What are the key features that make Python a popular programming language?

- Python is popular for its simple syntax, large standard library, cross-platform support, strong community, and support for multiple programming paradigms.

Explain the difference between mutable and immutable data types in Python.

- Mutable data types (like lists) can be changed after creation, while immutable data types (like tuples) cannot.

What is the purpose of Python's `__init__` method in classes?

- `__init__` is a constructor method that initializes object attributes when an instance of a class is created.

How does Python handle memory management and garbage collection?

- Python uses a private heap space and an automatic garbage collector to manage memory and clean up unused objects.

What is the difference between a shallow copy and a deep copy in Python?

- A shallow copy copies the references to objects, while a deep copy creates independent copies of objects, including nested ones.

Explain the difference between a list and a tuple in Python.

- Lists are mutable and defined with `[]`, while tuples are immutable and defined with `()`.

What are Python decorators, and when would you use them?

- Decorators are functions that modify other functions, used to add functionality like logging or access control.

What is a lambda function, and how is it different from a regular function in Python?

- A lambda function is a one-line, anonymous function defined with `lambda`. It's used for small operations without needing a full function definition.

How does exception handling work in Python? Describe the `try`, `except`, `else`, and `finally` blocks.

- `try` handles exceptions, `except` catches errors, `else` runs if no exceptions, and `finally` always executes at the end.

What is a generator, and how does it differ from a regular function in Python?

- A generator uses `yield` to return values one at a time, making it memory efficient compared to regular functions.

What are list comprehensions, and what advantages do they offer?

- List comprehensions provide a concise way to create lists, making code shorter and more readable.

Explain the purpose of `self` in Python class methods.

- `self` represents the instance of a class, allowing access to its attributes and methods within the class.

What is the difference between `==` and `is` in Python?

- `==` checks value equality, while `is` checks if two variables point to the same object in memory.

How does Python's `with` statement (context manager) work, and why is it useful?

- `with` handles resources automatically (like files), ensuring they're cleaned up (closed) after use.

What are `@staticmethod` and `@classmethod` in Python? How do they differ?

- `@staticmethod` doesn't take `self` or `cls`, while `@classmethod` takes `cls`, allowing access to class-level data.

What is the purpose of the `__name__ == "__main__"` statement in Python scripts?

- It allows code to run only when the script is executed directly, not when imported as a module.

Describe the Global Interpreter Lock (GIL) and its impact on Python's multi-threading.

- The GIL restricts Python to execute only one thread at a time, limiting performance in CPU-bound multi-threaded programs.

What is the difference between `*args` and `**kwargs` in function definitions?

- `*args` passes variable arguments as a tuple, while `**kwargs` passes keyword arguments as a dictionary.

Explain the concept of namespaces and scope in Python.

- A namespace is a container for variable names, while scope is the region in which a variable can be accessed.

What are modules and packages in Python? How do they differ?

- A module is a single Python file, while a package is a directory with multiple modules and an `__init__.py` file.