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?

Greedy means that it tries to match as many characters to the syntax as possible and non-greedy will try to match as few instances as possible in the string/ sentence.

Just place a question mark after \* i.e. after the pattern

We will introduce the character question mark (?) or double question mark (??)

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?

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?

We can just find the location of the patterns in which we are interested in but we cannot change them.

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

We can find the non-desired patterns in the lines and use them to convert to the other pattern

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.

Lookahead condition just finds the finds the indexes of the characters that matches the defined pattern. These do not return any value of match or not matched.

This could help a programmer to find the positions of the desired patterns

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

Positive lookahead means that it will first match a condition but only if the certain match is find while negative lookahead means that it will find a match only if the certain match is not present in the match found.

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

Benefit of referring to groups by name is that names can be remembered by the people and python also has a standard of naming starting with name rather than numbers

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

Yes, we can identify the repeated items in strings using named groups

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?

The scanner object finds the match in full string while the re.findall matches only if a string starts with the match.

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

No we can name it with any other name we like