**Q1. Describe three applications for exception processing.**

**Answer 1:** Three applications for exception processing include error handling, graceful program termination, and resource cleanup.

1. **Error Handling:** Exception processing is commonly used for handling errors in a program. When unexpected situations or errors occur during execution, exceptions provide a mechanism to gracefully capture and respond to these issues.
2. **Graceful Program Termination:** Exceptions are useful for ensuring that a program terminates gracefully in the face of unforeseen issues.
3. **Resource Cleanup:** Exception handling is crucial for releasing resources in a clean way, especially in scenarios where resources like files, network connections, or database connections are involved.

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

**Answer 2:** If you don't handle an exception, it will propagate up the call stack, potentially leading to program termination and providing an error message.

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

**Answer 3:** Options for recovering from an exception include using try-except blocks to catch and handle specific exceptions, logging the exception information, and providing alternative paths in the code.

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

**Answer 4:**

1. *Using raise Statement:* The raise statement is used to manually trigger exceptions in a script. You can raise built-in exceptions or create custom ones. For example:

raise ValueError("This is a custom exception")

This allows you to signal that a particular condition has occurred, and the normal flow of the program should be interrupted.

2. *Invoking Functions or Methods:* Certain functions or methods may be designed to raise exceptions under specific conditions. For instance, attempting to open a non-existent file with the open() function can raise a FileNotFoundError:

try:

file = open("nonexistent\_file.txt", "r")

except FileNotFoundError as e:

print(f"File not found: {e}")

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

**Answer 5:** Two methods for specifying actions at termination time are using the finally block in a try-except-finally structure and the atexit module, which allows you to register functions to be executed when the script exits, whether an exception occurs.