Q1. What are the two latest user-defined exception constraints in Python 3.X?

Sol: 1. All user-defined exceptions should be derived from the base class “Exception”.

2. All user-defined exception classes should define the “str” method to provide a human-readable error message when the exception is raised.

Q2. How are class-based exceptions that have been raised matched to handlers?

Sol In Python when a class-based exception is raised it is matched to the appropriate handler by checking the exception hierarchy.

Q3. Describe two methods for attaching context information to exception artefacts.

Sol: “raise…from…” syntax: The “from” keyword allows you to attach a new exception to an existing one which creates a chain of exceptions that provides additional context.

“Contextvars” module: This will allows you to attach context information to exceptions, tracebacks, and log records using context variables.

Q4. Describe two methods for specifying the text of an exception object's error message.

Sol: passing a string argument to the exception class constructor: when you create a new exception object you can pass a string argument to the constructor to specify the error message.

Implementing the “str” method: All exception classes in python inherit from the BaseException calss which defines a str method that returns a string representation of the exception.

Q5. Why do you no longer use string-based exceptions?

Sol: String based exceptions are not vary informative.

String based exceptions are not type-safe.

String based exceptions are not internationalization-friendly.