题目：

Given an absolute path for a file (Unix-style), simplify it.

For example,

path = `"/home/"`, => `"/home"`

path = `"/a/./b/../../c/"`, => `"/c"`

click to show corner cases.

Corner Cases:

- Did you consider the case where **path** = `"/../"`?

  In this case, you should return `"/"`.
- Another corner case is the path might contain multiple slashes `'/'` together, such as `"/home//foo/"`.

  In this case, you should ignore redundant slashes and return `"/home/foo"`.

Subscribe to see which companies asked this question.

1.时间：O（N）; 空间：O（1） -->>错误

```
class Solution {
public:
    string simplifyPath(string path) {
        if (path.empty()) return std::string();
        /* 找到最后一个.或者.. */
        int start_index = path.size() - 1;
        for (; start_index >= 0; --start_index){
```

```
            if (start_index == '.') break;
        }
        start_index = start_index == -1 ? 0 : start_index;
        std::string result = path.substr(start_index);
        /*  去除冗余的'/' */
        int index = 0;
        for (int i = 0; i < result.size(); ++i){
            if (result[i] != '/'){
                result[index++] = result[i];
            } else if (i < result.size() - 1 && result[i + 1] != '/'){
                result[index++] = result[i];
            }
        }
        return result;
    }
};
```

2.时间：O（N）；空间：O（N）

```
class Solution {
public:
    string simplifyPath(string path) {
        if (path.empty()) return std::string();
        std::vector<std::string> strs;
```

```
        for (int i = 0; i < path.size();){
            /* 跳过'/' */
            while (i < path.size() && path[i] == '/') ++i;
            std::string str;
            while (i < path.size() && path[i] != '/') str.push_back(path[i++]);
            /* 分析 str 中的内容 */
            if (str == ".." && !strs.empty()) strs.pop_back();
            if (str != "" && str != "." && str != "..") strs.push_back(str);
        }
        if (strs.empty()) return "/";
        std::string result;
        for (int i = 0; i < strs.size(); ++i){
            result += "/" + strs[i];
        }
        return result;
    }
};
```