

Python (Basic !' Advanced) Top 50 One-Liner Interview Questions & Answers

1. What is Python? – A high-level, interpreted, general-purpose programming language.
2. What does “interpreted” mean in Python? – Code runs via an interpreter (bytecode executed by a VM) rather than being fully compiled to machine code ahead of time.
3. What are Python’s key features? – Readable syntax, dynamic typing, rich standard library, and huge ecosystem.
4. What is PEP 8? – The official style guide for writing clean, consistent Python code.
5. What is a virtual environment? – An isolated environment to manage project-specific dependencies.
6. What is pip? – Python’s package installer used to install libraries from PyPI.
7. List vs tuple? – List is mutable; tuple is immutable.
8. What is a dictionary? – A key-value data structure with fast average lookup by key.
9. What is a set? – An unordered collection of unique elements.
10. What is slicing? – Extracting parts of sequences using start:stop:step.
11. What is indentation used for? – It defines code blocks (instead of braces).
12. What is None? – A singleton representing “no value” / absence of a value.
13. What is the difference between `is` and `==`? – `is` checks identity; `==` checks equality.
14. What are mutable vs immutable types? – Mutable can change in-place (list, dict); immutable cannot (int, str, tuple).
15. What is a function in Python? – A reusable block of code defined using `def` (or `lambda`).
16. What is a lambda? – A small anonymous function written as an expression.
17. What are *args* and *kwargs*? – *args* collects positional args; ***kwargs* collects keyword args.
18. What is unpacking? – Expanding iterables into variables or arguments using *and* ***.

19. What is a module? – A Python file containing code (functions/classes/variables).
20. What is a package? – A collection of modules organized in a directory structure.
21. What is an exception? – An error event that interrupts normal flow and can be handled with try/except.
22. Difference between except and finally? – except handles errors; finally runs no matter what.
23. What is a context manager? – A construct (with) that manages setup/cleanup reliably.
24. What does name == "main" do? – Runs code only when the file is executed directly.
25. What is list comprehension? – A concise way to build lists: [expr for x in iterable if cond].
26. Generator vs list comprehension? – Generators are lazy (memory efficient); lists are eager.
27. What is an iterator? – An object implementing iter() and next() for sequential access.
28. What is a generator? – A function using yield that produces values lazily.
29. What is yield? – It returns a value and pauses function state for the next iteration.
30. What is a decorator? – A function that wraps another function to add behavior without modifying it.
31. What is OOP in Python? – Programming with classes/objects to model data + behavior.
32. What is self? – The instance reference passed to instance methods.
33. What is init? – The constructor method called when an object is created.
34. Class method vs static method? – @classmethod gets cls; @staticmethod gets neither self nor cls.
35. What is inheritance? – A class deriving behavior/attributes from another class.
36. What is polymorphism? – Same interface, different implementations (e.g., method overriding).

37. What is encapsulation? – Hiding internal state and exposing controlled access.
38. What are dunder methods? – Special methods like `str`, `len`, `add` enabling operator behavior.
39. What is a dataclass? – A decorator (`@dataclass`) that auto-generates boilerplate like `init` and `repr`.
40. What is type hinting? – Adding optional types (e.g., `def f(x: int) -> str`) for clarity and tooling.
41. What is the GIL? – A CPython lock that allows only one thread to execute Python bytecode at a time.
42. Threading vs multiprocessing? – Threading shares memory (limited by GIL for CPU-bound); multiprocessing uses separate processes for true parallelism.
43. What is `async/await`? – Syntax for writing non-blocking asynchronous code using an event loop.
44. When to use `asyncio`? – For I/O-bound concurrency (network calls, many sockets) with high throughput.
45. What is pickling? – Serializing Python objects to bytes (and back) using `pickle`.
46. What is `deepcopy` vs `shallow copy`? – Shallow copies references to nested objects; deep copies recursively copy them.
47. How do you avoid SQL injection in Python? – Use parameterized queries/prepared statements, never string concatenation.
48. What is unit testing in Python? – Testing isolated components using `unittest` or `pytest`.
49. What is mocking? – Replacing real dependencies with fake objects to test behavior in isolation.
50. How do you optimize Python code? – Use better algorithms, reduce allocations, leverage built-ins, profile (`cProfile`), and use vectorization/compiled libs when needed.