Q1. Describe three applications for exception processing.

**Ans:**  There are three types of errors; syntax errors, logic errors and exceptions.

Q2. What happens if you do not do something extra to treat an exception?

**Ans:** An exception may cause the program to stop if it is not properly "caught" (i.e., handled correctly). If you think that your program might raise an exception when executed, you will find it useful to use try/except to handle them.

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

**Ans:** Put try/except structure more in-wards. Otherwise, when you get an error, it will break all the loops. Perhaps after the first for-loop, add the try/except. Then if an error is raised, it will continue with the next file.

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

**Ans**: To avoid such a scenario, there are two methods to handle Python exceptions:

1. Try – This method catches the exceptions raised by the program.
2. 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:** A finally block contains all the crucial statements that must be executed whether exception occurs or not. The statements present in this block will always execute regardless of whether exception occurs in try block or not such as closing a connection, stream etc.