Binding is a binding in which name can be associated with the class during execution time, and it is also called as Late Binding.

---

46) How many instances can be created for an abstract class?

Zero instances will be created for an abstract class. In other words, you cannot create an instance of an Abstract Class.

---

47) Which keyword can be used for overloading?

Operator keyword is used for overloading.

---

48) What is the default access specifier in a class definition?

Private access specifier is used in a class definition.

---

49) Which OOPS concept is used as a reuse mechanism?

Inheritance is the OOPS concept that can be used as a reuse mechanism.

---

50) Which OOPS concept exposes only the necessary information to the calling functions?

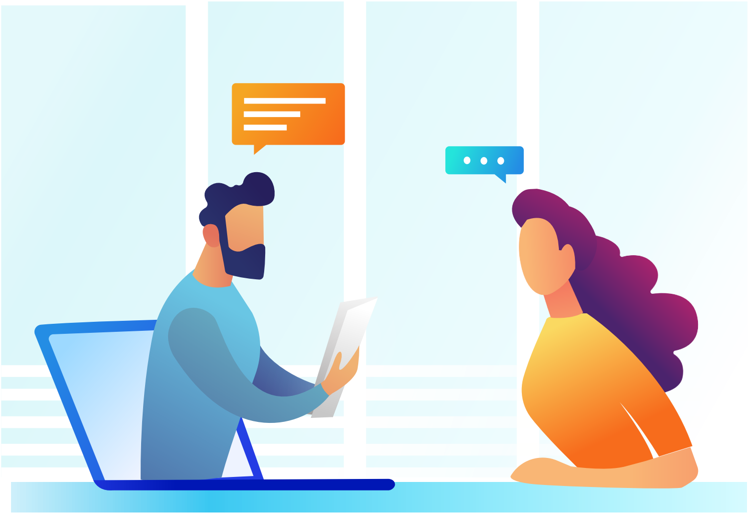
Encapsulation

Thanks

Top 100 Python Interview Questions You Must Prepare In 2019

Last updated on Aug 14,2019 *379.7K Views*

Aayushi Johari
A technophile who likes writing about different technologies and spreading knowledge.



Python Certification is the most sought-after skill in programming domain. In this Python Interview Questions blog, I will introduce you to the most frequently asked questions in Python interviews. Our Python Interview Questions is the one-stop resource from where you can boost your interview preparation. We have 100+ questions on **Python Programming** basics which will help you with different expertise levels to reap the maximum benefit from our blog.

Edureka 2019 Tech Career Guide is out! Hottest job roles, precise learning paths, industry outlook & more in the guide. ***Download*** now.
Let us start by taking a look at some of the most frequently asked Python interview questions,

Q1. What is the difference between list and tuples in Python?
Q2. What are the key features of Python?
Q3. What type of language is python?
Q4. How is Python an interpreted language?
Q5. What is pep 8?
Q6. How is memory managed in Python?
Q7. What is name space in Python?
Q8. What is PYTHON PATH?
Q9. What are python modules?
Q10. What are local variables and global variables in Python?

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

- Basic Interview Questions
- OOPS Interview Questions
- Basic Python Programs
- Python Libraries Interview Questions
- Web Scraping Interview Questions
- Data Analysis Interview Questions
- Multiple Choice Questions (MCQ)

Before moving ahead, you may go through the recording of Python Interview Questions where our instructor has shared his experience and expertise that will help you to crack any Python Interview:

Python Interview Questions And Answers | Python Training | Edureka

Python Interview Questions And Answers | Python Interview Preparati…

If you have other doubts regarding Python, feel free to post them in our **QnA Forum**. Our expert team will get back to you at the earliest.

Basic Python Interview Questions

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

LIST	TUPLES
Lists are mutable i.e they can be edited.	Tuples are immutable (tuples are lists which can't be edited).
Lists are slower than tuples.	Tuples are faster than list.
Syntax: list_1 = [10, 'Chelsea', 20]	Syntax: tup_1 = (10, 'Chelsea' , 20)

LIST vs TUPLES

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`, `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 general sense, it is considered as a general-purpose programming language. To know more about Scripting, you can refer to the Python Scripting Tutorial.

Q4.How is Python an interpreted language?

Ans: An interpreted language is any programming language which 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. How is memory managed in Python?

Ans:

1. Memory management in python is managed by ***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 and so that it can be made available to the heap space.

Q7. What is namespace in Python?

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

Q8. 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.

Q9. What are 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

Q10.What are local variables and global variables in Python?

Global Variables:

Variables declared outside a function or in 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.

Q11. Is python case sensitive?

Ans: Yes. Python is a case sensitive language.

Q12.What is type conversion in Python?

Ans: Type conversion refers to the conversion of one data type iinto 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 integer to octal

tuple() – This function is used to convert to a tuple.

set() – This function returns the type after converting to set.

list() – This function is used to convert any data type to a list type.

dict() – This function is used to convert a tuple of order (key,value) into a dictionary.

str() – Used to convert integer into a string.

complex(real,imag) – This functionconverts real numbers to complex(real,imag) number.

Q13. How to install Python on Windows and set path variable?

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%

Q14. Is indentation required in python?

Ans: Indentation is necessary for Python. It specifies a block of code. All code within loops, classes, functions, etc 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.

Q15. 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 elements whereas lists can hold any data type elements.

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]

Q16. What are functions in Python?

Ans: A function is a block of code which 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

Q17.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.

Here is an example of how to use it.

```
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

Q18.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

Q19. 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.

Q20. How does break, continue and pass work?

Break	Allows loop termination when some condition is met and the 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.

Q21. What does [::-1} do?

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

Q22. 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']
```

Q23. What are python iterators?

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

Q24. 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 creates a different instance of individual threads. The other random generators that are used in this are:

1. randrange(a, b): it chooses an integer and define 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 that is used and instantiated creates an independent multiple random number generators.

Q25. What is the difference between range & 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 like 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 generators. That means that if you have a really gigantic range 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 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.

Q26. 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:

```
#Comments in Python start like this
print("Comments in Python start with a #")
```

Output: Comments in Python start with a #

Q27. 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 dump function, this process is called pickling. While the process of retrieving original Python objects from the stored string representation is called unpickling.

Q28. What are the generators in python?

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

Q29. 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.

Q30. How will you convert a string to all lowercase?

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

Example:

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

Output: abcd

Q31. 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.

Q32.What are docstrings in Python?

Ans: Docstrings are not actually comments, but, they are ***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

Q33. 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

Q34. What is the usage of help() and dir() function 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, etc.
2. Dir() function: The dir() function is used to display the defined symbols.

Q35. Whenever Python exits, why isn't all the memory de-allocated?

Ans:

1. Whenever Python exits, especially those Python modules which are having circular references to other objects or the objects that are referenced from the global namespaces 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.

Q36. What is a dictionary in Python?

Ans: The built-in datatypes in Python is called dictionary. It defines 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 & PM. Their corresponding values are India, Delhi and Modi respectively.

```
dict={'Country':'India','Capital':'Delhi','PM':'Modi'}
```

```
print dict[Country]
```

```
India
```

```
print dict[Capital]
```

```
Delhi
```

```
print dict[PM]
```

```
Modi
```

Q37. How can the ternary operators be used in python?

Ans: The Ternary operator is the operator that is used to show the 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.

Q38. What does this mean: *args, **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.

Q39. What does len() do?

Ans: It is used to determine the length of a string, a list, an array, etc.

Example:

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

Q40. Explain split(), sub(), 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 no. of replacements.

Q41. What are negative indexes and why are they used?

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

The index for the negative number starts from '-1' that 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 allow 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 correct order.

Q42. What are Python packages?

Ans: Python packages are namespaces containing multiple modules.

Q43.How can files be deleted in Python?

Ans: To delete a file 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")
```

Q44. 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

Q45. 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 mean 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 allow 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, etc.

Q46. How to add values to a python array?

Ans: Elements can be added to an array using the **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])

Q47. 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])

Q48. 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 procedural as well as structural language.

Q49. 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 in 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 the 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.

Q50. How is Multithreading achieved in Python?

Ans:

1. Python has a multi-threading 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 execute 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.

Q51. What is the process of compilation and linking in python?

Ans: The compiling and linking allows the new extensions to be compiled properly without any error and the linking can be done only when it passes the compiled procedure. If the 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 the 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 getting 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'.

Q52. 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.

Q53. 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']

Q54. 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 Interview Questions

Q55. 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, makes it easier to create and maintain an application. The class from which we are inheriting is called super-class and the class that is inherited is called a derived / child class.

They are different types of inheritance supported by Python:

1. Single Inheritance – where a derived class acquires the members of a single super class.
2. Multi-level inheritance – a derived class d1 in inherited from base class base1, and d2 are 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.

Q56. 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)
```

Output: abc

Q57. What is monkey patching in Python?

Ans: In Python, the term monkey patch 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*.

Q58. Does python support multiple inheritance?

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

Q59. 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 also can have a method with the same name ABC having its own parameters and variables. Python allows polymorphism.

Q60. Define encapsulation in Python?

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

Q61. How do you do data abstraction in Python?

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

Q62.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.

Q63. 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 its executed. it's a null statement.

For example-

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

Output:

```
Name =  xyz
```

Q64. 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.

Basic Python Programs

Q65. Write a program in Python to execute the Bubble sort algorithm.

```
def bs(a):&nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp;# a = name of list
  &nbsp; b=len(a)-1&nbsp; &nbsp; &nbsp; &nbsp; &nbsp;# minus 1 because we always compare 2 adjacent values
  &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp;
  &nbsp; for x in range(b):
  &nbsp; &nbsp; &nbsp; for y in range(b-x):
  &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; if a[y]>a[y+1]:
  &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; a[y],a[y+1]=a[y+1],a[y]
  &nbsp; return a
a=[32,5,3,6,7,54,87]
bs(a)
```

Output: [3, 5, 6, 7, 32, 54, 87]

Q66. Write a program in Python to produce Star triangle.

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

Output:

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

Q67. Write a program to produce Fibonacci series in Python.

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

Output: Enter the terms 5 0 1 1 2 3

Q68. Write a program in Python to check if a number is prime.

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

Output:

enter number 3

Prime

Q69. Write a program in Python to check if a sequence is a Palindrome.

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

Output:

enter sequence 323 palindrome

Q70. 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())
```

Q71. 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)
```

Q72. Looking at the below code, write down the final values of A0, A1, …An.

```
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)
```

Ans: 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 Interview Questions

Q73. 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 its 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 basically 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. The user can modify the session if only it has the secret key Flask.secret_key.

Q74. Is Django better than Flask?

Ans: Django and Flask map the URL’s 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 own code, therefore, making it simpler to understand the code. Technically both are equally good and both contain their own pros and cons.

Q75. Mention the differences between Django, Pyramid and Flask.

Ans:

- Flask is a “microframework” primarily build for a small application with simpler requirements. In flask, you have to 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. Pyramid is heavy configurable.
- Django can also be used for larger applications just like Pyramid. It includes an ORM.

Q76. Discuss Django architecture.

Ans: Django MVT Pattern:

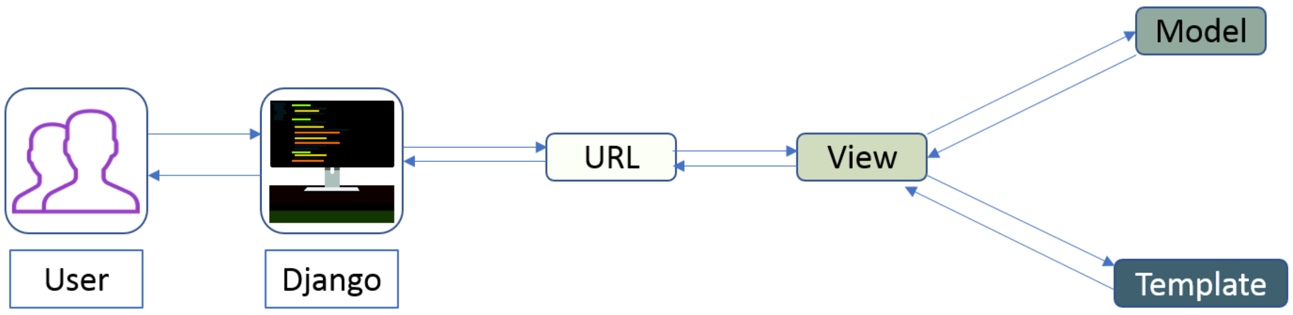


Figure: *Python Interview Questions – Django Architecture*

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.

Q77. 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 the case your database choice is different that you have to 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 the case if you are using SQLite as your database, in that case, 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, etc. 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'),
     }
}
```

Q78. Give an example 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 = "<html><body>It is now %s</body></html> % now
     return HttpResponse(html)
```

Returns the current date and time, as an HTML document

Q79. Mention what the Django templates consist of.

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

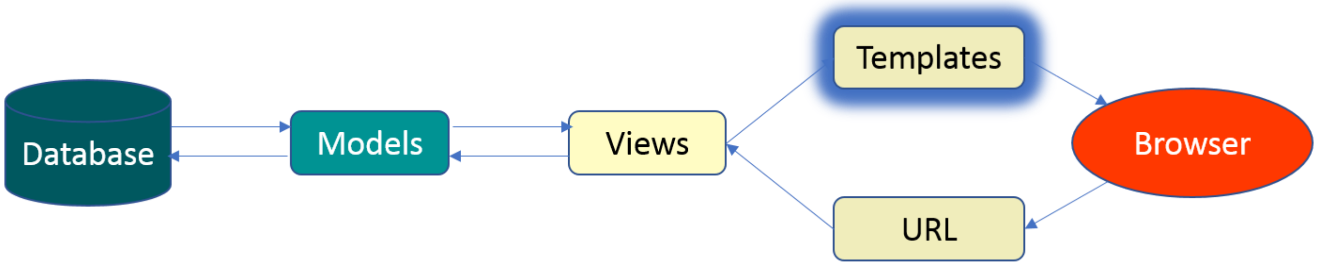


Figure: *Python Interview Questions – Django Template*

Q80. Explain the use of session in 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.

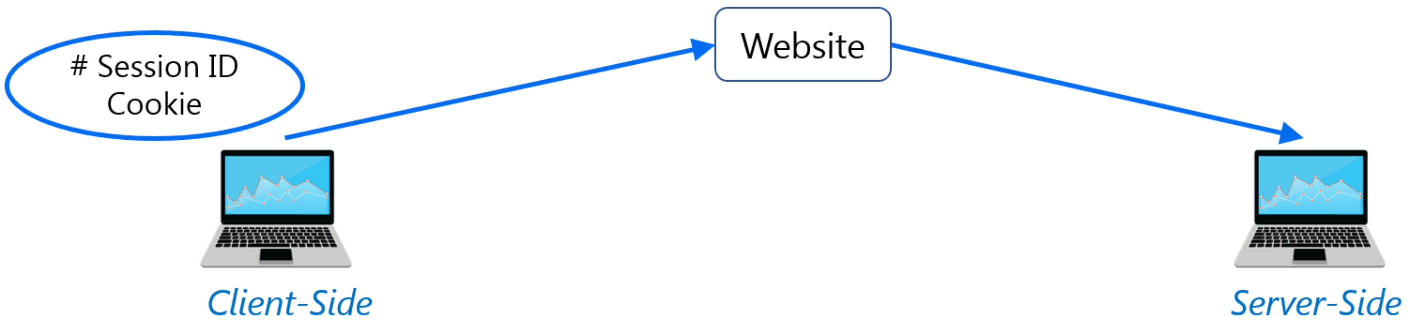


Figure: *Python Interview Questions – Django Framework*

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

Q81. 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 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 own 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

Q82. 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")
```

Q83. 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 Webcache age of edureka.co you'd use the following URL:

`http://webcache.googleusercontent.com/search?q=cache:edureka.co`

Q84. You are required to scrap data from IMDb top 250 movies page. It should only have fields movie name, year, and rating.

Ans: We will use the following lines of code:

```
from bs4 import BeautifulSoup

import requests
import sys

url = 'http://www.imdb.com/chart/top'
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 scrap data from IMDb's top 250 list

Data Analysis – Python Interview Questions

Q85. 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 in more than 1 arguments, then many iterables are given. #Follow the link to know more similar functions.

Q86. 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

For more information on these parameters, you can refer to this section – Numpy Vs List.

Q87. How to 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 ]
```

Q88. 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
```

Q89. What is the difference between NumPy and SciPy?

Ans:

1. In an ideal world, NumPy would contain nothing but the array data type and the most basic operations: indexing, sorting, reshaping, basic elementwise functions, et cetera.

2. All numerical code would reside in SciPy. However, one of NumPy's important goals is compatibility, so NumPy tries to retain all features supported by either of its predecessors.
3. Thus NumPy contains some linear algebra functions, even though these more properly belong in SciPy. In any case, SciPy contains more fully-featured versions of the linear algebra modules, as well as many other numerical algorithms.
4. If you are doing scientific computing with python, you should probably install both NumPy and SciPy. Most new features belong in SciPy rather than NumPy.

Q90. 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 (MCQ)

Q91. Which of the following statements create a dictionary? (Multiple Correct Answers Possible)

a) d = {}
b) d = {"john":40, "peter":45}
c) d = {40:"john", 45:"peter"}
d) d = (40:"john", 45:"50")

Answer: b, c & d.

Dictionaries are created by specifying keys and values.

Q92. 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 ex, 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

Q93. 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) None of the above

Identifiers can be of any length.

Q94. Why are local variable names beginning with an underscore discouraged?

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

Answer: a) they are used to indicate a private variable of a class

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

Q95. 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) a b c = 1000 2000 3000

Spaces are not allowed in variable names.

Q96. 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) invalid code

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

Q97. Suppose list1 is [2, 33, 222, 14, 25], What is list1[-1] ?

a) Error
b) None
c) 25
d) 2

Answer: c) 25

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

Q98. 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) The location contains double slashes ( ) and w is used to indicate that file is being written to.

Q99. 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) True

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

Q100. 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) when no exception occurs

The else part is executed when no exception occurs.

I hope this set of Python Interview Questions will help you in preparing for your interviews. All the best!

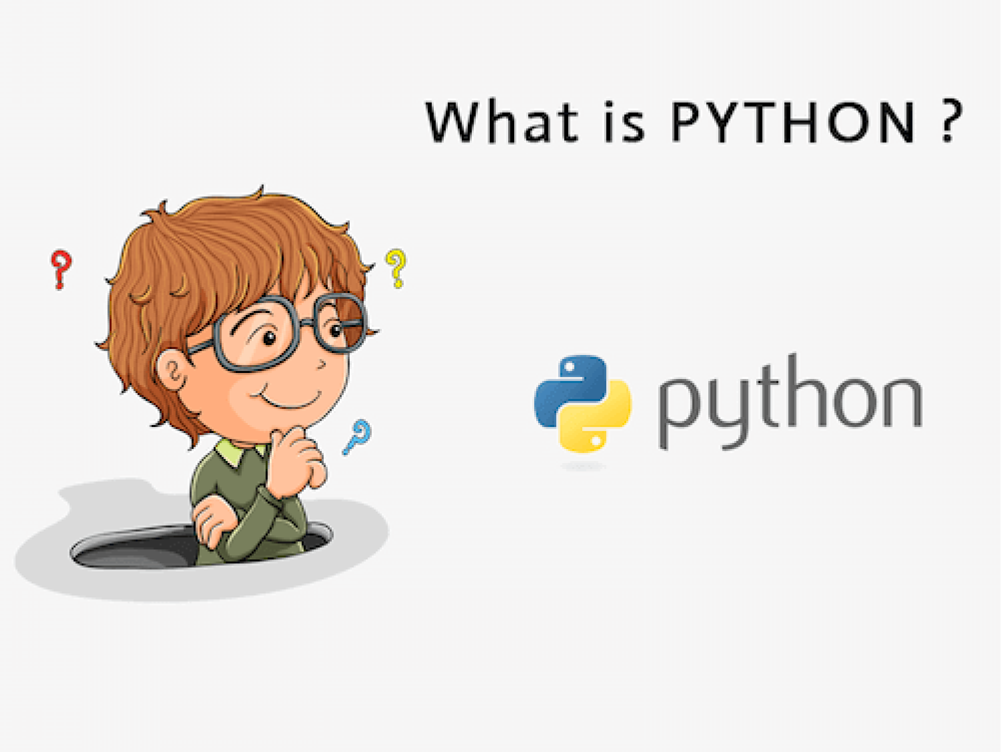
Got a question for us? Please mention it in the comments section and we will get back to you at the earliest.

If you wish to learn Python and gain expertise in quantitative analysis, data mining, and the presentation of data to see beyond the numbers by transforming your career into Data Scientist role, check out our interactive, live-online ***Python Certification Training****. You will use libraries like Pandas, Numpy, Matplotlib, Scipy, Scikit, Pyspark and master the concepts like Python machine learning, scripts, sequence, web scraping and big data analytics leveraging Apache Spark. The training comes with 24*7 support to guide you throughout your learning period.*

Python Interview Questions

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. Its high-level built-in data structures, combined with dynamic typing and dynamic binding. 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.

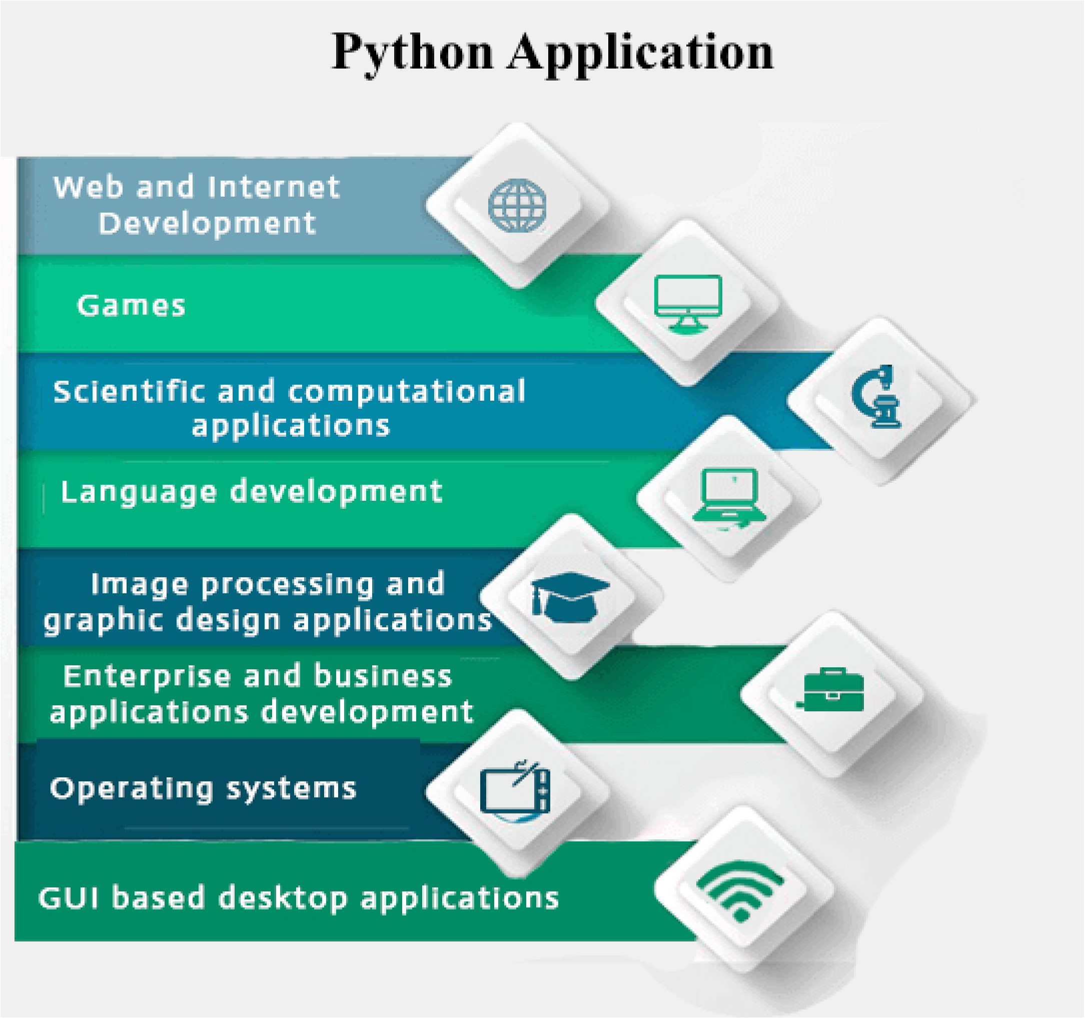
2) Why Python?

- Python is an interpreted, object-oriented, high-level programming language with dynamic semantics.
- 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.
- The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

3) 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 Application
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.

---

4) What are the advantages of Python?

Advantages of Python are:

- Python is **Interpreted** language

Interpreted: Python is an interpreted language. It does not require prior compilation of code and executes instructions directly.

- It is Free and open source

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

- It is **Extensible**

Extensible: It is very flexible and extensible with any module.

- Object-oriented

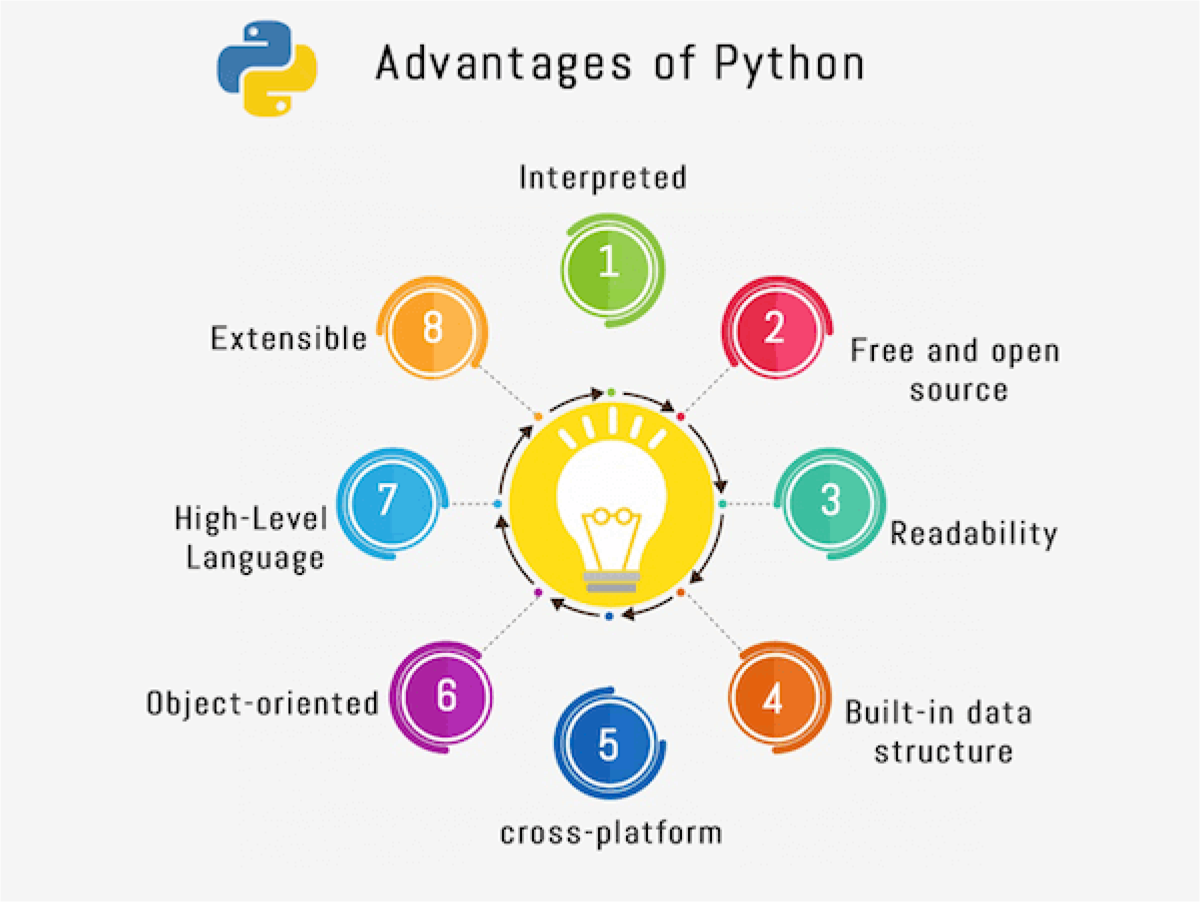
Object-oriented: Python allows to implement the Object-Oriented concepts to build application solution.

- It has Built-in data structure

Built-in data structure: Tuple, List, and Dictionary are useful integrated data structures provided by the language.

- Readability
- High-Level Language
- Cross-platform

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



5) What is PEP 8?

PEP 8 stands for **Python Enhancement Proposal**, it can be defined as a document that helps us to provide the guidelines on how to write the Python code. It is basically a set of rules that specify how to format Python code for maximum readability. It was written by Guido van Rossum, Barry Warsaw and Nick Coghlan in 2001.

6) 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.

Example:

```
# in single quotes
single = 'JavaTpoint'
# in double quotes
double = "JavaTpoint"
# multi-line String
multi = '''Java
           T
              point'''

print(single)
print(double)
print(multi)
```

Output:

```
JavaTpoint
JavaTpoint
Java
           T
               point
```

Numeric Literals

Python supports three types of numeric literals integer, float and complex.

Example:

```
# Integer literal
a = 10
#Float Literal
b = 12.3
#Complex Literal
x = 3.14j
print(a)
print(b)
print(x)
```

Output:

```
10
12.3
3.14j
```

Boolean Literals

Boolean literals are used to denote Boolean values. It contains either True or False.

Example:

```
p = (1 == True)
q = (1 == False)
r = True + 3
s = False + 7

print("p is", p)
print("q is", q)
print("r:", r)
print("s:", s)
```

Output:

```
p is True
q is False
r: 4
s: 7
```

Special literals

Python contains one special literal, that is, **'None'**. This special literal is used for defining a null variable. If 'None' is compared with anything else other than a 'None', it will return false.

Example:

```
word = None
print(word)
```

Output:

```
None
```

7) 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 three 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.
- **Anonymous functions:** These functions are also known as lambda functions because they are not declared with the standard def keyword.

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

```
def function_name(parameters list):
    #--- statements---
    return a_value
```

8) 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

```
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.

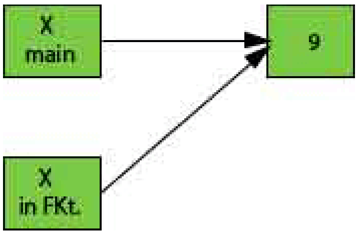
> Note: If the given lists are of different lengths, zip stops generating tuples when the first list ends. It means two lists are having 3, and 5 lengths will create a 3-tuple.

9) What is Python's parameter passing mechanism?

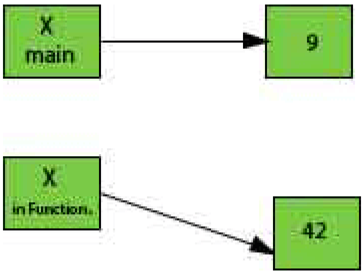
There are two parameters passing mechanism in Python:

- 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.

10) 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.

Example:

```
class student:
    def __init__(self, name):
        self.name = name
    def __init__(self, name, email):
        self.name = name
        self.email = email

# This line will generate an error
#st = student("rahul")

# This line will call the second constructor
st = student("rahul", "rahul@gmail.com")
print("Name: ", st.name)
print("Email id: ", st.email)
```

Output:

```
Name:  rahul
Email id:  rahul@gmail.com
```

11) What is the difference between remove() function and del statement?

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

Example:

```
list_1 = [ 3, 5, 7, 3, 9, 3 ]
print(list_1)
list_1.remove(3)
print("After removal: ", list_1)
```

Output:

```
[3, 5, 7, 3, 9, 3]
After removal: [5, 7, 3, 9, 3]
```

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

Example:

```python
list_1 = [ 3, 5, 7, 3, 9, 3 ]
print(list_1)
del list_1[2]
print("After deleting: ", list_1)
```

Output:

```
[3, 5, 7, 3, 9, 3]
After deleting: [3, 5, 3, 9, 3]
```

> 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.

12) 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.

Example:

```python
string = "IT IS IN LOWERCASE."
print(string.swapcase())

string = "it is in uppercase."
print(string.swapcase())
```

Output:

```
it is in lowercase.
IT IS IN UPPERCASE.
```

13) 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.

Example:

```
string = "  javatpoint "
string2 = "    javatpoint        "
string3 = "        javatpoint"
print(string)
print(string2)
print(string3)
print("After stripping all have placed in a sequence:")
print(string.strip())
print(string2.strip())
print(string3.strip())
```

Output:

```
javatpoint
    javatpoint
       javatpoint
After stripping all have placed in a sequence:
Javatpoint
javatpoint
javatpoint
```

14) 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.

Example:

```
string = "  javatpoint "
string2 = "    javatpoint        "
print(string)
print(string2)
print("After stripping all leading whitespaces:")
print(string.lstrip())
print(string2.lstrip())
```

Output:

```
javatpoint
    javatpoint
After stripping all leading whitespaces:
javatpoint
javatpoint
```

		j	a	v	a	t	p	o	i	n	t	

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

j	a	v	a	t	p	o	i	n	t

15) 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.

Example:

```
str = "Rohan"
str2 = "ab"
# Calling function
str2 = str.join(str2)
# Displaying result
print(str2)
```

Output:

```
aRohanb
```

16) 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.

Example:

```
# import the random module
import random
# declare a list
sample_list1 = ['Z', 'Y', 'X', 'W', 'V', 'U']
print("Original LIST1: ")
print(sample_list1)
# first shuffle
random.shuffle(sample_list1)
print("\nAfter the first shuffle of LIST1: ")
print(sample_list1)
# second shuffle
random.shuffle(sample_list1)
print("\nAfter the second shuffle of LIST1: ")
print(sample_list1)
```

Output:

```
Original LIST1:
['Z', 'Y', 'X', 'W', 'V', 'U']

After the first shuffle of LIST1:
['V', 'U', 'W', 'X', 'Y', 'Z']

After the second shuffle of LIST1:
['Z', 'Y', 'X', 'U', 'V', 'W']
```

17) What is the use of break statement?

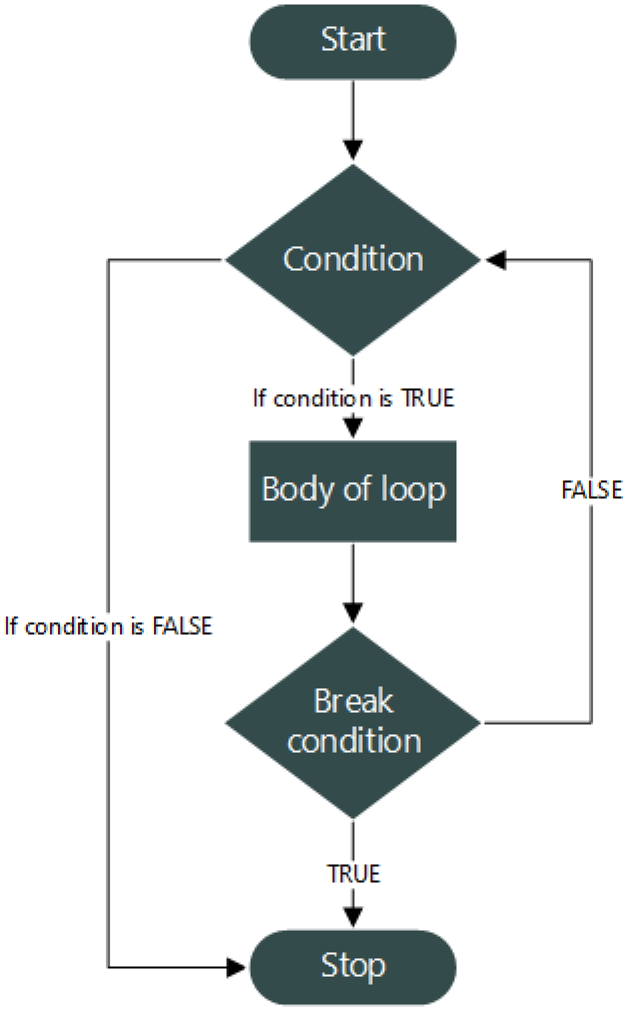
The break statement is used to terminate the execution of the current loop. Break always breaks the current execution and transfer control to outside the current block. If the block is in a loop, it exits from the loop, and if the break is in a nested loop, it exits from the innermost loop.

Example:

```
list_1 = ['X', 'Y', 'Z']
list_2 = [11, 22, 33]
for i in list_1:
    for j in list_2:
        print(i, j)
        if i == 'Y' and j == 33:
            print('BREAK')
            break
        else:
            continue
        break
```

Output:

```
2
X 11
X 22
X 33
Y 11
Y 22
Y 33
BREAK
```



Python Break statement flowchart.

18) 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.

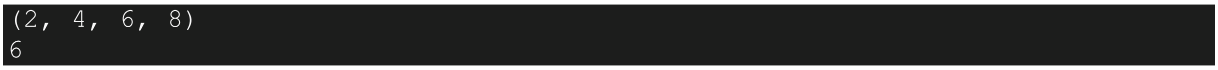
To create a tuple, we can declare it as below.

Example:

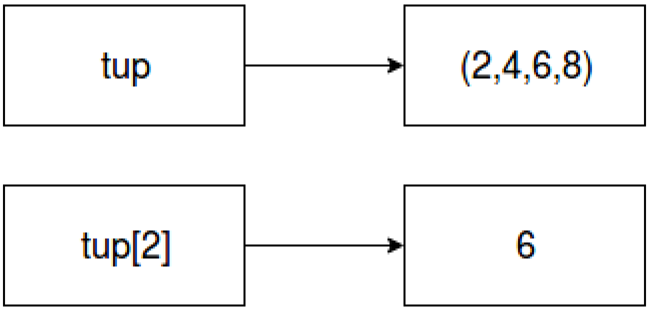
```
# Declaring tuple
tup = (2,4,6,8)
# Displaying value
print(tup)

# Displaying Single value
print(tup[2])
```

Output:



```
(2, 4, 6, 8)
6
```



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

Example:

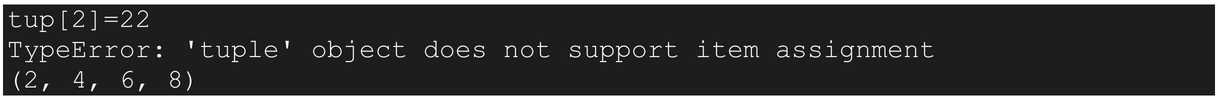
```
# Declaring tuple
tup = (2,4,6,8)
# Displaying value
print(tup)

```

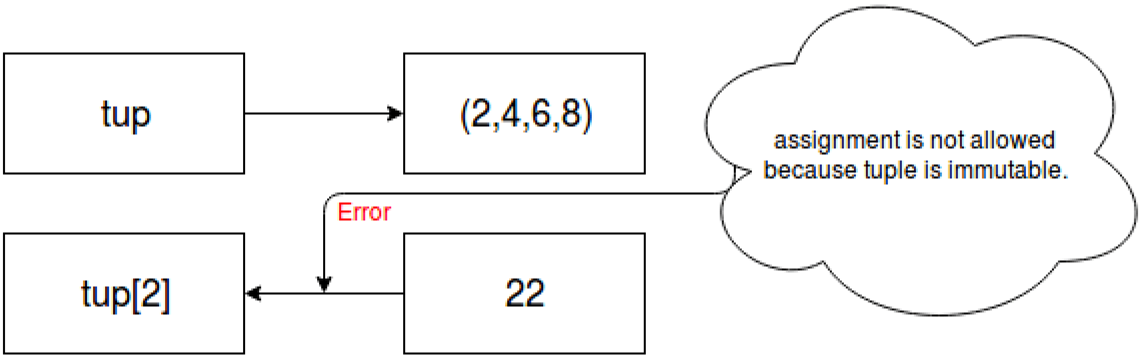
```
# Displaying Single value
print(tup[2])

# Updating by assigning new value
tup[2]=22
# Displaying Single value
print(tup[2])
```

Output:



```
tup[2]=22
TypeError: 'tuple' object does not support item assignment
(2, 4, 6, 8)
```



Updating value at index 2

19) 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.

20) What are the different file processing modes supported by Python?

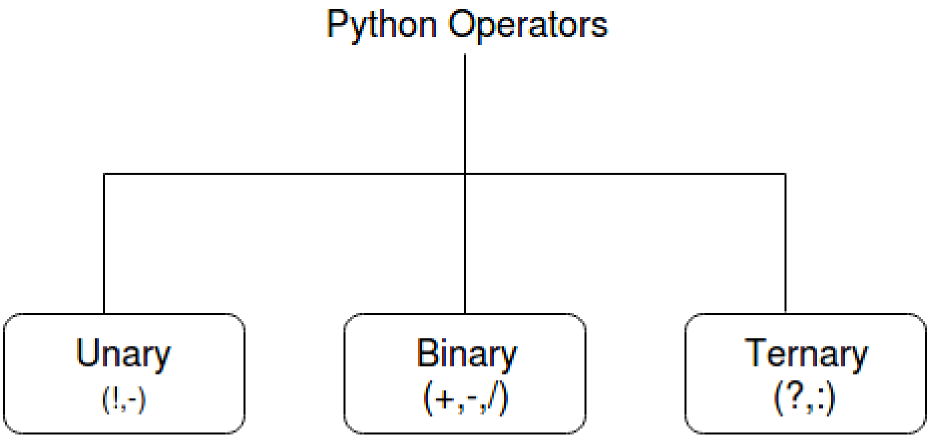
Python provides **four** modes to open files. The read-only (r), write-only (w), read-write (rw) and append mode (a). 'r' is used to open a file in read-only mode, 'w' is used to open a file in write-only mode, 'rw' is used to open in reading and write mode, 'a' is used to open a file in append mode.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

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

---

21) 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 requires 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.**



Example:

```
1. # Unary Operator
2. A = 12
3. B = -(A)
4. print (B)
5. # Binary Operator
```

```
A = 12
B = 13
print (A + B)
print (B * A)
#Ternary Operator
A = 12
B = 13
min = A if A < B else B

print(min)
```

Output:

```
# Unary Operator
-12
# Binary Operator
25
156
# Ternary Operator
12
```

22) 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

python Operators
Arithmetic Operator
Relational Operator
Assignment Operator
Logical Operator
Bitwise Operator
Membership Operator
Identity Operator

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

Example:

```
# Adding two values
print(12+23)
# Subtracting two values
print(12-23)
# Multiplying two values
print(12*23)
# Dividing two values
print(12/23)
```

Output:

```
35
-11
276
0.5217391304347826
```

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

Example:

```
a, b = 10, 12
print(a==b) # False
print(a<b) # True
print(a<=b) # True
print(a!=b) # True
```

Output:

```
False
True
True
True
```

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

Example:

```
# Examples of Assignment operators
a=12
print(a) # 12
a += 2
print(a) # 14
a -= 2
print(a) # 12
a *=2
print(a) # 24
a **=2
print(a) # 576
```

Output:

```
12
14
12
24
576
```

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

Example:

```
# Logical operator examples
a = True
b = False
print(a and b) # False
print(a or b) # True
print(not b) # True
```

Output:

```
False
True
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.

Example:

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

Output:

```
False
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.

Example:

```
# Identity operator example
a = 10
b = 12
print(a is b) # False
print(a is not b) # True
```

Output:

```
False
True
```

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

Example:

```
# Identity operator example
a = 10
b = 12
print(a & b) # 8
print(a | b) # 14
print(a ^ b) # 6
print(~a) # -11
```

Output:

```
8
14
6
-11
```

23) 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.

Example:

```
unicode_1 = ("\u0123", "\u2665", "\U0001f638", "\u265E", "\u265F", "\u2168")
print (unicode_1)
```

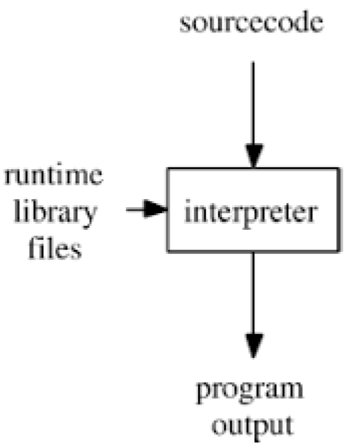
Output:

```
unicode_1: ('ģ', '♥', '😸', '♞', '♟', 'Ⅸ')
```

24) 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.



25) How is memory managed in Python?

Memory is managed in Python in the following ways:

- Memory management in python is managed by 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.
- 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.
- Python also has an inbuilt garbage collector, which recycles all the unused memory and so that it can be made available to the heap space.

26) What is the Python decorator?

Decorators are very powerful and a useful tool in Python that allows the programmers to add functionality to an existing code. This is also called metaprogramming because a part of the program tries to modify another part of the program at compile time. It allows the user to wrap another function to extend the behaviour of the wrapped function, without permanently modifying it.

Example:

```
def function_is_called():
    def function_is_returned():
        print("JavaTpoint")
    return function_is_returned
new_1 = function_is_called()
# Outputs "JavaTpoint"
new_1()
```

Output:

```
JavaTpoint
```

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.

---

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

Global Variables:

- Variables declared outside a function or in global space are called global variables.
- 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.
- Global variables are accessible anywhere in the program, and any function can access and modify its value.

Example:

```
A = "JavaTpoint"
def my_function():
    print(A)
my_function()
```

Output:

```
JavaTpoint
```

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.
- If a variable is assigned a new value anywhere within the function's body, it's assumed to be a local.
- Local variables are accessible within local body only.

Example:

```
def my_function2():
    K = "JavaTpoint Local"
    print(K)
my_function2()
```

Output:

```
JavaTpoint Local
```

28) 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.

29) 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).

In simple words: Iterators are objects which can be traversed though or iterated upon.

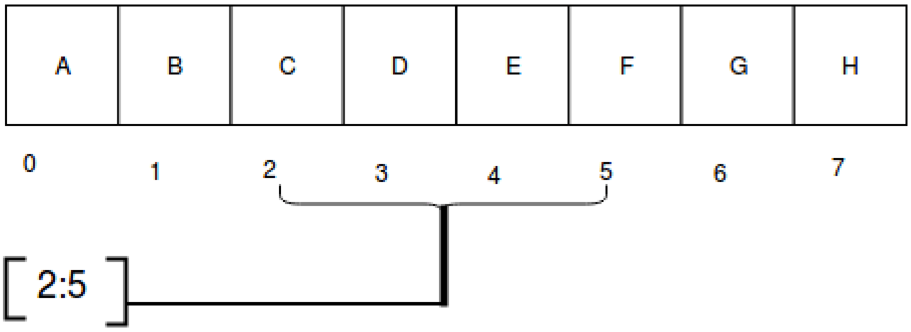
30) What is a generator in Python?

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.

31) 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.



Example:

```
Q = "JavaTpoint, Python Interview Questions!"
print(Q[2:25])
```

Output:

```
vaTpoint, Python Interv
```

32) 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.

Example:

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

```
dict = {'Country': 'India', 'Hero': 'Modi', 'Cartoon': 'Rahul'}
print ("Country: ", dict['Country'])
print ("Hero: ", dict['Hero'])
print ("Cartoon: ", dict['Cartoon'])
```

Output:

```
Country:  India
Hero:  Modi
Cartoon:  Rahul
```

33) 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, the pass keyword helps to pass the control without error.

Example:

```
class Student:
    pass # Passing class
class Student:
    def info():
        pass # Passing function
```

34) 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:

```
# One-line docstrings
def hello():
    """A function to greet."""
    return "hello"
```

35) What is a negative index in Python and why are they used?

The sequences in Python are indexed and it consists of the positive as well as negative numbers. The numbers that are positive uses '0' that is uses as first index and '1' as the second index and the process go on like that.

The index for the negative number starts from '-1' that 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 allow 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 correct order.

36) 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**.

37) 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

38) 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.

39) 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.

40) 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.

41) 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:

Example:

```
with open("FILE NAME", "r") as fp:
    fileData = fp.read()
# To print the contents of the file
print(fileData)
```

Output:

```
"The data of the file will be printed."
```

42) 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.

Example:

```
list_1 = ["A","B","C"]
s_1 = "Javatpoint"
# creating enumerate objects
object_1 = enumerate(list_1)
object_2 = enumerate(s_1)

print ("Return type:",type(object_1))
print (list(enumerate(list_1)))
print (list(enumerate(s_1)))
```

Output:

```
Return type:
[(0, 'A'), (1, 'B'), (2, 'C')]
[(0, 'J'), (1, 'a'), (2, 'v'), (3, 'a'), (4, 't'), (5, 'p'), (6, 'o'), (7,
'i'), (8, 'n'), (9, 't')]
```

43) 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.

44) What is type conversion in Python?

Type conversion refers to the conversion of one data type iinto 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 integer to octal

tuple() - This function is used to convert to a tuple.

set() - This function returns the type after converting to set.

list() - This function is used to convert any data type to a list type.

dict() - This function is used to convert a tuple of order (key,value) into a dictionary.

str() - Used to convert integer into a string.

complex(real,imag) - This functionconverts real numbers to complex(real,imag) number.

---

45) 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.

Example:

```
import smtplib
# Calling SMTP
s = smtplib.SMTP('smtp.gmail.com', 587)
# TLS for network security
s.starttls()
# User email Authentication
s.login("sender@email_id", "sender_email_id_password")
# Message to be sent
message = "Message_sender_need_to_send"
# Sending the mail
s.sendmail("sender@email_id ", "receiver@email_id", message)
```

---

46) What is the difference between Python Arrays and lists?

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

Example:

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

Output:

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

47) 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**.

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

Lambda forms in Python does not have the statement because it is used to make the new function object and return them in runtime.

49) What are functions in Python?

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

Example:

```
def New_func():
    print ("Hi, Welcome to JavaTpoint")
New_func() #calling the function
```

Output:

```
Hi, Welcome to JavaTpoint
```

50) What is __init__?

The __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_1:
    def __init__(self, name, age,salary):
        self.name = name
        self.age = age
        self.salary = 20000
E_1 = Employee_1("pqr", 20, 25000)
# E1 is the instance of class Employee.
#__init__ allocates memory for E1.
print(E_1.name)
print(E_1.age)
print(E_1.salary)
```

Output:

```
pqr
20
25000
```

51) What is self in Python?

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.

52) How can you generate random numbers in Python?

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 creates a different instance of individual threads. The other random generators that are used in this are:

randrange(a, b): it chooses an integer and define 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.

uniform(a, b): it chooses a floating point number that is defined in the range of [a,b).Iyt returns the floating point number

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.

The Random class that is used and instantiated creates independent multiple random number generators.

53) What is PYTHONPATH?

PYTHONPATH 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.

54) What are python modules? Name some commonly used built-in modules in Python?

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

55) What is the difference between range & xrange?

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 like 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 generators. That means that if you have a really gigantic range 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 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.

56) What advantages do NumPy arrays offer over (nested) Python lists?

- 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.
- 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 mean that Python must store type information for every element, and must execute type dispatching code when operating on each element.
- NumPy is not just more efficient; it is also more convenient. We get a lot of vector and matrix operations for free, which sometimes allow one to avoid unnecessary work. And they are also efficiently implemented.

- NumPy array is faster and we get a lot built in with NumPy, FFTs, convolutions, fast searching, basic statistics, linear algebra, histograms, etc.

57) Mention what the Django templates consist of.

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

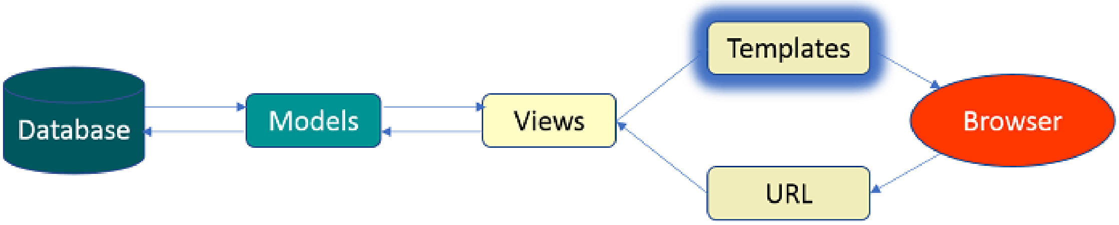


Figure: Python Interview Questions – Django Template

58) Explain the use of session in Django framework?

Django provides a session that lets the user 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.

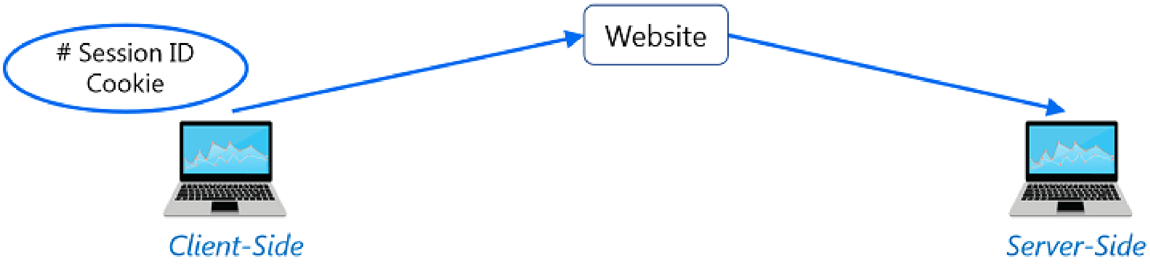


Figure: Python Interview Questions – Django Framework

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

Python Most Asked Interview Qns & Ans

1) What is Python? What are the benefits of using Python?

Python is a programming language with objects, modules, threads, exceptions, and automatic memory management. The benefits of pythons are that it is simple and easy, portable, extensible, build-in data structure, and it is open-source.

2) Explain namespace in Python

In Python, every name introduced has a place where it lives and can be hooked for. This is known as a namespace. It is like a box where a variable name is mapped to the object placed. Whenever the variable is searched out, this box will be searched to get the corresponding object.

3) What is lambda in Python?

It is a single expression anonymous function often used as inline function.

4) Why lambda forms in python do not have statements?

A lambda form in python does not have statements as it is used to make new function object and then return them at runtime.

5) Explain pass in Python

Pass means no-operation Python statement, or in other words, it is a place holder in a compound statement, where there should be a blank left, and nothing has to be written there.

6) In Python what are iterators?

In Python, iterators are used to iterate a group of elements, containers like a list.

7) What is the unittest in Python?

A unit testing framework in Python is known as unittest. It supports sharing of setups, automation testing, shutdown code for tests, aggregation of tests into collections, etc.

8) Explain slicing in Python?

A mechanism to select a range of items from sequence types like list, tuple, strings etc., is known as slicing.

9) What are generators in Python?

The way of implementing iterators are known as generators. It is a normal function except that it yields expression in the function.

10) What is docstring in Python?

A Python documentation string is known as docstring, it is a way of documenting Python functions, modules, and classes.

11) What are built-in type does python provides?

Python provides two built-in types: 1) Mutable and 2) Immutable.

Mutable built-in types are:

- List
- Sets
- Dictionaries
- Immutable built-in types
- Strings
- Tuples
- Numbers

Immutable built-in types are:

- Strings
- Tuples
- Numbers

12) What is PEP 8?

PEP 8 is a coding convention, a set of recommendation, about how to write your Python code more readable.

13) What is pickling and unpickling?

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

14) How is Python interpreted?

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

15) How is memory managed in Python?

Python memory is managed by Python private heap space. All Python objects and data structures are located in a private heap. The programmer does not have an access to this private heap, and the interpreter takes care of this Python private heap.

The allocation of Python heap space for Python objects is done by the Python memory manager. The core API gives access to some tools for the programmer to code.

Python also has an inbuilt garbage collector, which recycles all the unused memory and frees the memory and makes it available to the heap space.

16) What are the tools that help to find bugs or perform the static analysis?

PyChecker is a static analysis tool that detects the bugs in Python source code and warns about the style and complexity of the bug. Pylint is another tool that verifies whether the module meets the coding standard.

17) What are Python decorators?

A Python decorator is a specific change that we make in Python syntax to alter functions easily.

18) What is the difference between list and tuple?

The difference between list and tuple is that list is mutable while tuple is not. Tuple can be hashed, for example., as a key for dictionaries.

19) How are arguments passed by value or by reference?

Everything in Python is an object, and all variables hold references to the objects. The reference values are according to the functions. Therefore, you cannot change the value of the references. However, you can change the objects if it is mutable.

20) What is Dict and List comprehensions are?

They are syntax constructions to ease the creation of a Dictionary or List based on existing iterable.

21) How can you copy an object in Python?

To copy an object in Python, you can try a copy.copy () or copy.deepcopy() for the general case. You cannot copy all objects but most of them.

22) What is negative index in Python?

Python sequences can be index in positive and negative numbers. For positive index, 0 is the first index, 1 is the second index, and so forth. For the negative index, (-1) is the last index, and (-2) is the second last index, and so forth.

23) How can you convert a number to a string?

In order to convert a number into a string, use the inbuilt function str(). If you want a octal or hexadecimal representation, use the inbuilt function oct() or hex().

24) What is the difference between xrange and range?

Xrange returns the xrange object while range returns the list and uses the same memory and no matter what the range size is.

25) What is module and package in Python?

In Python, module is the way to structure a program. Each Python program file is a module, which imports other modules like objects and attributes.

The folder of Python program is a package of modules. A package can have modules or subfolders.

Top 50

Python

Interview Questions Answers

CONTENTS

21. What is the significance of functions that start and end with _ symbol in Python?

22. What is the difference between xrange and range in Python?

23. What is lambda expression in Python?

24. How will you copy an object in Python?

25. What are the main benefits of using Python?

26. What is a metaclass in Python?

27. What is the use of frozenset in Python?

28. What is Python Flask?

29. What is None in Python?

30. What is the use of zip() function in Python?

31. What is the use of // operator in Python?

32. What is a Module in Python?

33. How can we create a dictionary with ordered set of keys in Python?

34. Python is an Object Oriented programming language or a functional programming language?

35. How can we retrieve data from a MySQL database in a Python script?

36. What is the difference between append() and extend() functions of a list in Python?

37. How will you handle an error condition in Python code?

38. What is the difference between split() and slicing in Python?

39. How will you check in Python, if a class is subclass of another class?

40. How will you debug a piece of code in Python?

41. How do you profile a Python script?

42. What is the difference between ‘is’ and ‘==’ in Python?

43. How will you share variables across modules in Python?

44. How can we do Functional programming in Python?

45. What is the improvement in enumerate() function of Python?

46. How will you execute a Python script in Unix?

47. What are the popular Python libraries used in Data analysis?

48. What is the output of following code in Python?

49. What is the output of following code in Python?

50. If you have data with name of customers and their location, which data type will you use to store it in Python?

ACKNOWLEDGMENTS

We thank our readers who constantly send feedback and reviews to motivate us in creating these useful books with the latest information!

INTRODUCTION

This book contains basic to expert level Python interview questions that an interviewer asks. Each question is accompanied with an answer so that you can prepare for job interview in short time.

We have compiled this list after attending dozens of technical interviews in top-notch companies like- Google, Facebook, Netflix, Amazon etc.

Often, these questions and concepts are used in our daily programming work. But these are most helpful when an Interviewer is trying to test your deep knowledge of Python.

The difficulty rating on these Questions varies from a Fresher level software programmer to a Senior software programmer.

Once you go through them in the first pass, mark the questions that you could not answer by yourself. Then, in second pass go through only the difficult questions.

After going through this book 2-3 times, you will be well prepared to face a technical interview on Python for an experienced programmer.

Python Interview Questions

1. How will you improve the performance of a program in Python?

There are many ways to improve the performance of a Python program. Some of these are as follows:

i. **Data Structure**: We have to select the right data structure for our purpose in a Python program.

ii. **Standard Library**: Wherever possible, we should use methods from standard library. Methods implemented in standard library have much better performance than user implementation.

iii. **Abstraction**: At times, a lot of abstraction and indirection can cause slow performance of a program. We should remove the redundant abstraction in code.

iv. **Algorithm**: Use of right algorithm can make a big difference in a program. We have to find and select the suitable algorithm to solve our problem with high performance.

2. What are the benefits of using Python?

Python is strong that even Google uses it. Some of the benefits of using Python are as follows:

i. **Efficient**: Python is very efficient in memory management. For a large data set like Big Data, it is much easier to program in Python.

ii. **Faster**: Though Python code is interpreted, still Python has very fast performance.

iii. **Wide usage**: Python is widely used among different organizations for different projects. Due to this wide usage, there are thousands of add-ons available for use with Python.

iv. **Easy to learn**: Python is quite easy to learn. This is the biggest benefit of using Python. Complex tasks can be very easily implemented in Python.

3. How will you specify source code encoding in a Python source file?

By default, every source code file in Python is in UTF-8 encoding. But we can also specify our own encoding for source files. This can be done by adding following line after #! line in the source file.

```
# -*- coding: encoding -*-
```

In the above line we can replace encoding with the encoding that we want to use.

4. What is the use of PEP 8 in Python?

PEP 8 is a style guide for Python code. This document provides the coding conventions for writing code in Python. Coding conventions are about indentation, formatting, tabs, maximum line length, imports organization, line spacing etc. We use PEP 8 to bring consistency in our code. We consistency it is easier for other developers to read the code.

5. What is Pickling in Python?

Pickling is a process by which a Python object hierarchy can be converted into a byte stream. The reverse operation of Pickling is Unpickling.

Python has a module named pickle. This module has the implementation of a powerful algorithm for serialization and de-serialization of Python object structure.

Some people also call Pickling as Serialization or Marshalling.

With Serialization we can transfer Python objects over the network. It is also used in persisting the state of a Python object. We can write it to a file or a database.

6. How does memory management work in Python?

There is a private heap space in Python that contains all the Python objects and data structures. In CPython there is a memory manager responsible for managing the heap space.

There are different components in Python memory manager that handle segmentation, sharing, caching, memory pre-allocation etc.

Python memory manager also takes care of garbage collection by using Reference counting algorithm.

7. How will you perform Static Analysis on a Python Script?

We can use Static Analysis tool called PyChecker for this purpose. PyChecker can detect errors in Python code.

PyChecker also gives warnings for any style issues.

Some other tools to find bugs in Python code are pylint and pyflakes.

8. What is the difference between a Tuple and List in Python?

In Python, Tuple and List are built-in data structures.

Some of the differences between Tuple and List are as follows:

I. **Syntax**: A Tuple is enclosed in parentheses:
E.g. myTuple = (10, 20, “apple”);
A List is enclosed in brackets:
E.g. myList = [10, 20, 30];

II. **Mutable**: Tuple is an immutable data structure. Whereas, a List is a mutable data structure.

III. **Size**: A Tuple takes much lesser space than a List in Python.

IV. **Performance**: Tuple is faster than a List in Python. So it gives us good performance.

V. **Use case**: Since Tuple is immutable, we can use it in cases like Dictionary creation. Whereas, a List is preferred in the use case where data can alter.

9. What is a Python Decorator?

A Python Decorator is a mechanism to wrap a Python function and modify its behavior by adding more functionality to it. We can use @ symbol to call a Python Decorator function.

10. How are arguments passed in a Python method? By value or by reference?

Every argument in a Python method is an Object. All the variables in Python have reference to an Object. Therefore arguments in Python method are passed by Reference.

Since some of the objects passed as reference are mutable, we can change those objects in a method. But for an Immutable object like String, any change done within a method is not reflected outside.

11. What is the difference between List and Dictionary data types in Python?

Main differences between List and Dictionary data types in Python are as follows:

I. **Syntax**: In a List we store objects in a sequence. In a Dictionary we store objects in key-value pairs.

II. **Reference**: In List we access objects by index number. It starts from 0 index. In a Dictionary we access objects by key specified at the time of Dictionary creation.

III. **Ordering**: In a List objects are stored in an ordered sequence. In a Dictionary objects are not stored in an ordered sequence.

IV. **Hashing**: In a Dictionary, keys have to be hashable. In a List there is no need for hashing.

12. What are the different built-in data types available in Python?

Some of the built-in data types available in Python are as follows:

Numeric types: These are the data types used to represent numbers in Python.

int: It is used for Integers

long: It is used for very large integers of non-limited length.

float: It is used for decimal numbers.

complex: This one is for representing complex numbers

Sequence types: These data types are used to represent sequence of characters or objects.

str: This is similar to String in Java. It can represent a sequence of characters.

bytes: This is a sequence of integers in the range of 0-255.

byte array: like bytes, but mutable (see below); only available in Python 3.x

list: This is a sequence of objects.

tuple: This is a sequence of immutable objects.

Sets: These are unordered collections.

set: This is a collection of unique objects.

frozen set: This is a collection of unique immutable objects.

Mappings: This is similar to a Map in Java.

dict: This is also called hashmap. It has key value pair to store information by using hashing.

13. What is a Namespace in Python?

A Namespace in Python is a mapping between a name and an object. It is currently implemented as Python dictionary.

E.g. the set of built-in exception names, the set of built-in names, local names in a function

At different moments in Python, different Namespaces are created. Each Namespace in Python can have a different lifetime.

For the list of built-in names, Namespace is created when Python interpreter starts.

When Python interpreter reads the definition of a module, it creates global namespace for that module.

When Python interpreter calls a function, it creates local namespace for that function.

14. How will you concatenate multiple strings together in Python?

We can use following ways to concatenate multiple string together in Python:

I. use + operator:

E.g.

```
>>> fname="John"
>>> lname="Ray"
>>> print fname+lname
JohnRay
```

II. use join function:

E.g.

```
>>> ''.join(['John','Ray'])
'JohnRay'
```

15. What is the use of Pass statement in Python?

The use of Pass statement is to do nothing. It is just a placeholder for a statement that is required for syntax purpose. It does not execute any code or command.

Some of the use cases for pass statement are as follows:

I. Syntax purpose:

```
>>> while True:

... pass # Wait till user input is received
```

II. **Minimal Class:** It can be used for creating minimal classes:

```
>>> class MyMinimalClass:

... pass
```

III. Place-holder for TODO work:

We can also use it as a placeholder for TODO work on a function or code that needs to be implemented at a later point of time.

```
>>> def initialization():

... pass # TODO
```

16. What is the use of Slicing in Python?

We can use Slicing in Python to get a substring from a String.

The syntax of Slicing is very convenient to use.

E.g. In following example we are getting a substring out of the name John.

```
>>> name="John"
>>> name[1:3]
'oh'
```

In Slicing we can give two indices in the String to create a Substring. If we do not give first index, then it defaults to 0.

E.g.

```
>>> name="John"
>>> name[:2]
'Jo'
```

If we do not give second index, then it defaults to the size of the String.

```
>>> name="John"
>>> name[3:]
'n'
```

17. What is the difference between Docstring in Python and Javadoc in Java?

A Docstring in Python is a string used for adding comments or summarizing a piece of code in Python.

The main difference between Javadoc and Docstring is that docstring is available during runtime as well. Whereas, Javadoc is removed from the Bytecode and it is not present in .class file.

We can even use Docstring comments at run time as an interactive help manual.

In Python, we have to specify docstring as the first statement of a code object, just after the def or class statement.

The docstring for a code object can be accessed from the '__doc__' attribute of that object.

18. How do you perform unit testing for Python code?

We can use the unit testing modules **unittest** or **unittest2** to create and run unit tests for Python code.

We can even do automation of tests with these modules. Some of the main components of unittest are as follows:

I. **Test fixture**: We use test fixture to create preparation methods required to run a test. It can even perform post-test cleanup.

II. **Test case**: This is main unit test that we run on a piece of code. We can use **Testcase** base class to create new test cases.

III. **Test suite**: We can aggregate our unit test cases in a Test suite.

IV. **Test runner**: We use test runner to execute unit tests and produce reports of the test run.

19. What is the difference between an Iterator and Iterable in Python?

An Iterable is an object that can be iterated by an Iterator.

In Python, Iterator object provides _iter_() and next() methods.

In Python, an Iterable object has _iter_ function that returns an Iterator object.

When we work on a map or a for loop in Python, we can use next() method to get an Iterable item from the Iterator.

20. What is the use of Generator in Python?

We can use Generator to create Iterators in Python. A Generator is written like a regular function. It can make use yield statement to return data during the function call. In this way we can write complex logic that works as an Iterator.

A Generator is more compact than an Iterator due to the fact that _iter_() and next() functions are automatically created in a Generator.

Also within a Generator code, local variables and execution state are saved between multiple calls. Therefore, there is no need to add extra variables like self.index etc to keep track of iteration.

Generator also increases the readability of the code written in Python. It is a very simple implementation of an Iterator.

21. What is the significance of functions that start and end with _ symbol in Python?

Python provides many built-in functions that are surrounded by _ symbol at the start and end of the function name. As per Python documentation, double _ symbol is used for reserved names of functions.

These are also known as System-defined names.

Some of the important functions are:

Object._new_

Object._init_

Object._del_

22. What is the difference between xrange and range in Python?

In Python, we use range(0,10) to create a list in memory for 10 numbers.

Python provides another function xrange() that is similar to range() but xrange() returns a sequence object instead of list object. In xrange() all the values are not stored simultaneously in memory. It is a lazy loading based function.

But as per Python documentation, the benefit of xrange() over range() is very minimal in regular scenarios.

As of version 3.1, xrange is deprecated.

23. What is lambda expression in Python?

A lambda expression in Python is used for creating an anonymous function.

Wherever we need a function, we can also use a lambda expression.

We have to use lambda keyword for creating a lambda expression. Syntax of lambda function is as follows:

lambda argumentList: expression

E.g. lambda a,b: a+b

The above mentioned lambda expression takes two arguments and returns their sum.

We can use lambda expression to return a function.

A lambda expression can be used to pass a function as an argument in another function.

24. How will you copy an object in Python?

In Python we have two options to copy an object. It is similar to cloning an object in Java.

I. **Shallow Copy**: To create a shallow copy we call copy.copy(x). In a shallow copy, Python creates a new compound object based on the original object. And it tries to put references from the original object into copy object.

II. **Deep Copy**: To create a deep copy, we call copy.deepcopy(x). In a deep copy, Python creates a new object and recursively creates and inserts copies of the objects from original object into copy object. In a deep copy, we may face the issue of recursive loop due to infinite recursion.

25. What are the main benefits of using Python?

Some of the main benefits of using Python are as follows:

I. **Easy to learn**: Python is simple language. It is easy to learn for a new programmer.

II. **Large library**: There is a large library for utilities in Python that can be used for different kinds of applications.

III. **Readability**: Python has a variety of statements and expressions that are quite readable and very explicit in their use. It increases the readability of overall code.

IV. **Memory management**: In Python, memory management is built into the Interpreter. So a developer does not have to spend effort on managing memory among objects.

V. **Complex built-in Data types**: Python has built-in Complex data types like list, set, dict etc. These data types give very good performance as well as save time in coding new features.

26. What is a metaclass in Python?

A metaclass in Python is also known as class of a class. A class defines the behavior of an instance. A metaclass defines the behavior of a class.

One of the most common metaclass in Python is type. We can subclass type to create our own metaclass.

We can use metaclass as a class-factory to create different types of classes.

27. What is the use of frozenset in Python?

A frozenset is a collection of unique values in Python. In addition to all the properties of set, a frozenset is immutable and hashable.

Once we have set the values in a frozenset, we cannot change. So we cannot use and update methods from set on frozenset.

Being hashable, we can use the objects in frozenset as keys in a Dictionary.

28. What is Python Flask?

Python Flask is a micro-framework based on Python to develop a web application.

It is a very simple application framework that has many extensions to build an enterprise level application.

Flask does not provide a data abstraction layer or form validation by default. We can use external libraries on top of Flask to perform such tasks.

29. What is None in Python?

None is a reserved keyword used in Python for null objects. It is neither a null value nor a null pointer. It is an actual object in Python. But there is only one instance of None in a Python environment.

We can use None as a default argument in a function.

During comparison we have to use “is” operator instead of “==” for None.

30. What is the use of zip() function in Python?

In Python, we have a built-in function zip() that can be used to aggregate all the Iterable objects of an Iterator.

We can use it to aggregate Iterable objects from two iterators as well.

E.g.

```
list_1 = ['a', 'b', 'c']

list_2 = ['1', '2', '3']

for a, b in zip(list_1, list_2):

   print a, b
```

Output:

```
a1

b2

c3
```

By using zip() function we can divide our input data from different sources into fixed number of sets.

31. What is the use of // operator in Python?

Python provides // operator to perform floor division of a number by another. The result of // operator is a whole number (without decimal part) quotient that we get by dividing left number with right number.

It can also be used floordiv(a,b).

E.g.

10// 4 = 2

-10//4 = -3

32. What is a Module in Python?

A Module is a script written in Python with import statements, classes, functions etc. We can use a module in another Python script by importing it or by giving the complete namespace.

With Modules, we can divide the functionality of our application in smaller chunks that can be easily managed.

33. How can we create a dictionary with ordered set of keys in Python?

In a normal dictionary in Python, there is no order maintained between keys. To solve this problem, we can use OrderDict class in Python. This class is available for use since version 2.7.

It is similar to a dictionary in Python, but it maintains the insertion order of keys in the dictionary collection.

34. Python is an Object Oriented programming language or a functional programming language?

Python uses most of the Object Oriented programming concepts. But we can also do functional programming in Python. As per the opinion of experts, Python is a multi-paradigm programming language.

We can do functional, procedural, object-oriented and imperative programming with the help of Python.

35. How can we retrieve data from a MySQL database in a Python script?

To retrieve data from a database we have to make use of the module available for that database. For MySQL database, we import MySQLdb module in our Python script.

We have to first connect to a specific database by passing URL, username, password and the name of database.

Once we establish the connection, we can open a cursor with cursor() function. On an open cursor, we can run fetch() function to execute queries and retrieve data from the database tables.

36. What is the difference between append() and extend() functions of a list in Python?

In Python, we get a built-in sequence called list. We can call standard functions like append() and extend() on a list.

We call append() method to add an item to the end of a list.

We call extend() method to add another list to the end of a list.

In append() we have to add items one by one. But in extend() multiple items from another list can be added at the same time.

37. How will you handle an error condition in Python code?

We can implement exception handling to handle error conditions in Python code. If we are expecting an error condition that we cannot handle, we can raise an error with appropriate message.

E.g.

```
>>> if student_score < 0: raise ValueError(“Score can not be negative”)
```

If we do not want to stop the program, we can just catch the error condition, print a message and continue with our program.

E.g. In following code snippet we are catching the error and continuing with the default value of age.

```
#!/usr/bin/python
try:
   age=18+'duration'
except:
   print("duration has to be a number")
age=18
print(age)
```

38. What is the difference between split() and slicing in Python?

Both split() function and slicing work on a String object. By using split() function, we can get the list of words from a String.

E.g. 'a b c '.split() returns [‘a’, ‘b’, ‘c’]

Slicing is a way of getting substring from a String. It returns another String.

E.g. >>> 'a b c'[2:3] returns b

39. How will you check in Python, if a class is subclass of another class?

Python provides a useful method issubclass(a,b) to check whether class a is a subclass of b.

E.g. int is not a subclass of long

```
>>> issubclass(int,long)
False
```

bool is a subclass of int

```
>>> issubclass(bool,int)
True
```

40. How will you debug a piece of code in Python?

In Python, we can use the debugger pdb for debugging the code. To start debugging we have to enter following lines on the top of a Python script.

```
import pdb

pdb.set_trace()
```

After adding these lines, our code runs in debug mode. Now we can use commands like breakpoint, step through, step into etc for debugging.

41. How do you profile a Python script?

Python provides a profiler called cProfile that can be used for profiling Python code.

We can call it from our code as well as from the interpreter.

It gives use the number of function calls as well as the total time taken to run the script.

We can even write the profile results to a file instead of standard out.

42. What is the difference between ‘is’ and ‘==’ in Python?

We use ‘is’ to check an object against its identity.

We use ‘==’ to check equality of two objects.

E.g.

```
>>> lst = [10,20, 20]

>>> lst == lst[:]

True

>>> lst is lst[:]

False
```

43. How will you share variables across modules in Python?

We can create a common module with variables that we want to share.

This common module can be imported in all the modules in which we want to share the variables.

In this way, all the shared variables will be in one module and available for sharing with any new module as well.

44. How can we do Functional programming in Python?

In Functional Programming, we decompose a program into functions. These functions take input and after processing give an output. The function does not maintain any state.

Python provides built-in functions that can be used for Functional programming. Some of these functions are:

I. Map()
II. reduce()
III. filter()

Event iterators and generators can be used for Functional programming in Python.

45. What is the improvement in enumerate() function of Python?

In Python, enumerate() function is an improvement over regular iteration. The enumerate() function returns an iterator that gives (0, item[0]).

E.g.

```
>>> thelist=['a','b']
>>> for i,j in enumerate(thelist):
... print i,j
...
0 a
1 b
```

46. How will you execute a Python script in Unix?

To execute a Python script in Unix, we need to have Python executor in Unix environment.

In addition to that we have to add following line as the first line in a Python script file.

```
#!/usr/local/bin/python
```

This will tell Unix to use Python interpreter to execute the script.

47. What are the popular Python libraries used in Data analysis?

Some of the popular libraries of Python used for Data analysis are:

I. **Pandas**: Powerful Python Data Analysis Toolkit
II. **SciKit**: This is a machine learning library in Python.
III. **Seaborn**: This is a statistical data visualization library in Python.
IV. **SciPy**: This is an open source system for science, mathematics and engineering implemented in Python.

48. What is the output of following code in Python?

```
>>> thelist=['a','b']

>>> print thelist[3:]
```

Ans: The output of this code is following:

```
[]
```

Even though the list has only 2 elements, the call to thelist with index 3 does not give any index error.

49. What is the output of following code in Python?

```
>>>name=’John Smith’

>>>print name[:5] + name[5:]
```

Ans: Output of this will be

```
John Smith
```

This is an example of Slicing. Since we are slicing at the same index, the first name[:5] gives the substring name upto 5th location excluding 5th location. The name[5:] gives the rest of the substring of name from the 5th location. So we get the full name as output.

50. If you have data with name of customers and their location, which data type will you use to store it in Python?

In Python, we can use dict data type to store key value pairs. In this example, customer name can be the key and their location can be the value in a dict data type.

Dictionary is an efficient way to store data that can be looked up based on a key.