Ericsson Python Interview Questions

Q1. What is __init__?

Answer: When a new object or instance is formed, Python's __init__ constructor method is immediately invoked to allocate memory. Every class has a related __init__ method. It aids in separating local variables from a class's methods and properties.

Q2. What's the difference between Python arrays and lists?

Answer:

- Python arrays can only have items of the same data type; that is, the array's data type must be homogeneous. It uses a lot less memory than lists and is a thin wrapper around arrays in the C programming language.
- Python lists can have elements of many data types; in other words, lists' data types can be heterogeneous. Its drawback is that it uses a lot of memory.

Q3. Describe the process of making a Python script run on Unix.

Answer: The Script file must start with #!/usr/bin/env python.

Q4. What are global, protected, and private attributes in Python?

Answer:

- Public variables that are defined globally are known as global variables. The global keyword is used to use the variable in the global scope inside a function.
- Protected attributes are those that have an underscore (_sara, for example) before their identification. Although a competent developer should avoid doing so, they can nevertheless be accessed and changed from outside the class in which they are created.
- Private attributes, such as __ansh, have a double underscore before their identifier. They cannot be directly accessed or changed from the outside, and attempting to do so would result in an AttributeError.

Q5. What are Python packages and modules?

Answer: Python modular programming is made possible by two mechanisms: Python packages and Python modules. There are various benefits to modularisation.

- Simplicity: You can concentrate on a comparatively tiny part of the issue at hand by working on a single module. Development becomes simpler and less prone to mistakes as a result.
- Maintainability: The logical boundaries between various problem domains are enforced by the modules' design. Modifications to a module are less likely to affect

other program components if they are written in a way that minimizes interdependency.
- Reusability: Other application components can readily utilise functions defined in a module.
- Scoping: In order to prevent misunderstanding between identifiers from different sections of the program, modules usually declare a distinct namespace.

Q5. What is pass in Python?

Answer: In Python, a null operation is represented by the pass keyword. It is typically used to fill in blank code blocks that may run during runtime but have not yet been written. If the following code is executed without the pass statement, we can encounter several issues.

Q6. What are the common built-in data types in Python?

Answer: Python comes with several built-in data types. Type errors are likely to occur if the knowledge of data types and their compatibility with one another is overlooked, even though Python does not need data types to be stated explicitly during variable declarations. To determine the type of these variables, Python offers the type() and isinstance() methods.

Q7. What are lists and tuples? What is the key difference between the two?

Answer: Python sequence data types that can hold a collection of objects are lists and tuples. The data types of the objects recorded in the two sequences may differ. Square brackets are used to indicate lists ['sara', 6, 0.19], while parentheses are used to indicate tuples ('ansh', 5, 0.97).
However, what is the true distinction between the two? The primary distinction between the two is that tuples are immutable objects, whereas lists are mutable. This implies that whereas tuples are immutable and cannot be altered in any way, lists can be changed, appended, or sliced on the fly.

Q8. What is PEP 8, and why is it important?

Answer: Python Enhancement Proposal is referred to as PEP. A PEP is an official design document that describes a new feature for Python or its procedures, or gives information to the Python community. Given that it outlines the style standards for Python code, PEP 8 is particularly significant. It appears that you must truly and rigorously adhere to these style rules to contribute to the Python open-source community.

Q9. What is an interpreted language?

Answer: Line by line, the sentences in an interpreted language are carried out. Among the best examples of interpreted languages are Python, JavaScript, R, PHP, and Ruby. Interpreted language programs execute straight from the source code without the need for a compilation step in between.

Q10. What is a language that is dynamically typed?

Answer: We should first learn what typing is before we can comprehend a dynamically typed language. In programming languages, typing is synonymous with type-checking. Since strongly-typed languages, like Python, do not provide "type-coercion" (implicit conversion of data types), "1" + 2 will produce a type error. Conversely, a language with weak typing, like JavaScript, will just produce "12" as the result.

Two steps can be taken to type-check:

- Static: Before being executed, data types are verified.
- Dynamic: During execution, data types are verified.

Because Python is an interpreted language, type-checking is done as it is being executed, because each statement is carried out line by line. Python is a dynamically typed language as a result.

Q11. What is Python? What are Python's advantages?

Answer: Python is a general-purpose, high-level, interpreted programming language. With the correct tools and libraries, this general-purpose language may be used to create nearly any kind of application. Furthermore, Python facilitates the modelling of real-world issues and the development of applications to address them by supporting objects, modules, threads, exception-handling, and automatic memory management.

Advantages of using Python:

- Python is a general-purpose programming language that emphasises readability and lowers program maintenance costs because of its straightforward, simple grammar. Additionally, the language is fully open-source, scriptable, and compatible with third-party packages that promote code reuse and modularity.
- Its dynamic typing, dynamic binding, and high-level data structures draw a large developer community for rapid application development and deployment.

Q12. What are decorators in Python?

Answer: Decorators are essentially functions in Python that increase the functionality of an existing function without changing the function's structure. They are represented by the @decorator_name and are called from the bottom up in Python.

Q13. What are Python namespaces? Why are they used?

Answer: In Python, a namespace guarantees that program object names are distinct and compatible with one another. Python uses dictionaries to implement these namespaces, mapping a corresponding 'object as value' to a 'name as key'. This makes it possible for several namespaces to map the same name to different objects. Here are some instances of namespaces:

- Within a function, local names are part of the local namespace. During a function call, the namespace is momentarily formed and then deleted upon the function's return.

- Names from different imported packages and modules that are being utilised in the present project are included in the Global Namespace. When the package is imported into the script, this namespace is created, and it remains there until the script is executed.
- Built-in Namespace contains built-in names for different kinds of exceptions as well as built-in functions of core Python.

The range of objects to which a namespace is mapped determines its lifespan. The lifecycle of that namespace finishes when an object's scope does. Therefore, objects in an inner namespace cannot be accessed from an outer namespace.

Q14. How are arguments passed by value or by reference in Python?

Answer:

- Pass by value: A duplicate of the real object is sent. The value of the original object will remain unchanged if the value of the duplicate is altered.
- Pass via reference: The real object is referenced. The value of the original object will change if the value of the new object changes.

Q15. What is PYTHONPATH in Python?

Answer: You can add more directories where Python will search for modules and packages by setting the environment variable PYTHONPATH. Maintaining Python libraries that you don't want to install in the global default location is made much easier with this.