Q1. What is the purpose of the try statement?

The purpose of the try statement in Python is to define a block of code where exceptions might occur. It allows you to handle potential exceptions and control the flow of the program in response to those exceptions.

Q2. What are the two most popular try statement variations?

1. try-except: This variation allows you to handle specific exceptions that may occur within the try block. You can specify one or more except blocks following the try block to catch and handle different types of exceptions
2. try-finally: This variation allows you to define a finally block that will be executed regardless of whether an exception occurs or not. The code within the finally block is executed at the termination of the try block, whether an exception is raised or not

Q3. What is the purpose of the raise statement?

The raise statement in Python is used to manually raise exceptions in a program. Its purpose is to interrupt the normal flow of execution and indicate that an exceptional condition or error has occurred.

The raise statement in Python is used to manually raise exceptions in a program. Its purpose is to interrupt the normal flow of execution and indicate that an exceptional condition or error has occurred

Q4. What does the assert statement do, and what other statement is it like?

The assert statement in Python is used as a debugging aid to check if a given condition is true. It is typically used to verify assumptions or conditions that should always be true during the execution of the program. If the condition evaluates to False, the assert statement raises an AssertionError exception.

Q5. What is the purpose of the with/as argument, and what other statement is it like?

The with/as statement in Python is used to create a context manager. Its purpose is to simplify the management of resources, such as files or network connections, by automatically taking care of their setup and cleanup operations.

The with/as statement is often referred to as the "context manager" statement. It ensures that certain actions are performed before and after a block of code is executed. It guarantees that the resources associated with the context manager are properly managed and cleaned up, even if exceptions occur during the execution of the block.