0139. 单词拆分

ITCharge 大约 2 分钟

- 标签：字典树、记忆化搜索、数组、哈希表、字符串、动态规划
- 难度：中等

题目链接

- 0139. 单词拆分 - 力扣

题目大意

描述：给定一个非空字符串 s 和一个包含非空单词的列表 $wordDict$ 作为字典。

要求：判断是否可以利用字典中出现的单词拼接出 s 。

说明：

- 不要求字典中出现的单词全部都使　　并且字典中的单词可以重复使用。
- $1 \leq s.length \leq 300$。
- $1 \leq wordDict.length \leq 1000$。
- $1 \leq wordDict[i].length \leq 20$。
- s 和 $wordDict[i]$ 仅有小写英文字母组成。
- $wordDict$ 中的所有字符串互不相同。

示例：

- 示例 1：

```py
输入：s = "leetcode", wordDict = ["leet", "code"]
输出：true
解释：返回 true 因为 "leetcode" 可以由 "leet" 和 "code" 拼接成。
```

- 示例 2：

```py
输入：s = "applepenapple", wordDict = ["apple", "pen"]
输出：true
解释：返回 true 因为 "applepenapple" 可以由 "apple" "pen" "apple" 拼接成。
     注意，你可以重复使用字典中的单词。
```

解题思路

思路 1：动态规划

1. 划分阶段

按照单词结尾位置进行阶段划分。

2. 定义状态

s 能否拆分为单词表的单词，可以分解为：

- 前 i 个字符构成的字符串，能否分解为单词。
- 剩余字符串，能否分解为单词。

定义状态 $dp[i]$ 表示：长度为 i 的字符串 $s[0:i]$ 能否拆分成单词，如果为 $True$ 则表示可以拆分，如果为 $False$ 则表示不能拆分。

3. 状态转移方程

- 如果 $s[0:j]$ 可以拆分为单词（即 $dp[j] == True$），并且字符串 $s[j:i]$ 出现在字典中，则 `dp[i] = True`。
- 如果 $s[0:j]$ 不可以拆分为单词（即 $dp[j] == False$），或者字符串 $s[j:i]$ 没有出现在字典中，则 `dp[i] = False`。

4. 初始条件

- 长度为 0 的字符串 $s[0:i]$ 可以拆分为单词，即 $dp[0] = True$。

5. 最终结果

根据我们之前定义的状态，$dp[i]$ 表示：长度为 i 的字符串 $s[0:i]$ 能否拆分成单词。则最终结果为 $dp[size]$，$size$ 为字符串长度。

思路 1：代码

```py
class Solution:
    def wordBreak(self, s: str, wordDict: List[str]) -> bool:
        size = len(s)
        dp = [False for _ in range(size + 1)]
        dp[0] = True
        for i in range(size + 1):
            for j in range(i):
                if dp[j] and s[j: i] in wordDict:
                    dp[i] = True
        return dp[size]
```

思路 1：复杂度分析

- **时间复杂度**：$O(n^2)$，其中 n 为字符串 s 的长度。
- **空间复杂度**：$O(n)$。