Q1. What is the purpose of the try statement?

The try statement allows you to define a block of code to be tested for errors while it is being executed.

The catch statement allows you to define a block of code to be executed, if an error occurs in the try block.

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

The “try” and “else” statements are used in Python for flow control.

An Exception, also known as run time error, is an event which occurs during the execution of a Python program. When that error occurs, it interrupts the Python program and generates an error message by printing a “Traceback” to the console with information about the exception and how it was raised.

Q3. What is the purpose of the raise statement?

The raise keyword is used to raise an exception.

You can define what kind of error to raise, and the text to print to the user.

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

In Python, the assert statement is a powerful tool that helps with debugging and handling errors during development. It allows you to make assumptions about the code and catch potential issues early on.

The assert statements can be useful when testing functions or methods, as it allows you to ensure that the function is behaving as expected. Additionally, assert statements can be used to enforce certain conditions on the inputs or outputs of a function, such as ensuring that a parameter is within a certain range or that a return value meets certain criteria.

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

In Python, the with statement replaces a try-catch block with a concise shorthand. More importantly, it ensures closing resources right after processing them. A common example of using the with statement is reading or writing to a file. A function or class that supports the with statement is known as a context manager.