Q1. Describe three applications for exception processing.

Applications for Exception Processing:

**Error Handling**: Exception processing is commonly used for handling errors that may occur during the execution of a program. This helps in gracefully responding to unexpected situations, preventing program crashes, and providing meaningful error messages.

**Input Validation**: Exceptions can be used to validate user input. For example, catching a ValueError when converting user input to an integer allows the program to handle invalid input without crashing.

**Resource Management**: Exception processing is essential for proper resource management. It allows the release of resources (such as file handles or network connections) in the event of an exception, ensuring that resources are not left open.

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

If an exception is not explicitly handled, the program will terminate, and an error message will be displayed. This can make it challenging to identify and fix issues.

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

**Try-Except Blocks**: Using try-except blocks allows the program to catch and handle exceptions. Recovery actions can be specified in the except block to perform alternative operations or provide fallback behaviour.

try:

# Code that may raise an exception

except SomeException as e:

# Handle the exception and perform recovery actions

**Finally Block:** The finally block is executed whether an exception occurs or not. It is often used for cleanup operations, ensuring that certain actions are taken regardless of whether an exception is raised.

try:

# Code that may raise an exception

except SomeException as e:

# Handle the exception

finally:

# Code in this block will be executed regardless of exceptions

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

**Raise Statement:** The raise statement is used to manually trigger exceptions at specific points in the code. This can be useful for signalling errors or exceptional conditions.

**Assertion Error**: The assert statement is another way to trigger an exception based on a condition. It is commonly used for debugging and validating assumptions.

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

**Atexit Module**: The atexit module provides a way to register functions that will be called when a program is about to exit. This allows for actions to be performed regardless of whether an exception occurred.

**Context Managers (with Statement)**: Context managers can be used with the with statement to ensure certain actions are taken before exiting a block of code. This is commonly used for resource management.