Top Accenture Python Interview Questions

Q1. What is Python? List some popular applications of Python in the world of technology.

Answer: Python is a widely-used general-purpose, high-level programming language. It was created by Guido van Rossum in 1991 and further developed by the Python Software Foundation. And it was designed with an emphasis on code readability, and its syntax allows programmers to express their concepts in fewer lines of code.
Python is used for:

- System Scripting
- Web Development
- Game Development
- Software Development
- Complex Mathematics

Q2. Importance of self-keywords in Python?

Answer: The first parameter used when defining a class's instance methods is the keyword "self." Stated differently, it can be said that it represents the class instance itself. While we can use any name as the first parameter, using self is considered standard and is used in Python, even though it is a convention.

Q3. Define NumPy in Python?

Answer: Numerical Python, or NumPy, is a library used for numerical operations, data analysis, and scientific computing. Data analysis, machine learning, and mathematical operations are a few typical use cases and applications for NumPy.

Q4. What does "name mangling" in Python mean?

Answer: To make the attribute private and prohibit direct access from outside the class, name mangling is used. This method indicates that an attribute can only be used internally within a class and cannot be altered or obtained through the use of external code.

Q5. Why is Python unable to support function overloading?

Answer: Python does not enable function overloading, also known as function loading, to the same extent as certain other programming languages like Java or C++. Function overloading is the capacity to define many functions with the same name but different argument lists.

In Python, the most recent definition takes precedence over previous definitions when you declare multiple functions with the same name. Python does not consider the types or

quantity of arguments when determining which function to call. Python, on the other hand, has a unique approach called "duck typing," which prioritises object behaviour over object types.

Q6. For machine learning, which Python library is used?

Answer: The most widely used Python machine learning library is called scikit-learn, or sklearn. A free, open-source machine learning framework called Scikit-learn offers effective, user-friendly tools for data mining and analysis.

Along with tools for feature extraction, data preparation, model selection and evaluation, it offers a broad range of machine learning algorithms for applications including regression, classification, clustering, and dimensionality reduction.

Q7. Does Python is case sensitive?

Answer: Python is a case-sensitive language, to answer your question. This indicates that Python's syntax makes a distinction between capital and lowercase letters. For instance, in Python, the variables "myVar" and "myvar" are regarded as distinct entities, and using them in the incorrect case can result in coding mistakes.

Q8. In Python, what is a tuple?

Answer: To put it another way, a tuple in Python is an ordered set of values that resembles a list. Tuples, on the other hand, cannot have their values altered once they are constructed since they are immutable. When related items are not anticipated to change, tuples are frequently used to group them together. In dictionaries, if immutable objects are needed as keys, they can also be utilised as keys.

Q9. Which various file processing modes does Python support?

Answer: Multiple file processing modes that specify how a file can be opened, read, or written to are supported by Python. Python offers several modes for processing files, including:

- **"r" (Read mode):** The default mode for opening files for reading is "r" (read mode). The file is read-only, meaning you cannot edit its contents.
- **"w" (Write mode):** A file can be opened for writing in the "w" (Write mode) mode. Its contents will be deleted if the file already exists. It will be produced if the file doesn't already exist.
- **"a" (Append mode):** A file can be opened in "a" (append mode) in order to append data. New data will be appended to the end of the file, should it already exist. It will be produced if the file doesn't already exist.
- **"x" (Exclusive creation mode):** A new file can be created using this mode. An error message will be displayed if the file is already there.
- **"b" (Binary mode):** To open a file in binary mode, use the mode "b" (Binary mode). This indicates that the file will be read or written in binary format. When working with non-text files, such pictures, audio files, and movies, this mode is utilised.

- **"t" (Text mode):** Text mode ("t"): This mode is used to open a file in text mode, which enables text-based writing and reading. When working with text files, like.txt files, this mode is utilised.

Q10. Do Python database packages have any interfaces?

Answer: Yes, Python database packages have a wide variety of interfaces. Numerous database-related modules are available in Python, making it simple to communicate with a wide range of database systems from within the language.

- **SQLite3**: This module offers an interface to the file-based, lightweight SQLite database system, which is frequently utilised for small-scale applications.
- One of the most popular open-source relational database systems, MySQL Connector/Python offers an interface to the MySQL database system.
- **psycopg2**: This module offers an interface to another well-liked open-source relational database system, PostgreSQL.
- **Oracle Database**: Several large-scale systems use the Oracle Database system, a commercial relational database system, to which this module offers an interface.
- **PyMongo**: This module offers an interface to the document-oriented MongoDB NoSQL database, which is utilised to store substantial amounts of unstructured data.

Q11. In Python, how is memory managed?

Answer:

- Python's private heap space is in charge of memory management. A private heap holds all Python objects and data structures. This secret heap is not accessible to the programmer. Instead, the Python interpreter takes care of it.
- Python also includes a built-in garbage collector, which recycles all unused memory and makes it available to the heap space.
- Python's memory management is in charge of allocating heap space for Python objects. The core API allows programmers access to some programming tools.

Q12. Explain PYTHONPATH.

Answer: It's an environment variable that is used when you import a module. When a module is imported, PYTHONPATH is checked to see if the imported modules are present in various folders. It is used by the interpreter to determine which module to load.

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

Answer:

- Download Python from https://www.python.org/downloads/
- Install it on your computer. Using your command prompt, look for the location where PYTHON is installed on your computer by typing cmd python.
- Then, in advanced system settings, create a new variable called PYTHON_NAME and paste the copied path into it.

- Search the path variable, choose its value and select ‘edit’.
- If the value doesn't have a semicolon at the end, add one, and then type %PYTHON HOME%.

Q14. What are the types of literals in Python?

Answer: For primitive data types, a literal in Python source code indicates a fixed value. Following are the 5 types of literal in Python:

- String Literal: A string literal is formed by assigning some text to a variable that is contained in single or double-quotes. Assign the multiline text encased in triple quotes to produce multiline literals.
- Numeric Literal: They may contain numeric values that are floating-point values, integers, or complex numbers.
- Character Literal: It is made by putting a single character in double-quotes.
- Boolean Literal: True or False
- Literal Collections: There are four types of literals such as list collections, tuple literals, set literals, dictionary literals, and set literals.

Q15. Differentiate between range and xrange.

Answer: In terms of functionality, xrange and range are essentially the same. They both provide you the option of generating a list of integers to use whatever you want. The sole difference between range and xrange is that range produces a Python list object whereas x range returns an xrange object.

This is especially true if you are working with a machine that requires a lot of memory, such as a phone because range will utilize as much memory as it can to generate your array of numbers, which can cause a memory error and crash your program. It is a beast with a memory problem.

Q16. How do you copy an object in Python?

Answer: The assignment statement (= operator) in Python does not copy objects. Instead, it establishes a connection between the existing object and the name of the target variable. The copy module is used to make copies of an object in Python. Furthermore, the copy module provides two options for producing copies of a given object –

Deep Copy: Deep Copy recursively replicates all values from source to destination object, including the objects referenced by the source object.

```
from copy import copy, deepcopy

list_1 = [1, 2, [3, 5], 4]

## shallow copy

list_2 = copy(list_1)

list_2[3] = 7
```

```
list_2[2].append(6)

list_2    # output => [1, 2, [3, 5, 6], 7]

list_1    # output => [1, 2, [3, 5, 6], 4]

## deep copy

list_3 = deepcopy(list_1)

list_3[3] = 8

list_3[2].append(7)

list_3    # output => [1, 2, [3, 5, 6, 7], 8]

list_1    # output => [1, 2, [3, 5, 6], 4]
```

Shallow Copy: A bit-wise copy of an object is called a shallow copy. The values in the copied object are identical to those in the original object. If one of the values is a reference to another object, only its reference addresses are copied.

Q17. In Python, are arguments provided by value or reference?

Answer:

Pass by value: The actual item's copy is passed. Changing the value of the object's copy has no effect on the original object's value.

Pass by reference: The actual object is passed as a reference. The value of the old object will change if the value of the new object is changed.

Arguments are passed by reference in Python.

```
def appendNumber(arr):

arr.append(4)

arr = [1, 2, 3]

print(arr)  #Output: => [1, 2, 3]

appendNumber(arr)

print(arr)  #Output: => [1, 2, 3, 4]
```

Q18. Why isn't all the memory de-allocated when Python exits?

Answer:

- When Python quits, some Python modules, especially those with circular references to other objects or objects referenced from global namespaces, are not necessarily freed or deallocated.
- Python would try to de-allocate/destroy all other objects on exit because it has its own efficient cleanup mechanism.
- It is difficult to de-allocate memory that has been reserved by the C library.

Q19. What is a dictionary in Python?

Answer: Dictionary is one of Python's built-in datatypes. It establishes a one-to-one correspondence between keys and values. Dictionary keys and values are stored in pairs in dictionaries. Keys are used to index dictionaries.

Q20. Explain the split(), sub(), and subn() methods of the Python "re" module.

Answer: Python's "re" module provides three ways for modifying strings. They are:

split (): a regex pattern is used to "separate" a string into a list

subn(): It works similarly to sub(), returning the new string as well as the number of replacements.

sub(): identifies all substrings that match the regex pattern and replaces them with a new string

Q21. What are negative indexes and why do we utilize them?

Answer: Python sequences are indexed, and they include both positive and negative values. Positive numbers are indexed with '0' as the first index and '1' as the second index, and so on.

The index for a negative number begins with '-1,' which is the last index in the sequence, and ends with '-2,' which is the penultimate index, and the sequence continues like a positive number. The negative index is used to eliminate all new-line spaces from the string and allow it to accept the last character S[:-1]. The negative index can also be used to represent the correct order of the string.

Q22. Is there an object-oriented Programming (OOps) concept in Python?

Answer: Python is a computer language that focuses on objects. This indicates that by simply constructing an object model, every program can be solved in Python. Python, on the other hand, may be used as both a procedural and structured language.

Q23. Differentiate between deep and shallow copy.

Answer:

When a new instance type is formed, a shallow copy is used to maintain the values that were copied in the previous instance. Shallow copy is used to copy reference pointers in the same

way as values are copied. These references refer to the original objects, and any modifications made to any member of the class will have an impact on the original copy. Shallow copy enables faster program execution and is dependent on the size of the data being utilized.

Deep copy is a technique for storing previously copied values. The reference pointers to the objects are not copied during deep copy. It creates a reference to an object and stores the new object that is referenced to by another object. The changes made to the original copy will have no effect on any subsequent copies that utilize the item. Deep copy slows down program performance by creating many copies of each object that is called.

Q24. How multithreading is achieved in Python?

Answer:

- Although Python includes a multi-threading module, it is usually not a good idea to utilize it if you want to multi-thread to speed up your code.
- As this happens so quickly, it may appear to the human eye that your threads are running in parallel, but they are actually sharing the same CPU core.
- The Global Interpreter Lock is a Python concept (GIL). Only one of your 'threads' can execute at a moment, thanks to the GIL. A thread obtains the GIL, performs some work, and then passes the GIL to the following thread.

Q25. What are the different types of inheritance in Python?

Answer: The following are the various types of inheritance in Python:

- **Single inheritance**: The members of a single super class are acquired by a derived class.
- **Multiple inheritance**: More than one base class is inherited by a derived class.
- **Muti-level inheritance**: D1 is a derived class inherited from base1 while D2 is inherited from base2.
- **Hierarchical Inheritance**: You can inherit any number of child classes from a single base class.