**150+ python interview questions for 2025**

### **Basic Python (30 Questions)**

1. What are Python’s key features?
2. How is Python interpreted?
3. What are Python’s data types?
4. How is memory managed in Python?
5. Explain the difference between list, tuple, and set.
6. What is the difference between is and ==?
7. What is slicing in Python?
8. What are Python’s reserved keywords?
9. What is a lambda function?
10. What are \*args and \*\*kwargs?
11. What is the use of the id() function?
12. How is indentation used in Python?
13. What is the use of the pass statement?
14. What are Python namespaces?
15. What is the with statement?
16. Explain mutable vs immutable types.
17. What is the difference between range and xrange?
18. How is exception handling done in Python?
19. What are list comprehensions?
20. How is type casting done in Python?
21. What are Python modules?
22. What is the purpose of the \_\_name\_\_ == "\_\_main\_\_" check?
23. What is pip in Python?
24. How to install third-party packages in Python?
25. How do you read and write files in Python?
26. What is the difference between shallow and deep copy?
27. What is a Python docstring?
28. How to reverse a string in Python?
29. How to merge two dictionaries in Python 3.9+?
30. What is unpacking in Python?

### **Intermediate Python (30 Questions)**

1. What are decorators in Python?
2. How do generators work?
3. What is a closure in Python?
4. Explain the concept of iterators and iterables.
5. What is the difference between map, filter, and reduce?
6. What are Python’s data structures?
7. How does Python manage memory for lists and dicts?
8. What is the GIL (Global Interpreter Lock)?
9. Explain the use of zip() in Python.
10. How do you handle circular imports?
11. What is monkey patching?
12. What are metaclasses?
13. What is the use of the any() and all() functions?
14. What are Python’s magic methods?
15. What does \_\_str\_\_ vs \_\_repr\_\_ do?
16. How does super() work in Python?
17. Explain the @staticmethod and @classmethod.
18. What is the difference between dir() and vars()?
19. What is the difference between a shallow copy and a deep copy?
20. What are the differences between Python 2 and Python 3?
21. What are Python sets and how are they used?
22. How to use enumerate() in Python?
23. Explain frozenset() in Python.
24. How to handle JSON data in Python?
25. What is context management in Python?
26. What are the benefits of using type hints?
27. What is the dataclasses module?
28. How to use the collections module?
29. What are namedtuples and how are they useful?
30. What is a WeakSet or WeakKeyDictionary?

### **Object-Oriented Programming in Python (15 Questions)**

1. What is object-oriented programming?
2. What is a class and an object?
3. What is inheritance? Types?
4. What is encapsulation?
5. What is polymorphism?
6. What are dunder methods?
7. How is multiple inheritance handled in Python?
8. What is MRO (Method Resolution Order)?
9. How is abstraction implemented in Python?
10. How does composition differ from inheritance?
11. What is the purpose of \_\_init\_\_()?
12. What is \_\_slots\_\_?
13. How does Python handle private and protected variables?
14. What is the difference between a class method and an instance method?
15. How do you implement interfaces in Python?

### **Advanced Python (25 Questions)**

1. What is the purpose of Python’s asyncio?
2. What is the difference between threading and multiprocessing?
3. What is the event loop in async Python?
4. What are coroutines?
5. How does the yield keyword work?
6. What is a descriptor in Python?
7. Explain metaprogramming in Python.
8. How is a singleton implemented in Python?
9. What is the use of functools.lru\_cache?
10. What are Python wheels?
11. How to handle memory leaks in Python?
12. What are Cython and PyPy?
13. What is the difference between classmethod, staticmethod, and instance method?
14. What are weak references?
15. How does Python handle garbage collection?
16. How to profile performance in Python?
17. What is duck typing?
18. What are abstract base classes?
19. Explain Python’s import system.
20. What are Python descriptors?
21. How does Python handle namespaces?
22. How is dynamic typing implemented in Python?
23. What is \_\_call\_\_()?
24. What is the difference between exec() and eval()?
25. What are memoryviews?

### **Python for Data Science & Libraries (20 Questions)**

1. What is NumPy and how is it used?
2. How does broadcasting work in NumPy?
3. What are Pandas DataFrames?
4. Difference between iloc and loc in Pandas?
5. How to handle missing data in Pandas?
6. What are groupby operations in Pandas?
7. How to merge/join DataFrames in Pandas?
8. What is vectorization?
9. Explain the use of matplotlib.
10. What is seaborn used for?
11. What is the role of scikit-learn?
12. What are pipelines in scikit-learn?
13. How does train\_test\_split() work?
14. What are confusion matrix and classification report?
15. What is overfitting and underfitting?
16. What is one-hot encoding?
17. What is the use of joblib and pickle?
18. How is TensorFlow different from PyTorch?
19. What is feature engineering?
20. How do you evaluate a machine learning model?

### **Python for Web Development (10 Questions)**

1. What is Flask and how does it work?
2. Difference between Flask and Django?
3. How to define routes in Flask?
4. How does middleware work in Django?
5. How to handle forms in Flask?
6. What is ORM in Django?
7. What is Jinja2 template engine?
8. How to handle sessions and cookies in Flask?
9. How to manage static files in Django?
10. How to build REST APIs using Flask?

### **Testing, Tools, and Best Practices (10 Questions)**

1. How do you write unit tests in Python?
2. What is the use of pytest?
3. How to mock objects in Python?
4. What is coverage in testing?
5. What are fixtures in pytest?
6. How to use virtual environments in Python?
7. What is flake8 and black?
8. How to handle logging in Python?
9. What is the use of the unittest module?
10. What are some best practices in Python coding?

### **Tricky and Conceptual (10 Questions)**

1. What happens when you modify a list while iterating over it?
2. Can you explain Python’s pass-by-object-reference?
3. How does Python handle recursion limits?
4. What is the use of \_\_getattr\_\_ and \_\_setattr\_\_?
5. What are common anti-patterns in Python?
6. Can you explain the difference between shallow and deep immutability?
7. What happens if you use a mutable default argument?
8. How to avoid circular imports?
9. Why is Python considered dynamically typed?
10. Can you sort a dict by values?
11. How would you implement a custom iterable?