

Experiment No. - 8

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 Remove Duplicate Letter.

<https://leetcode.com/problems/remove-duplicate-letters/>

2. Apparatus / Simulator Used:

- Windows 7 or above
- Google Chrome

3. Objective:

- To understand the concept of Greedy Approach.
- To implement the concept of Remove Duplicate Letter from String.

4. Code:

```
class Solution {
    public String removeDuplicateLetters(String S) {
        Stack<Integer> stack = new Stack<>();
        int[] last = new int[26], seen = new int[26];
        for (int i = 0; i < S.length(); ++i)
            last[S.charAt(i) - 'a'] = i;
        for (int i = 0; i < S.length(); ++i) {
            int c = S.charAt(i) - 'a';
            if (seen[c]++ > 0) continue;
            while (!stack.isEmpty() && stack.peek() > c && i < last[stack.peek()])
                seen[stack.pop()] = 0;
            stack.push(c);
        }
        StringBuilder sb = new StringBuilder();
        for (int i : stack) sb.append((char)('a' + i));
        return sb.toString();
    }
}
```

5. Result/Output/Writing Summary:

The screenshot shows the LeetCode interface for the problem "Remove Duplicate Letters". The problem description states: "Given a string *s*, remove duplicate letters so that every letter appears once and only once. You must make sure your result is the **smallest in lexicographical order** among all possible results."

Example 1:
Input: *s* = "bcabc"
Output: "abc"

Example 2:
Input: *s* = "cbacdcbc"
Output: "acdb"

Constraints:

- $1 \leq s.length \leq 10^4$
- s* consists of lowercase English letters.

Note: This question is the same as 1081: <https://leetcode.com/problems/smallest-subsequence-of-distinct-characters/>

Accepted: 228,140 | Submissions: 505,809

Seen this question in a real interview before? ☐ Yes ☐ No

Companies: [Google](#) [Amazon](#) [Facebook](#) [Microsoft](#) [Apple](#) [LinkedIn](#) [Uber](#) [Twitter](#) [Netflix](#) [Airbnb](#) [Dropbox](#) [Instagram](#) [Pinterest](#) [Slack](#) [Zoom](#) [Twitter](#) [Facebook](#) [Google](#) [Amazon](#) [Microsoft](#) [Apple](#) [LinkedIn](#) [Uber](#) [Twitter](#) [Netflix](#) [Airbnb](#) [Dropbox](#) [Instagram](#) [Pinterest](#) [Slack](#) [Zoom](#)

Related Topics: [Stack](#) [String](#) [Greedy](#)

Similar Questions: [Remove Duplicate Letters II](#) [Smallest Subsequence of Distinct Characters](#) [Shortest Palindrome](#)

Java Solution:

```
class Solution {
    public String removeDuplicateLetters(String S) {
        Stack<Integer> stack = new Stack<>();
        int[] last = new int[26];
        for (int i = 0; i < S.length(); ++i) {
            last[S.charAt(i) - 'a'] = i;
        }
        for (int i = 0; i < S.length(); ++i) {
            int c = S.charAt(i) - 'a';
            if (seen[c] > 0) continue;
            while (!stack.isEmpty() && stack.peek() > c && i < last[stack.peek()]) {
                seen[stack.pop()] = 0;
            }
            stack.push(c);
            seen[c] = 1;
        }
        StringBuilder sb = new StringBuilder();
        for (int i : stack) sb.append((char)('a' + i));
        return sb.toString();
    }
}
```

Testcase: Run Code Result: Debbuger

Accepted Runtime: 0 ms

Your input: "bcabc"

Output: "abc"

Expected: "abc"

Diff

Run Code Submit

The screenshot shows the LeetCode submission page for the problem "Remove Duplicate Letters". The submission was successful.

Success Details

Runtime: 2 ms, faster than 98.92% of Java online submissions for Remove Duplicate Letters.

Memory Usage: 42.4 MB, less than 51.78% of Java online submissions for Remove Duplicate Letters.

Next challenges: [Smallest K-Length Subsequence With Occurrences of a Letter](#)

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

Time Submitted	Status	Runtime	Memory	Language
04/26/2023 10:49	Accepted	2 ms	42.4 MB	java
04/26/2023 10:39	Accepted	3 ms	42.7 MB	java

Java Solution:

```
class Solution {
    public String removeDuplicateLetters(String S) {
        Stack<Integer> stack = new Stack<>();
        int[] last = new int[26];
        for (int i = 0; i < S.length(); ++i) {
            last[S.charAt(i) - 'a'] = i;
        }
        for (int i = 0; i < S.length(); ++i) {
            int c = S.charAt(i) - 'a';
            if (seen[c] > 0) continue;
            while (!stack.isEmpty() && stack.peek() > c && i < last[stack.peek()]) {
                seen[stack.pop()] = 0;
            }
            stack.push(c);
            seen[c] = 1;
        }
        StringBuilder sb = new StringBuilder();
        for (int i : stack) sb.append((char)('a' + i));
        return sb.toString();
    }
}
```

Testcase: Run Code Result: Debbuger

Accepted Runtime: 0 ms

Your input: "bcabc"

Output: "abc"

Expected: "abc"

Diff

Run Code Submit

1. Aim/Overview of the practical:

Q.2 Assign Cookies.

<https://leetcode.com/problems/assign-cookies/>

2. Apparatus / Simulator Used:

- Windows 7 or above
- Google Chrome

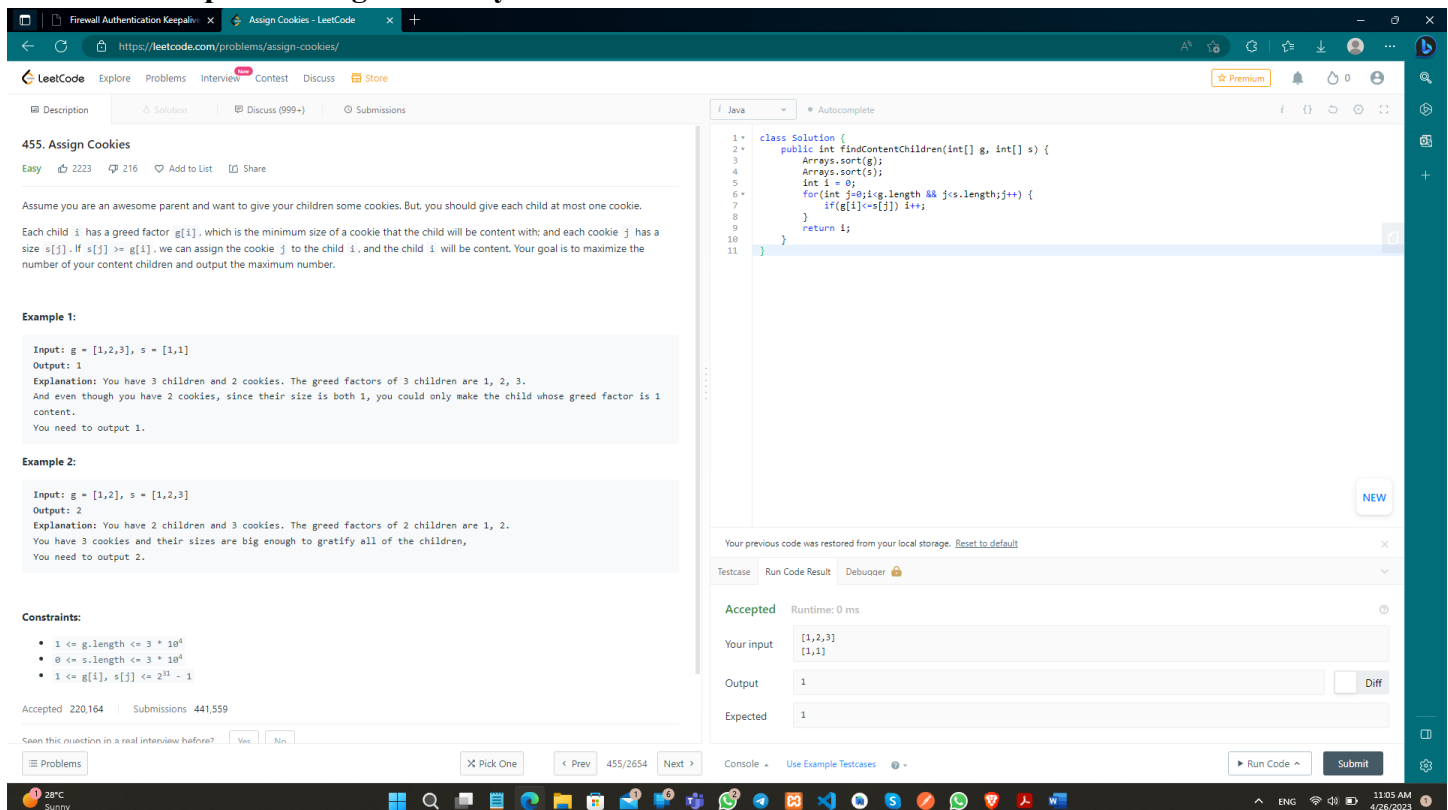
3. Objective:

- To understand the concept of Greedy Algorithm.
- To implement the concept of Assign Cookies.

4. Code:

```
class Solution {
    public int findContentChildren(int[] g, int[] s) {
        Arrays.sort(g);
        Arrays.sort(s);
        int i = 0;
        for(int j=0;j<g.length && j<s.length;j++) {
            if(g[i]<=s[j]) i++;
        }
        return i;
    }
}
```

5. Result/Output/Writing Summary:



The screenshot shows the LeetCode interface for the problem "455. Assign Cookies". The problem description states: "Assume you are an awesome parent and want to give your children some cookies. But, you should give each child at most one cookie. Each child i has a greed factor $g[i]$, which is the minimum size of a cookie that the child will be content with; and each cookie j has a size $s[j]$. If $s[j] \geq g[i]$, we can assign the cookie j to the child i , and the child i will be content. Your goal is to maximize the number of your content children and output the maximum number."

Example 1:
Input: $g = [1,2,3]$, $s = [1,1]$
Output: 1
Explanation: You have 3 children and 2 cookies. The greed factors of 3 children are 1, 2, 3. And even though you have 2 cookies, since their size is both 1, you could only make the child whose greed factor is 1 content. You need to output 1.

Example 2:
Input: $g = [1,2]$, $s = [1,2,3]$
Output: 2
Explanation: You have 2 children and 3 cookies. The greed factors of 2 children are 1, 2. You have 3 cookies and their sizes are big enough to gratify all of the children, You need to output 2.

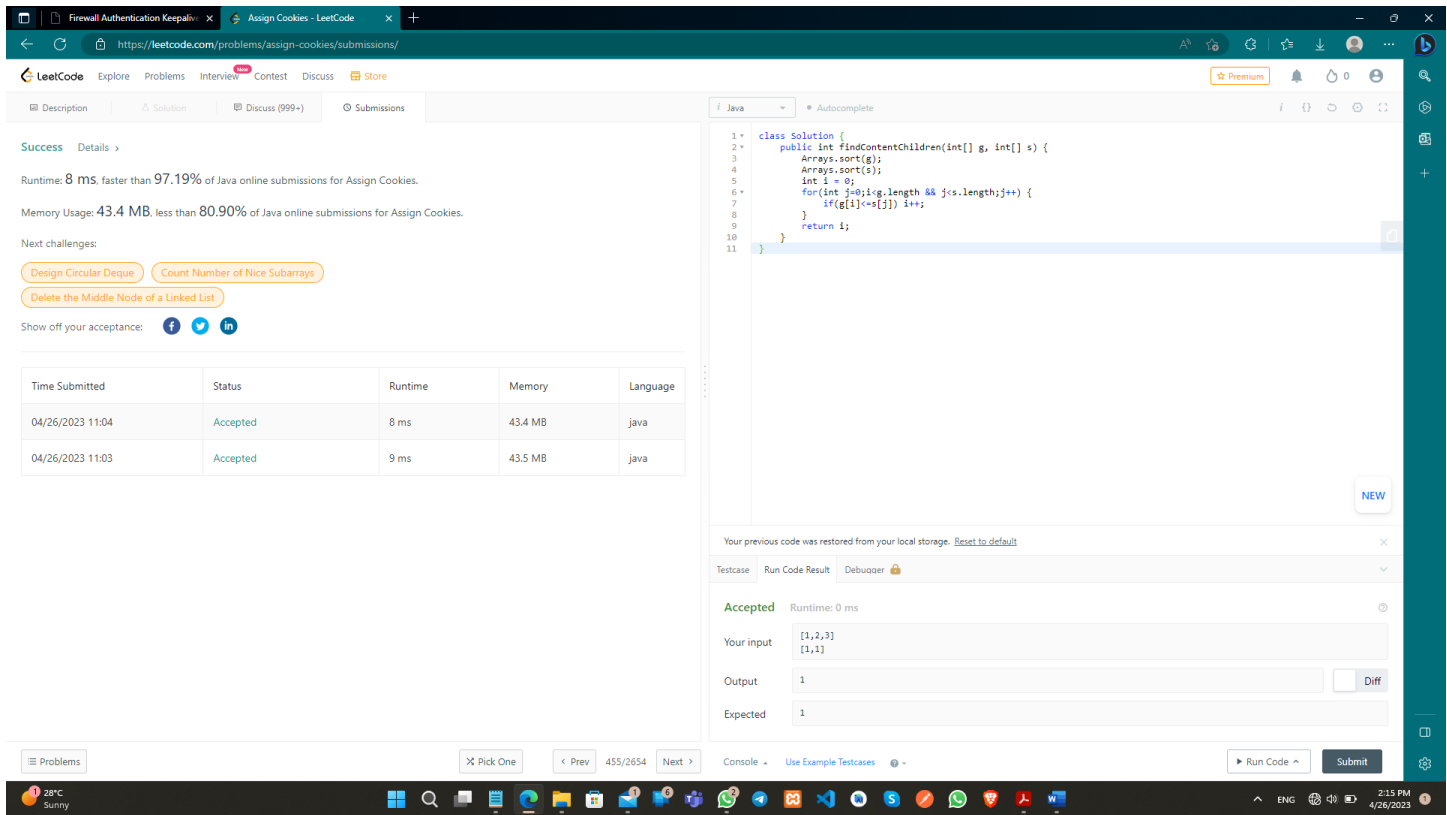
Constraints:

- $1 \leq g.length \leq 3 \times 10^4$
- $0 \leq s.length \leq 3 \times 10^4$
- $1 \leq g[i], s[j] \leq 2^{31} - 1$

The solution code in Java is as follows:

```
class Solution {
    public int findContentChildren(int[] g, int[] s) {
        Arrays.sort(g);
        Arrays.sort(s);
        int i = 0;
        for(int j=0;j<g.length && j<s.length;j++) {
            if(g[i]<=s[j]) i++;
        }
        return i;
    }
}
```

The test results show that the code is "Accepted" with a runtime of 0 ms. The input is $[1,2,3]$ and $[1,1]$, the output is 1, and the expected result is 1.



Success Details >

Runtime: 8 ms, faster than 97.19% of Java online submissions for Assign Cookies.

Memory Usage: 43.4 MB, less than 80.90% of Java online submissions for Assign Cookies.

Next challenges:

- Design Circular Deque
- Count Number of Nice Subarrays
- Delete the Middle Node of a Linked List

Show off your acceptance: [f](#) [t](#) [in](#)

Time Submitted	Status	Runtime	Memory	Language
04/26/2023 11:04	Accepted	8 ms	43.4 MB	java
04/26/2023 11:03	Accepted	9 ms	43.5 MB	java

```

1 class Solution {
2     public int findContentChildren(int[] g, int[] s) {
3         Arrays.sort(g);
4         Arrays.sort(s);
5         int i = 0;
6         for(int j=0; j<g.length && j<s.length; j++) {
7             if(g[j]<=s[j]) i++;
8         }
9         return i;
10    }
11 }

```

Your previous code was restored from your local storage. [Reset to default](#)

Testcase Run Code Result Debuquer

Accepted Runtime: 0 ms

Your input: [1,2,3], [1,1]

Output: 1

Expected: 1

Run Code Submit

Learning outcomes (What I have learnt):

- Learned the concept of Greedy Algorithm.
- Learnt about Remove Duplicate Letter & Assign Cookies problems.