

## DSA Practice – 8

### 1.3 Sum Closest

```
class Solution {  
    public int threeSumClosest(int[] nums, int t) {  
        int result=nums[0]+nums[1]+nums[nums.length-1];  
        Arrays.sort(nums);  
        for (int i=0;i<nums.length-2;i++) {  
            int s=i+1,e=nums.length-1;  
            while(s<e) {  
                int sum=nums[i]+nums[s]+nums[e];  
                if(sum>t) e--;  
                else s++;  
                if (Math.abs(sum-t)<Math.abs(result-t)) result=sum;  
            }  
        }  
        return result;  
    }  
}
```

The screenshot displays a code editor interface for a problem titled "3Sum Closest". The left sidebar shows the "Accepted" status and submission details for "Prethika A" submitted on Nov 20, 2024 at 13:50. The "Runtime" section indicates 14 ms with a 77.12% beat rate, and the "Memory" section shows 43.33 MB with a 25.62% beat rate. The main editor area contains the Java code for the solution, which is identical to the code provided in the previous block. The code is syntax-highlighted and includes line numbers. The bottom right corner shows the "Test Result" section, indicating the solution is "Accepted" with a runtime of 0 ms for the test cases.

## 2.Jump Game II

```
class Solution {  
    public int jump(int[] nums) {  
        int n = nums.length;  
        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;  
    }  
}
```

The screenshot shows a code editor interface with the following components:

- Problem List:** Shows the problem name "Jump Game II" and the status "Accepted".
- Code Editor:** Displays the Java code for the solution. The code is as follows:  

```
1 class Solution {  
2     public int jump(int[] nums) {  
3         int n = nums.length;  
4         int jumps = 0;  
5         int currentEnd = 0;  
6         int farthest = 0;  
7         for (int i = 0; i < n - 1; i++) {  
8             farthest = Math.max(farthest, i + nums[i]);  
9             if (i == currentEnd){  
10                 jumps++;  
11                 currentEnd = farthest;  
12                 if (currentEnd >= n - 1) {  
13                     break;  
14                 }  
15             }  
16         }  
17         return jumps;  
18     }  
19 }
```
- Runtime:** Shows "1 ms" and "Beats 99.19%".
- Memory:** Shows "45.07 MB" and "Beats 54.36%".
- Testcase:** Shows "Accepted" and "Runtime: 0 ms".
- Case 1:** Shows "Input" and "Output".

### 3.Group Anagrams

The screenshot shows a LeetCode submission for the 'Group Anagrams' problem. The submission is 'Accepted' and was made by 'Prethika A' on Nov 21, 2024, at 19:05. The runtime is 7 ms, which beats 64.18% of other submissions. The memory usage is 47.84 MB, which beats 44.58% of other submissions. The Java code is as follows:

```
1 class Solution {
2     public List<List<String>> groupAnagrams(String[] strs) {
3         Map<String, List<String>> anagramsMap = new HashMap<>();
4         for (String str : strs) {
5             char[] charArray = str.toCharArray();
6             Arrays.sort(charArray);
7             String sortedStr = new String(charArray);
8             anagramsMap.putIfAbsent(sortedStr, new ArrayList<>());
9             anagramsMap.get(sortedStr).add(str);
10        }
11        return new ArrayList<>(anagramsMap.values());
12    }
13 }
```

The test result shows 'Accepted' with a runtime of 0 ms for all three test cases.

### 4.Best Time to Buy and Sell Stock II

The screenshot shows a LeetCode submission for the 'Best Time to Buy and Sell Stock II' problem. The submission is 'Accepted' and was made by 'Prethika A' on Nov 21, 2024, at 19:24. The runtime is 1 ms, which beats 92.14% of other submissions. The memory usage is 45.74 MB, which beats 47.46% of other submissions. The Java code is as follows:

```
1 class Solution {
2     public int maxProfit(int[] prices) {
3         int maxProfit = 0;
4         for (int i = 1; i < prices.length; i++) {
5             if (prices[i] > prices[i - 1]) {
6                 maxProfit += prices[i] - prices[i - 1];
7             }
8         }
9         return maxProfit;
10    }
11 }
```

The test result shows 'Accepted' with a runtime of 0 ms for all three test cases.