1. The re.compile() function returns Regex objects.

2. Raw strings are used so that backslashes do not have to be escaped

3. The search() method returns Match objects.

4. The group() method returns strings of the matched text.

5. Group 0 is the entire match, group 1 covers the first set of parentheses, and group 2 covers the second set of parentheses.

6. Periods and parentheses can be escaped with a backslash: ., (, and ).

7. If the regex has no groups, a list of strings is returned. If the regex has groups, a list of tuples of strings is returned.

8. The | character signifies matching “either, or” between two groups

9. The ? character can either mean “match zero or one of the preceding group” or be used to signify nongreedy matching.

10. The + matches one or more. The \* matches zero or more.

11. The {4} matches exactly four instances of the preceding group. The {4,5} matches between four and five instances.

12. The \d, \w, and \s shorthand character classes match a single digit, word, or space character, respectively.

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

14. The . performs a greedy match, and the .? performs a nongreedy match.

15. Either [0-9a-z] or [a-z0-9]

16. Passing re.I or re.IGNORECASE as the second argument to re.compile() will make the matching case insensitive.

17. The . character normally matches any character except the newline character. If re.DOTALL is passed as the second argument to re.compile(), then the dot will also match newline characters

18. 'X drummers, X pipers, five rings, X hens'

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

20. '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)

Answer:-re.compile(r'^\d{1,3}(,\d{3})\*$') will create this regex, but other regex strings can produce a similar regular expression.

21. '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)

Answer:- re.compile(r'\s([A-Z]\w+)\s(Watanabe)')

22. '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.'

Answer:- re.compile(r'(Alice|Bob|Carol)\s(eats|pets|throws)\s(apples|cats|baseballs).', re.IGNORECASE)