第三周习题刷题记录

01.01.判断字符是否唯一

题目： 实现一个算法，确定一个字符串 `s` 的所有字符是否全部不同

算法：

1. 假设给出的字符集是ASCII字符，因此可以将ASCII码的范围0-127为数组下标，因此可以开辟一个128的数组。若出现一次将该下标对应的值设定为1，若再次出现则说明该字符串有相同的字符。

```
bool isUnique(char* astr){
    int array[128];
    memset(array, 0, sizeof(array));
    bool ret = true;
    int len = strlen(astr);
    for(int i = 0; i < len; i++) {
        if(array[astr[i]] == 1) {
            ret = false;
            break;
        } else {
            array[astr[i]] = 1;
        }
    }
    return ret;
}
```

93 .最长上升子序列

题目：给定一个字符串，只包含数字。请解码出所有合法的IP地址。

算法：

1. 暴力枚举dfs
    - 合法的IP地址由四个0到255的整数组成。我们直接枚举四个整数的位数，然后判断每个数的范围是否在0到255。

```
class Solution {
public:
    vector<string> res;//存储的结果
    vector<int> part;

    vector<string> restoreIpAddresses(string s) {
        dfs(0, 0, s);
        return res;
    }

    // index表示枚举到的字符串下标，nums表示当前截断的IP个数，s表示原字符串
    void dfs(int index, int nums, string &s)
    {
        if(nums > 4) {
            return;
        }
        if (index == s.size()) {
```

```
            if (nums == 4) {
                string ip = to_string(part[0]);
                for (int i = 1; i < 4; i++ )
                    ip += '.' + to_string(part[i]);
                res.push_back(ip);
            }
            return;
        }
        unsigned t = 0;//有的测试案例数字超长故用无符号数
        for (int i = index; i < s.size(); i++ ) {
            t = t * 10 + s[i] - '0';
            if (t >= 0 && t < 256) {
                part.push_back(t);
                dfs(i + 1, nums + 1, s);
                part.pop_back();
            }
            //这里是排除的是00,01,001等有前导零的数，这类数不合法
            if (!t) {
                break;
            }
        }
    }
};
```