Test 1

These are very simple technical questions to give us a bit overview on your basic programming understanding, skill, and style.

Please use a programing PHP or JS for programing language. Some questions have challenge case that you can skip, but will have more points if you can finish the challenge correctly.

Question 1

Please write code containing at least one main function/method called **sum_deep** that receives one array that its structure like a tree and one character. The function/method will return a positive integer of sum of node level which any nodes contain the second parameter. Level of the root node is 1. Assump the parameter never empty (array always at least have one node as root node and the second parameter always one character) and the height of the tree is finite. Example test case:

a. Input: [“AB”, [“XY”], [“YP”]], ‘Y’ Output: 4
b. Input: [“”, [“”, [“XXXXX”]]], ‘X’ Output: 3
c. Input: [“X”], ‘X’ Output: 1
d. Input: [“”], ‘X’ Output: 0
e. Input: [“X”, [“”, [“”, [“Y”], [“X”]]], [“X”, [“”, [“Y”], [“Z”]]]], ‘X’ Output: 7
f. Input: [“X”, [“”], [“X”], [“X”], [“Y”, [“”]], [“X”]], ‘X’ Output: 7

Note: First parameter input in example test case above is array with JSON-like format.

CHALLENGE FOR QUESTION 1

Use question number 1. However, the second parameter may have two characters or more so the function must sum up all sum of node level of every character in it. Another thing is the level number for each of the second parameter will multiply by number position of the character itself.

Example test case:

Input: [“ABAH”, [“CIRCA”], [“CRUMP”, [“IRA”]], [“ALI”]], “ACI”

Calculate (for illustration):

= sum_a*position_a + sum_c*position_c + sum_i*position_i

= (1+2+3+2)*1 + (2+2)*2 + (2+3+2)*3 Output: 37

Note: First parameter input in example test case above is array with JSON-like format.

Question 2

Please write code containing at least one main function/method called **get_scheme**that

receives a string containing simple HTML tags like <div>, <b> , <i> , <u> with attributes . Any attribute that contains scheme prefix **sc-** is tag with scheme. The function/method will return an array that contain of scheme name. Assump the HTML tags are always have correct open and close tag and have no branch.

Example test case:

a. Input: “<i sc-root>Hello</i>” Output: [“root”]

b. Input: “<div><div sc-rootbear title=”Oh My”>Hello <i sc-org>World</i></div></div>” Output: [“rootbear”, “org”]

CHALLENGE FOR QUESTION NUMBER 2

Use question number 2. But, in this case the tree structure of the input will have multiple scheme in a tag and may have a value. The function must return a json string that structured like a tree. The first element of the result node contains an array that it is a list of scheme parsed from input parameter. The rest element is branches of node.

Example test case:

1. Input: `“<i sc-root=”Hello”>World</i>”`
   Output: `[{“root”:”Hello”}]`

2. Input (1 line):
   ```
    “<div sc-prop sc-alias=”” sc-type=”Organization”><div sc-name=”Alice”>Hello <i
   sc-name=”Wonderland”>World</i></div></div>”
   ```

   Output (aligned for readable purpose):

   ```
   [
      {“prop”:””, alias:””, “type”:”Organization”},
      [
        {“name”:”Alice”},
        [
          {“name”:”Wonderland”}
        ]
      ]
    ]
   ```

Question 3

Please write code containing at least one main function/method called pattern_countthat receives two strings or two array of characters with length between 0 and 100 characters. First parameter is the text and second parameter is the pattern. This function will return a number how many pattern is inside the text. Assume the input parameters are always not null. Your solution must not use any predefined helper function like substr_countin PHP or regex match length in JavaScript.

Example:

- Input: “palindrom”, “ind”
  Output: 1
- Input: “abakadabra”, “ab”
  Output: 2
- Input: “hello”,
  “” Output: 0
- Input: “ababab”, “aba”
  Output: 2
- Input: “aaaaaa”, “aa”
  Output: 5
- Input: “hell”, “hello”
  Output: 0

Question 4

Simple Question Class OOP

A shipyard wants to make an application to store the data/ship chart that has been made.

With the concept of object-oriented programming, create classes along with abstractions, functions and properties for several types of ships (motor boats, sailboats & yachts) and implement some basic concepts such as encapsulation & polymorphism.