Q1. Describe three applications for exception processing.

ANS:

* SyntaxError: This exception is raised when the interpreter encounters a syntax error in the code, such as a misspelled keyword, a missing colon, or an unbalanced parenthesis.
* TypeError: This exception is raised when an operation or function is applied to an object of the wrong type, such as adding a string to an integer.
* NameError: This exception is raised when a variable or function name is not found in the current scope.

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

ANS:

if we don't handle exceptions

When an exception occurred, if we don't handle it, the program terminates abruptly and the code past the line that caused the exception will not get executed.

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

ANS:

In Python, exceptions can be handled by two new methods: Try: Catches exceptions raised by Python or a program. Raise: A custom exception that triggers an exception manually

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

ANS:

To avoid such a scenario, there are two methods to handle Python exceptions: Try – This method catches the exceptions raised by the program. Raise – Triggers an exception manually using custom exceptions.

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

ANS:

An exception is an error which happens at the time of execution of a program. However, while running a program, Python generates an exception that should be handled to avoid your program to crash. In Python language, exceptions trigger automatically on errors, or they can be triggered and intercepted by your code.