438. Find All Anagrams in a String

Description | Hints | Submissions | Discuss | Solution

Pick One

Given a string **s** and a **non-empty** string **p**, find all the start indices of **p**'s anagrams in **s**.

Strings consists of lowercase English letters only and the length of both strings **s** and **p** will not be larger than 20,100.

The order of output does not matter.

Example 1:

```
Input:
s: "cbaebabacd" p: "abc"

Output:
[0, 6]

Explanation:
The substring with start index = 0 is "cba", which is an anagram of "abc".
The substring with start index = 6 is "bac", which is an anagram of "abc".
```

Example 2:

```
Input:
s: "abab" p: "ab"

Output:
[0, 1, 2]

Explanation:
The substring with start index = 0 is "ab", which is an anagram of "ab".
The substring with start index = 1 is "ba", which is an anagram of "ab".
The substring with start index = 2 is "ab", which is an anagram of "ab".
```

```java
public class L438 {
    /*
     * 这个题目利用滑动窗口和hash的思想，这个sArr和pArr里面存储的是字符的个数，
     * pArr[0]是‘a’的个数，pArr[1]是‘b’的个数
     */
     public List<Integer> findAnagrams(String s, String p) {
         List<Integer> list = new ArrayList<>();

         if(s == null || s.length() == 0)
             return list;

         int sLen = s.length();
         int pLen = p.length();

         int [] sArr = new int [256];
         int [] pArr = new int [256];

         for(int i = 0; i < pLen; i ++) {
             pArr[p.charAt(i) - 'a']++;
         }

         for(int i = 0; i < sLen; i ++) {
             sArr[s.charAt(i) - 'a']++;
             //这里就相当于一个滑动窗口，来了一个字符后，先添加进去，然后去掉末尾一个
             if(i >= pLen) {
                 sArr[s.charAt(i - pLen) - 'a']--;
             }
             if(Arrays.equals(pArr, sArr));
             list.add(i - pLen + 1);
         }
         return list;
     }
}
```