Q1. Explain the difference between greedy and non-greedy syntax with visual terms in as few words as possible. What is the bare minimum effort required to transform a greedy pattern into a non-greedy one? What characters or characters can you introduce or change?

The greedy match tries to match maximum number of possible repetitions of the quantified pattern. The non-greedy match matches minimum number of possible repetitions of the quantified pattern.

**import** re

print(re**.**findall('a?', 'aaaa'))

print(re**.**findall('a??', 'aaaa'))

['a', 'a', 'a', 'a', '']

['', 'a', '', 'a', '', 'a', '', 'a', '']

In the first instance, we use the zero-or-one version 'a?'. It’s greedy so it matches longest possible string,i.e., one 'a' character if possible.

In the second instance, you use the non-greedy zero-or-one version 'a??'. Sp it matches zero 'a's (shortest possible string).

Changing ? to ?? can change greedy to non greedy regex. Similarly, other small changes can be made.

Q2. When exactly does greedy versus non-greedy make a difference?  What if you're looking for a non-greedy match but the only one available is greedy?

Ans.We can use filtering or pattern matching methods of regex if non-greedy is not available.

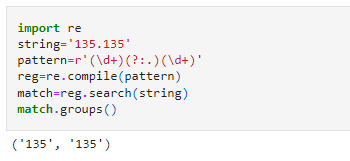
Q3. In a simple match of a string, which looks only for one match and does not do any replacement, is the use of a nontagged group likely to make any practical difference?

Ans

. 

Q4. Describe a scenario in which using a nontagged category would have a significant impact on the program's outcomes.

Ans. The non tagged group will not make any diffrence in this case.



Here the decimal "." is not tagged or captured. It will useful in scenarios where the separator of value in a string is of no use and we need to capture only the values.

Q5. Unlike a normal regex pattern, a look-ahead condition does not consume the characters it examines. Describe a situation in which this could make a difference in the results of your programme.

Ans. In cases where we need the count of the number of multiple lines or sentences in a string, the positive look ahead makes a difference over normal regex pattern, by giving us the correct count.

Q6. In standard expressions, what is the difference between positive look-ahead and negative look-ahead?

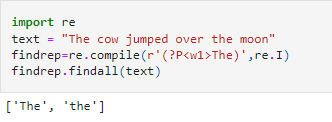
Ans. Positive look ahead is an assertion continuing the search and extending the string. e.g.pattern= r'abc(?=[A-Z])''.Here after 'abc', ? is extending the search and says that in the remaining string, first identify the next charater should be capitalized character between A and Z, but doesnt consume it.

Negative look head is also an assertion to exclude certain patterns. e.g. pattern = r'abc(?!abc)', means identify a substring containing'abc' which is not followed by another 'abc'

Q7. What is the benefit of referring to groups by name rather than by number in a standard expression?

Ans. The first benefit is the code structure is clear. Secondly , it is easier to maintain the code.

Q8. Can you identify repeated items within a target string using named groups, as in "The cow jumped over the moon"?

Ans. 

Q9. When parsing a string, what is at least one thing that the Scanner interface does for you that the re.findall feature does not?

Ans.

Q10. Does a scanner object have to be named scanner?

Ans. No it doesn't. It can be assigned any name.