50+ Python Interview Questions, Evaluation Checklist & Hiring Scorecard

A Complete Hiring Toolkit for Recruiters & Hiring Managers

How to Use This Guide

This guide is designed to help recruiters, hiring managers, and HR leaders run **structured, bias-free, and high-signal Python interviews**.

4-Step Hiring Workflow

1. **Screen →** Use resume checklist (Section 10)
2. **Assess →** Pick questions by experience level
3. **Evaluate →** Use checklist (Section 7)
4. **Decide →** Scorecard + clear hire bar

Suggested 60-Min Interview Plan

- **0-10 min →** Fundamentals
- **10-25 min →** Core concepts
- **25-40 min →** Coding/problem-solving
- **40-55 min →** Real-world scenarios
- **55-60 min →** Candidate questions + wrap

Tip: Keep at least **30% time for problem-solving + scenarios**—this is where real signal comes from.

SECTION 1: BEGINNER QUESTIONS (0–2 YEARS)

1. What are Python data types?

Answer: int, float, str, list, tuple, dict, set, bool
Good Answer Looks Like: Gives examples + when to use each
Score Guide:

- 5/5: Explains with examples + use cases
- 3/5: Lists types only
- 1/5: Misses key types

2. List vs Tuple

Answer: Lists are mutable; tuples are immutable
Follow-up: When would you use tuple over list?

3. What is a dictionary?

Answer: Key-value pair data structure (hash map)
Red Flag: Cannot explain key uniqueness

4. What is indentation in Python?

Answer: Defines code blocks instead of braces

5. What is dynamic typing?

Answer: Variable type decided at runtime

6. What is slicing?

Answer: Accessing subsets of sequences using `[start:end:step]`

7. What is a set?

Answer: Unordered collection with unique elements

8. What are functions in Python?

Answer: Reusable blocks of code using `def`

9. What is None?

Answer: Represents absence of value

10. What is type casting?

Answer: Converting data types (e.g., int to str)

SECTION 2: INTERMEDIATE QUESTIONS (2–5 YEARS)

11. What are decorators?

Answer: Functions that modify behavior of other functions
Example: Logging, authentication

12. Deep Copy vs Shallow Copy

Answer: Shallow copies references; deep copies nested objects

13. What is GIL?

Answer: Global Interpreter Lock restricts multi-thread execution in CPython

14. Generators vs Iterators

Answer: Generators use `yield`; iterators implement `__iter__()`

15. List vs NumPy Array

Answer: NumPy is faster for numerical ops, supports vectorization

16. What is exception handling?

Answer: try-except blocks to manage runtime errors

17. What are lambda functions?

Answer: Anonymous single-line functions

18. What is list comprehension?

Answer: Concise way to create lists

19. What is init?

Answer: Constructor method in classes

20. What is inheritance?

Answer: Class derives properties from another class

SECTION 3: ADVANCED QUESTIONS (5+ YEARS)

21. How does Python memory management work?

Answer: Reference counting + garbage collector

22. Multithreading vs Multiprocessing

Answer: Threads share memory; processes don't

23. How do you optimize a slow Python script?

Answer: Profiling, caching, vectorization, async

24. What is async programming?

Answer: Non-blocking execution using async/await

25. How would you design a scalable system?

Answer: Microservices, caching, load balancing, queues

26. What is monkey patching?

Answer: Modifying code at runtime

27. What are metaclasses?

Answer: Classes of classes

28. Explain context managers

Answer: `with` statement for resource handling

29. What is dependency injection?

Answer: Passing dependencies externally

30. What is serialization?

Answer: Converting objects to JSON/bytes

SECTION 4: CODING QUESTIONS

31. Reverse a string without slicing

Expected: Loop or stack logic

32. Find duplicates in a list

Expected: Set or hashmap

33. Check palindrome

34. Implement LRU cache

35. Find first non-repeating character

36. Merge two sorted lists

37. Detect cycle in list

38. Count frequency of elements

39. Flatten nested list

40. Two-sum problem

Evaluation Criteria:

- Logic clarity
- Time complexity
- Code readability

SECTION 5: REAL-WORLD SCENARIOS

41. API latency increased by 3x

Strong Answer: Profiling → logs → DB queries → caching

42. Process 1M records daily

Strong Answer: Batching, multiprocessing, streaming

43. Debug production issue

Strong Answer: Logs, rollback, RCA

44. System crash under load

Strong Answer: Load testing, scaling, queues

45. Improve codebase quality

Strong Answer: Refactoring, tests, linting

SECTION 6: EVALUATION CHECKLIST

Skill	What Great (5/5) Looks Like	What Avg (3/5) Looks Like	What Poor (1/5) Looks Like
Core Python	Strong fundamentals + examples	Basic knowledge	Weak basics
Problem Solving	Structured + optimized	Works but inefficient	Struggles
Debugging	Root cause approach	Trial & error	No method
System Thinking	Scalable design	Limited thinking	None
Communication	Clear + structured	Understandable	Confusing

SECTION 7: HIRING SCORECARD

Candidate Name:
Role:
Experience:

Criteria	Scorecard (__ /10)
Technical Skills	
Problem Solving	
Practical Knowledge	
Communication	
Culture Fit	

Final Recommendation:

- ✅ Hire (40+)
- 🤔 Consider (30–39)
- ❌ Reject (<30)

SECTION 8: COMMON HIRING MISTAKES

- Over-indexing on theory
- Ignoring debugging ability
- Not testing real-world thinking
- Hiring based on frameworks, not fundamentals

SECTION 9: ROLE-BASED MAPPING

Role	Focus Area
Backend Dev	APIs, DB, performance
Data Analyst	Pandas, SQL
Data Scientist	ML, stats
Automation	Testing, scripting

SECTION 10: BONUS

Resume Screening Checklist

- Real-world Python usage
- GitHub projects
- Problem-solving exposure

Final Round Questions

- Biggest challenge solved?
- How do you debug failures?

SECTION 11: INDIA HIRING INSIGHTS (2025–26)

Salary Benchmarks (Approx)

- 0–2 yrs: INR 3–8 LPA
- 2–5 yrs: INR 8–18 LPA
- 5+ yrs: INR 18–40+ LPA

Talent Trends

- High demand for **Python + AI/ML + Data roles**
- Strong shift toward **practical coding + system thinking**
- Companies prioritizing **problem-solving over degrees**

Hiring Challenges

- Candidate inflation (resume vs real skill gap)
- Offer drop rates in tech roles
- Speed vs quality trade-offs

Need help hiring Python talent faster, better, and at scale? Talk to our experts today.

End of Guide