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

Ans: if the string class object is ‘str’ then a few of the method examples are below:

Str.upper()

Str.lower()

Str.title()

Str.capitalise()

Str.casefold()

Str.center()

Str.count()

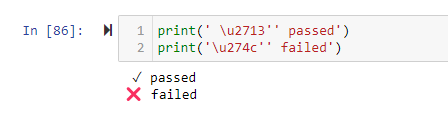
Q2. How do the string forms in Python 3.X vary in terms of operations?

Ans: **below is the example how different operators used on strings results in different results:**



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

Ans: **we can use Unicode escape character in the form of \u0123 in your string. For example:**



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

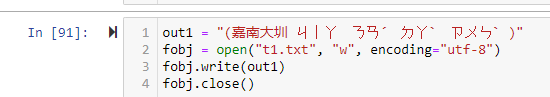
Ans: **Text file contains human readable characters and binary mode files contain non-human readable characters and symbols.**

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

Ans: **you have to use decode method off a string to decode the text.**

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

Ans: **below is an example of create a unicode text file with an encoding format:**



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

Ans:  **The first 128 unicode characters point to ASCII characters. ASCII is a subset of Unicode and ASCII is just UTF-8.**

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

Ans: **Python by default uses Unicode and if we change it to ASCII then it takes less memory to store Roman alphabet character but in totality it doesn’t have much of impact.**