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: non-greedy matching, also known as lazy matching, tries to match as little text as possible.

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 : Greedy matching can return unexpected results when a regular expression contains repeating or optional elements.

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 : match()

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

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.

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

Ans: Positive lookahead: (?= «pattern») matches if pattern matches what comes after the current location in the input string. Negative lookahead: (?! «pattern») matches if pattern does not match what comes after the current location in the input string.

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

Ans: saves future developers' time and effort

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

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?

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