1. 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: Greedy match tries to match as many repetitions of the quantified pattern as possible whereas non greedy match tries to match as few repetitions of the quantified pattern as possible.
re.findall("a*", "aaa")) # Greedy Match Synatx
re.findall("a*?", "aaa")) # Non Greddy Syntax
Introducing *? Converts a greedy pattern to a non-greedy one.

2. 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 match tries to match as many repetitions of the quantified pattern as possible whereas non greedy match tries to match as few repetitions of the quantified pattern as possible. If only a non-greedy match is available, we can use other filtering or pattern matching methods of regex to identify the required pattern.

3. 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 doesn't make a difference in this case.

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

Ans: In case there are many tagged expressions, or in case we want to maximize the readability of the program, it's handy to be able to refer to variables by name and use nontagged category.

5. 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: While counting the number of multiple lines or multiple sentences in a string the positive look ahead makes a difference, without which we won't get the correct count of lines or sentences in a string.

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

Ans: Positive lookahead allows to add a condition for what follows. Negative lookahead allows to match a pattern only if there’s something before it.

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

Ans: Referring to groups by name rather than by number in a standard expression helps to keep the code clear and easy to understand.

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

Ans: string = "The cow jumped over the moon"
```
regobj=re.compile(r'(?P<w1>The)',re.I)
regobj.findall(string)
```

9. 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() module searches for “all” occurrences that match a given pattern while search() module will only return the first occurrence that matches the specified pattern. findall() will iterate over all the lines of the file and will return all non-overlapping matches of pattern in a single step

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

Ans: A scanner object can be named any name.