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

Ans:- re.compile() is used for generating Regex objects.

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

Ans:- Raw string is used when a string needs to contain a backslash(\).

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

Ans:- search() method returns matched objects in the string.

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

Ans:-group() method returns matched string from a group of string.

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:- In the Regex r'(\d\d\d)-(\d\d\d-\d\d\d\d)' the zero group covers the entire pattern match whereas the first group cover (\d\d\d) and the second group cover (\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:- The \. \( and \) escape characters in the raw string passed to re.compile() will match actual parenthesis characters.

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 pattern has no groups, a list of strings matched is returned. if the regex pattern has groups, a list of tuple of strings is returned.

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

Ans:- In standard expressions , the | character means Or.

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

Ans:- In regular Expressions, ? characters represents zero or one match of the preceding group.

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

Ans:- The + matches one or more, the \* matches zero or more of the preceding group.

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

Ans:- {4} means that its preceding group should repeat 4 times. whereas {4,5} means that its preceding group should repeat minimum 4 times and maximum 5 times.

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

Ans:- \d, \w and \s are special sequences in regular expressions in python:

\d – Matches digit character equivalent to [0-9]

\w – Matches a word character equivalent to [a-zA-Z0-9]

\s – Matches whitespace character (space, tab, newline, etc.)

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

Ans:- \D, \W and \S are special sequences in regular expressions in python:

\D – Matches any non-digit character, this is equivalent to the set class [^0-9]

\W – Matches any non-alphanumeric character equivalent to [^a-zA-Z0-9]

\S – Matches any non-whitespace character

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

Ans:- .\* is a Greedy mode, which returns the longest string that meets the condition. Whereas .\*? is a non greedy mode which returns the shortest string that meets the condition.

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

Ans:- The Syntax for matching both numbers and lower case letters is Either [a-z0-9] or [0-9a-z].

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

Ans:- re.IGNORECASE as a flag to make a normal expression 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 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?

Ans:- The output of the string 'X drummers, X pipers, five rings, X hens'.

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

Ans:- re.VERBOSE will allow to add whitespace and comments to 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)

Ans:- import re

pattern = r'^\d{1,3}(,\d{3})\*$'

pagex = re.compile(pattern)

for ele in ['42','1,234', '6,368,745','12,34,567','1234']:

print('Output:',ele, '->', pagex.search(ele))

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:- pattern = r'[A-Z]{1}[a-z]\*\sWatanabe'

**import** re

pattern **=** r'[A-Z]{1}[a-z]\*\sWatanabe'

namex **=** re**.**compile(pattern)

**for** name **in** ['Haruto Watanabe','Alice Watanabe','RoboCop Watanabe','haruto Watanabe','Mr. Watanabe','Watanabe','Haruto watanabe']:

print('Output: ',name,'->',namex**.**search(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.'

Ans:- pattern = r'(Alice|Bob|Carol)\s(eats|pets|throws)\s(apples|cats|baseballs)\.'

**import** re

pattern **=** r'(Alice|Bob|Carol)\s(eats|pets|throws)\s(apples|cats|baseballs)\.'

casex **=** re**.**compile(pattern,re**.**IGNORECASE)

**for** ele **in** ['Alice eats apples.','Bob pets cats.','Carol throws baseballs.','Alice throws Apples.','BOB EATS CATS.','RoboCop eats apples.'

,'ALICE THROWS FOOTBALLS.','Carol eats 7 cats.']:

print('Output: ',ele,'->',casex**.**search(ele))