Python Fundamentals

1. What is Python and why is it popular?
2. What are the key features of Python?
3. What is the difference between Python 2 and Python 3?
4. What is PEP 8 and why is it important?
5. How does Python execute code internally?
6. What are Python's main applications?
7. What are Python's advantages and limitations?
8. What are Python's built-in data types?
9. What are Python's memory management techniques?
10. What are Python namespaces?

Data Types and Structures

11. What are mutable and immutable data types in Python?
12. What is the difference between list and tuple?
13. What is the difference between shallow copy and deep copy?
14. What are Python sets and their uses?
15. What are Python dictionaries and their characteristics?
16. How do you remove duplicates from a list?
17. What are frozensets?
18. What are Python ranges and how are they used?
19. How does Python handle Boolean values?
20. What is the None type in Python?

Control Flow and Functions

21. What control flow statements are available in Python?
22. What is the difference between break, continue, and pass?
23. How do you define a function in Python?
24. What are default arguments in functions?
25. What is the difference between *args and **kwargs?
26. What are lambda functions?
27. What is function recursion?
28. What is the difference between return and yield?
29. What is function scope in Python?
30. What is the difference between local and global variables?

Object-Oriented Programming

31. What are the four pillars of OOP?
32. What is inheritance in Python?
33. What is multiple inheritance?
34. What is method overriding?
35. What are instance methods, class methods, and static methods?
36. What are Python magic methods (dunder methods)?
37. What is encapsulation in Python?
38. What is polymorphism in Python?
39. What is abstraction in Python?
40. How do you implement private variables in Python?

Advanced Concepts

41. What are Python decorators?
42. What are Python closures?
43. What are generators?
44. What is the difference between iterators and generators?
45. What are list comprehensions?
46. What are dictionary comprehensions?
47. What is a context manager in Python?
48. What is monkey patching?
49. What are metaclasses in Python?
50. What is the difference between shallow copy and deep copy?

Exception Handling

51. What is the difference between errors and exceptions?
52. How do you handle exceptions in Python?
53. What is the difference between try-except-else-finally?
54. What are built-in exceptions in Python?
55. How do you create a custom exception class?
56. What is exception chaining?
57. What is the purpose of the raise keyword?
58. What happens if an exception is not handled?
59. What is the purpose of the assert statement?
60. What is the difference between SyntaxError and RuntimeError?

File Handling and I/O

61. How do you read and write files in Python?
62. What are different file modes in Python?
63. What is the with statement in file handling?
64. How do you read CSV files in Python?
65. How do you write to a CSV file in Python?
66. How do you read JSON files in Python?
67. What is the difference between binary and text modes?
68. How do you handle file exceptions in Python?
69. What is file pointer and how does seek() work?
70. How do you check if a file exists in Python?

Memory Management and Performance

71. How does Python manage memory?
72. What is reference counting in Python?
73. What is garbage collection in Python?
74. What is the Global Interpreter Lock (GIL)?
75. How does Python handle multithreading?
76. What is the difference between threading and multiprocessing?
77. How do you optimize Python code performance?
78. What is memoization in Python?
79. What are weak references in Python?
80. What is Python's memoryview object?

Modules and Packages

81. What is the difference between a module and a package?
82. How do you import a module in Python?
83. What is the purpose of __init__.py?
84. What are relative and absolute imports?
85. What is the difference between import module and from module import?
86. What is the purpose of __name__ == "__main__" in Python?
87. How do you create a custom Python package?
88. What is pip and how is it used?
89. What is virtualenv in Python?
90. How do you publish a Python package to PyPI?

Data Structures and Algorithms

91. How do you reverse a list in Python?
92. What is the time complexity of list operations?
93. How do you detect duplicates in a list?
94. How do you implement a stack in Python?
95. How do you implement a queue in Python?
96. How do you implement a priority queue in Python?
97. How do you find the maximum and minimum in a list?
98. How do you sort a list of tuples by the second element?
99. What is the difference between deep copy and shallow copy in data structures?
100. How do you find the intersection of two lists?

String Manipulation

101. How do you reverse a string in Python?
102. How do you check if a string is a palindrome?
103. What are f-strings in Python?
104. How do you concatenate strings in Python?
105. What are raw strings in Python?
106. How do you remove whitespace from a string?
107. How do you split and join strings in Python?
108. How do you find substrings in a string?
109. What is string immutability?
110. How do you format numbers inside strings?

Best Practices and Common Pitfalls

111. What are Python naming conventions?
112. What is the difference between append() and extend()?
113. Why should mutable default arguments be avoided in functions?
114. What are Python's common anti-patterns?
115. What is the difference between is and ==?
116. Why should wildcard imports be avoided?
117. What is the difference between shallow copy and assignment?
118. What is the Zen of Python?
119. How do you handle circular imports?
120. What are Python type hints and annotations?

JSON, API, and Automation

121. How do you parse JSON data in Python?
122. How do you convert Python objects into JSON format?
123. How do you call REST APIs in Python without external libraries?
124. What is the difference between json.load() and json.loads()?
125. What are common use cases of JSON in Python?
126. What is the difference between REST API and SOAP API?
127. How do you send GET and POST requests in Python?
128. How do you handle API authentication in Python?
129. What is an API rate limit and how to handle it in Python?
130. How do you parse XML responses in Python?

Boto3 and AWS with Python

Basics (Beginner Level)

1. What is Boto3 and why is it used in Python?
2. How do you install Boto3 in a Python environment?
3. What are the two main interfaces provided by Boto3? (client vs resource)
4. How do you create a Boto3 client for S3? Show syntax.
5. How do you configure AWS credentials for Boto3?
6. What is the difference between using environment variables and ~/.aws/credentials file for authentication?
7. How do you list all buckets in S3 using Boto3?
8. How do you upload a file to S3 using Boto3?
9. How do you download a file from S3 using Boto3?
10. What's the difference between client.upload_file() and put_object() in Boto3?

Intermediate (Practical Use Cases)

11. How do you paginate through large lists of objects in S3 using Boto3?
12. How can you check if a bucket exists using Boto3?
13. How do you handle exceptions in Boto3 (e.g., NoSuchBucket, AccessDenied)?
14. How do you delete multiple objects from an S3 bucket at once?
15. How do you generate a pre-signed URL for S3 with Boto3?
16. How do you enable server-side encryption while uploading to S3 with Boto3?
17. How do you copy an object from one S3 bucket to another using Boto3?
18. How do you create an EC2 instance using Boto3?
19. How do you stop and start an EC2 instance using Boto3?
20. How do you fetch the status of an EC2 instance using Boto3?

Advanced (Architecture + Best Practices)

21. How do you assume an IAM role in Boto3?
22. What is a Boto3 Session, and why would you use it instead of the default session?
23. How do you enable retries in Boto3 when throttling errors occur?
24. How do you work with DynamoDB using Boto3 (insert, read, update, delete)?
25. How do you use Boto3 to publish a message to an SNS topic?
26. How do you trigger a Lambda function using Boto3?
27. How do you use Boto3 with AWS STS (Security Token Service)?
28. How do you configure Boto3 to use a custom endpoint (e.g., LocalStack)?
29. How do you handle large file uploads with Boto3 using multipart upload?
30. How do you debug API calls made by Boto3 (logging requests & responses)?

Ansible with Python

141. How does Ansible use Python internally?
142. How do you execute Python scripts from Ansible playbooks?
143. How do you write a custom Ansible module in Python?
144. What is ansible-runner in Python?
145. How do you run an Ansible playbook programmatically from Python?
146. How do you parse Ansible inventory files in Python?
147. What are Ansible facts and how to access them in Python?
148. How do you debug Ansible modules written in Python?
149. What are idempotency checks in Ansible modules?
150. How do you use Python with Ansible for dynamic inventory management?
