139. Word Break

Description | Hints | Submissions | Discuss | Solution

Pick One

Given a **non-empty** string *s* and a dictionary *wordDict* containing a list of **non-empty** words, determine if *s* can be segmented into a space-separated sequence of one or more dictionary words.

Note:

- The same word in the dictionary may be reused multiple times in the segmentation.
- You may assume the dictionary does not contain duplicate words.

Example 1:

```
Input: s = "leetcode", wordDict = ["leet", "code"]
Output: true
Explanation: Return true because "leetcode" can be segmented as "leet code".
```

Example 2:

```
Input: s = "applepenapple", wordDict = ["apple", "pen"]
Output: true
Explanation: Return true because "applepenapple" can be segmented as "apple pen apple".
             Note that you are allowed to reuse a dictionary word.
```

Example 3:

```
Input: s = "catsandog", wordDict = ["cats", "dog", "sand", "and", "cat"]
Output: false
```

```java
public class L127 {

    /*
     * 这是一个动态规划的问题。
     * res[i+1]的意思是从0到i的字符串能不能被包含在wordDict中
     * str.deleteCharAt是一步试探。
     * 当res[j] = true 表示前j个字符组成的字符串可以被包含，wordDict.contains(str.toString())表示后面的也可包含
     * 即代表从0到i组成的字符串都可以被包含，即res[i + 1] = true。
     */
    public boolean wordBreak(String s, List<String> wordDict) {

        if(s == null || s.length() == 0)
            return false;
        boolean [] res = new boolean[s.length() + 1];
        res[0] = true;

        for(int i = 0; i < s.length(); i++) {
            StringBuilder str = new StringBuilder(s.subSequence(0, i + 1));
            for(int j = 0; j <= i; i++) {
                if(res[j] && wordDict.contains(str.toString())) {
                    res[i + 1] = true;
                    break;
                }
                str.deleteCharAt(0);
            }
        }
        return res[s.length()];
    }
}
```