

Leetcode Solution

This screenshot shows the LeetCode submission interface for the problem "Partition Array for Maximum Sum". The left sidebar contains a table of submissions:

Time Submitted	Status	Runtime	Memory	Language
07/31/2022 20:53	Accepted	12 ms	42.7 MB	java
07/31/2022 20:53	Compile Error	N/A	N/A	java
07/31/2022 20:52	Compile Error	N/A	N/A	java
07/31/2022 20:51	Compile Error	N/A	N/A	java

The main area displays the Java code for the solution, which uses a recursive approach with memoization to find the maximum sum of the array after partitioning it into sub-arrays of size at most k.

```
1 class Solution {
2     Integer[] memo;
3     public int maxSumAfterPartitioning(int[] arr, int k) {
4         if (arr == null || arr.length == 0) return 0;
5
6         int n = arr.length;
7         memo = new Integer[n];
8         return dfs(arr, k, 0);
9     }
10
11     private int dfs(int[] arr, int k, int index) {
12         if (index == arr.length)
13             return 0;
14         if (memo[index] != null)
15             return memo[index];
16
17         int local = Integer.MIN_VALUE, max = Integer.MIN_VALUE;
18         for (int i = index; i < arr.length && i < index + k; i++) {
19             local = Math.max(local, arr[i]);
20             max = Math.max(max, local * (i - index + 1) + dfs(arr, k, i + 1));
21         }
22         memo[index] = max;
23         return max;
24     }
25 }
```

The bottom of the interface shows a status bar with system information like temperature (31°C) and time (11:39).

This screenshot shows the LeetCode submission interface for the problem "Is Subsequence". The left sidebar contains a table of submissions:

Time Submitted	Status	Runtime	Memory	Language
07/31/2022 20:51	Accepted	1 ms	42 MB	java

The main area displays the Java code for the solution, which uses a two-pointer technique to check if string t is a subsequence of string s.

```
1 class Solution {
2     public boolean isSubsequence(String s, String t) {
3         if (s.length() == 0)
4             return true;
5         int indexS = 0, indexT = 0;
6         while (indexT < t.length()) {
7             if (s.charAt(indexS) == t.charAt(indexT)) {
8                 indexS++;
9             }
10            indexT++;
11            if (indexS == s.length())
12                return true;
13        }
14        return false;
15    }
16 }
```

The bottom of the interface shows a status bar with system information like temperature (31°C) and time (11:39).