增强的 strstr

题目描述：

C 语言有一个库函数： char *strstr(const char *haystack, const char *needle) ，实现在字符串 haystack 中查找第一次出现字符串 needle 的位置，如果未找到则返回 null。

现要求实现一个 strstr 的增强函数，可以使用带可选段的字符串来模糊查询，与 strstr 一样返回首次查找到的字符串位置。

可选段使用“[]”标识，表示该位置是可选段中任意一个字符即可满足匹配条件。比如“a[bc]”表示可以匹配“ab”或“ac”。

注意目标字符串中可选段可能出现多次。

输入描述：

与 strstr 函数一样，输入参数是两个字符串指针，分别是源字符串和目标字符串。

输出描述：

与 strstr 函数不同，返回的是源字符串中，匹配子字符串相对于源字符串地址的偏移（从 0 开始算），如果没有匹配返回 -1。

补充说明：

源字符串中必定不包含‘[]’；目标字符串中‘[]’必定成对出现，且不会出现嵌套。

输入的字符串长度在[1,100]之间。

示例 1

输入：

```
abcd
b[cd]
```

输出：

```
1
```

说明：

相当于是在源字符串中查找 bc 或者 bd，bc 子字符串相对于 abcd 的偏移是 1

```
def enhanced_strstr(s1, s2):
    m = len(s2)
    l, r = [], []
    for i in range(m):
        if s2[i] =='[':
            l.append(i)
        if s2[i] == ']':
            r.append(i)

    n = len(l)
    if n == 0:
        try:
            return s1.index(s2)
        except ValueError:
            return -1

    prev = s2[:l[0]]
    g = []
    for i in range(l[0]+1, r[0]):
        g.append(str(prev + s2[i]))

    for i in range(1, n):
        left, right = l[i], r[i]
        tem = ''
        for j in range(r[i-1]+1, left):
            tem += s2[j]
        arr = []
        for j in range(left+1, right):
            for k in range(len(g)):
                arr.append(str(g[k] + tem + s2[j]))
        g = arr

    res = 100000
    for s in g:
        # print(s)
        try:
            index = s1.index(s)
            if index >= 0:
                res = min(res, index)
        except ValueError:
            continue
```

```
        if res == 100000:
            return -1

        return res

s1 = input()
s2 = input()
print(enhanced_strstr(s1, s2))
```