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

Answer 1: The feature responsible for generating Regex objects is the re.compile() function.

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

Answer 2: Raw strings (prefixed with 'r') are often used in Regex objects to treat backslashes as literal characters. This is helpful in avoiding unintentional escape sequences.

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

Answer 3: The return value of the search() method is a Match object if a match is found, and None if no match is found.

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

Answer 4: To get the actual strings that match the pattern from a Match object, you can use the group() method. For example, if matchObj is the Match object, matchObj.group() returns the entire match.

**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?**

Answer 5: In the regex created from r'(\d\d\d)-(\d\d\d-\d\d\d\d)', group 0 covers the entire match, group 2 covers the second set of parentheses, and group 1 covers the first set of parentheses.

**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?**

Answer 6: To tell a regex that you want it to fit real parentheses and periods, you can use the backslash (‘\’) to escape them. For example, to match a literal period, use ‘\.’.

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

Answer 7: The findall() method returns a list of string tuples if the regex pattern contains groups, otherwise, it returns a list of strings.

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

Answer 8: In standard expressions, the | character means "OR". It allows you to match either the expression on its left or the one on its right.

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

Answer 9: In regular expressions, the . character stands for any character except a newline.

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

Answer 10: In regular expressions, the + character means "one or more occurrences," while the \* character means "zero or more occurrences."

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

Answer 11: {4} in a regular expression means exactly four occurrences, while {4,5} means between four and five occurrences.

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

Answer 12: In regular expressions, \d signifies any digit, \w signifies any word character (alphanumeric + underscore), and \s signifies any whitespace character.

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

Answer 13: \D in regular expressions matches any non-digit, \W matches any non-word character, and \S matches any non-whitespace character.

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

Answer 14: .\*? is a non-greedy match, meaning it matches as little text as possible, while .\* is a greedy match, meaning it matches as much text as possible.

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

Answer 15: The syntax for matching both numbers and lowercase letters with a character class is [0-9a-z] or [a-z0-9].

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

Answer 16: To make a regular expression case-insensitive, you can use the re.IGNORECASE or re.I flag when compiling the regex.

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

Answer 17: The . character normally matches any character except a newline. If re.DOTALL is passed as the second argument in re.compile(), it matches any character, including a newline.

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

Answer 18: numRegex.sub('X', '11 drummers, 10 pipers, five rings, 4 hen') will return the string 'X drummers, X pipers, five rings, X hen'.

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

Answer 19: Passing re.VERBOSE as the second argument to re.compile() allows the use of whitespace and comments within the regular expression for better readability.

**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)**

Answer 20: To match a number with commas for every three digits, you can use the regex 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)**

Answer 21: To match the full name of someone whose last name is Watanabe, you can use the regex r'^[A-Z][a-zA-Z]\* Watanabe$'.

**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.'**

Answer 22: To match a sentence with specific words in a case-insensitive manner, you can use the regex r'^(Alice|Bob|Carol) (eats|pets|throws) (apples|cats|baseballs)\.$'.