# 93. Restore IP Addresses

| | |
|---|---|
| Created | @June 14, 2021 7:28 PM |
| Tags | Medium |
| link | https://leetcode.com/problems/restore-ip-addresses/ |

### Description

Given a string `s` containing only digits, return all possible valid IP addresses that can be obtained from `s`. You can return them in **any** order.

A **valid IP address** consists of exactly four integers, each integer is between `0` and `255`, separated by single dots and cannot have leading zeros. For example, "0.1.2.201" and "192.168.1.1" are **valid** IP addresses and "0.011.255.245", "192.168.1.312" and "192.168@1.1" are **invalid** IP addresses.

### Approach

- We can implement a recursive solution
- Insert '.' in possible places in start and make recursive call for remaining substring.
- Complexity O(n)

```
class Solution {
public:

    // to store result ips
    vector<string> results;

    // check if substring 's' is valid part of IP
    bool validPiece(string s) {
        if (s.length() <= 0 or s.length() > 3) return false;

        if (s[0] == '0' and s.length() > 1) return false;

        if (s[0] == '0' and stoi(s) > 0) return false;

        if (stoi(s) > 255) return false;
        return true;
    }

    int countAllIps(string s, string ip, int k) {
        // No more '.'s to place, append if valid ip else return
        if (k == 0) {
            if (validPiece(s)) {
                results.push_back(ip + s);
                return 1;
            } else {
                return -1;
            }
        }

        // Place '.' and check number of possible IPs in remaining string
        int allCount = 0;
        for (int i = 0; i < 3 && i <= s.length(); i++) {
            string first = s.substr(0, i+1);
            if (i+1 >= s.length())
                continue;
            string rem = s.substr(i+1, s.length() - i - 1);

            if (validPiece(first)) {
                int count = countAllIps(rem, ip + first + ".", k-1);
                if (count != -1) allCount += count;
            }
        }

        return allCount;
    }


    vector<string> restoreIpAddresses(string s) {
        countAllIps(s, "", 3);
        return results;
    }
};
```