# 50. 第一个只出现一次的字符位置

## 题目链接

[牛客网](https://www.nowcoder.com/practice/1c82e8cf713b4bbeb2a5b31cf5b0417c?tpId=13&tqId=11187&tPage=1&rp=1&ru=/ta/coding-interviews&qru=/ta/coding-interviews/question-ranking&from=cyc_github)

## 题目描述

在一个字符串中找到第一个只出现一次的字符，并返回它的位置。字符串只包含 ASCII 码字符。

Input: abacc
Output: b

## 解题思路

最直观的解法是使用 HashMap 对出现次数进行统计：字符做为 key，出现次数作为 value，遍历字符串每次都将 key 对应的 value 加 1。最后再遍历这个 HashMap 就可以找出出现次数为 1 的字符。

考虑到要统计的字符范围有限，也可以使用整型数组代替 HashMap。ASCII 码只有 128 个字符，因此可以使用长度为 128 的整型数组来存储每个字符出现的次数。

public int FirstNotRepeatingChar(String str) {
 int[] cnts = new int[128];
 for (int i = 0; i < str.length(); i++)
 cnts[str.charAt(i)]++;
 for (int i = 0; i < str.length(); i++)
 if (cnts[str.charAt(i)] == 1)
 return i;
 return -1;
}

以上实现的空间复杂度还不是最优的。考虑到只需要找到只出现一次的字符，那么需要统计的次数信息只有 0,1,更大，使用两个比特位就能存储这些信息。

public int FirstNotRepeatingChar2(String str) {
 BitSet bs1 = new BitSet(128);
 BitSet bs2 = new BitSet(128);
 for (char c : str.toCharArray()) {
 if (!bs1.get(c) && !bs2.get(c))
 bs1.set(c); // 0 0 -> 0 1
 else if (bs1.get(c) && !bs2.get(c))
 bs2.set(c); // 0 1 -> 1 1
 }
 for (int i = 0; i < str.length(); i++) {
 char c = str.charAt(i);
 if (bs1.get(c) && !bs2.get(c)) // 0 1
 return i;
 }
 return -1;
}