Regular Expressions

Here’s a complete list of metacharacters:

```
.  ^  $  *  +  ?  {  }  [  ]  \  |  (  )
```

Some reminders on what each can do (this is not exhaustive):

- "^" matches the position at the beginning of string (unless used for negation "[^\ ]")
- "$" matches the position at the end of string character.
- "?" match preceding literal or sub-expression 0 or 1 times.
- "+" match preceding literal or sub-expression one or more times.
- "*" match preceding literal or sub-expression zero or more times.
- "." match any character except new line.
- "[ ]" match any one of the characters inside, accepts a range, e.g., "[a-c]".

Some useful `re` package functions:

- `re.split(pattern, string)` split the string at substrings that match the pattern. Returns a list.
- `re.sub(pattern, replace, string)` apply the pattern to string replacing matching substrings with replace. Returns a string.
- `re.findall(pattern, string)` Returns a list of all matches for the given pattern in the string.
Regular Expressions

1. Which strings contain a match for the following regular expression, "1+1$"? The character "." represents a single space.

- What_is_1+1
- Make_a_wish_at_11:11
- 111_Ways_to_Succeed

2. Given the text:

"<record>Josh_Hug<hug@cs.berkeley.edu>Faculty</record>"
"<record>Manana_Hakobyan<manana.hakobyan@berkeley.edu>TA</record>"

Which of the following matches exactly to the email addresses (including angle brackets)?

- <.*@.*>
- <[^>]+@[^>]+>
- <.*@[a-z1,9]+>

3. For each pattern specify the starting and ending position of the first match in the string. The index starts at zero and we are using closed intervals (both endpoints are included).

<table>
<thead>
<tr>
<th>Pattern</th>
<th>abcdefg</th>
<th>abcs!</th>
<th>ab_abcd</th>
<th>abc,_[123]</th>
</tr>
</thead>
<tbody>
<tr>
<td>abc*</td>
<td>[0,2]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[^\s]+</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ab.*c</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[a-z1,9]+</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4. Write a regular expression that matches strings (including the empty string) that only contain lowercase letters and numbers.

5. Write a regular expression that matches strings that contain exactly 5 vowels.

6. Given that address is a string, use re.sub to replace all vowels with a lowercase letter "o". For example "123_Orange_Street" would be changed to "123_orongo_Street".
7. Given that sometext is a string, use `re.sub` to replace all clusters of non-vowel characters with a single period. For example "a big moon, between us..." would be changed to "a.i.o.o.e.e.e.u.".

8. Given `sometext = "I’ve got 10 eggs, 20 gooses, and 30 giants."`, use `re.findall` to extract all the items and quantities from the string. The result should look like `['10 eggs', '20 gooses', '30 giants']`. You may assume that a space separates quantity and type, and that each item ends in s.

9. Given the following text in a variable `log`:


   Fill in the regular expression in the variable `pattern` below so that after it executes, day is 26, month is Jan, and year is 2014.

   ```python
   pattern = ...
   matches = re.findall(pattern, log)
   day, month, year = matches[0]
   ```