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

* re.compile()

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

* Raw Strings are able to handle special characters, and consider them as part of data instead of escaping it. Thats is reason why we use raw strings in regex

Ex :

input : print(r”Hi \t is used for tab space” )

output : “Hi \t is used for tab space”

input : print(”Hi \t is used for tab space” )

output : “Hi is used for tab space”

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

* It returns regex object if the pattern matches otherwise it returns None.

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

* Using group()

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?

* group zero gives the string with matching regex **r'(\d\d\d)-(\d\d\d-\d\d\d\d)'**
* group 1 gives the string with matching regex within first () brackets, i.e, **(\d\d\d)**
* group 2 gives the string with matching regex within second () brackets, i.e, (**\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?

* To fit real one, i will put backslah before the parentheses in regex part
* To fit the period, i wont put backslash.

txt = "aa{bcd}"

print(txt)

x = re.search(r'a{2}\{bcd}', txt)

x.string

output : aa{bcd}

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

* findall() method returns all the matching strings wrt regex, so it will returns string list or list of string tuples.

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

* | like a **or operator** in regex.

Ex :

hi | hello -> used in regex returns the strings matches either with hi or hello

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

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

* + tells regex either one or more occurances
* \* tells regex either zero or more occurances

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

* {4} -> matches a character for 4 Repetitions
* {4,5} -> matches min 4 and max 5 repetitions only

ex:

txt = "afaaaaaaf"

regex : a{4}f -> "afaaaaaaf" -> matches index from (4, 9)

regex : a{4,5} -> "afaaaaaaf"-> matches index from (3, 9)

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

* \d -> matches strings which contain numbers
* \w -> matches strings which contain alphabets, digits, and underscore
* \s -> matches strings which contain whitespace characters

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

* \D -> matches strings which doesnt contain numbers
* \W -> matches strings which doesnt contain any word, numbers and underscore
* \S -> matches strings which doesnt contain whitespace characters

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

* .\* -> matches character with zero or more occurances.
* .\*? -> last character in regex part is optional to match the string

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

* [0-9a-z] or [a-z0-9]

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

* Using parameter **re.IGNORECASE** .

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

* Dot(.) matches all characters except newline, re.dotall extents its functionality by matching newline character in the string

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

* 'X drummers, X pipers, five rings, X hen'

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

* Allows to write the regex in more readable format

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)

* txt="123,456"

pattern = re.compile("(^\d{1,3})(,\d{3}){0,}")

x = pattern.fullmatch(txt)

x.string

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)

* txt = 'RoboCop Watanabe'

pattern = re.compile(r'[A-Z][a-z]\*\sWatanabe')

x = pattern.search(txt)

print(x.string)

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.'

* txt = 'Carol eats 7 cats.'

pattern = re.compile(r"^(alice|bob|carol) (eats|pets|throws) (apples|cats|baseballs)\.$", re.IGNORECASE)

x = pattern.search(txt)

print(x.string)