Python Interview Questions

A list of top frequently asked **Python interview questions** and answers are given below.

1) What is Python?

Python is a high level object-oriented programming language with objects, modules, threads, exceptions and automatic memory management. It is a simple yet powerful programming language. It can run equally on different platforms such as Windows, Linux, UNIX, Macintosh etc. Thus, Python is a portable language.

2) What are the advantages of Python?

Following are the main advantages of using Python.

- Free and open source
- Portable
- Extensible
- Object oriented
- Built-in data structure

3)What is PEP 8?

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

4)What is used to create Unicode string in Python?

You should use "Unicode" before the string. For example: Unicode (text).

5) Explain how Python is interpreted?

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

6) How memory is managed in Python?

Memory is managed in Python in following way:

- Memory is managed in Python by 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 interpreter takes care of this Python private heap.

- Python memory manager is responsible for allocating Python heap space for Python objects.
- Python also have an inbuilt garbage collector, which recycle all the unused memory and frees the memory and makes it available to the heap space.

7) What is Python decorator?

A Python decorator is a specific change made within Python syntax to alter functions easily.

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

In Python, variables that are only referenced inside a function are called implicitly global. If a variable is assigned a new value anywhere within the function's body, it's assumed to be a local. If a variable is ever assigned a new value inside the function, the variable is implicitly local, and you need to explicitly declare it as 'global'.

9) What is namespace in Python?

In Python, every name has a place where it lives and can be hooked for. This is known as 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 corresponding object.

10) What are iterators in Python?

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

11) What is generator in Python?

In Python, generator is a way that specifies how to implement iterators. It is a normal function except that it yields expression in the function.

12) What is slicing in Python?

Slicing is a mechanism used to select a range of items from sequence type like list, tuple, string etc.

13) What is dictionary in Python?

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 Hero & Cartoon. Their corresponding values are India, Modi and Rahul respectively.

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

14) What is Pass in Python?

Pass specifies a Python statement without operations. It is a place holder in a compound statement, where there should be a blank left and nothing has to be written there.

15) Explain docstring in Python?

A Python documentation string is called docstring. It is used for documenting Python functions, modules and classes.

16) What is negative index in Python?

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

17)What is pickling and unpickling in Python?

Pickling is a process in which a pickle module acceptsany

Python object, converts it into a string representation and dumps it into a file by using dump() function.
Unpickling is a process of retrieving original Python object from the stored string representation for use.
Pickle is a standard module which serializes and de-serializes a Python object structure.

18) How can you make forms in Python?

You have to import cgi module to access form fields using FieldStorage class.
Attributes of class FieldStorage for form:
form.name: The name of the field, if specified.
form.filename: If an FTP transaction, the client-side filename.
form.value: The value of the field as a string.
form.file: file object from which data can be read.
form.type: The content type, if applicable.
form.type_options: The options of the 'content-type' line of the HTTP request, returned as a dictionary.
form.disposition: The field 'content-disposition'; None, if unspecified.
form.disposition_options: The options for 'content-disposition'.
form.headers: All of the HTTP headers returned as a dictionary.
Example

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

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

Python 2.x is an older version of Python. It is a legacy now. Python 3.x is newer. It is the present and future of this language.
The most visible difference between them is in print statement. In Python 2 it is print ?Hello? and in Python 3, it is

print (?Hello?).

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

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

Example

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