```
/*
给定一个只包含数字的字符串，复原它并返回所有可能的 IP 地址格式。

示例:

输入: "25525511135"
输出: ["255.255.11.135", "255.255.111.35"]

来源：力扣（LeetCode）
链接：https://leetcode-cn.com/problems/restore-ip-addresses
著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。
*/
```

分析:

- 一个byte一个byte的填,当取数据长度为3时要注意对应的整数是否大于了255;
- 若取的数据长度大于1,要去掉第一个数据是否0;
- 方法一:递归写法;
- 方法二:非递归写法.

方法一:C++_递归写法

```
class Solution
{
    private:
        void helper(    vector<string>&     vs          ,
                        string&             s           ,
                        string              cur_str     ,
                        int                 str_idx     ,
                        int                 count
                    )
        {
            if(count == 4 && str_idx==s.size())
            {
                vs.push_back(cur_str);
                return;
            }

            for(int length = 1 ; length <=3 ; length++ )
            {
                if(s.size() <str_idx+length)
                {
                    break;
                }
                string sub = s.substr(str_idx,length);
                /*去掉前导0的可能性*/
                if(length > 1 && sub[0] =='0')
                {
                    continue;
                }
```

```
                if(stoi(sub) > 255)
                {
                    break;
                }
                if(count==3)
                {
                    helper(vs,s,cur_str+sub,str_idx+length,count+1);
                }
                else
                {
                    helper(vs,s,cur_str+sub+".",str_idx+length,count+1);
                }

            }



        }
    public:
        vector<string> restoreIpAddresses(string s)
        {
            vector<string> vs;
            if(s.size() < 4 || s.size() > 12)
            {
                return vs;
            }

            helper(vs,s,string(),0,0);

            return vs;


        }
};

/*
执行结果：
通过
显示详情
执行用时 :8 ms, 在所有 cpp 提交中击败了46.22% 的用户
内存消耗 :9.3 MB, 在所有 cpp 提交中击败了10.32%的用户
*/
```

方法二:C++_非递归写法

```
class Solution
{
    public:
    vector<string> restoreIpAddresses(string s)
    {
        vector<string> vs;
        if(s.size() < 4 || s.size() > 12)
        {
            return vs;
        }
```

```
11
12          int length[4]   = {0,0,0,0}     ;
13          int idx[5]      = {0,0,0,0,0}   ;
14          int flag        = 0             ;
15          int i           = 0             ;
16
17          for(length[0] = 1; length[0] <=3 ;length[0]++)
18          {
19              for( length[1] = 1; length[1] <=3 ;length[1]++)
20              {
21                  for( length[2] = 1; length[2] <=3 ;length[2]++)
22                  {
23                      for( length[3] = 1; length[3] <=3 ;length[3]++)
24                      {
25
26                          flag    =   1                       ;
27                          idx[0]  =   0                       ;
28                          idx[1]  =   length[0] + idx[0]      ;
29                          idx[2]  =   length[1] + idx[1]      ;
30                          idx[3]  =   length[2] + idx[2]      ;
31                          idx[4]  =   length[3] + idx[3]      ;
32
33                          if(idx[4] == s.size())
34                          {
35                              string cur_s;
36                              for( i = 0 ; i < 4 ; i++)
37                              {
38                                  if(   (length[i] > 1 && s[idx[i]]=='0')
39                                      ||(stoi(s.substr(idx[i],length[i])) >
    255)
40                                  )
41                                  {
42                                      flag = 0;
43                                      break;
44                                  }
45
46                                  if(i<3)
47                                  {
48                                      cur_s = cur_s +
    s.substr(idx[i],length[i]) + '.';
49                                  }
50                                  else // i == 3
51                                  {
52                                      cur_s = cur_s +
    s.substr(idx[i],length[i]) ;
53                                  }
54                              }
55                              if(flag)
56                              {
57                                  vs.push_back(cur_s);
58                              }
59                          }
60                      }
61                  }
62              }
63          }
64          return vs;
65      }
```

```
};


/*
执行结果：
通过
显示详情
执行用时 :8 ms, 在所有 cpp 提交中击败了46.22% 的用户
内存消耗 :8.4 MB, 在所有 cpp 提交中击败了82.74%的用户
*/

```

AlimyBreak
2019.12.16