1. What is the name of the feature responsible for generating Regex objects?

- re.compile()
- It takes a raw string of some pattern with or without quantifiers

2. Why do raw strings often appear in Regex objects?

- Backslashes are used to define the pattern, if these backslashes are passed inside a normal string, it considers them to be escape characters. So, raw strings are used for regex objects.

3. What is the return value of the search() method?

- It returns the match object with span and matching value for the first appearance of the given pattern in the given text.

4. From a Match item, how do you get the actual strings that match the pattern?

- By using .group() method to the match object.

5. In the regex which created from the r'(\d\d\d)-(\d\d\d-\d\d\d\d)', what does group zero cover? Group 2? Group 1?

- None of them
- The ordering for groups start from 1
- And, .group(0) returns the entire match just like .group()

6. In standard expression syntax, parentheses and intervals have distinct meanings. How can you tell a regex that you want it to fit real parentheses and periods?

- By using a backslash before period or parenthesis in our pattern. Or
- The pattern for non-alphanumeric characters **\W**.
- It can also be used in patterns for parentheses and periods.

7. The findall() method returns a string list or a list of string tuples. What causes it to return one of the two options?

- Parenthesis are used for grouping parts of the pattern to extract them individually. If they are used .findall() returns a list of string tuples. If not, it just returns a string list.

8. In standard expressions, what does the | character mean?

- It's like an **or** operator between two patterns.

9. In regular expressions, what does the character stand for?

- << that character is not mentioned in the question. QUESTION IS NOT COMPLETED. Considering ‘a character’ instead of ‘the character’ to answer the question.>>
- Characters such as a digit or a single string have different codes that represent them.
- These can be used to build up a pattern string.
- This requires usage of backslashes thus we use raw strings.
- The general syntax is r’pattern’
- This pattern contains different characters, that are used to identify different characters in the given text like digits, alphanumerics, white spaces, non-digits, non-alphanumeric values, non-whitespaces etc.

10.In regular expressions, what is the difference between the + and * characters?

- + : occurs one or more times
- * : occurs zero or more times

11. What is the difference between {4} and {4,5} in regular expression?

- {4} : occurs exactly four times
- {4, 5}: occurs 4 to 5 times i.e. occurs 4 or 5 times in this case.

12. What do you mean by the \d, \w, and \s shorthand character classes signify in regular expressions?

- \d: A digit
- \w: An alphanumeric
- \s: A white space

13. What do means by \D, \W, and \S shorthand character classes signify in regular expressions?

- \D: A non digit
- \W: A non-alphanumeric
- \S: Non-whitespace

14. What is the difference between .*? and .*?

- << QUESTION IS NOT PROPERLY FRAMED. There is no difference between .*? and .*? They both are dot-star-questionmark. But, to answer the question, I am considering the last question mark is for the entire question and not to find the difference with the object inside it. >>
- .*?: This is a non-greed match or reluctant match. Because, by until * it matches all the matching characters, then with this question mark it will condense it down to only zero or one match.
- .*: This is a greedy match. Because, they try to match as many reps as possible, and when this doesn’t work and they have to backtrack, they try to match one or fewer rep at a time, until a match of the whole pattern is found. As a result, when a match finally happens, a greedy repetition would match as many reps as possible.

15. What is the syntax for matching both numbers and lowercase letters with a character class?

- [0-9a-z]

16. What is the procedure for making a normal expression in regax case insensitive?

- By passing re.IGNORECASE along with the pattern in re.compile
- re.compile(pattern, re.IGNORECASE)

17. What does the . character normally match? What does it match if re.DOTALL is passed as 2nd argument in re.compile()?

- . character normally matches any character except \n
- If re.DOTALL is passed as a 2nd argument in re.compile, it will consider \n also for matching.

18. If numReg = re.compile(r'\d+'), what will numRegex.sub('X', '11 drummers, 10 pipers, five rings, 4 hen') return?

- It will place X in place of all the numbers matching the given pattern in the given the text
- It will return: “X drummers, X pipers, five rings, X hen”

19. What does passing re.VERBOSE as the 2nd argument to re.compile() allow to do?

- It allows to add whitespace and comments to the string passed to re.

20. How would you write a regex that match a number with comma for every three digits? It must match the given following:

'42'

'1,234'

'6,368,745'

but not the following:

'12,34,567' (which has only two digits between the commas)

'1234' (which lacks commas)

- re.compile(r'^\d{1,3}(?:,\d{3})*$')

21. How would you write a regex that matches the full name of someone whose last name is Watanabe? You can assume that the first name that comes before it will always be one word that begins with a capital letter. The regex must match the following:

'Haruto Watanabe'

'Alice Watanabe'

'RoboCop Watanabe'

but not the following:

'haruto Watanabe' (where the first name is not capitalized)

'Mr. Watanabe' (where the preceding word has a nonletter character)

'Watanabe' (which has no first name)

'Haruto watanabe' (where Watanabe is not capitalized)

- pattern = re.compile(r'^[A-Z][A-Za-z]*\sWatanabe')

22. How would you write a regex that matches a sentence where the first word is either Alice, Bob, or Carol; the second word is either eats, pets, or throws; the third word is apples, cats, or baseballs; and the sentence ends with a period? This regex should be case-insensitive. It must match the following:

'Alice eats apples.'

'Bob pets cats.'

'Carol throws baseballs.'

'Alice throws Apples.'

'BOB EATS CATS.'

but not the following:

'RoboCop eats apples.'

'ALICE THROWS FOOTBALLS.'

'Carol eats 7 cats.'

- re.compile(r"^(alice|bob|carol)\s(eats|pets|throws)\s(apples|cats|baseballs)\W$", re.IGNORECASE)