# Python Interview Questions for Experienced with PDF

By **Meenakshi Agarwal**

Last updated: Sep 06, 2024 3:33 pm

Python is one of the most popular programming languages that many people use today. For developers with several years of experience, interviews can become more challenging, as employers expect you to have a deeper understanding of advanced Python concepts, libraries, and best practices. In this guide, we'll explore the most commonly asked Python interview questions for experienced candidates, along with explanations.

## Python Interview Questions Guide for Experienced (PDF)

If you're preparing for an interview, you can also download a PDF version of this guide to keep for future reference. Let's dive in!

### Python Interview Questions Guide Includes

This Python interview PDF guide for experienced professionals covers questions on the following important topics.

## Python Basics (but Deeper)

Even for experienced developers, interviewers often begin by assessing your foundational knowledge. They will ask questions that seem basic but require a deeper explanation.

▼ What are Python's built-in data types?
Some of the main data types in Python are int, float, str, list, tuple, set, and dict. Each one is used for different kinds of information.

▼ Explain the difference between lists and tuples.
While both are used to store collections of items, lists are mutable, meaning they can be changed, while tuples are immutable. This immutability makes tuples faster and more memory-efficient.

▼ What is list comprehension in Python?

List comprehension is a quick way to create lists by writing the logic inside square brackets. It's simpler and faster than using a loop.

▼ How are list comprehensions used?

List comprehensions provide a concise way to create lists. For example, squares = [x**2 for x in range(10)] will create a list of square numbers from 0 to 9.

▼ What are Python decorators?

Decorators are functions that modify other functions. They help add extra features to existing code without changing the original function.

▼ How does Python handle memory management?

Python has an automatic garbage collector that manages memory allocation and deallocation. It uses a combination of reference counting and cycle detection.

## Object-Oriented Programming (OOP)

Experienced Python developers need a strong grasp of OOP principles, as these are critical for writing modular and reusable code. Expect interviewers to dig into your knowledge of classes, objects, and design patterns.

▼ What is the difference between **init** and **new**?

The **init** method initializes a newly created object, while **new** is responsible for creating the object itself. Normally, you only override **init**, but sometimes **new** is useful for custom object creation.

▼ Explain multiple inheritance in Python.

Multiple inheritance allows a class to inherit from more than one base class. It is powerful but can lead to complications like the "diamond problem." Python uses the C3 Linearization algorithm (Method Resolution Order – MRO) to resolve such issues.

▼ What are static methods and class methods?

Static methods don't take self or cls as their first parameter and act like regular functions. Class methods, on the other hand, take cls as the first parameter and can modify class

state.

▼ What is inheritance in Python?

Inheritance allows a class to use properties and methods from another class. This helps reuse code and keep it organized. Check here for more on inheritance in Python.

▼ What is a method overriding in Python?

Method overriding happens when a subclass has a method with the same name as the one in the parent class. It allows the subclass to have its own version of that method.

▼ What are dunder (double underscore) methods?

Dunder methods, like **init** or **str**, are special methods used to define the behavior of Python objects. They help in customizing how objects work.

## Advanced Python Concepts

At this level of interview, as an experienced person, you should able to answer questions on Python decorators, generators, and context managers. These are essential for writing efficient Python code.

▼ How are decorators used in Python programs?

Decorators are functions that modify the behavior of another function or method. They are often used for logging, authentication, or modifying the output of a function. Example:

```
def my_decorator(func):
    def wrapper():
        print("Something before the function.")
        func()
        print("Something after the function.")
    return wrapper

@my_decorator
def say_hello():
    print("Hello!")
```

▼ How do Python generators work?

Generators are functions that return items one at a time, which saves memory. They use the yield keyword instead of return.

▼ What are context managers?

Context managers are used to properly manage resources like file streams or database connections. The with statement is a common way to implement context managers, ensuring resources are properly closed or released after use.

▼ How does Python manage memory?

Python uses a technique called garbage collection to free up memory that is no longer used. It automatically manages memory using reference counting and a garbage collector.

▼ How does Python handle errors?

Python uses try-except blocks to manage errors. You put code that might cause an error in a try block, and handle the error in the except block.

▼ What are closures in Python?

Closures happen when an inner function remembers variables from the outer function, even after the outer function has finished executing.

## Python Libraries and Frameworks

Experienced developers often work with various libraries and frameworks. Interviewers will want to know how well you understand Python's ecosystem.

▼ What is Django used for in Python?

Django is a web framework that helps you build websites quickly and securely. It provides tools for handling things like databases, routing, and security.

▼ How does Flask differ from Django?

Flask is a smaller and simpler framework than Django. It's more flexible but needs more work to set up. Django comes with many built-in features, while Flask lets you choose what you want to add.

▼ What is Pandas used for?

Pandas is a library used for data analysis. It helps you work with large datasets and makes it easy to manipulate and visualize data.

▼ What is NumPy?
NumPy is a tool in Python for doing math with numbers. It helps you work with arrays and matrices, which are like lists of numbers. People often use NumPy for science and math projects because it makes calculations easier.

▼ How do you work with external libraries in Python?
You can install libraries using pip and manage them in a requirements.txt file or a virtual environment. Example: pip install requests installs the requests library for handling HTTP requests.

## Error Handling and Debugging

Good error handling is crucial for writing robust code. Interviewers will ask you about how you manage exceptions and debug your code.

▼ What is the difference between try-except and try-finally?
try-except handles exceptions that occur in the try block, while try-finally ensures that the finally block runs no matter what happens in the try block, making it useful for cleanup.

▼ What are some common exceptions in Python?
Some common exceptions include IndexError, KeyError, ValueError, TypeError, and AttributeError. Handling these properly ensures your program doesn't crash unexpectedly.

▼ How do you debug a Python program?
You can use the built-in pdb module for step-by-step debugging. You can also use IDEs like PyCharm or VS Code, which come with graphical debuggers.

## File Handling and Data

Reading from and writing to files is a common task in Python, especially when dealing with data.

▼ How do you open a file in Python?

Use the open() function to open a file. Example:

```
with open('file.txt', 'r') as file:
    content = file.read()
```

▼ How do you work with CSV files in Python?

The csv module can be used to read and write CSV files. Alternatively, you can use the pandas library for more advanced operations.

## Concurrency and Parallelism

Handling multiple tasks at the same time is an important skill for experienced Python developers.

▼ What is the Global Interpreter Lock (GIL) in Python?

The GIL is a mechanism that prevents multiple threads from running Python code at the same time. It makes multi-threading slower for CPU-bound tasks.

▼ What is multithreading in Python?

Multithreading in Python allows your program to run several threads at once, which is helpful for tasks like reading files or downloading data from the internet.

▼ What is the difference between multithreading and multiprocessing?

Multithreading runs multiple threads in a single process, while multiprocessing runs multiple processes, each with its own memory. Multiprocessing is useful for CPU-heavy tasks.

▼ How do you achieve concurrency in Python?

You can use threading for I/O-bound tasks and multiprocessing for CPU-bound tasks. AsyncIO is another option for non-blocking, I/O-heavy operations.

▼ What is asyncio, and how is it used?

The asyncio is a library for asynchronous I/O. It lets you write code that can perform tasks concurrently without the complexity of threads.

▼ How do async and await work in Python?

The async is used to declare asynchronous functions, and await is used to wait for an asynchronous operation to complete. Example:

```
async def main():
    await asyncio.sleep(1)
```

## Databases in Python

Knowing how to work with databases is crucial, especially in web development.

▼ How does Python interact with databases?

Python uses libraries like SQLite or MySQL with Python to connect to databases. You can send SQL commands from your Python code to query and modify the database.

▼ What is SQLAlchemy?

SQLAlchemy is a Python library that helps manage databases using object-relational mapping (ORM). It lets you interact with databases using Python classes instead of raw SQL.

▼ How would you use a database with Django?

Django has a built-in ORM, so you can define models for your database tables. Django handles the database connection and queries automatically.

## Testing in Python

In a Python interview for experienced, interviewers expect candidates to answer questions related to testing. This is crucial because writing tests helps to ensure the code is high quality and finds any mistakes or issues.

▼ What is unittest in Python?

The Python unittest is a built-in module for writing and running tests. It helps check if your code works as expected by writing test cases.

▼ What is the use of a Data Provider in TestNG?

A Data Provider in TestNG allows a test method to run multiple times with different sets of data. This helps in testing various scenarios without duplicating code.

▼ What is Pytest?

Pytest is a popular testing framework in Python. It simplifies writing test cases and has many useful plugins for extending its functionality.

## Python in Real-World Projects

This section focuses on applying Python skills in practical scenarios.

▼ How do you use Python for web scraping?

We can use libraries like BeautifulSoup or Scrapy to extract data from websites. Python is often used to gather information from the web and store it for analysis.

▼ How is Python used in machine learning?

Python is popular for machine learning because of libraries like TensorFlow, Keras, and Scikit-learn. These libraries provide pre-built tools to build and train machine learning models.

▼ How does Python integrate with cloud platforms?

Python can be used to automate tasks on cloud platforms like AWS or Google Cloud using their SDKs. You can write Python scripts to manage cloud resources, deploy applications, or handle data.

## Security in Python

Security is an important topic, especially for developers working on larger systems.

▼ How do you secure a web application in Python?

For securing web apps, you can use Django's built-in security features like CSRF protection, SQL injection prevention, and HTTPS support.

▼ What is input validation in Python?

Input validation ensures that only correct data is passed into your program. You can check data types, formats, and ranges to prevent harmful inputs.

▼ How do you optimize the performance of a Python program?

Some techniques include using list comprehensions, reducing function calls, using Cython to compile Python code into C, and utilizing built-in libraries like multiprocessing.

▼ how does GIL affect multithreading?

The Global Interpreter Lock (GIL) stops more than one native thread from running Python code at the same time. This makes multithreading less effective because even if there are multiple threads, only one can run Python bytecode at a time. However, there are ways to get around this problem. You can use the multiprocessing module, which allows you to create separate processes that can run in parallel.

**Before You Leave**

This guide covers a wide range of topics related to Python, focusing on questions that experienced developers may face. You now have a deeper understanding of Python's core concepts, libraries, and real-world applications. For more details, download the "Python interview questions for experienced PDF" and use it for your next interview preparation.

Finally, to keep our site free, we need your support. If you found this tutorial helpful, please share it on Linkedin or YouTube.

**Happy Learning,**
**TechBeamers.**