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

In Python, Strings are **arrays of bytes representing Unicode characters**. A string is a collection of one or more characters put in a single quote, double-quote or triple quote. In python there is no character data type, a character is a string of length one. It is represented by str class.

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

String is an array of bytes that represent the Unicode characters in Python. Python does not support character datatype. A single character also works as a string. Python supports writing the string within a single quote('') and a double quote(""). Use the join() and concatenate() operations etc.

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

1. The source file must be saved using the correct encoding in your text editor as well.
2. In **Python** 2, the unicode literal must have a u before it, as in s. replace(u"Â ", u"") But in **Python** 3, just use quotes. ...
3. s. replace(u"Â ", u"") will also fail if s is **not** a unicode string.
4. string.

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

* When a file is opened in ***text mode***, reading its data automatically decodes its content (per a platform default or a provided encoding), and returns it as a str; writing takes a str, and automatically encodes it before transferring to the file. Text mode files also support universal end-of-line translation, and encoding specification arguments.
* When a file is opened in ***binary mode*** by adding a "b" to the mode string argument in the open() call, reading its data does not decode it in any way, and simply returns its content raw and unchanged, as a bytes object; writing takes a bytes object and transfers it to the file unchanged. Binary-mode files also accept a bytearray object for the content to be written to the file.

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

Read text file in python2, you should decode line to get unicode string, encode line before writing to a file. In python3, open function supports encoding keyword parameter, decoding/encoding can happen under the hood automatically. You can just work with unicode string.

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

write() to write unicode text to a text file. Call str. encode(encoding) with encoding set to "utf8" to encode str . Call open(file, mode) to open a file with mode set to "wb" .

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

ASCII has its equivalent in Unicode. The difference between ASCII and Unicode is that **ASCII represents lowercase letters (a-z), uppercase letters (A-Z), digits (0–9) and symbols such as punctuation marks** while Unicode represents letters of English, Arabic, Greek etc.

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

Its return the Typeerror to change in string types in python3.X