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

Answer=

The feature responsible for generating regular expression (Regex) objects in Python is the re module

The function which returns Regex objects is = re.compile()

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

Answer=

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

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

Answer=

It matches objects in the return value of the search() method

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

Answer=

By using The group() method in which get the actual strings that match the pattern of returns strings by the matched 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?

Answer=

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

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?

Answer=

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

Answer=

The | character signifies matching "either, or" between two groups.

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

Answer=

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

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

Answer=

In regular expressions The + matches one or more. And The \* matches zero or more.

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

Answer=

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

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

Answer=

In regular expressions, the \d shorthand character class signifies any digit (0-9), the \w shorthand character class signifies any word character (a-z, A-Z, 0-9, \_), and the \s shorthand character class signifies any whitespace character (space, tab, newline, etc.). These shorthand character classes can be used to match specific types of characters within a string. The \d, \w, and \s shorthand character classes match a single digit, word, or space character, respectively.

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

Answer=

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

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

Answer=

The .performs a greedy match, and the .? performs a non-greedy match.

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

Answer=

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

Answer=

Passing re.I or re.IGNORECASE as the second argument to 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()?

Answer=

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

Answer=

If numReg = re.compile(r'\d+'), then numReg.sub('X', '11 drummers, 10 pipers, five rings, 4 hen') will return the string "X drummers, X pipers, five rings, X hen". The sub() method will replace any sequences of one or more digits in the input string with the character 'X'.

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

Answer=

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

Answer=

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

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=

The regular expression that matches the full name of someone whose last name is Watanabe, assuming that the first name that comes before it will always be one word that begins with a capital letter, would be:

"[A-Z][a-z]+\sWatanabe"

The first part of the regex "[A-Z][a-z]+" will match one or more lower case letters preceded by a capital letter, which will match the first name. The "\s" will match a whitespace character. And the last part of the regex "Watanabe" will match the last name.

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=

The regular expression 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 would be:

"(Alice|Bob|Carol)\s(eats|pets|throws)\s(apples|cats|baseballs).$"

The first part of the regex "(Alice|Bob|Carol)" will match the first word, the second part of the regex "\s(eats|pets|throws)" will match the second word, the third part of the regex "\s(apples|cats|baseballs)" will match the third word, and the last part of the regex ".$" will match the end of the sentence.