**50 Cleaned Python Interview Questions (From Your File) with 1-Line Answers**

1. **What is a Python decorator?**  
   → A decorator is a function that modifies another function’s behavior.
2. **What is the difference between mutable and immutable types?**  
   → Mutable objects can be changed in place; immutable ones cannot.
3. **What is a namespace in Python?**  
   → A namespace is a mapping between names and objects.
4. **What is the difference between is and ==?**  
   → is checks object identity; == checks value equality.
5. **How to check if a string contains a substring in Python?**  
   → Use 'substring' in 'string'.
6. **What are the differences between data types in Python?**  
   → Python has primitive types (int, float, etc.) and collection types (list, dict, etc.).
7. **What does the eval() statement do in Python?**  
   → It evaluates a string as a Python expression.
8. **What is a lambda function in Python?**  
   → An anonymous, single-expression function defined with lambda.
9. **What is encapsulation in Python?**  
   → Wrapping data and functions into a single class and hiding details.
10. **What is inheritance in Python and its types?**  
    → A way to reuse code; types: single, multiple, multilevel, hierarchical, hybrid.
11. **What are Python class methods used for?**  
    → Methods that act on the class, not instances, using @classmethod.
12. **How does Python handle memory management?**  
    → Using reference counting and garbage collection.
13. **What is data abstraction in Python?**  
    → Hiding implementation and showing only relevant details.
14. **What are Python iterators?**  
    → Objects with \_\_iter\_\_() and \_\_next\_\_() methods for iteration.
15. **What is the difference between shallow and deep copy?**  
    → Shallow copy copies references; deep copy copies all nested elements.
16. **What is PEP 8?**  
    → Python Enhancement Proposal for style guide and coding conventions.
17. **How do Python libraries work internally?**  
    → They are modules/packages with reusable code stored in .py files.
18. **What is a global variable in Python?**  
    → A variable declared outside functions and accessible everywhere.
19. **What is the difference between absolute and relative import?**  
    → Absolute uses full module path; relative uses current module hierarchy.
20. **What is an interpreter in Python?**  
    → It executes Python code line by line.
21. **How to read and write files in Python?**  
    → Using open(), read(), and write() with appropriate modes ('r', 'w', etc.).
22. **What are Python’s data types?**  
    → Includes int, float, bool, str, list, tuple, dict, set.
23. **What is list comprehension in Python?**  
    → A concise way to generate lists using [expression for item in iterable].
24. **What is duck typing in Python?**  
    → If it behaves like the expected type, it is acceptable regardless of actual type.
25. **What is the difference between shallow copy and deep copy?**  
    → Shallow copy duplicates references; deep copy duplicates full structure.
26. **What is the Global Interpreter Lock (GIL)?**  
    → A mutex in CPython that prevents multiple native threads from executing Python bytecode simultaneously.
27. **What is Python’s None keyword?**  
    → A special constant representing null or no value.
28. **What is the difference between == and is?**  
    → == compares values; is compares object identity.
29. **What is a Python package?**  
    → A collection of Python modules in a directory with \_\_init\_\_.py.
30. **What is the difference between Python 2 and 3?**  
    → Python 3 supports modern syntax and is not backward-compatible with Python 2.
31. **What is multithreading in Python?**  
    → Executing multiple threads concurrently, but limited by GIL in CPython.
32. **What is multiple inheritance?**  
    → When a class inherits from more than one parent class.
33. **What is the use of super() in Python?**  
    → It calls methods of the parent class.
34. **What is the purpose of \_\_init\_\_()?**  
    → Initializes object state when a class is instantiated.
35. **What are built-in types in Python?**  
    → Common types include int, str, list, dict, bool, set, tuple.
36. **What is tuple unpacking?**  
    → Assigning multiple variables at once using tuple syntax.
37. **What is Python's garbage collector?**  
    → Automatically deallocates memory for unreachable objects.
38. **What is the use of pass in Python?**  
    → A null operation placeholder.
39. **What is the difference between continue and break?**  
    → continue skips to next loop iteration; break exits the loop.
40. **What are Python’s control flow statements?**  
    → if, elif, else, for, while, break, continue, pass.
41. **What is a Python dictionary?**  
    → A key-value mapping data structure.
42. **How do you create a set in Python?**  
    → Using {} or set() with unique, unordered elements.
43. **What is slicing in Python?**  
    → Extracting parts of sequences using [start:stop:step].
44. **What is the difference between append() and extend() in lists?**  
    → append() adds a single element; extend() adds elements from another iterable.
45. **What are Python's string methods?**  
    → Include .upper(), .lower(), .replace(), .split(), .join(), etc.
46. **What is a comprehension in Python?**  
    → Syntactic sugar for generating sequences: list, dict, or set.
47. **What are \*args and \*\*kwargs?**  
    → Used to pass variable-length positional and keyword arguments to functions.
48. **What is the use of try-except in Python?**  
    → Handles exceptions to prevent program crashes.
49. **What are Python literals?**  
    → Fixed values like numbers, strings, and boolean values used in code.
50. **What is the difference between isinstance() and type()?**  
    → isinstance() checks inheritance; type() checks exact type.

**1. What is the purpose of the \_\_init\_\_ method?**  
It's the constructor in Python classes, used to initialize object attributes when an instance is created.

**2. What is the difference between mutable and immutable data types?**

* **Mutable**: Can be changed (e.g., list, dict).
* **Immutable**: Cannot be changed after creation (e.g., tuple, str, int).

**3. What is a Python decorator?**  
A function that modifies another function's behavior, typically using the @ symbol.

**4. What is a namespace in Python?**  
A container that maps names to objects, like a dictionary of variable names and values.

**5. What is the use of slicing in Python?**  
To extract parts of sequences like lists or strings using [start:stop:step].

**6. What is pickling in Python?**  
The process of converting a Python object into a byte stream for storage or transmission (pickle module).

**7. What is the difference between == and is?**

* ==: Compares values.
* is: Compares memory locations (identity).

**8. What is the use of \*args and \*\*kwargs?**

* \*args: For variable-length positional arguments.
* \*\*kwargs: For variable-length keyword arguments.

**9. How do you check if a string contains a specific substring?**  
Using the in keyword: "py" in "python" → True.

**10. What are control statements in Python?**  
Statements that change execution flow: if, else, elif, break, continue, pass.

**11. How do you pass arguments to a function in Python?**  
Using positional arguments, keyword arguments, or a combination of both.

**12. What is a lambda function in Python?**  
A small anonymous function: lambda x: x + 1.

**13. Why is Python called an interpreted language?**  
Python code is executed line-by-line by the interpreter rather than being compiled to machine code.

**14. What is the use of frozenset in Python?**  
An immutable set—useful when a set needs to be used as a key in a dictionary.

**15. How do you handle errors in Python?**  
Using try, except, else, finally blocks.

**16. What is the difference between staticmethod and classmethod?**

* staticmethod: No access to class or instance (self or cls).
* classmethod: Takes cls as first argument, has access to class state.

**17. What is encapsulation?**  
Restricting direct access to object data using access modifiers (e.g., \_protected, \_\_private).

**18. What is inheritance and its types?**  
Inheritance allows a class to inherit properties from another. Types:

* Single
* Multiple
* Multilevel
* Hierarchical
* Hybrid

**19. What is self in Python?**  
Refers to the current instance of the class; used to access instance variables and methods.

**20. What are Python iterators?**  
Objects with a \_\_next\_\_() method used to iterate over iterable containers like lists or tuples.

**21. What is split() used for?**  
To divide a string into a list based on a delimiter:

python

CopyEdit

"hello world".split() # ['hello', 'world']

**22. How do you do data abstraction in Python?**  
Using abstract base classes and hiding internal details with private members.

**23. What is the role of map() function?**  
Applies a function to every item in an iterable:

python

CopyEdit

map(str.upper, ['a', 'b']) → ['A', 'B']

**24. What is recursion in Python?**  
A function calling itself to solve smaller instances of a problem.

**25. What is enumerate() in Python?**  
Adds counter to an iterable:

python

CopyEdit

for i, val in enumerate(['a', 'b']): print(i, val)

**26. What are the common built-in data types?**  
int, float, str, list, tuple, dict, set, bool, bytes, NoneType

**27. How does Python handle memory management?**  
Using a private heap space and garbage collection via reference counting and the gc module.

**28. What is the difference between yield and return?**

* return: Exits function and sends back a value.
* yield: Returns a generator object and pauses the function.

**29. What is list comprehension?**  
Concise way to create lists:

python

CopyEdit

[x for x in range(5) if x % 2 == 0]

**30. What are Python magic methods?**  
Special methods with \_\_double\_underscores\_\_ used to define behavior (e.g., \_\_init\_\_, \_\_str\_\_, \_\_len\_\_).

**31. What is duck typing in Python?**  
"If it looks like a duck and quacks like a duck, it’s a duck" — Type checking is based on behavior, not class.

**32. What is the difference between append() and extend()?**

* append(): Adds a single element.
* extend(): Adds multiple elements from another iterable.

**33. What is the difference between shallow copy and deep copy?**

* **Shallow copy**: Copies references.
* **Deep copy**: Recursively copies all objects.

**34. What are PEPs and why are they important?**  
Python Enhancement Proposals — documents that describe new features or standards for Python.

**35. How do dictionaries work internally?**  
They use hash tables for fast lookup using key hashes.

**36. What is the purpose of the raise keyword?**  
To trigger exceptions manually in code.

**37. What is the Global Interpreter Lock (GIL)?**  
A mutex in CPython that prevents multiple native threads from executing Python bytecodes at once.

**38. What are relative and absolute imports?**

* **Absolute**: From the root project: import package.module
* **Relative**: From current file: from . import module

**39. How do you read and write files in Python?**  
Using open() with modes like 'r', 'w', 'a', and 'with' blocks for safety.

**40. What is Pythonic code?**  
Code that follows idiomatic and readable Python principles. Example:

python

CopyEdit

# Pythonic

squares = [x\*\*2 for x in range(10)]

# Not Pythonic

squares = []

for x in range(10):

squares.append(x\*\*2)

**41. What is polymorphism in Python?**  
Ability to use the same function or method name with different behaviors based on context.

**42. What is a docstring in Python?**  
A string literal used to document functions, classes, or modules.

**43. What are iterations in Python?**  
Repeating execution over a sequence using loops or comprehensions.

**44. What is the use of help() and dir()?**

* help(): Shows documentation.
* dir(): Lists attributes and methods of an object.

**45. How is multithreading achieved in Python?**  
Using the threading module, but limited by GIL in CPU-bound tasks.

**46. How is @property used in Python?**  
To turn a method into a read-only attribute.

**47. What is inheritance in Python?**  
Allows a class to derive features from another class (Covered in Q18).

**48. What is None in Python?**  
Represents the absence of a value or a null value.

**49. What is the difference between xrange and range?**  
In Python 2:

* range: Returns a list.
* xrange: Returns a generator (lazy evaluation).  
  In Python 3, only range exists and behaves like xrange.

**50. What is monkey patching in Python?**  
Dynamically changing or extending code at runtime, usually by modifying classes or modules.

51.**What is the collections module?**

A Python standard library module that provides **specialized data types** like Counter, defaultdict, namedtuple, deque, and OrderedDict for efficient data handling.