# 844 比较含退格的字符串

label: 栈
给定 S 和 T 两个字符串，当它们分别被输入到空白的文本编辑器后，判断二者是否相等，并返回结果。 # 代表退格字符。
注意：如果对空文本输入退格字符，文本继续为空。

> 输入：S = "ab#c", T = "ad#c"
> 输出：true
> 解释：S 和 T 都会变成 “ac”。

- 两个栈

```
class Solution {
    public boolean backspaceCompare(String S, String T) {
        Stack<Character> stackS = new Stack<>();
        Stack<Character> stackT = new Stack<>();
        // 去除退格
        char[] sArr = S.toCharArray();
        char[] tArr = T.toCharArray();
        for(int i=0;i<sArr.length;i++){
            if(sArr[i] == '#'){
                if(stackS.isEmpty())
                    continue;

                stackS.pop();
            }else{
                stackS.push(sArr[i]);
            }
        }
        for(int i=0;i<tArr.length;i++){
            if(tArr[i] == '#'){
                if(stackT.isEmpty())
                    continue;
                stackT.pop();
            }else{
                stackT.push(tArr[i]);
            }
        }

        // 构建字符串
        String strS = "";
        String strT = "";
        while(!stackS.isEmpty())strS+=stackS.pop();
        while(!stackT.isEmpty())strT+=stackT.pop();

       return strS.euqals(strT)
    }
}
```

- StringBuffer

```
class Solution {
    public boolean backspaceCompare(String S, String T) {
        return build(S).equals(build(T));
    }

    public String build(String str) {
        StringBuffer ret = new StringBuffer();
        int length = str.length();
        for (int i = 0; i < length; ++i) {
            char ch = str.charAt(i);
            if (ch != '#') {
                ret.append(ch);
            } else {
                if (ret.length() > 0) {
                    ret.deleteCharAt(ret.length() - 1);
                }
            }
        }
        return ret.toString();
    }
}
```