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

**Ans:** re.complie()

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

**Ans:** Raw string notation (r"text") keeps regular expressions meaningful and confusion-free. Without it, every backslash ('\') in a regular expression would have to be prefixed with another one to escape it.

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

**Ans:** If Search() method is successful then it will return the matched text, if search() methods fails then it will return None.

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

**Ans:** Match.group() is used to get the actual strings that match the pattern.

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:** 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?

**Ans**: Intervals 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?

**Ans:** 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?

**Ans:** Character | behaves like “Either, Or “between two groups.

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

**Ans:** 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?

**Ans:** The + matches one or more. The \* matches zero or more.

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

**Ans:** {4} -- matches exactly 4 instances

{4,5} – matches between 4 and 5 instances

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

**Ans:** \d – matches single digit numbers 0-9

\w matches alphabetical word

\s – matches space (tab, newline)

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

**Ans:** \D– matches non single digit number

\W – matches non alphabetical character

\S – matches non space

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

**Ans: .\*** is a greedy mode, which tends to get the longest string that meets the conditions.  
 **.\*?** is a non-greedy mode, which tends to get the shortest string that can meet the conditions.

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

**Ans:** pattern = '[a-z0-9]\*' or pattern = '[0-9a-z]\*'

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

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

And passing (?i) in pattern also makes 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  matches with any character excluding the new line but using DOTALL flag  we can extend its functionality. With the help of DOTALL flag the '. ' character can match any character including newline.

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

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

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

**Ans:** Allows us to add comments right inside the pattern for later reference using the hash sign (#).

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:** re.compile(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)

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

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:**

import re

string = input()

pattern = re.compile(r'(Alice|Bob|Carol)\s(eats|pets|throws)\s(apples|cats|baseballs)\.', re.I)

match = pattern.match(string)

if match:

print("found",match)

else:

print("not found")