|  |  |
| --- | --- |
|  | Chapter 7 Practice Questions – Pattern Matching with Regular Expressions |
| **Q1** | **What is the function that creates Regex objects?** |
| A | The function is re.compile. Passing a string value representing your regular expression to re.compile() returns a Regex patterns object e.g.:  Re.compile(r’\d\d\d-\d\d\d-\d\d\d\d’) should be able to pick up the telephone number in the string: ‘My number is 415-555-4242.’ |
| **Q2** | **Why are raw strings often used when creating Regex objects?** |
| A | By putting an r before the first quote of the regex string value-pattern, the string is marked as raw string which means escape characters such as (\) are not needed if you want to include a include the regex pattern for a single digit match such as (\d) |
| **Q3** | **What does the search() method return?** |
| A | A regex object’s search() method searches the string it is passed for any matches to the regex pattern e.g.:  adefinedregex.search(‘My number is 415-555-4242.’) returns 415-555-4242 |
| **Q4** | **How do you get the actual strings that match the pattern from a Match object?** |
| A | Using the search() method, if a match is found matching the regex pattern, a search object is returned (assigned to a variable). The group() function can then be called on match object (the variable) as matchobject.group(). Using a print function would then return the matching string.  The group() method basically returns strings of the matched text but inside the group() method parenthesis the parts that match the regex expression (each surrounded by parenthesis in the regex expression can be returned) |
| **Q5** | **In the regex created from r’(\d\d\d)-(d\d\d-\d\d\d\d)’, what does group 0 cover? Group 1? Group2?** |
| A | In the regex pattern using parenthesis creates groups in matching. So the first set of parenthesis in a regex string will be group 1 (\d\d\d), the second group 2 (d\d\d-\d\d\d\d). Group 0 will represent the entire matched text |
| **Q6** | **Parentheses and periods have specific meanings in regular expression syntax. How would you specify that you want a regex to match actual parentheses and period characters?** |
| A | If you want a regex to match actual parentheses and period characters, then those characters must be escaped within the regex string using the (\) as in \( and \) e.g. r’(\(\d\d\d\)) will match 3 digits as well as the parentheses. The same principle applies to the period character. It would need to be escaped. |
| **Q7** | **The findall() method returns a list of strings or a list of tuples of strings. What makes it return one or the other?** |
| A | The findall() method unlike the search() method returns strings of every match in the searched string. If there are groups in the regular expression, then findall() will return a *list* of tuples with each tuple representing a found match e.g. [(‘415’,’555’,’1122’),(‘212’,’555’,’0000’)]  From (r’\d\d\d-\d\d\d-\d\d\d\d’) regex expression |
| **Q8** | **What does the | character signify in regular expressions?** |
| A | It is used in the regex match string to match either pattern separated by it. The first occurrence of the pattern is returned if say both patterns separated by | are found |
| **Q9** | **What two things does the ? character signify in regular expressions?** |
| A | The ? character after the pattern enclosed in parenthesis in a regex expression signifies that the pattern is optional e.g. Bat(wo)?man meaning if the search string is Batman then that is fine but will consider also a string called Batwoman and return Batwoman.  It also works like a nongreedy match i.e. matching the shortest string possible |
| **Q10** | **What is the difference between the + and \* characters in a regular expression?** |
| A | The \* character after the pattern enclosed in parenthesis in a regex expression says that the pattern should be matched 0 or more times e.g. Ba(wo)man on ‘Batwowowowoman’ will match 4 instances of (wo).  The + character after the pattern enclosed in parenthesis in a regex expression says that the pattern should be matched 1 or more times e.g. Bat(wo)man on ‘Batman’. The returned object is None = True i.e. the (wo) part has not appeared at least once. |
| **Q11** | **What is the difference between {3} and {3,5} in regular expressions?** |
| A | The {3} is used to indicate in the regex expression how many times a search string should be repeated e.g. (Ha){3} should match on the sting Ha repeated 3 times like ‘HaHaHa’; rather than having to type out the repeating string in its entirety  The {3,5} is used to indicate in the Regex expression the range of the search string repetition e.g. (Ha){3,5} means search ‘HaHaHa’, ‘HaHaHaHa’ and ‘HaHaHaHaHa’ |
| **Q12** | **What do the \d, \w, and \s shorthand character classes signify in regular expressions?** |
| A | In the Regex object:  \d means: Any numeric digit from 0 to 9  \w means: Any letter, numeric digit, or the underscore character i.e. word characters  \s means: Any space, tab, or newline character |
| **Q13** | **What do the \D, \W and \s shorthand character classes signify in regular expressions?** |
| A | Are used in the Regex expressions to define the match criteria:  \D: Any character that is not a numeric digit from 0 to 9  \W: Any character that is not a letter, numeric digit, or the underscore character  \S: Any character not a space, tab, or newline |
| **Q14** | **How do you make a regular expression case-insensitive?** |
| A | Use IGNORECASE or I as a second argument to re.compile() when creating the regex expression e.g.  regex1 = re.compile(r’robocop’, re.I) |
| **Q15** | **What does the . character normally match? What does it match if re.DOTALL is passed as the second argument to re.compile()?** |
| A | The . character in a regular expression is a wildcard which will match any character except for a newline e.g.  re.compile(r’.at’) will match words end in “at”  The DOTALL argument in addition to the . character matches all characters including the newline character e.g.  re.compile(‘.\*’, re.DOTALL) |
| **Q16** | **What is the difference between .\* and .\*? ?** |
| A | .\* is used in a regex expression to match anything e.g. to match the string ‘First Name:’ followed by whatever comes after it e.g.  re.compile(r’First Name: (.\*)’)  .\*? matches in a non-greedy (or lazy) fashion meaning that it would only match the first occurrences of the matching character e.g.  re.compile(‘<.\*?>’) and re.compile(‘<.\*>’) on (‘<To serve man> for dinner.>’) the non-greedy match would find ‘<To serve man>’ whilst the greedy match would find ‘<To serve man> for dinner.>’ |
| **Q17** | **What is the character class syntax to match all numbers and lowercase letters?** |
| A | a-z0-9 |
| **Q18** | **If numRegex= re.compile(r’\d+’), what will numRegex.sub(’X’, ’12 drummers, 11 pipers, five rings, 3 hens’) return?** |
| A | numRegex.sub will show/return ‘ x dummies, x pipers, five rings, x hens’ because the sub method replaces one string with another. In this case the regex is matching one or more of the preceding groups of a digit  NB – there’s only one group here in the regex expression |
| **Q19** | **What does passing re.VERBOSE as the second argument to re.compile() allow you to do?** |
| A | Tells regex to ignore whitespace and comments in long and convoluted regular expressions |
| **Q20** | **How would you write a regex that matches a number with commas for every three digits? It must match the following:**   * **‘42’** * **‘1,234’** * **‘6,368,745’**   **But nit the following:**   * **’12,34,567’ (which has only two digits between the commas)** * **‘1234’ (which lacks commas)** |
| A | commRegex = re.comile(r’W\d{3}’)  commRegex = re.compile(r’^\d{1,3}(,\d{3})\*$’)  The caret symbol symbolises that the subsequent regex patterns must be matched at the beginning of the search text  The dollar symbol indicates that the searched string must end with the regex pattern |
| **Q21** | **How would you write a regex that matches the full name of someone whose last name is Nakamoto? You can assume that the first name that comes before it will always be one word that begins with a capital letter. The regex must match the following:**   * **‘Satoshi Nakamoto’** * **‘Alice Nakamoto’** * **‘Robocop Nakamoto’**   **But not the following:**   * **‘satoshi Nakamoto’ (where the first name is not capitalized)** * **‘Mr . Nakamoto’ (where the preceding word has a nonletter character)** * **‘Nakamoto’ (which has no first name)** * **‘Satoshi nakamoto’ (where Nakamoto is not capitalized)** |
| A | nameRegex = re.compile(r’^[A-Z].\*, ‘Nakamoto$’)  nameRegex = re.compile(r’[A-Z][a-z]\*\sNakamoto’)  NB - \s is space, \* means match 0 or more or an occurrence of words with capital starting and the rest lower case |
| **Q22** | **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.’** |
| A | wordmatch = re.compile(r’^Alice|Bob|Carol, eats|pets|throws,apples|cats|baseballs$’, re.IGNORECASE)  wordmatch = re.compile(‘(Alice|Bob|Carol)\s(eats|pets|throws)\ s(apples|cats|baseballs)\. ‘, re.IGNORECASE) |