# 【题目】

给定一个字符串，请你找出其中不含有重复字符的 **最长子串**的长度。

**示例 1:**

**输入:** "abcabcbb"

**输出:** 3

**解释:** 因为无重复字符的最长子串是 "abc"，所以其长度为 3。

**示例 2:**

**输入:** "bbbbb"

**输出:** 1

**解释:** 因为无重复字符的最长子串是 "b"，所以其长度为 1。

**示例 3:**

**输入:** "pwwkew"

**输出:** 3

**解释:** 因为无重复字符的最长子串是 "wke"，所以其长度为 3。

# 【分析】

首先该题目要求的是求不重复的最长“子串”，不是子序列，这一点不能搞错。

该题目首先可能首先想到的是暴力求解，就是遍历所有的子串，并判断字串中是不是有重复的，如果没有的话，记录该子串的长度，最后获得所有不重复的最长子串。

求一个字符串所有的子串的时间复杂度为O(n^2),那么在子串的基础上判断是否有重复的字符时间复杂度为O(n),所以这种暴力求解的时间复杂度会有O(n^3)。

那有没有更少复杂度的方法了？答案是当然有了，我们来看一下滑动窗法，有点像TCP协议中流量控制的滑动窗。我们可以举一个例子：

abcbace，来看一下滑动窗该怎么用：

Step1：原始的字符串数组和滑动窗



Step2:此时滑动窗内为空，字符串的第一个字符a不在滑动窗内，所以改动滑动窗，此时滑动窗窗口长度为1，题目的结果为1：



Step3:同理后续的b、c和滑动窗内的字符都不重复，继续滑动滑动窗，此时滑动窗窗口长度为3，题目的结果为3



Step4:来看一下后续的b(第4个字符,表示为b(4))和滑动窗内的第二个字符重复了，现在滑动窗该怎么滑动呢？敲黑板，重点来了，我们换个角度看，某个字符前面有多少和自身不同的连续字符。a(1)就自己，b(2)有ab两个，c(3)有abc三个，b(4)有cb两个，想到滑动窗该怎么滑动了吧，此时滑动窗窗口长度为2，题目的结果为3。



Step5:后续会遇到a(5),滑动窗如你所预期，继续后向后滑动一个，此时滑动窗窗口长度为3，题目的结果为3：



但是为什么这里还有拿出来单独说了，因为这里有一个编码的技巧点，需要在这里强调一下，滑动窗我们可以使用HashMap来实现key我们保存各个字符，Value保存字符的下标，想降低时间复杂度，拿空间过来换，没办法！

所以前面Step2~Step3 HashMap的变化：



在Ste4时由于b已经在窗口中可以直接更新b的Value



但是此时我们并没有按照示意图中的窗口删除a，因为我们可以在实现上将其在逻辑上删除，即窗口的左边界始终取的是重复字符的下标和当前左边界最大值，故a的存在没有任何影响。

# 【代码实现】

xxxx

# 【扩展】

但我看到这个题目，下意识的想到了另外一个比较经典的题目，即两个字符串的最长公共子串

例如：

子串1：abcdef

子串2：xyabcz

最长的公共子串为abc，长度为3

这个题目可以用动态规划进行即