Subdomain Visit Count

A website domain like "discuss.leetcode.com" consists of various subdomains. At the top level, we have "com", at the next level, we have "leetcode.com", and at the lowest level, "discuss.leetcode.com". When we visit a domain like "discuss.leetcode.com", we will also visit the parent domains "leetcode.com" and "com" implicitly.

Now, call a "count-paired domain" to be a count (representing the number of visits this domain received), followed by a space, followed by the address. An example of a count-paired domain might be "9001 discuss.leetcode.com".

We are given a list `cpdomains` of count-paired domains. We would like a list of count-paired domains, (in the same format as the input, and in any order), that explicitly counts the number of visits to each subdomain.

```
Example 1:
Input: 
["9001 discuss.leetcode.com"]
Output: 
["9001 discuss.leetcode.com", "9001 leetcode.com", "9001 com"]
Explanation: 
We only have one website domain: "discuss.leetcode.com". As discussed above, the subd
omain "leetcode.com" and "com" will also be visited. So they will all be visited 9001
 times.


Example 2:
Input: 
["900 google.mail.com", "50 yahoo.com", "1 intel.mail.com", "5 wiki.org"]
Output: 
["901 mail.com","50 yahoo.com","900 google.mail.com","5 wiki.org","5 org","1 intel.ma
il.com","951 com"]
Explanation: 
We will visit "google.mail.com" 900 times, "yahoo.com" 50 times, "intel.mail.com" onc
e and "wiki.org" 5 times. For the subdomains, we will visit "mail.com" 900 + 1 = 901
times, "com" 900 + 50 + 1 = 951 times, and "org" 5 times.
```

Notes:

- The length of `cpdomains` will not exceed `100`.
- The length of each domain name will not exceed `100`.
- Each address will have either 1 or 2 "." characters.
- The input count in any count-paired domain will not exceed `10000`.

Solution 1

```
Example 1:
Input:
["900 google.mail.com", "50 yahoo.com", "1 intel.mail.com", "5 wiki.org"]
Output:
["9001 discuss.leetcode.com", "9001 leetcode.com", "9001 com"]
Explanation:
We only have one website domain: "discuss.leetcode.com". As discussed above, the sub
domain "leetcode.com" and "com" will also be visited. So they will all be visited 90
01 times.
```

written by 0xFFFFFFFF original link here

Solution 2

```
// General idea:
// - first split your cpdomain into int n and string s
// - then parse s backwards looking for '.' and do a += n to a hashmap with the subs
tring for domain
// - make sure to add the whole domain to hasmap when you hit beginning of line
// - create vector of strings at the end from hashmap and return

vector<string> subdomainVisits(vector<string>& cpdomains)
{
    unordered_map<string, int> m;

    for (const auto& word : cpdomains)
    {
        int i    = word.find (" ");
        int n    = stoi (word.substr (0, i));
        string s = word.substr (i+1, word.size ()-i-1);

        for (int i = s.size ()-1; i >= 0; i--) {
            if (s[i] == '.') m[s.substr (i+1, s.size ()-i-1)] += n;
            else if (i == 0) m[s.substr (i,   s.size ()-i)  ] += n;
        }
    }

    vector<string> v;
    for (const auto& e : m) v.push_back (to_string (e.second) + " " + e.first);
    return v;
}
```

written by code_report original link here

Solution 3

```java
class Solution {
    public List<String> subdomainVisits(String[] cpdomains) {
        Map<String, Integer> map = new HashMap();

        for(String str : cpdomains){
            String[] line = str.split(" ");
            int count = Integer.valueOf(line[0]);
            String[] domains = line[1].split("\\.");
            String temp = "";
            for(int i = domains.length - 1;i >= 0;i--){
                temp = domains[i] + (temp.equals("") ? temp : "." + temp);
                if(!map.containsKey(temp)){
                    map.put(temp, count);
                }else{
                    map.put(temp, map.get(temp) + count);
                }
            }
        }

        List<String> res = new ArrayList();
        for(String str : map.keySet()){
            res.add(map.get(str) + " " + str);
        }

        return res;
    }
}
```

written by kevincongcc original link here