# Top Python Interview Questions & Answers

## What is Python, and what are some of its key features?

Python is a high-level, interpreted, general-purpose programming language. Key features: dynamic typing, garbage collection, large standard library, readability, and support for multiple paradigms (OOP, functional, procedural).

## What are Python lists and tuples, and how do they differ?

Lists are mutable, ordered collections, defined with square brackets []. Tuples are immutable, ordered collections, defined with parentheses ().

## What is \_\_init\_\_() in Python and what is its purpose?

\_\_init\_\_ is a constructor method in Python classes. It initializes object attributes when a new instance is created.

## What is the difference between mutable and immutable data types in Python?

Mutable types (list, dict, set) can be modified after creation. Immutable types (int, str, tuple) cannot be changed once created.

## Can you explain list, dictionary, and tuple comprehension with examples?

Comprehensions provide a concise way to create collections. Example: [x\*x for x in range(5)] for a list of squares.

## How would you remove duplicates from a sorted array in Python while maintaining order?

Use a loop or collections.OrderedDict to filter duplicates while maintaining order. Example: list(dict.fromkeys(arr)).

## Given a list of positive integers from 1 to n with one number missing, how do you find the missing number?

Use the formula n(n+1)/2 - sum(arr) to find the missing value.

## Write a Python function to determine if a given string is a palindrome.

Check equality of the string with its reverse after normalizing case and removing non-alphanumeric characters.

## Given a list of stock prices, how would you determine the best buy-sell pair to maximize profit?

Iterate through prices, track the minimum price seen so far, and calculate maximum profit using current price - min\_price.

## Define and differentiate between a lambda function, an iterator, and a generator in Python.

Lambda: anonymous function defined using 'lambda' keyword.  
Iterator: an object implementing \_\_iter\_\_() and \_\_next\_\_() to iterate over elements.  
Generator: a special iterator created using a function with 'yield' or generator expressions, producing values lazily.

## Why would you use an else clause in a try/except construct?

The else clause runs if no exception occurs in the try block. It separates normal execution from exception handling.

## What is the difference between deep copy and shallow copy?

Shallow copy: creates a new object but references the same nested objects.  
Deep copy: creates a new object and recursively copies all nested objects.

## Explain Python’s Global Interpreter Lock (GIL).

GIL is a mutex that allows only one thread to execute Python bytecode at a time. It simplifies memory management but limits multi-threaded CPU-bound performance.

## What are Python’s context managers and how are they implemented?

Context managers handle resource setup and cleanup using \_\_enter\_\_() and \_\_exit\_\_() methods or 'with' statement.

## What is monkey patching in Python, and when would you use it?

Monkey patching: dynamically modifying a class or module at runtime.  
Use cases: fixing bugs in third-party libraries, patching for testing purposes.

## Explain Python’s decorators with an example.

Decorators are functions that take another function and extend its behavior without explicitly modifying it.  
Example: @timer to measure execution time of a function.

## What are Python’s coroutines and how do they differ from threads?

Coroutines are functions that can pause and resume their execution using 'async' and 'await'.  
Unlike threads, coroutines are cooperative and run in a single thread but can handle concurrency efficiently.

## Explain Python’s generators and the yield statement.

Generators allow lazy evaluation of sequences. 'yield' returns a value and pauses the function, resuming where it left off on next() call.

## What is Python’s multiple inheritance and how does Method Resolution Order (MRO) work?

Python supports multiple inheritance. MRO determines the order in which base classes are searched when executing a method, using C3 linearization algorithm.