字符串匹配2: IP地址 ☆☆

题目描述

给定一个字符串 s，请你判断它是否是一个有效的 IPv4 地址。
一个有效的 IPv4 地址由四个整数构成，这些整数位于 0 到 255 之间，用 '.' 分隔。
每个整数不能有前导零（除非该整数本身就是 0），并且每个部分必须是非空的。

输入输出格式

输入：输入包含多组测试数据。每组测试数据占一行，包含一个字符串 s ($1 \leq |s| \leq 15$)。

输入以文件结束符（EOF）结束。

输出：对于每组测试数据，如果输入的字符串 s 是一个有效的 IPv4 地址，则输出 "YES"；否则输出 "NO"。

每个输出占一行。

输入示例	输出示例
192.168.1.1	YES
192.168.1.01	NO
0.0.0.0	YES
255.255.255.255	YES

样例解释

- **192.168.1.1**: 所有部分都在 0-255 范围内，没有前导零，格式正确，输出 YES
- **192.168.1.01**: 第四部分 "01" 有前导零，不符合要求，输出 NO
- **0.0.0.0**: 所有部分都是 0，允许单个 0，输出 YES
- **255.255.255.255**: 所有部分都在 0-255 范围内，没有前导零，格式正确，输出 YES

算法分析

1. 验证条件
   IPv4 地址验证需满足：
   - 恰好包含 3 个点号 '.'，分成 4 个部分
   - 每部分为 0-255 的整数且无前导零
2. 实现思路

（1） 按 '.' 分割字符串为 4 部分

（2） 验证每部分是否都是数字且大小在 0-255 范围内

（3） 验证无前导零

代码实现：

判断逻辑：

```
bool check (string s) {
    if (s.size() > 3 || s.size() == 0) return false;
    //超出3位数，或者是空的直接不满足[0,255]条件
    for (int i = 0; i < s.size(); i++) {
        if (s[i] > '9' || s[i] < '0') return false;
        //验证都是数字
    }
    for (int i = 0; i < s.size(); i++) { //验证无前导零
        if (s[i] == '0' && s.size() > 1) {
            return false;
        } else if (s.size() > 1) {
            break;
        }
    }

    int num = stoi(s); //字符串转换为数字
    if (num < 0 || num > 255) return false; //判断是否在[0,255]范围内
    return true;
};
```

字符串处理：

```
while (cin >> s) {
    string t;
    bool ok = true;
    for (int i = 0; i < s.size(); i++) {
        if (s[i] == '.') {  // 遇到分割符号，检查当前部分是否满足要求
            ok &= check(t);
            t.clear();
        } else {
            t += s[i];
        }
    }

    ok &= check(t);  // 检查最后一部分

    if (ok) {
        cout << "YES\n";
    } else {
        cout << "NO\n";
    }
}
```

拓展思考：

- 给定任意一个字符串和一个分割的字符集，如何实现一个通用的字符串分割函数?