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

**Answer**: re.compile() method of re module

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

**Answer**: In regex when we create an object which is normally in the form of \d where it represents a digit character. So there can be n number of \ i.e backslash and for each \ we need one escape character. Thus to avoid all this complexity we use the raw string.

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

**Answer:**Normally it return the **match** as specified in re.compile()**.  However,** if the match is not found it may return **None**.

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

**Answer:** The **group()** or **groups()** method will get us the actual string 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?**

**Answer**:

|  |  |  |
| --- | --- | --- |
| group(0) | (\d\d\d)-(\d\d\d-\d\d\d\d) | It will return the match as it is. I.e. **complete match** |
| group(1) | (\d\d\d) | It will return the value match in **first** **paranthesis** |
| group(2) | (\d\d\d-\d\d\d\d) | It will return the value match in **second** **paranthesis** |

**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**: We need to use**\ i.e. escape them**

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

|  |  |
| --- | --- |
| If there are **groups** in regex | If there are **no groups** in regex |
| It returns the **tuple** | It returns the **list** |

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

**Answer**: | means pipe and it in standard **sense used to match either of many options** available.

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

**Answer**: -----------Did not understand the question-------

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

**Answer:**

|  |  |
| --- | --- |
| **+** | **\*** |
| Whatever precedes the + character must **atleast appear once** in the text. Else the regex returns NoneType | Whatever precedes **\* does not require to appear** for the match. **i.e. None or more** and still it will return the match. |

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

**Answer:**

|  |  |
| --- | --- |
| **{4}** | **{4,5}** |
| Whatever precedes {4} has to be repeated four times(4X) to Match else it will return NoneType | Whatever precedes **{4,5}** has to be repeated in **range of 4-5** to return the match else it will return None type |

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

|  |  |
| --- | --- |
| \d | Any numeric digit from 0 to 9 |
| \w | Any alpha-numeric character or underscore |
| \s | Any space tab or new line (whitespace) |

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

|  |  |
| --- | --- |
| \D | Anything that is **not** numeric digit from 0 to 9 |
| \W | Anything that is **not** alpha-numeric character or underscore |
| \S | Anything that is **not** space tab or new line |

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

**Answer**:

|  |  |
| --- | --- |
| **.\*** | **.\*?** |
| Match Everything: as much as possible | It will try to match less or minimum |
| Greedy Match | Non-Greedy Match |

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

**Answer**: alnum= re.compile**(r'[0-9a-z]')**

#please see there is no space. If space is included then it will match space as well.

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

**Answer**: We canpass **re.I** as **second** **argument** to re.compile()

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

|  |  |
| --- | --- |
| **. character** | **re.DOTALL** |
| Dot character normally **match one** character before the specified pattern. And it matches **any character except the new line.** | By passing re.DOTALL as second argument to re.compile() the dot character **will match all the character including the new line.** |

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

**Answer:**

'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**: It will allow user to write the comments and ignore the whitespaces. Just like what one does with ‘’’ ‘’’ three inverted commas. Ultimately It helps reduce the complexity of the expression and make it look more simple.

**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:

import re

regex\_digit=re.compile(r'''

^(\d{1}|\d{2}|\d{3})

(\,\d{3})\*$''',re.VERBOSE)

ans=regex\_digit.match('30,000,000')

print(ans.group())

**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**:

#Watanabe is the surname

name\_regex=re.compile(r'(\W)?(\w\*)?(\s)?Watanabe')

line=str(input('Please enter the input: '))

ans=name\_regex.search(line)

print(ans.group())

**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.'**

import re

sentence\_regex=re.compile(r'(Alice|Bob|Carol)(\s?)(eats|pets|throws)(\s?)(apples|cats|baseballs)\.',re.IGNORECASE)

line=str(input('Please input: '))

ans=sentence\_regex.search(line)

print(ans.groups())