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

**Ans:** The re.compile() function returns the regex objects.

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

**Ans:** Python raw string treats the backslash character(\) as a literal character. Raw string is useful when a string needs to contain a backslash, such as for a regular expression or windows directory path, and you don’t want it to be treated as an escape character.

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

**Ans:** The search() method returns the Match objects.

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

**Ans:** The group() method returns the strings of the match text.

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?

**Ans:** Group 0 cover entire match ‘(\d\d\d)-(\d\d\d-\d\d\d\d)’ , Group 1 cover first set of parentheses '(\d\d\d)’ and Group 2 cover second set of parentheses ‘(\d\d\d-\d\d\d\d)'.

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?

**Ans:** Periods and parentheses can be escaped with a backslash: \. , \( and \).

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

**Ans:** If the regex has no groups then a list of strings is returned, if the regex has groups the list of string tuple is returned.

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

**Ans:** The | character signifies matching “either, or “ between two groups.

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

**Ans:** All characters, except those having special meaning in regex, matches themselves. E.g., the regex x matches substring "x" ; regex 9 matches "9" ; regex = matches "=" ; and regex @ matches "@" .

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

**Ans:** The + matches one or more. The \* matches zero or more.

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

**Ans:** The {4} matches exactly 4 instances of the preceding group. The {4,5} matches the instances between 4 and 5 of the preceding group.

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

**Ans:** The \d,\w,\s shorthand character classes match a single digit, word and space character respectively.

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

**Ans:** The \D,\W,\S shorthand character classes match a single character that is not a digit, word and space character respectively.

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

**Ans:** The . and \* performs the greedy match and the ? performs the non greedy match.

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

**Ans:** Either [a-z0-9] or [0-9a-z]

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

**Ans:** Passing re.I or re.IGNORECASE as the second argument in the re.compile() will make the matching case insensitive.

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

**Ans:** The . character generally matches any character except the new line character. If we pass re.DOTALL as a second argument in re.compile() it will match the new line characters also.

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

**Ans:** X drummers, X pipers, five rings, X hen

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

Ans: The re.VERBOSE argument allows you to add the whitespace and the comments to the string passed to the re.compile().

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)

**Ans:** re.compile(r’^\d{1,3}(,\d{3})\*$’) will create this regex.

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)

**Ans:** re.compile(r’^[A-Z][a-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.'

**Ans:** re.compile(r'^(Alice|Bob|carol)\s(eats|pets|throws)\s(apples|cats|baseballs)(.)$',re.IGNORECASE)