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?

Ans=>

Greediness refers to the quantity of times the regex engine will try to match certain set of characters. The way to state the "greediness" of a regex expression is using the special characters

str = "asdfasdfbbbb"

r1 = /b/

r2 = /(asdf)\*/

r3 = /b{3}/

r4 = /.\*/

In greedy approach regex pattern tend to consume maximum characters in a source string.

textstr = "bcabdcab"

textstr.gsub!(/(.\*)ab/, "xxx")

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=> A non-greedy match means that the regex engine matches as few characters as possible—so that it still can match the pattern in the given string. For example, the regex 'a+?' will match as few 'a' s as possible in your string 'aaaa' . Thus, it matches the first character 'a' and is done with it.

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=>no it does not

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

Ans=>

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=>it will make difference in results of program when we don’t want that characters to check in string.

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

Ans=> The engine notes that the regex inside the lookahead failed. Because the lookahead is negative, this means that the lookahead has successfully matched at the current position. At this point, the entire regex has matched

Lookahead is used as an assertion in Python regular expressions to determine success or failure whether the pattern is ahead i.e to the right of the parser’s current position. They don’t match anything

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

Ans=>it will look for whole string inside name.

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

Ans=>import regex

s = "foo bar"

rx = regex.compile(r"(?P<n>foo)|(?P<n>bar)")

print([x.group("n") for x in rx.finditer(s)])

// => ['foo', 'bar']

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=> findall Return all non-overlapping matches of pattern in string, as a list of strings. The string is scanned left-to-right, and matches are returned in the order found.

with scanner we can use multiple expressions

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

Ans=>no