Top 50 Python Interview Question h Answers

python™

NEXT

1. What is Python?

Python is a high-level, interpreted programming language with dynamic semantics, known for its readability and ease of learning.

2. What are the key features of Python?

Python's key features include its easy-to-read syntax, dynamic typing, automatic memory management, and a vast standard library.

3. How is memory managed in Python?

Memory in Python is managed by the Python memory manager. Data structures and objects are stored in a private heap, with a built-in garbage collector recycling unused memory.

4. What are decorators in Python

Decorators in Python are a design pattern that allows users to modify the behavior of functions or methods without altering their code.

5. What is PEP 8?

PEP 8 is a Python Enhancement Proposal that outlines guidelines and best practices for writing Python code to maintain readability and consistency.

6. What is a lambda function in Python?

A lambda function is a small anonymous function defined using the `lambda` keyword, which can take any number of arguments but contains only a single expression.

7. What is the difference between a list and a tuple?

Lists in Python are mutable, meaning their elements can be changed, while tuples are immutable, so their elements cannot be altered after creation.

8. How does Python handle memory deallocation?

Python automatically handles memory deallocation using a built-in garbage collector that reclaims unused memory to make it available for future use.

9. What is slicing in Python?

Slicing is a technique used to extract a portion of a sequence (like lists, tuples, or strings) by specifying a start and end index.

10. What are Python modules?

Python modules are files with a `.py` extension containing Python code, which can be imported and reused in other Python programs.

11. What is the difference between Python arrays and lists?

Python arrays can only store elements of the same data type, while lists can store elements of different data types.

12. What is the difference between `deepcopy` and `copy`?

`deepcopy` creates a new compound object and recursively copies all objects found in the original, while `copy` creates a new compound object but only copies the references to objects found in the original.

13. What is a namespace in Python?

A namespace in Python is a system that ensures names are unique, thereby preventing naming conflicts.

14. What is a dictionary in Python?

A dictionary in Python is an unordered collection of key-value pairs used to store data values in a map-like structure.

15. What is the difference between `xrange` and `range`?

`xrange` returns an xrange object (an iterable), while `range` returns a list. `xrange` is more memory-efficient as it generates numbers on the fly.

16. What is pickling and unpickling?

Pickling is the process of converting a Python object hierarchy into a byte stream, while unpickling is the reverse process, converting a byte stream back into a Python object hierarchy.

17. What are Python generators?

Generators are a simple way to create iterators that yield items one at a time and can be iterated over lazily.

18. What is `__init__` in Python?

`__init__` is a special method in Python known as a constructor. It is automatically called when a new instance of a class is created.

19. What is `self` in Python?

`self` represents the instance of a class and is used to access attributes and methods within the class.

20. What is `__str__`?

`__str__` is a special method in Python that returns a string representation of an object when `print()` or `str()` is called on it.

21. What is the difference between `append()` and `extend()` methods?

`append()` adds its argument as a single element to the end of a list, while `extend()` adds each element of its argument (which should be an iterable) to the list.

22. What is a docstring in Python?

A docstring is a string literal that occurs as the first statement in a module, function, class, or method definition, used for documenting the object.

23. What is the difference between global and local variables?

Global variables are accessible throughout the entire program, while local variables are accessible only within the function or block where they are defined.

24. What is the `pass` statement in Python?

The `pass` statement is a placeholder that does nothing when executed, often used in situations where a statement is syntactically required but no action is needed.

25. What is the difference between `==` and `is`?

`==` checks for value equality between two objects, while `is` checks whether two references point to the same object in memory.

26. What is a session in Python?

A session in Python allows you to persist certain parameters or data across multiple requests within a web application.

27. What is the difference between `break`, `continue`, and `pass`?

`break` terminates the nearest enclosing loop, `continue` skips the current iteration and moves to the next one, and `pass` does nothing and is often used as a placeholder.

28. What is `*args` and `kwargs`?

`*args` is used to pass a variable number of non-keyword arguments to a function, while `kwargs` is used to pass a variable number of keyword arguments.

29. What is the difference between `isinstance()` and `type()`?

`isinstance()` checks if an object is an instance of a class or a subclass, while `type()` returns the type of the object.

30. What is the difference between `.py` and `.pyc` files?

`.py` files contain Python source code, while `.pyc` files contain the compiled bytecode that can be executed by the Python interpreter.

31. What is `__name__` in Python?

`__name__` is a special built-in variable that evaluates to the name of the current module or `"__main__"` if the module is being run as the main program.

32. What are metaclasses in Python?

Metaclasses are classes that define the behavior and rules for other classes, essentially acting as "classes of classes."

33. What is monkey patching in Python?

Monkey patching is a technique to dynamically modify or extend the behavior of libraries or classes at runtime.

34. What is the `with` statement in Python?

The `with` statement simplifies exception handling by managing resources such as file streams or database connections, ensuring they are properly acquired and released.

35. What is the difference between `staticmethod` and `classmethod`?

`staticmethod` does not receive any reference to the class or instance, while `classmethod` receives a reference to the class as its first argument.

36. What is the difference between `.py` files and `.pyw` files? `.py` files are standard Python source files, while `.pyw` files are Python scripts meant to be executed on Windows without opening a command prompt window.

37. What is the difference between `assert` and `raise`?

`assert` is used for debugging purposes to test if a condition is true, while `raise` is used to explicitly raise an exception in a program.

38. What is the `enumerate` function in Python?

`enumerate` is a built-in function that adds a counter to an iterable, returning it as an enumerate object of pairs (index, value).

39. What is the difference between `@staticmethod` and `@classmethod`?

`@staticmethod` defines a static method that does not receive any implicit arguments, while `@classmethod` defines a class method that receives the class as an implicit first argument.

40. What is the difference between `__new__` and `__init__`?

`__new__` is a static method that is called to create a new instance of a class, while `__init__` is the constructor that initializes the newly created instance.

41. What is the difference between `__getattr__` and `__getattribute__`?

`_getattr_` is invoked when an attribute is not found in the usual places, while `_getattribute_` is called for every attribute access and is used for custom behavior when accessing attributes.

42. What is the `global` keyword in Python?

The `global` keyword is used to declare that a variable inside a function is global, allowing it to be modified outside the function scope.

43. What is the difference between `__call__` and `__init__`?

`__call__` allows an instance of a class to be called as a function, while `__init__` is the constructor method used to initialize instances.

44. What is the difference between `__dict__` and `__dir__`?

`__dict__` is a dictionary holding an object's writable attributes, while `__dir__` is used to list all attributes of an object.

45. What is the `super` function in Python?

`super` is used to call methods from a parent or sibling class, often in the context of inheritance, to ensure that the correct method is called.

46. What is the difference between `__str__` and `__repr__`?

`__str__` is meant to return a readable and informal string representation of an object, while `_repr__` aims to provide a precise and formal string representation, often used for debugging.

47. What is the `zip` function in Python?

`zip` is a built-in function that aggregates elements from multiple iterables (like lists or tuples) and returns an iterator of tuples.

48. What are unit tests in Python?

Unit tests are small tests written to verify the correctness of individual functions or methods within a codebase, ensuring each part works as expected.

49. What is the Global Interpreter Lock (GIL) in Python?

The GIL is a mutex that prevents multiple native threads from executing Python bytecodes at once, ensuring that only one thread runs in the interpreter at any time.

50. What are function annotations in Python?

Function annotations allow developers to associate parts of a function (like parameters and return values) with metadata, often used for type hints, which are not enforced by the interpreter but can aid in code readability and tooling.