Q1. What are the two latest user-defined exception constraints in Python 3.X?
Answer: raise and assert statement are two user defined exceptions constraints in Python 3.x

Q2. How are class-based exceptions that have been raised matched to handlers?
Answer: In Python, users can define custom exceptions by creating a new class. This exception class has to be derived, either directly or indirectly, from the built-in Exception class. This new exception class, like other exceptions, can be raised using the raise statement with an optional error message.

Q3. Describe two methods for attaching context information to exception artefacts.
Answer: The **process()** method of **LoggerAdapter** is where the contextual information is added to the logging output. It's passed the message and keyword arguments of the logging call, and it passes back (potentially) modified versions of these to use in the call to the underlying logger.
Other method that can be used is exception(),Logs a message with level ERROR on this logger. The arguments are interpreted as for debug(). Exception info is added to the logging message.

Q4. Describe two methods for specifying the text of an exception object's error message.
Answer: raise statement is used to trigger exception, if certain condition is not as per condition requirement of programmer. It helps in triggering exception as per need of programmer and its program logic.
There are few assertions that programmer always want to be true to avoid code failure. This type of requirement is fulfilled by assert statement. This statement takes a Boolean condition output of which if true, further program executes. If output of assert statement is false it raises an Assertion Error.

Q5. Why do you no longer use string-based exceptions?
Answer: String-based exceptions doesn't inherit from exceptions, so plain exceptions catch all exceptions and not only system.