028 Implement strStr()

2018年3月29日 14:02

Question:

Implement strStr().

Return the index of the first occurrence of needle in haystack, or **-1** if needle is not part of haystack.

Example 1:

Input: haystack = "hello", needle = "ll"

Output: 2

Example 2:

Input: haystack = "aaaaa", needle = "bba"

Output: -1

来自 <https://leetcode.com/problems/implement-strstr/description/>

返回蕴含在 haystack 中的 needle 的第一个字符的索引，如果 needle 不是 haystack 的一部分则返回 -1 。

Solution for Python3:

```
class Solution1:
    def strStr(self, haystack, needle):
        """
        :type haystack: str
        :type needle: str
        :rtype: int
        """
        n, m = len(haystack), len(needle)
        for i in range(n - m + 1):
            for j in range(m):
                if haystack[i + j] != needle[j]:
                    break
                if j == m -1:
                    return i
        return 0 if m == 0 else -1

class Solution2:
    def strStr(self, haystack, needle):
        """
        :type haystack: str
        :type needle: str
        :rtype: int
        """
        return haystack.find(needle)

class Solution3:
    def strStr(self, haystack, needle):
        """
        :type haystack: str
        :type needle: str
        :rtype: int
        """
        for i in range(len(haystack) - len(needle) + 1):
            if haystack[i:i+len(needle)] == needle:
                return i
        return -1
```

Solution for C++:

```
class Solution1 {
public:
    int strStr(string haystack, string needle) {
        int m = haystack.length(), n = needle.length();
        if (!n) {
            return 0;
        }
        for (int i = 0; i <= m - n; i++) {
            int j = 0;
            for (int j = 0; j < n; j++) {
                if (haystack[i + j] != needle[j]) {
                    break;
                }
                if (j == n - 1)
                  return i;
            }
        }
        return -1;
    }
};

class Solution2 {
public:
    int strStr(string haystack, string needle) {
       int m = haystack.length(), n = needle.length()
       if (!n)
           return 0;
        for (int i = 0; ; i++) {
           for (int j = 0; ; j++) {
              if (j == n)
                  return i;
              if (i + j == m)
                  return -1
              if (needle[j] != haystack[i + j])
                  break;
           }
        }
    }
};
```

Appendix:

Python3：find()

Return the lowest index in the string where substring *sub* is found within the slice s[start:end]. Optional arguments *start* and *end* are interpreted as in slice notation. Return -1 if *sub* is not found.

来自 <https://docs.python.org/3/library/stdtypes.html?highlight=find#str.find>

The find() method should be used only if you need to know the position of *sub*. To check if *sub* is a substring or not, use the in operator:

```
>>> 'Py' in 'Python'
True
```

来自 <https://docs.python.org/3/library/stdtypes.html?highlight=find#str.find>

字符串匹配问题有个著名的匹配算法：