



Experiment 7

Student Name: Aman Gokul

UID: 20BCS5449

Branch: BE CSE

Section/Group: 607/A

Semester: 6th

Date of Performance: 21/04/2023

Subject Name: CC-2 Lab

Subject Code: 20CSP-351

1. Aim/Overview of the practical:

1-bit and 2-bit Characters

We have two special characters:

- The first character can be represented by one bit 0.
- The second character can be represented by two bits (10 or 11).

Given a binary array bits that ends with 0, return true if the last character must be a one-bit character.

<https://leetcode.com/problems/1-bit-and-2-bit-characters/>

2. Apparatus / Simulator Used:

- Windows 7 or above
- Google Chrome

3. Objective:

- To understand the concept of Divide and Conquer.

A divide-and-conquer algorithm recursively breaks down a problem into two or more sub-problems of the same or related type, until these become simple enough to be solved directly. The solutions to the sub-problems are then combined to give a solution to the original problem.

4. Code:

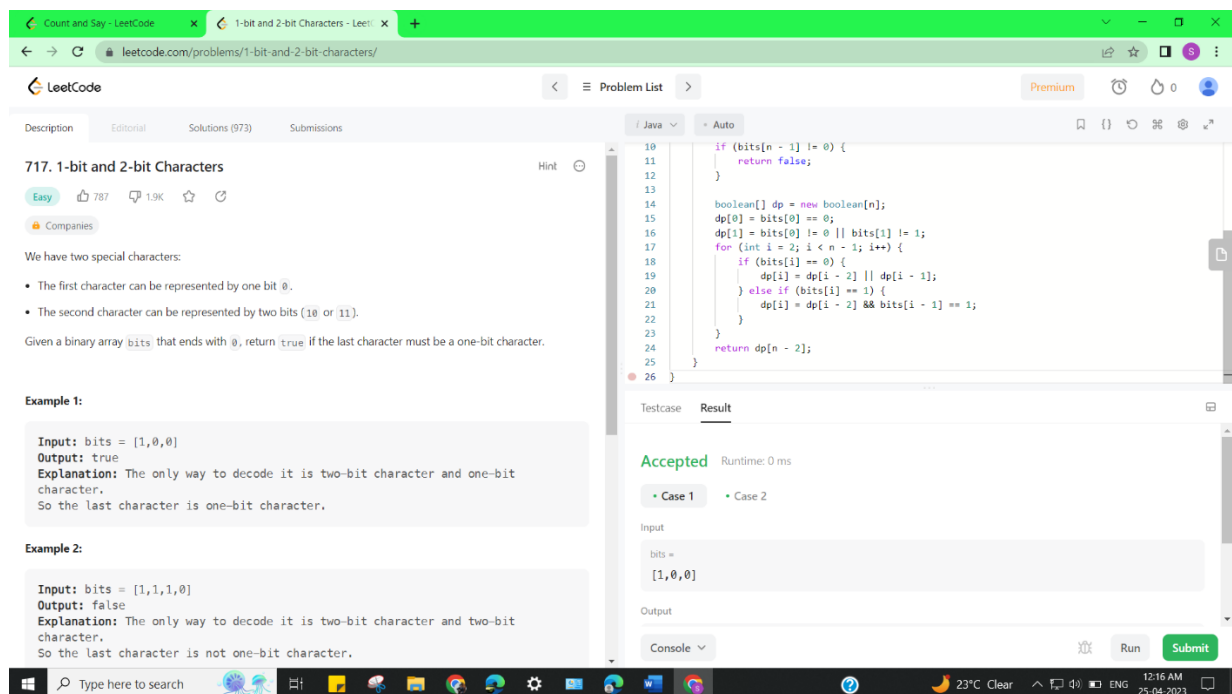
```
class Solution {
    public boolean isOneBitCharacter(int[] bits) {
        if (bits == null || bits.length == 0) {
            return false;
        }
        int n = bits.length;
        if (n == 1) {
            return bits[0] == 0;
        }
        if (bits[n - 1] != 0) {
            return false;
        }
        boolean[] dp = new boolean[n];
        dp[0] = bits[0] == 0;
        dp[1] = bits[0] != 0 || bits[1] != 1;
        for (int i = 2; i < n - 1; i++) {
            if (bits[i] == 0) {
```

```

        dp[i] = dp[i - 2] || dp[i - 1];
    } else if (bits[i] == 1) {
        dp[i] = dp[i - 2] && bits[i - 1] == 1;
    }
}
return dp[n - 2];
}
}

```

4. Result/Output/Writing Summary:



Count and Say - LeetCode | 1-bit and 2-bit Characters - LeetCode

leetcode.com/problems/1-bit-and-2-bit-characters/

LeetCode

Problem List

717. 1-bit and 2-bit Characters

Easy 787 1.9K

Companies

We have two special characters:

- The first character can be represented by one bit 0.
- The second character can be represented by two bits (10 or 11).

Given a binary array `bits` that ends with 0, return `true` if the last character must be a one-bit character.

Example 1:

Input: `bits = [1,0,0]`
Output: `true`
Explanation: The only way to decode it is two-bit character and one-bit character. So the last character is one-bit character.

Example 2:

Input: `bits = [1,1,1,0]`
Output: `false`
Explanation: The only way to decode it is two-bit character and two-bit character. So the last character is not one-bit character.

```

10  if (bits[n - 1] != 0) {
11      return false;
12  }
13
14  boolean[] dp = new boolean[n];
15  dp[0] = bits[0] == 0;
16  dp[1] = bits[0] != 0 || bits[1] != 1;
17  for (int i = 2; i < n - 1; i++) {
18      if (bits[i] == 0) {
19          dp[i] = dp[i - 2] || dp[i - 1];
20      } else if (bits[i] == 1) {
21          dp[i] = dp[i - 2] && bits[i - 1] == 1;
22      }
23  }
24  return dp[n - 2];
25  }
26  }

```

Testcase Result

Accepted Runtime: 0 ms

Case 1 Case 2

Input

bits =

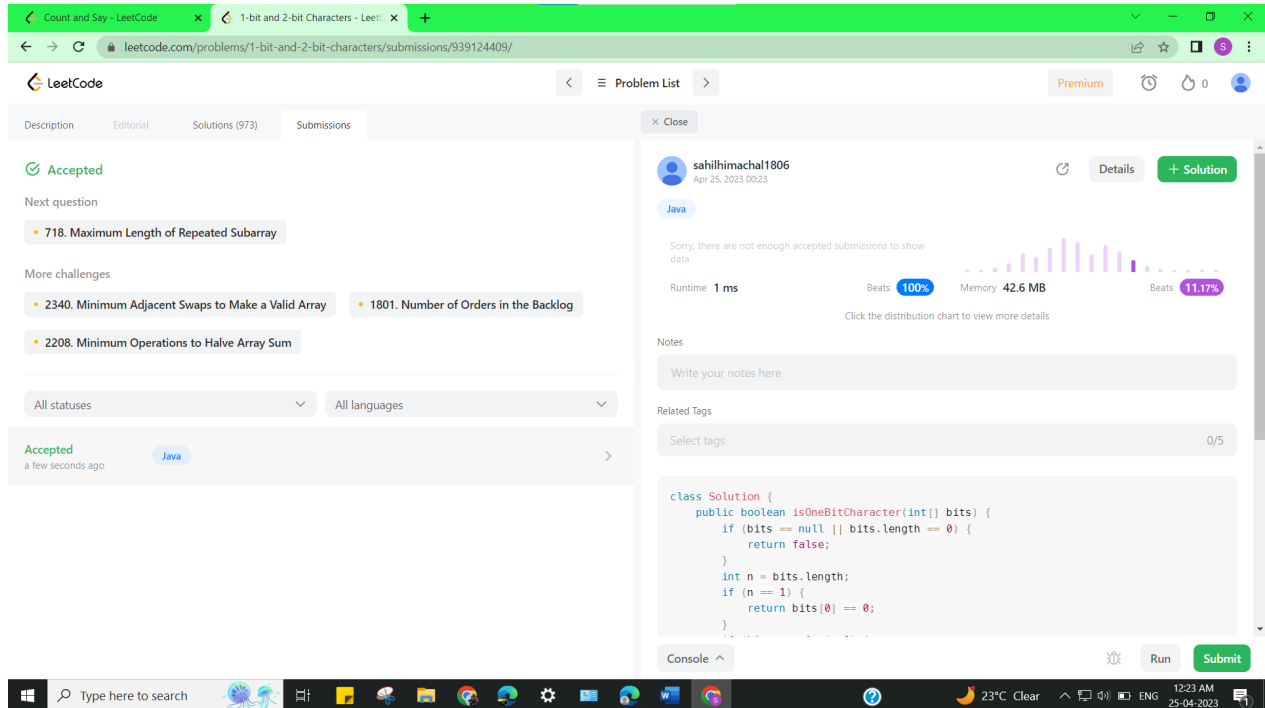
[1, 0, 0]

Output

Console

Run Submit

23°C Clear 12:16 AM 25-04-2023



Experiment 7.2

1. Aim/Overview of the practical:

Count and Say

The **count-and-say** sequence is a sequence of digit strings defined by the recursive formula:

- $\text{countAndSay}(1) = "1"$
- $\text{countAndSay}(n)$ is the way you would "say" the digit string from $\text{countAndSay}(n-1)$, which is then converted into a different digit string.

<https://leetcode.com/problems/count-and-say/>

2. Apparatus / Simulator Used:

- Windows 7 or above
- Google Chrome

3. Objective:

- To understand the concept of Divide and Conquer.

A divide-and-conquer algorithm recursively breaks down a problem into two or more sub-problems of the same or related type, until these become simple enough to be solved directly. The solutions to the sub-problems are then combined to give a solution to the original problem.

4. Code:

```
#include<bits/stdc++.h>
using namespace std;
class Solution {
public:
    string countAndSay(int n) {
        if(n==1){
```

```

        return "1";
    }
    string prev=countAndSay(n-1);
    cout<<n<<" "<<prev<<"\n";

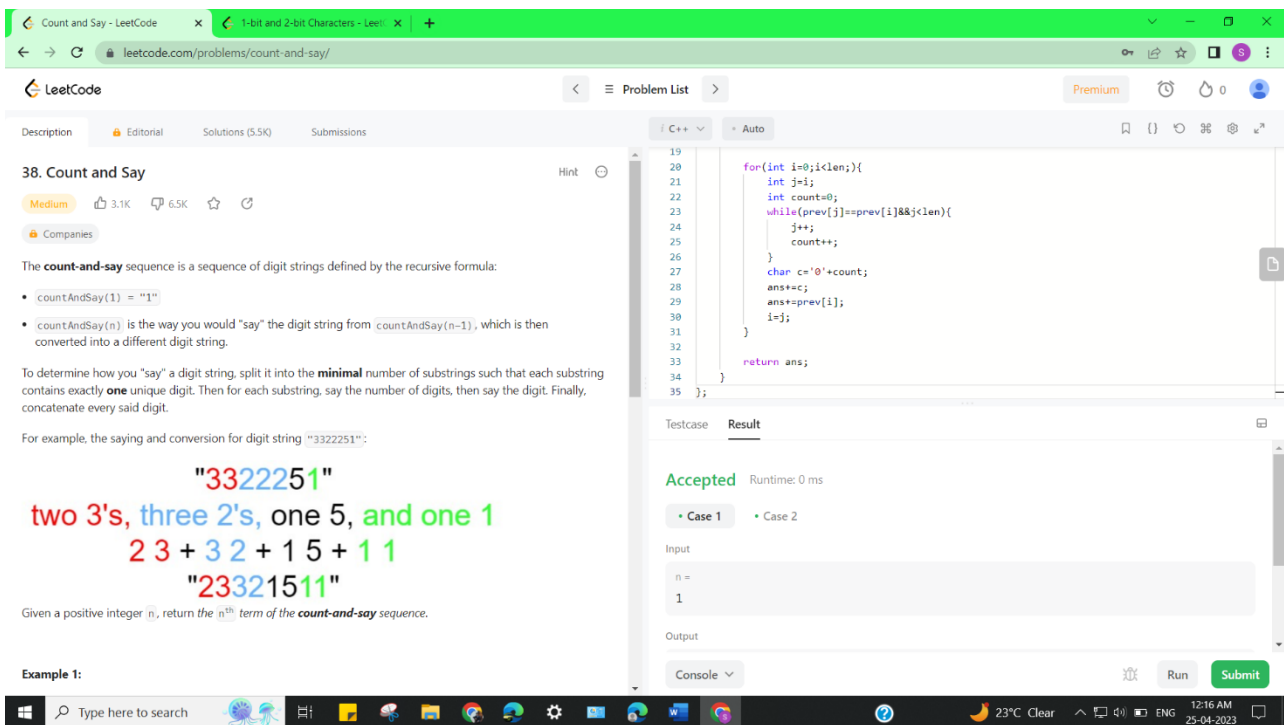
    string ans;
    int len=prev.length();

    for(int i=0;i<len;){
        int j=i;
        int count=0;
        while(prev[j]==prev[i]&&j<len){
            j++;
            count++;
        }
        char c='0'+count;
        ans+=c;
        ans+=prev[i];
        i=j;
    }

    return ans;
}
};

```

5. Result/Output/Writing Summary:



The screenshot displays the LeetCode interface for problem 38, "Count and Say". The problem description explains that the sequence is defined by a recursive formula: $\text{countAndSay}(1) = "1"$ and $\text{countAndSay}(n)$ is the way you would "say" the digit string from $\text{countAndSay}(n-1)$. An example shows the string "3322251" being said as "two 3's, three 2's, one 5, and one 1", resulting in the string "23321511".

The solution code in C++ is shown on the right, implementing the logic described in the problem statement. The code uses a loop to iterate through the previous string, counting consecutive identical digits and appending the count followed by the digit to the current string.

The test case results show that the solution is "Accepted" with a runtime of 0 ms. The input is $n = 1$ and the output is "1".



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Count and Say - LeetCode

1-bit and 2-bit Characters - LeetCode

leetcode.com/problems/count-and-say/submissions/939123786/

LeetCode

Problem List

Premium

Accepted

Next question

39. Combination Sum

More challenges

271. Encode and Decode Strings

443. String Compression

All statuses

All languages

Accepted

a few seconds ago

C++

sahilhimachal1806

Apr 25, 2023 00:22

Details

+ Solution

C++

Runtime 4 ms

Beats 85.34%

Memory 6.6 MB

Beats 60.41%

Click the distribution chart to view more details

Notes

Write your notes here

Related Tags

Select tags

0/5

```
#include<bits/stdc++.h>
using namespace std;

// @lc code=start
class Solution {
public:
    string countAndSay(int n) {
        // base case
        if(n==1){
            return "1";
        }
        string s = "1";
        for(int i=1; i<n; i++){
            string t = "";
            int count = 1;
            for(int j=1; j<s.length(); j++){
                if(s[j] == s[j-1]){
                    count++;
                } else {
                    t += to_string(count) + s[j-1];
                    count = 1;
                }
            }
            t += to_string(count) + s[s.length()-1];
            s = t;
        }
        return s;
    }
};

// @lc code=end
```

Console

Run

Submit

Type here to search

23°C Clear

12:22 AM

25-04-2023

Learning outcomes (What I have learnt):

- Learned the concept of Divide and Conquer.