



Experiment No. - 7

Student Name: Vivek Kumar
Branch: BE-CSE(LEET)
Semester: 6th
Subject Name: Competitive coding - II

UID: 21BCS8129
Section/Group: 20BCS-ST-801/B
Date of Performance: 25/04/2023
Subject Code: 20CSP-351

1. Aim/Overview of the practical:

Q.1 Water and Jug Problem.

<https://leetcode.com/problems/water-and-jug-problem/>

2. Apparatus / Simulator Used:

- Windows 7 or above
- Google Chrome

3. Objective:

- To understand the concept of Divide and Conquer.
- To implement the concept of Water and Jug Problem.

4. Code:

```
class Solution {  
    public boolean canMeasureWater(int x, int y, int z) {  
        if(x + y < z)  
            return false;  
        if( x == z || y == z || x + y == z )  
            return true;  
        return z%GCD(x, y) == 0;  
    }  
  
    public int GCD(int a, int b){  
        while(b != 0 ){  
            int temp = b;  
            b = a%b;  
            a = temp;  
        }  
        return a;  
    }  
}
```



5. Result/Output/Writing Summary:

The screenshot shows the LeetCode platform with the Java solution for the Water and Jug Problem. The code uses the Euclidean algorithm to determine if a target capacity can be measured using two jugs of given capacities. It includes examples and constraints. The submission was accepted with a runtime of 0 ms and memory usage of 39.3 MB.

```
class Solution {
    public boolean canMeasureWater(int x, int y, int z) {
        if(x + y < z)
            return false;
        if( x == z || y == z || x + y == z )
            return true;
        return z % GCD(x, y) == 0;
    }
    public int GCD(int a, int b){
        while(b != 0){
            int temp = b;
            b = a % b;
            a = temp;
        }
        return a;
    }
}
```

Example 1:
Input: jug1Capacity = 3, jug2Capacity = 5, targetCapacity = 4
Output: true
Explanation: The famous Die Hard example

Example 2:
Input: jug1Capacity = 2, jug2Capacity = 6, targetCapacity = 5
Output: false

Example 3:
Input: jug1Capacity = 1, jug2Capacity = 2, targetCapacity = 3
Output: true

Constraints:
• $1 \leq \text{jug1Capacity}, \text{jug2Capacity} \leq 10^3$

Testcase Run Code Result Debugger

Accepted Runtime: 0 ms

Your input: 3 5

Output: true

Expected: true

Run Code Submit

This screenshot shows the LeetCode submission details page for the Water and Jug Problem. It displays the accepted status, runtime, memory usage, and language of the submission. The user can also view the code editor, run the code, and access the testcases tab.

Runtime: 0 ms, faster than 100.00% of Java online submissions for Water and Jug Problem.

Memory Usage: 39.3 MB, less than 74.24% of Java online submissions for Water and Jug Problem.

Next challenges:
Freedom Trail, Minesweeper, Shortest Path to Get Food

Show off your acceptance: [Facebook](#) [Twitter](#) [LinkedIn](#)

Time Submitted	Status	Runtime	Memory	Language
04/25/2023 09:55	Accepted	0 ms	39.3 MB	java

Problems Pick One < Prev 365/2654 Next > Console Use Example Testcases Run Code Submit



1. Aim/Overview of the practical:

Q.2 Find and Replacement in String

<https://leetcode.com/problems/find-and-replace-in-string/>

2. Apparatus / Simulator Used:

- Windows 7 or above
- Google Chrome

3. Objective:

- To understand the concept of String.
- To implement the concept of Divide and Conquer.

4. Code:

```
class Solution {  
    public String findReplaceString(String S, int[] indexes, String[] sources, String[] targets) {  
        Map<Integer, Integer> table = new HashMap<>();  
        for (int i=0; i<indexes.length; i++) {  
            if (S.startsWith(sources[i], indexes[i])) {  
                table.put(indexes[i], i);  
            }  
        }  
        StringBuilder sb = new StringBuilder();  
        for (int i=0; i<S.length(); ) {  
            if (table.containsKey(i)) {  
                sb.append(targets[table.get(i)]);  
                i+=sources[table.get(i)].length();  
            } else {  
                sb.append(S.charAt(i));  
                i++;  
            }  
        }  
        return sb.toString();  
    }  
}
```



5. Result/Output/Writing Summary:

The screenshot shows the LeetCode platform with the following details:

- Problem:** 833. Find And Replace In String
- Language:** Java
- Code Snippet:**

```

1+ class Solution {
2+     public String findReplaceString(String S, int[] indexes, String[] sources, String[] targets) {
3+         Map<Integer, Integer> table = new HashMap<>();
4+         for (int i=0; i<indexes.length; i++) {
5+             if (S.startsWith(sources[i], indexes[i])) {
6+                 table.put(indexes[i], i);
7+             }
8+         }
9+         StringBuilder sb = new StringBuilder();
10+        for (int i=0; i<S.length(); ) {
11+            if (table.containsKey(i)) {
12+                i+=sources[table.get(i)].length();
13+            } else {
14+                sb.append(S.charAt(i));
15+                i++;
16+            }
17+        }
18+        return sb.toString();
19+    }
20+ }
```
- Testcase Results:**
 - Accepted: Runtime: 0 ms
 - Your input: "abcd" [0, 2]
 - Output: "eeebffff"
 - Expected: "eeebffff"

The screenshot shows the LeetCode platform with the following details:

- Success Details:** Runtime: 1 ms, faster than 100.00% of Java online submissions for Find And Replace in String.
- Memory Usage:** 42.4 MB, less than 64.75% of Java online submissions for Find And Replace in String.
- Next challenges:**
 - Closest Subsequence Sum
 - Maximum Number of Consecutive Values You Can Make
 - Make the Prefix Sum Non-negative
- Show off your acceptance:** [Facebook](#) [Twitter](#) [LinkedIn](#)
- Submission Details:**

Time Submitted	Status	Runtime	Memory	Language
04/25/2023 09:53	Accepted	1 ms	42.4 MB	java
- Testcase Results:**
 - Accepted: Runtime: 0 ms
 - Your input: "abcd" [0, 2]
 - Output: "eeebffff"
 - Expected: "eeebffff"

Learning outcomes (What I have learnt):

- Learned the concept of Divide and Conquer.
- Learnt about Water and Jug Problem & Find and Replacement in String.