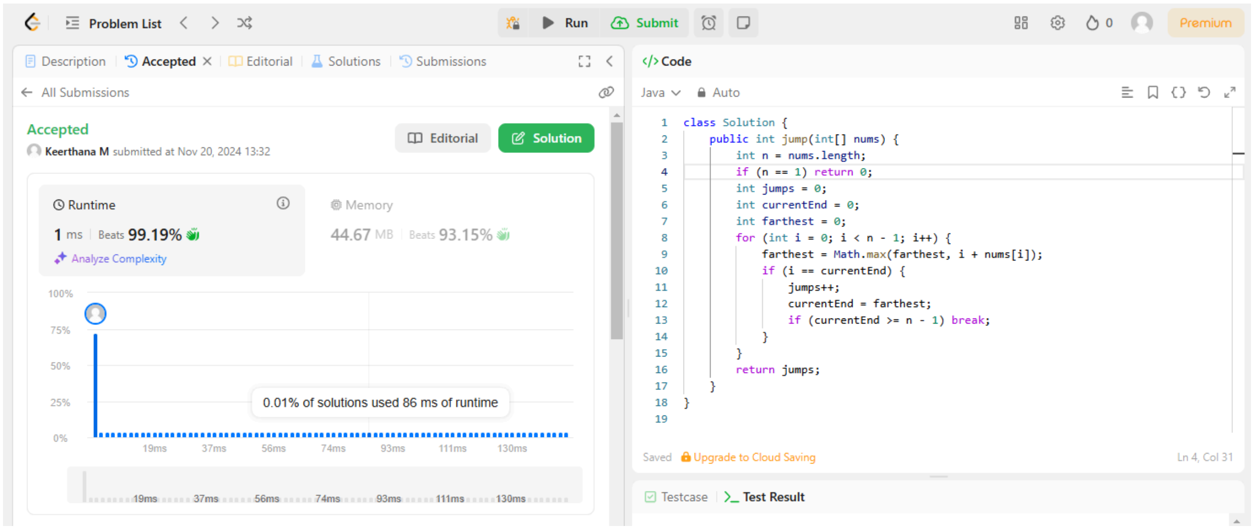
DSA Coding Practice-8

1. Jump Game II:



Accepted

Keerthana M submitted at Nov 20, 2024 13:32

Runtime: 1 ms | Beats 99.19%

Memory: 44.67 MB | Beats 93.15%

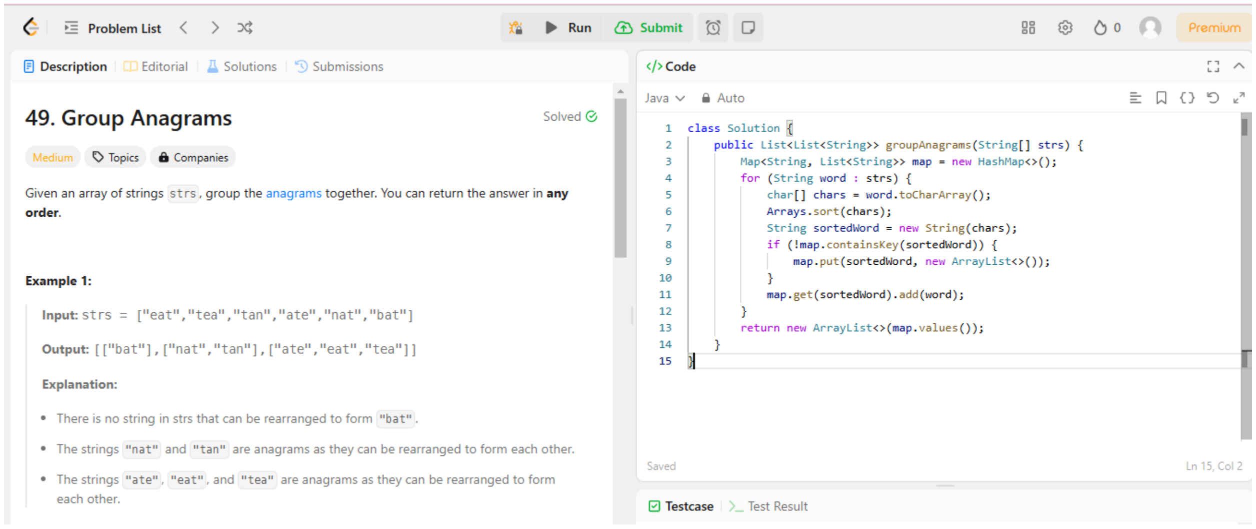
Analyze Complexity

0.01% of solutions used 86 ms of runtime

```java
class Solution {
    public int jump(int[] nums) {
        int n = nums.length;
        if (n == 1) return 0;
        int jumps = 0;
        int currentEnd = 0;
        int farthest = 0;
        for (int i = 0; i < n - 1; i++) {
            farthest = Math.max(farthest, i + nums[i]);
            if (i == currentEnd) {
                jumps++;
                currentEnd = farthest;
                if (currentEnd >= n - 1) break;
            }
        }
        return jumps;
    }
}
```

Saved | Upgrade to Cloud Saving | Ln 4, Col 31

2.Group Anagrams:



49. Group Anagrams — Solved

Medium | Topics | Companies

Given an array of strings `strs`, group the anagrams together. You can return the answer in **any order**.

Example 1:

Input: strs = ["eat","tea","tan","ate","nat","bat"]

Output: [["bat"],["nat","tan"],["ate","eat","tea"]]

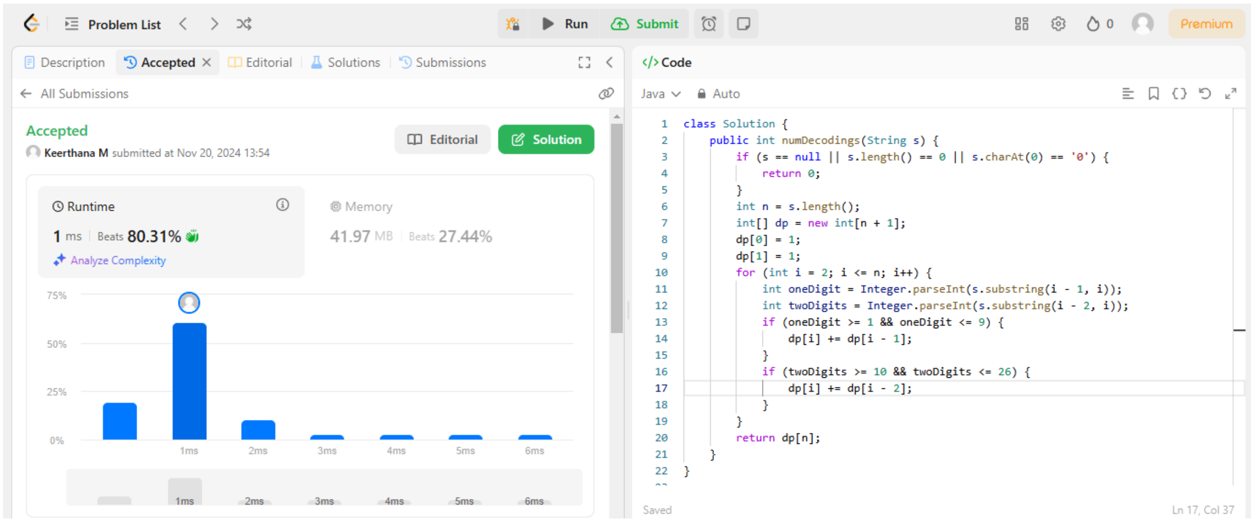
Explanation:

- There is no string in strs that can be rearranged to form `"bat"`.
- The strings `"nat"` and `"tan"` are anagrams as they can be rearranged to form each other.
- The strings `"ate"`, `"eat"`, and `"tea"` are anagrams as they can be rearranged to form each other.

```java
class Solution {
    public List<List<String>> groupAnagrams(String[] strs) {
        Map<String, List<String>> map = new HashMap<>();
        for (String word : strs) {
            char[] chars = word.toCharArray();
            Arrays.sort(chars);
            String sortedWord = new String(chars);
            if (!map.containsKey(sortedWord)) {
                map.put(sortedWord, new ArrayList<>());
            }
            map.get(sortedWord).add(word);
        }
        return new ArrayList<>(map.values());
    }
}
```

Saved | Ln 15, Col 2

3.Decode Ways:



Accepted
Keerthana M submitted at Nov 20, 2024 13:54

Runtime
1 ms | Beats 80.31%

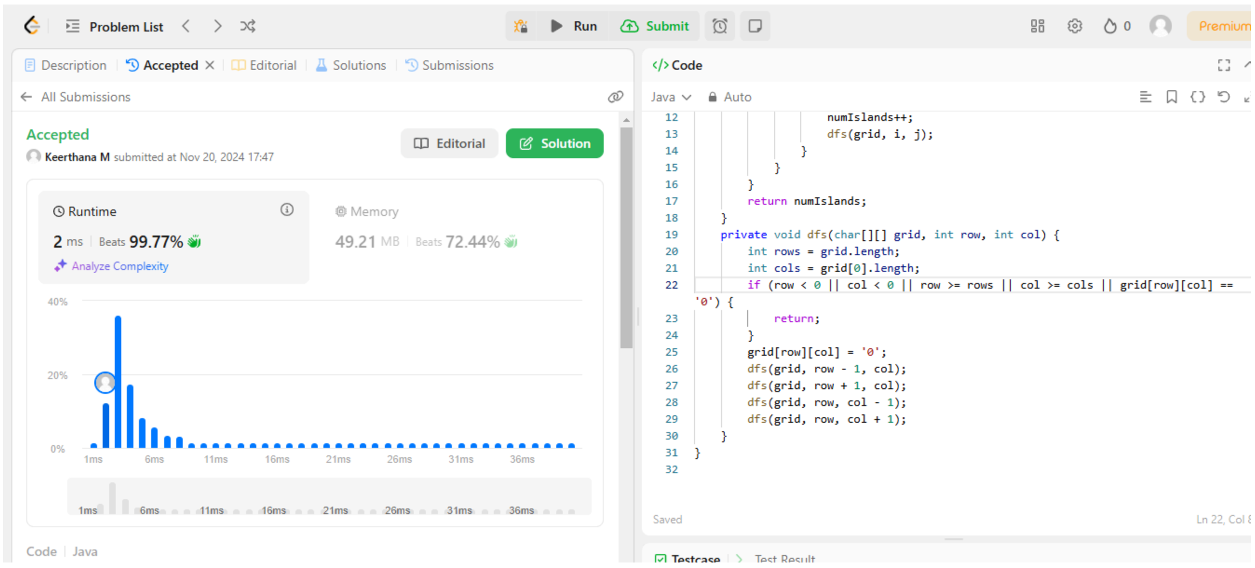
Memory
41.97 MB | Beats 27.44%

```java
class Solution {
    public int numDecodings(String s) {
        if (s == null || s.length() == 0 || s.charAt(0) == '0') {
            return 0;
        }
        int n = s.length();
        int[] dp = new int[n + 1];
        dp[0] = 1;
        dp[1] = 1;
        for (int i = 2; i <= n; i++) {
            int oneDigit = Integer.parseInt(s.substring(i - 1, i));
            int twoDigits = Integer.parseInt(s.substring(i - 2, i));
            if (oneDigit >= 1 && oneDigit <= 9) {
                dp[i] += dp[i - 1];
            }
            if (twoDigits >= 10 && twoDigits <= 26) {
                dp[i] += dp[i - 2];
            }
        }
        return dp[n];
    }
}
```

Ln 17, Col 37

4.Number of Islands:

```java
class Solution {
    public int numIslands(char[][] grid) {
        if (grid == null || grid.length == 0) {
            return 0;
        }
        int numIslands = 0;
        int rows = grid.length;
        int cols = grid[0].length;
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                if (grid[i][j] == '1') {
                    numIslands++;
                    dfs(grid, i, j);
                }
            }
        }
        return numIslands;
    }
    private void dfs(char[][] grid, int row, int col) {
        int rows = grid.length;
        int cols = grid[0].length;
        if (row < 0 || col < 0 || row >= rows || col >= cols || grid[row][col]
== '0') {
            return;
        }
        grid[row][col] = '0';
        dfs(grid, row - 1, col);
        dfs(grid, row + 1, col);
        dfs(grid, row, col - 1);
        dfs(grid, row, col + 1);
    }
}
```



Accepted

Keerthana M submitted at Nov 20, 2024 17:47

Runtime

2 ms | Beats 99.77%

Memory

49.21 MB | Beats 72.44%

Code | Java

```
                numIslands++;
                dfs(grid, i, j);
            }
        }
    }
    return numIslands;
}
private void dfs(char[][] grid, int row, int col) {
    int rows = grid.length;
    int cols = grid[0].length;
    if (row < 0 || col < 0 || row >= rows || col >= cols || grid[row][col] ==
'0') {
        return;
    }
    grid[row][col] = '0';
    dfs(grid, row - 1, col);
    dfs(grid, row + 1, col);
    dfs(grid, row, col - 1);
    dfs(grid, row, col + 1);
}
}
```