“**Experiment1.2”**

**Student Name: SUMIT KUMAR UID: 20BCS8226**

**Branch: CSE Section/Group: DM-720 A**

**Semester: 6th Date of Performance: 24-02-23**

**Subject Name: Competitive Coding-II Subject Code: 20CSP-351**

**Aim:**

**To demonstrate the concept of String Matching algorithms.**

**Objective:**

• The objective is to build problem solving capability and to learn the basic concepts of data structures.

• The implementation of Rotate String problem brushes up the concept of string matching.

• The implementation of Repeated String Match problem brushes up the concept of strings and string matching.

**Problem 1: “Rotate String”**

<https://leetcode.com/problems/rotate-string/>

Given two strings s and goal, return true if and only if s can become goal after some number of shifts on s.

A shift on s consists of moving the leftmost character of s to the rightmost position.

For example, if s = "abcde", then it will be "bcdea" after one shift.

**Code:**

class Solution {

public:

    bool rotateString(string s, string goal) {

        queue<char> q1, q2;         // SC: O(n)

        for(int i=0;i<s.size();i++) q1.push(s[i]);

        for(int i=0;i<goal.size();i++) q2.push(goal[i]);

        for(int i=0;i<s.size();i++){        // TC: O(n)

            char ch= q1.front();

            q1.pop();

            q1.push(ch);

            if(q1==q2) return true;

        }

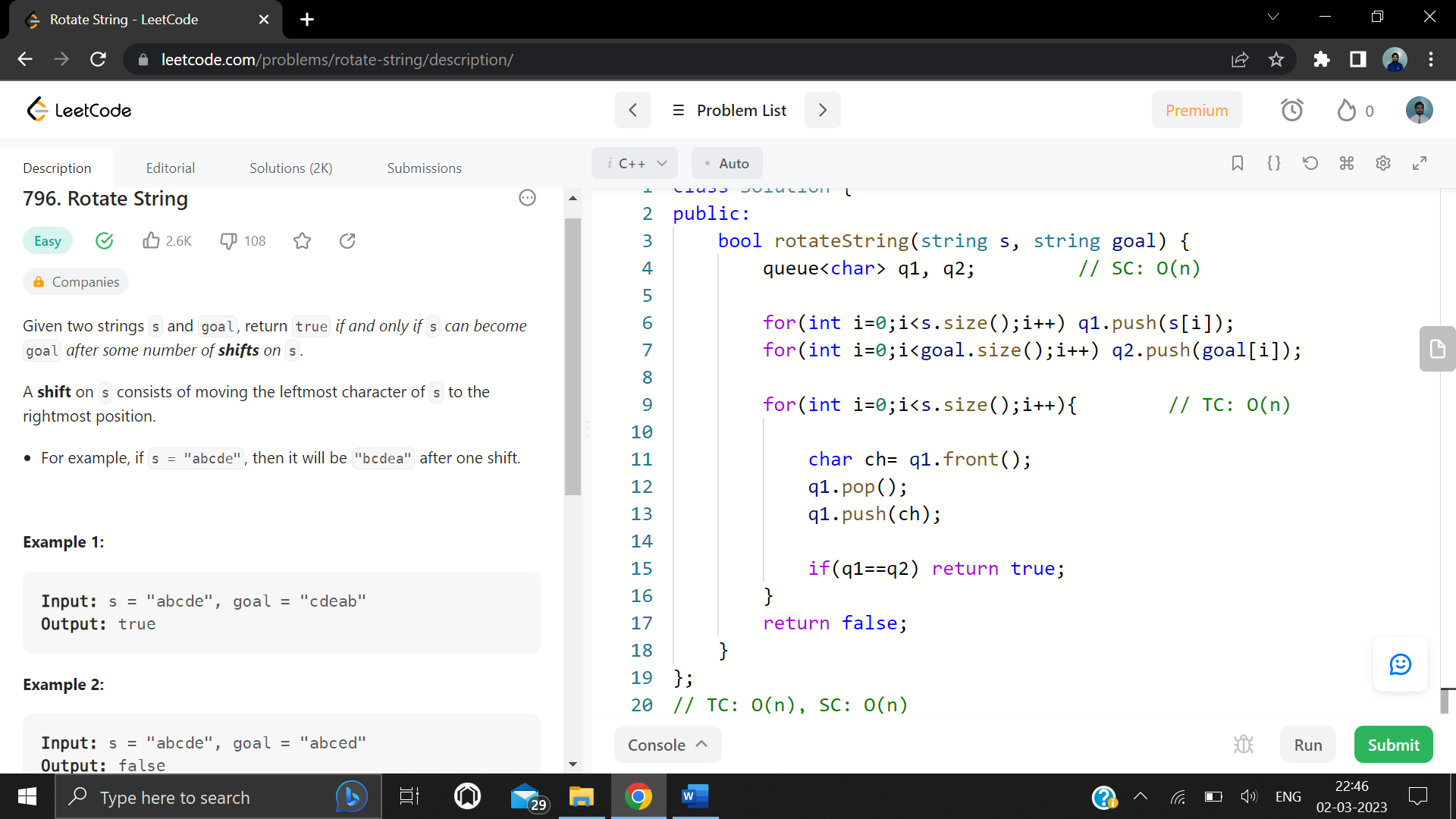
        return false;

    }

};

// TC: O(n), SC: O(n)

**Output:**



**Problem 2: “Repeated String Match”**

<https://leetcode.com/problems/repeated-string-match/>

Given two strings a and b, return the minimum number of times you should repeat string a so that string b is a substring of it. If it is impossible for b​​​​​​ to be a substring of a after repeating it, return -1.

Notice: string "abc" repeated 0 times is "", repeated 1 time is "abc" and repeated 2 times is "abcabc".

**Code:**

class Solution {

public:

    int repeatedStringMatch(string a, string b) {

        int cnt= 1;

        string temp= a;

        while(a.size() < b.size()){

            a += temp;

            cnt++;

        }

        if(a.find(b)!=-1) return cnt;

        a += temp;

        if(a.find(b)!=-1) return cnt+1;

        return -1;

    }

};

**Output:**

