Q1. In Python 3.X, what are the names and functions of string object types?

Ans. str (for Unicode text, including ASCII)

bytes(for binary data with absolute byte values)

bytearray (a mutable flavor of bytes)

Q2. How do the string forms in Python 3.X vary in terms of operations?

Ans. In 3.X, Unicode text is handled by str, byte-based data is handled by bytes

Q3. In 3.X, how do you put non-ASCII Unicode characters in a string?

Ans Non-ASCII Unicode characters can be coded in a string with both hex (\xNN) and Unicode (\uNNNN, \UNNNNNNNN) escapes.

Q4. In Python 3.X, what are the key differences between text-mode and binary-mode files?

Ans In 3.X,text-mode files assume their file content is Unicode text (even if it’s all ASCII) and automatically decode when reading and encode when writing.

With binary-mode files, bytes are transferred to and from the file unchanged

Q5. How can you interpret a Unicode text file containing text encoded in a different encoding than your platform's default?

Ans to read files encoded in a different encoding than the default for your platform,

# pass the name of the file’s encoding to the open function. This will decode the contents of the file read

Q6. What is the best way to make a Unicode text file in a particular encoding format?

Ans To create a Unicode text file in a specific encoding format, pass the desired encoding

# name to open function; strings will be encoded as per encoding when contents are written to the file.

Q7. What qualifies ASCII text as a form of Unicode text?

Ans ASCII is subset of the Unicode text, because its 7-bit range of values is a subset of most Unicode encodings

Q8. How much of an effect does the change in string types in Python 3.X have on your code?

Ans.