4月24日日报

本日学习内容

1. 完成hot 100 链表25 138 148 19 普通数组56 189 238 41，动态规划部分139，和写过的题一起总结为博客

今日算法题

题目1： 139. 单词拆分

139. 单词拆分

已解

中等 相关标签 相关企业

给你一个字符串 `s` 和一个字符串列表 `wordDict` 作为字典。如果可以利用字典中出现的一个或多个单词拼接出 `s` 则返回 `true`。

注意：不要求字典中出现的单词全部都使用，并且字典中的单词可以重复使用。

示例 1：

> **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：

> **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。

示例 3：

> **输入：** s = "catsandog", wordDict = ["cats", "dog", "sand", "and", "cat"]
> **输出：** false

提示：

- `1 <= s.length <= 300`
- `1 <= wordDict.length <= 1000`
- `1 <= wordDict[i].length <= 20`
- `s` 和 `wordDict[i]` 仅由小写英文字母组成
- `wordDict` 中的所有字符串 **互不相同**

```cpp
class Solution {
public:
    bool wordBreak(string s, vector<string>& wordDict) {
        unordered_set<string> dict(wordDict.begin(), wordDict.end());
        int n = s.size();
        vector<bool> dp(n + 1, false);
        dp[0] = true; // 空串可以拆分

        for (int i = 1; i <= n; ++i) {
            for (int j = 0; j < i; ++j) {
                if (dp[j] && dict.count(s.substr(j, i - j))) {
                    dp[i] = true;
                    break; // 找到一种拆分方式就可以停止了
                }
            }
        }

        return dp[n];
    }
};
```

题目2：41. 缺失的第一个正数

41. 缺失的第一个正数

已解答

困难 相关标签 相关企业 提示

给你一个未排序的整数数组 `nums` ，请你找出其中没有出现的最小的正整数。

请你实现时间复杂度为 `O(n)` 并且只使用常数级别额外空间的解决方案。

示例 1：

> **输入：** nums = [1,2,0]
> **输出：** 3
> **解释：** 范围 [1,2] 中的数字都在数组中。

示例 2：

> **输入：** nums = [3,4,-1,1]
> **输出：** 2
> **解释：** 1 在数组中，但 2 没有。

示例 3：

> **输入：** nums = [7,8,9,11,12]
> **输出：** 1
> **解释：** 最小的正数 1 没有出现。

提示：

- $1 \le nums.length \le 10^5$
- $-2^{31} \le nums[i] \le 2^{31} - 1$

```cpp
class Solution {
public:
    int firstMissingPositive(vector<int>& nums) {
        int n = nums.size();

        for (int i = 0; i < n; ++i) {
            while (nums[i] > 0 && nums[i] <= n && nums[nums[i] - 1] != nums[i]) {
                swap(nums[i], nums[nums[i] - 1]);
            }
        }

        for (int i = 0; i < n; ++i) {
            if (nums[i] != i + 1) {
                return i + 1;
            }
        }
```

```
            return n + 1;
        }
};
```

明日学习计划：

1. 总结一周的工作，完成周报
2. 写代码随想录二叉树6-10
3. 写hot100二叉树学过的部分