

1. Write a string for each case that would satisfy at least one match for the following regular expressions:

- a. .abc
- b. a+b?!1{4}
- c. .{3}a\b
- d. \w
- e. \s
- f. \d
- g. .
- h. [abc]
- i. (abc)
- j. [a-zA-Z_\\$\.\.]+[A-Za-z_\\$0-9\.\.]*@[a-zA-Z_\\$\.\.]+[a-zA-Z_\\$0-9\.\.]*\.(com|net|org){1}
- k. \([0oOn]{1}(_|s)[0oOn]{1}\)

Estimated Time: 2 hour.

2. Write a regular expression that can match:

- A. Date format <Month-string> <##day>, <####year>
 - a. examples:
 - i. September 29, 1972
 - ii. February 99, 0001
 - iii. June 04, 3000
- B. A letter followed OR preceded by a number
 - a. example
 - i. A52
 - ii. d747
 - iii. 27X
 - iv. v2
- C. txt, java, and cpp files with names consisting of only letters
 - a. example
 - i. test.java
 - ii. program.cpp
 - iii. newReport.txt
- D. A 5 character palindrome
 - a. example
 - i. abcba
 - ii. 12321
 - iii. _1a1_
- E. All words that consist of letters from b to y only
 - a. example
 - input:** "Bee zapp Crow Eagle Zorro mouse Ape you"
 - output:** ["Bee", "Crow", "mouse", "you"]

F. All the non nested tag elements in a string

a. example

input: "Is 4 < -1/12 true? The answer will
surprise you."

output: 4 < -1/12, answer, surprise

Estimated Time: 6 hour.

3. Write a program that will:

A. Shift cyclically every letter of the alphabet by one, and the numbers as well.

a. example

i. aBc = bCd

ii. xyz = yza

iii. aK89 = bL90

B. From a reasonably sized text, have a user defined string be replaced by that same string with a hashtag. That hashtag should be a link for a twitter search as well.

a. example

i. String: yolo

1. ... should never use [#yolo](#) for any reason...

C. Using regular expressions, create a function that will match all word palindromes of any size in a text.

Optional: Input a paragraph of text and translate it to 1337 \$|>34|<

Estimated Time: 8 hour.