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.

Efficient:
Garbage Collection
Garbage collection is a process in which the interpreter frees up the memory when not in use to make it available for other objects.
Assume a case where no reference is pointing to an object in memory i.e. it is not in use so, the virtual machine has a garbage collector that automatically deletes that object from the heap memory

Reference Counting
Reference counting works by counting the number of times an object is referenced by other objects in the system. When references to an object are removed, the reference count for an object is decremented. When the reference count becomes zero, the object is deallocated.

Python is an interpreted programming language. Your Python code actually gets compiled down to more computer-readable instructions called bytecode. These instructions get interpreted by a virtual machine when you run your code.

Python is both compiled as well as an interpreted language. This means when we run a python code, it is first compiled and then interpreted line by line. The compilation part is mostly hidden from the user. While running the code, Python generates a byte code internally, this byte code is then converted using a python virtual machine (p.v.m) to generate the output.

The compile part gets deleted as soon as the code gets executed so that the programmer doesn't get onto unnecessary complexity.

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.

Python Enterprise Proposal.

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.

Pickeling: writing datafile.txt in byte mode and converting to byte stream

```
import pickle
mylist = ['a', 'b', 'c', 'd']
with open('datafile.txt', 'wb') as fh:
   pickle.dump(mylist, fh)
```

Unpickeling:

```
import pickle
pickle_off = open ("datafile.txt", "rb")
emp = pickle.load(pickle_off)
print(emp)
```

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.

Tuples are stored in a single block of memory. Tuples are immutable so, It doesn't require extra space to store new objects.
Lists are allocated in two blocks: the fixed one with all the Python object information and a variable sized block for the data.

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.

```
def multiple_yield():

    str1 = "First String"
    yield str1

    str2 = "Second string"
    yield str2

    str3 = "Third String"
    yield str3


obj = multiple_yield()
print(next(obj))
print(next(obj))
print(next(obj))
```

Output:

```
First String
Second string
Third String
```

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 5^{th} location excluding 5^{th} location. The name[5:] gives the rest of the substring of name from the 5^{th} 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.