Q1. Describe three applications for exception processing.

Answer :

Exception processing is a fundamental aspect of error handling in programming languages.

1. Error Handling: Exception processing is primarily used for error handling. When unexpected or exceptional situations occur during the execution of a program, exceptions are raised to indicate that something went wrong. By catching and handling these exceptions, you can gracefully recover from errors, display meaningful error messages, and take appropriate actions to handle the exceptional situation.

2. Input Validation: Exception processing is often used for input validation to ensure that user input meets certain criteria or constraints. When input fails to meet the required conditions, an exception can be raised to indicate an invalid input.

3. Resource Management: Exception processing is essential for managing resources, such as files, network connections, database connections, or system handles. When acquiring or releasing resources, exceptions can be raised to handle exceptional cases or ensure proper cleanup.

Q2. What happens if you don't do something extra to treat an exception?

Answer : If you don't handle an exception, either by catching it or allowing it to propagate, it will result in an unhandled exception. When an unhandled exception occurs, the normal flow of the program is interrupted, and the exception propagates up the call stack until it is caught and handled, or if it reaches the top-level of the program, it will cause the program to terminate.

Q3. What are your options for recovering from an exception in your script?

Answer : Catch and Handle the Exception: You can use a try-except block to catch the exception and handle it within your script. By enclosing the code that may raise an exception within a try block, you can specify one or more except blocks to catch specific types of exceptions or a more general exception type. Inside the except block, you can implement error handling logic, such as displaying an error message, logging the exception, retrying the operation, or taking any necessary corrective action.

Q4. Describe two methods for triggering exceptions in your script.

Answer :

1. Using the assert Statement:

The assert statement is primarily used for debugging and checking the correctness of assumptions in your code. It raises an AssertionError exception if the given condition is false. You can use it to explicitly check for conditions that should always be true and raise an exception if they are not.

2. Using the raise Statement with a Built-in Exception:

You can use the raise statement to explicitly raise a specific built-in exception. This allows you to trigger exceptions based on specific conditions or error scenarios in your code.

Q5. Identify two methods for specifying actions to be executed at termination time, regardless of whether or not an exception exists.

Answer : The finally block is a construct that follows the try-except block. The code within the finally block is executed regardless of whether an exception occurred or not. It provides a way to define cleanup code that should always be executed, ensuring resources are properly released, connections are closed, or any necessary finalization steps are performed.