Python Interview Mastery

115 Essential Questions with Concise Answers

Pooja Pawar

Q: What are Python’s key features?
A: Python is high-level, interpreted, dynamically typed, object-oriented, and cross-platform, with a rich standard library and strong community support.

Q: What does it mean that Python is dynamically typed?
A: Variable types are determined at runtime, so you don’t need to declare data types explicitly.

Q: How is Python different from compiled languages?
A: Python code is compiled into bytecode and executed by the Python Virtual Machine, rather than directly compiled to machine code.

Q: What is PEP 8 and why is it important?
A: PEP 8 is Python’s official style guide that helps maintain readable, consistent, and clean code.

Pooja Pawar

Q: What are Python’s built-in data types?
A: Common types include int, float, complex, str, list, tuple, set, dict, bool, and NoneType.

Q: What is the difference between mutable and immutable objects?
A: Mutable objects can be changed after creation, while immutable objects cannot be modified once created.

Q: Why is indentation important in Python?
A: Indentation defines code blocks and is mandatory, unlike many other programming languages.

Q: How does Python manage memory?
A: Python uses a private heap, reference counting, and a garbage collector for automatic memory management.

Q: Why are tuples faster than lists?
A: Tuples have a fixed size and require less memory overhead.

Pooja Pawar

Q: What are namespaces in Python?
A: Namespaces map variable names to objects and help avoid naming conflicts between variables.

Q: What is the difference between is and == ?
A: is checks object identity, while == checks value equality.

Q: How does Python handle variable scope?
A: Python follows the LEGB rule: Local, Enclosing, Global, and Built-in scope.

Q: What is Python bytecode?
A: Bytecode is an intermediate representation of Python code executed by the Python Virtual Machine.

Q: What is None in Python?
A: None represents the absence of a value and is commonly used to indicate null or empty results.

Pooja Pawar

Q: Why is Python called an interpreted language?
A: Because Python code is executed line by line by the interpreter at runtime.

Q: What are Python's built-in data structures?
A: List, tuple, set, and dictionary are the core built-in data structures.

Q: What is the difference between a list and a tuple?
A: Lists are mutable, while tuples are immutable and generally faster.

Q: Why are sets useful in Python?
A: Sets store unique elements and provide fast membership testing.

Q: What is a dictionary in Python?
A: A dictionary stores data as key-value pairs and allows fast lookups using keys.

Pooja Pawar

Q: How is Python platform-independent?
A: Python programs run on any system with a compatible Python interpreter, without code changes.

Q: What is the difference between a list and a tuple?
A: A list is mutable and can be modified, while a tuple is immutable and cannot be changed after creation.

Q: When should you use a tuple instead of a list?
A: Use a tuple when data should remain constant, needs to be hashable, or when performance is slightly critical.

Q: What is a set in Python?
A: A set is an unordered collection of unique elements used for fast membership testing and duplicate removal.

Pooja Pawar

Q: How is a dictionary different from other data structures?
A: A dictionary stores data as key-value pairs and allows fast access using keys instead of indexes.

Q: How do dictionaries maintain fast lookups?
A: Dictionaries use hash tables internally to achieve average constant-time access.

Q: What is a frozen set?
A: A frozenset is an immutable version of a set and can be used as a dictionary key.

Q: What happens if duplicate keys are added to a dictionary?
A: The latest value overwrites the previous value for the same key.

Q: What is heapq used for?
A: It implements a priority queue using a heap-based algorithm.

Pooja Pawar

Q: What is slicing in Python?
A: Slicing extracts a portion of a sequence using start, end, and step values.

Q: How does negative indexing work in Python?
A: Negative indexes access elements from the end of a sequence.

Q: What is shallow copy?
A: A shallow copy creates a new object but references the same nested objects.

Q: What is deep copy?
A: A deep copy creates a completely independent copy, including all nested objects.

Q: How can duplicates be removed from a list?
A: By converting the list to a set or using dictionary-based techniques.

Q: What is list comprehension?
A: It is a compact syntax used to create lists using expressions and loops.

Pooja Pawar

Q: Why is list comprehension preferred over loops?
A: It improves readability and is often faster than traditional loops.

Q: What is the difference between is and == in collections?
A: is checks whether two variables reference the same object, while == checks content equality.

Q: How do you reverse a list efficiently?
A: By using slicing or built-in reversing utilities.

Q: How is a stack implemented in Python?
A: Using lists or specialized data structures for efficient push and pop operations.

Q: How is a queue implemented in Python?
A: Using collections designed for fast insertions and removals from both ends.

Pooja Pawar

Q: What is the difference between a function and a method?
A: A function is independent, while a method is associated with an object or class.

Q: What are arguments and parameters in Python?
A: Parameters are defined in the function signature, while arguments are values passed during a function call.

Q: What are positional arguments?
A: Positional arguments are passed based on their order in the function definition.

Q: What are keyword arguments?
A: Keyword arguments are passed using parameter names, making function calls clearer and order-independent.

Q: What are default arguments?
A: Default arguments have predefined values used when no argument is provided.

Pooja Pawar

Q: What are variable-length arguments?
A: They allow a function to accept an arbitrary number of arguments.

Q: Why are keyword arguments useful?
A: They improve readability and reduce errors caused by incorrect argument ordering.

Q: What is recursion?
A: Recursion is a technique where a function calls itself to solve a smaller part of the same problem.

Q: What is a base case in recursion?
A: The base case stops the recursive calls and prevents infinite recursion.

Q: What are lambda functions?
A: Lambda functions are small anonymous functions defined in a single expression.

Pooja Pawar

Q: How are generators different from normal functions?
A: Generators use yield instead of return and maintain state between calls.

Q: Why are generators memory-efficient?
A: They generate values on demand instead of storing all results in memory.

Q: What is object-oriented programming?
A: OOP is a programming paradigm based on objects that contain data and behavior.

Q: What is a class in Python?
A: A class is a blueprint for creating objects with attributes and methods.

Q: What is an object?
A: An object is an instance of a class that represents a real-world entity.

Pooja Pawar

Q: What is the purpose of the constructor?
A: The constructor initializes an object's attributes when it is created.

Q: What is inheritance?
A: Inheritance allows a class to reuse properties and methods of another class.

Q: What is polymorphism?
A: Polymorphism allows the same method name to behave differently in different contexts.

Q: What is encapsulation in Python?
A: Encapsulation bundles data and methods together and restricts direct access to internal details.

Q: What is abstraction in Python?
A: Abstraction hides complex implementation details and exposes only essential functionality.

Pooja Pawar

Q: What is multilevel inheritance?
A: Multilevel inheritance forms a chain where a class inherits from another derived class.

Q: What is Method Resolution Order (MRO)?
A: MRO defines the order in which Python searches for methods in an inheritance hierarchy.

Q: Why is MRO important?
A: It prevents ambiguity and ensures predictable method lookup in multiple inheritance.

Q: What is the diamond problem?
A: It occurs when multiple inheritance creates ambiguity in method resolution, which Python resolves using

Q: What are decorators in Python?
A: Decorators are functions that modify or extend the behavior of other functions or methods.

Pooja Pawar

Q: Why are decorators useful?
A: They allow adding functionality like logging or authentication without modifying core logic.

Q: What is a closure?
A: A closure is a function that retains access to variables from its enclosing scope.

Q: What are magic methods in Python?
A: Magic methods are special methods that enable built-in behavior like addition or string representation.

Q: What is the difference between str and repr?
A: str is user-friendly, while repr is developer-oriented and unambiguous.

Q: What is monkey patching?
A: Monkey patching dynamically modifies class or module behavior at runtime.

Pooja Pawar

Q: When should monkey patching be avoided?
A: It should be avoided in large systems due to maintainability and debugging risks.

Q: What is a static method?
A: A static method does not access class or instance data and behaves like a regular function.

Q: What is a class method?
A: A class method receives the class as its first argument and can modify class-level data.

Q: When should static methods be used?
A: When functionality is related to a class but doesn’t require instance or class data.

Q: What is operator overloading?
A: Operator overloading allows custom behavior for operators using magic methods.

Pooja Pawar

Q: What is the purpose of init?
A: It initializes object attributes when an instance is created.

Q: Can Python support interfaces?
A: Yes, through abstract base classes that enforce method implementation.

Q: What is an exception in Python?
A: An exception is an error that occurs during program execution and disrupts normal flow.

Q: What is the difference between syntax errors and exceptions?
A: Syntax errors occur before execution, while exceptions occur during runtime.

Q: What is exception handling?
A: Exception handling allows programs to handle runtime errors gracefully without crashing.

Pooja Pawar

Q: What is the purpose of try and except blocks?
A: They catch and handle exceptions to prevent program termination.

Q: What is the use of the else block in exception handling?
A: The else block runs when no exception occurs in the try block.

Q: What is the purpose of finally?
A: The finally block always executes, typically for cleanup tasks.

Q: Can multiple exceptions be handled in a single except block?
A: Yes, by grouping multiple exception types together.

Q: What is raising an exception?
A: Raising an exception forces an error condition intentionally using built-in or custom exceptions.

Pooja Pawar

Q: What is a custom exception?
A: A user-defined exception created to handle specific error scenarios.

Q: What is exception chaining?
A: Exception chaining links one exception to another for better debugging.

Q: What is the difference between raise and assert?
A: raise triggers exceptions intentionally, while assert is used for debugging checks.

Q: What happens if an exception is not handled?
A: The program terminates and displays a traceback.

Q: What is a traceback?
A: A traceback shows the sequence of function calls leading to an exception.

Pooja Pawar

Q: Why is exception handling important in production code?
A: It improves reliability, user experience, and system stability.

Q: What are built-in exceptions in Python?
A: Predefined exception classes like TypeError, ValueError, and IndexError.

Q: When should broad exception handling be avoided?
A: When it hides errors and makes debugging difficult.

Q: What is graceful error handling?
A: Handling errors in a controlled way without crashing the application.

Q: How does exception handling improve code quality?
A: It makes code robust, predictable, and easier to maintain.

Pooja Pawar

Q: What is a module in Python?
A: A module is a file that contains Python code such as functions, variables, and classes.

Q: What is a package in Python?
A: A package is a collection of related modules organized in a directory structure.

Q: Why are modules used in Python?
A: Modules improve code reusability, organization, and maintainability.

Q: What is the role of init in a package?
A: It marks a directory as a package and can initialize package-level code.

Q: What is the difference between import module and from module import name?
A: import module loads the entire module, while from imports specific components.

Pooja Pawar

Q: What are absolute imports?
A: Absolute imports specify the full path of a module from the project root.

Q: What are relative imports?
A: Relative imports refer to modules within the same package hierarchy.

Q: What is sys.path?
A: sys.path is a list of directories Python searches when importing modules.

Q: What is PYTHONPATH?
A: PYTHONPATH is an environment variable that adds directories to sys.path.

Q: What is a virtual environment?
A: A virtual environment isolates project dependencies to avoid version conflicts.

Q: Why are virtual environments important?
A: They ensure consistent dependencies across development, testing, and production.

Pooja Pawar

Q: What is pip?
A: pip is Python’s package manager used to install and manage external libraries.

Q: What is the difference between pip and conda?
A: pip manages Python packages, while conda manages both packages and environments.

Q: What is dependency management?
A: It ensures required libraries and correct versions are available for a project.

Q: What is a requirements file?
A: It lists project dependencies so they can be installed consistently.

Q: What happens if a module name conflicts with a built-in module?
A: The local module may override the built-in one, causing unexpected behavior.

Pooja Pawar

Q: What is import caching in Python?
A: Imported modules are cached to avoid reloading them multiple times.

Q: How does Python locate a module during import?
A: It searches built-in modules, then sys.path directories in order.

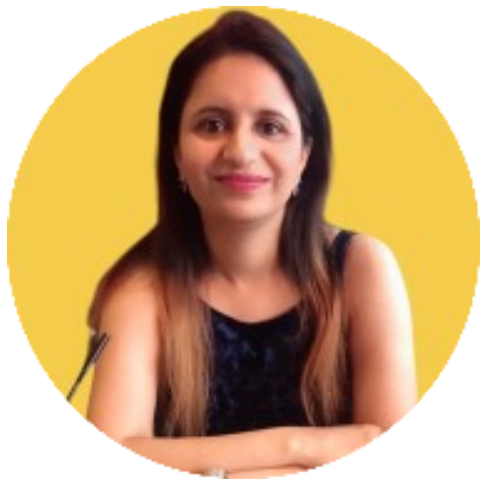
Q: Why should wildcard imports be avoided?
A: They reduce readability and can cause name conflicts.

Q: What is file handling in Python?
A: File handling allows Python to read from and write to files for data storage and retrieval.

Q: What is the difference between text mode and binary mode in file handling?
A: Text mode handles data as strings with encoding, while binary mode handles raw bytes.

Pooja Pawar

Follow for more data analytics content



Pooja Pawar