

## Experiment No. - 6

**Student Name:** Vivek Kumar

**Branch:** BE-CSE(LEET)

**Semester:** 6<sup>th</sup>

**Subject Name:** Competitive coding - II

**UID:** 21BCS8129

**Section/Group:** 20BCS-ST-801/B

**Date of Performance:** 11/04/2023

**Subject Code:** 20CSP-351

### 1. Aim/Overview of the practical:

#### Q.1 Is Graph Bipartite.

<https://leetcode.com/problems/is-graph-bipartite/>

### 2. Apparatus / Simulator Used:

- Windows 7 or above
- Google Chrome

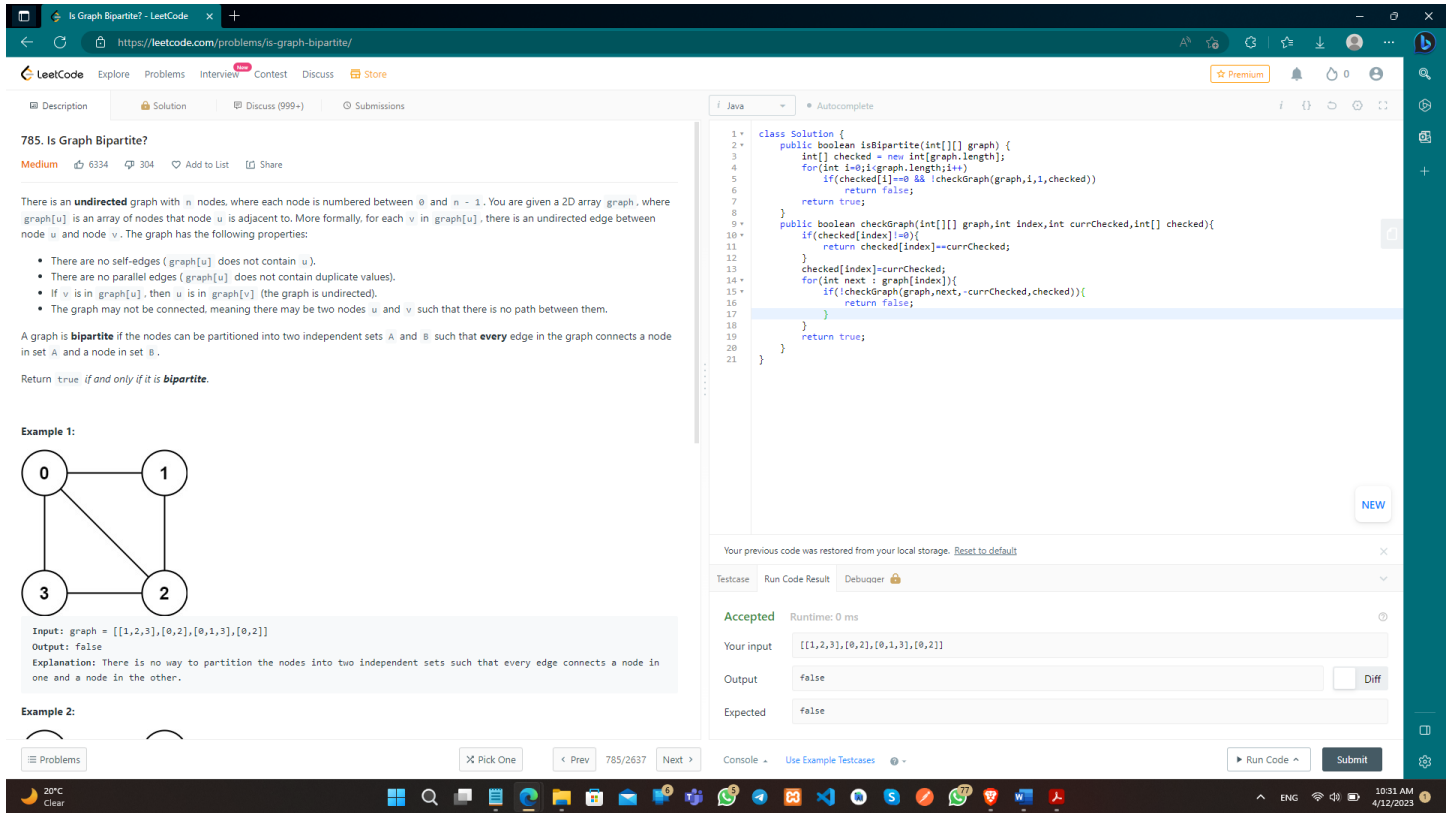
### 3. Objective:

- To understand the concept of Graph
- To implement the concept of Bipartite Graph.

### 4. Code:

```
class Solution {
    public boolean isBipartite(int[][] graph) {
        int[] checked = new int[graph.length];
        for(int i=0;i<graph.length;i++)
            if(checked[i]==0 && !checkGraph(graph,i,1,checked))
                return false;
        return true;
    }
    public boolean checkGraph(int[][] graph,int index,int currChecked,int[] checked){
        if(checked[index]!=0){
            return checked[index]==currChecked;
        }
        checked[index]=currChecked;
        for(int next : graph[index]){
            if(!checkGraph(graph,next,-currChecked,checked)){
                return false;
            }
        }
        return true;
    }
}
```

### 5. Result/Output/Writing Summary:



**785. Is Graph Bipartite?**  
Medium 6334 304 Add to List Share

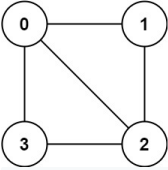
There is an **undirected** graph with  $n$  nodes, where each node is numbered between  $0$  and  $n - 1$ . You are given a 2D array `graph`, where `graph[u]` is an array of nodes that node `u` is adjacent to. More formally, for each `v` in `graph[u]`, there is an undirected edge between node `u` and node `v`. The graph has the following properties:

- There are no self-edges (`graph[u]` does not contain `u`).
- There are no parallel edges (`graph[u]` does not contain duplicate values).
- If `v` is in `graph[u]`, then `u` is in `graph[v]` (the graph is undirected).
- The graph may not be connected, meaning there may be two nodes `u` and `v` such that there is no path between them.

A graph is **bipartite** if the nodes can be partitioned into two independent sets `A` and `B` such that **every** edge in the graph connects a node in set `A` and a node in set `B`.


Return `true` if and only if it is **bipartite**.

**Example 1:**



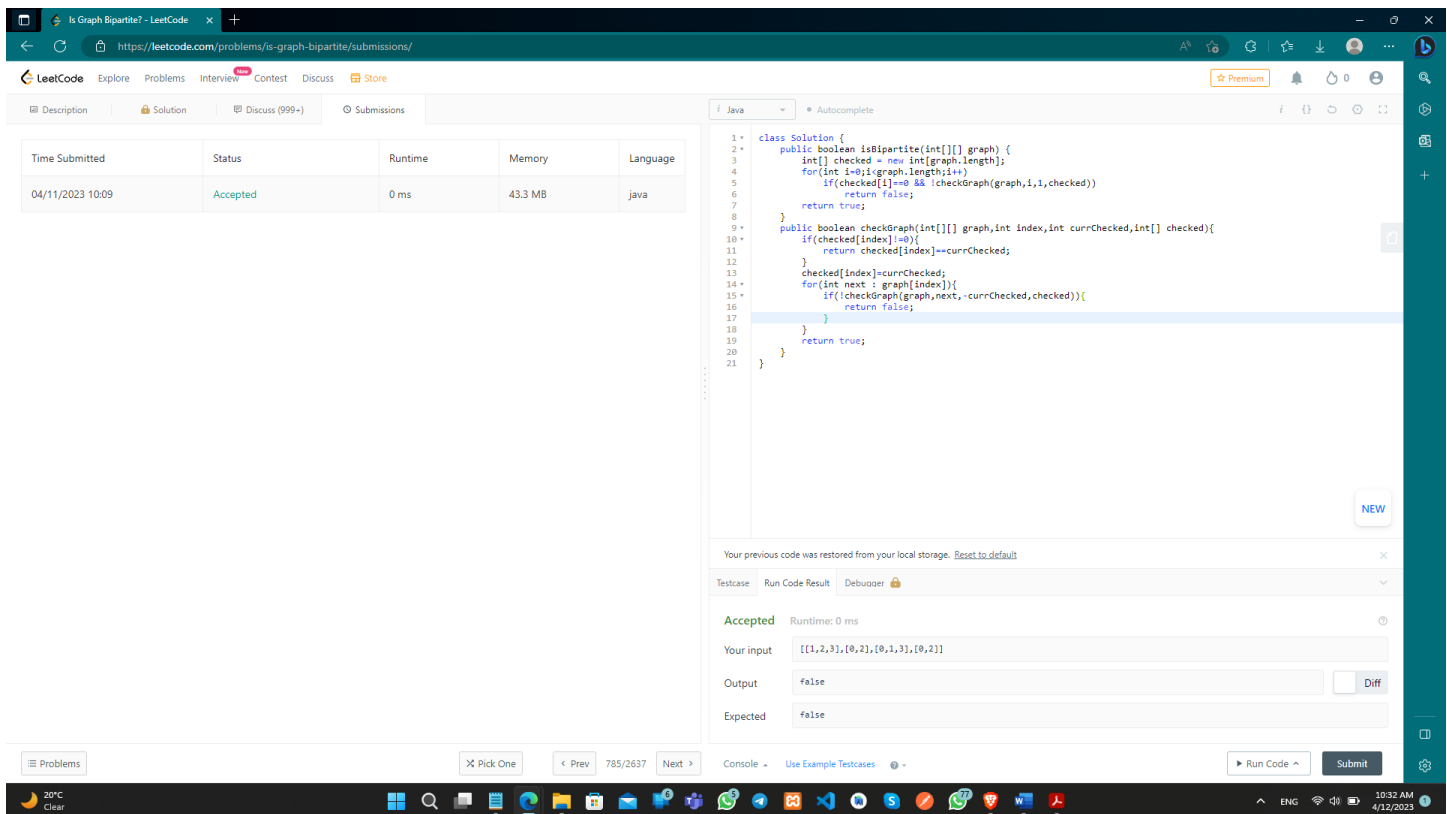
Input: `graph = [[1,2,3],[0,2],[0,1,3],[0,2]]`  
Output: `false`  
Explanation: There is no way to partition the nodes into two independent sets such that every edge connects a node in one and a node in the other.

**Example 2:**



**Java Solution:**

```
class Solution {
    public boolean isBipartite(int[][] graph) {
        int[] checked = new int[graph.length];
        for(int i=0;i<graph.length;i++)
            if(checked[i]==0 && !checkGraph(graph,i,checked))
                return false;
        return true;
    }
    public boolean checkGraph(int[][] graph,int index,int currChecked,int[] checked){
        if(checked[index]!=0){
            return checked[index]==currChecked;
        }
        checked[index]=currChecked;
        for(int next : graph[index]){
            if(!checkGraph(graph,next,-currChecked,checked)){
                return false;
            }
        }
        return true;
    }
}
```



**Submissions**

Time Submitted	Status	Runtime	Memory	Language
04/11/2023 10:09	Accepted	0 ms	43.3 MB	java

**Java Solution:**

```
class Solution {
    public boolean isBipartite(int[][] graph) {
        int[] checked = new int[graph.length];
        for(int i=0;i<graph.length;i++)
            if(checked[i]==0 && !checkGraph(graph,i,checked))
                return false;
        return true;
    }
    public boolean checkGraph(int[][] graph,int index,int currChecked,int[] checked){
        if(checked[index]!=0){
            return checked[index]==currChecked;
        }
        checked[index]=currChecked;
        for(int next : graph[index]){
            if(!checkGraph(graph,next,-currChecked,checked)){
                return false;
            }
        }
        return true;
    }
}
```

## 1. Aim/Overview of the practical:

### Q.2 Find The Difference

<https://leetcode.com/problems/find-the-difference/>

## 2. Apparatus / Simulator Used:

- Windows 7 or above
- Google Chrome

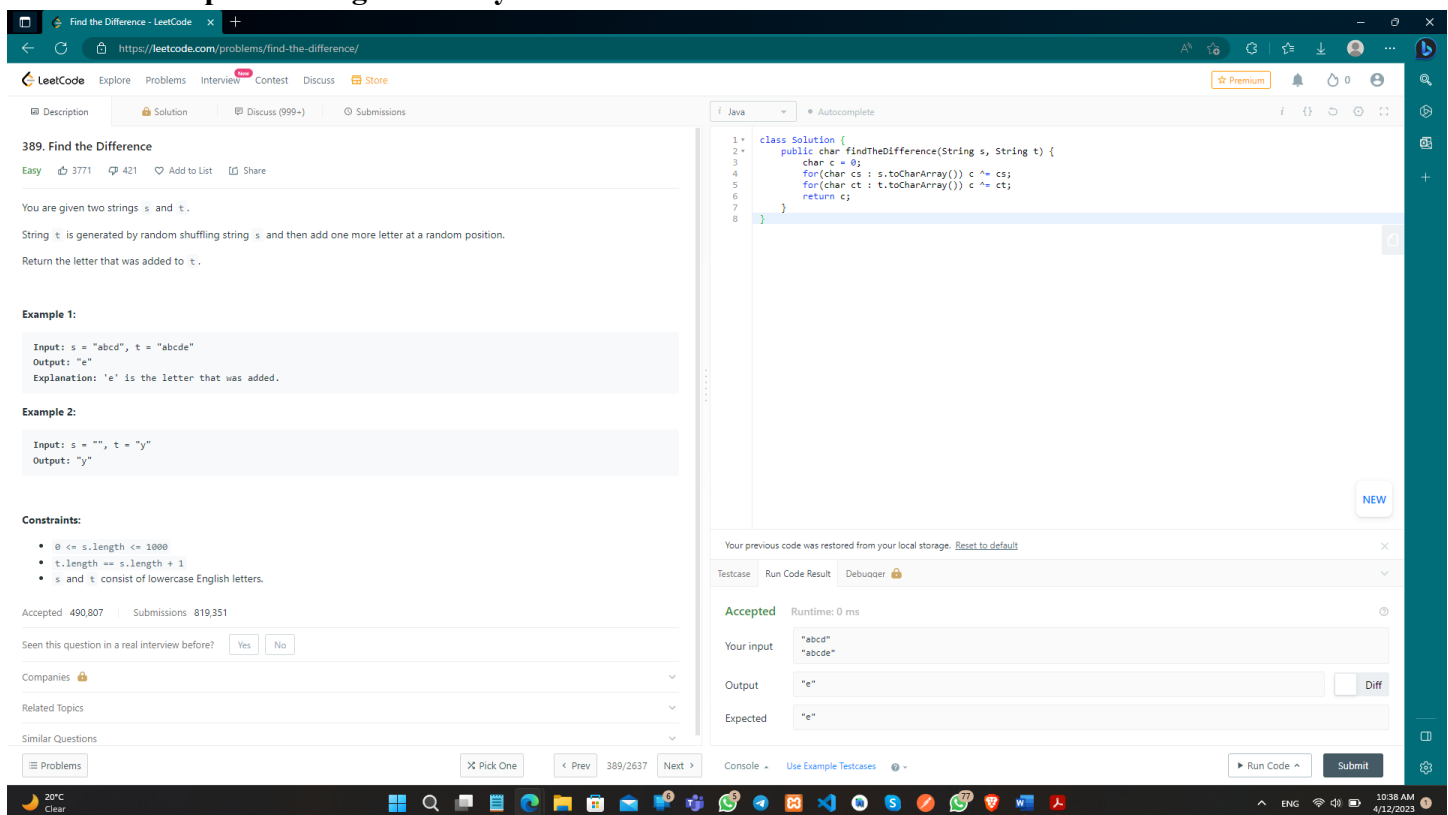
## 3. Objective:

- To understand the concept of String.
- To implement the concept of XOR with String.

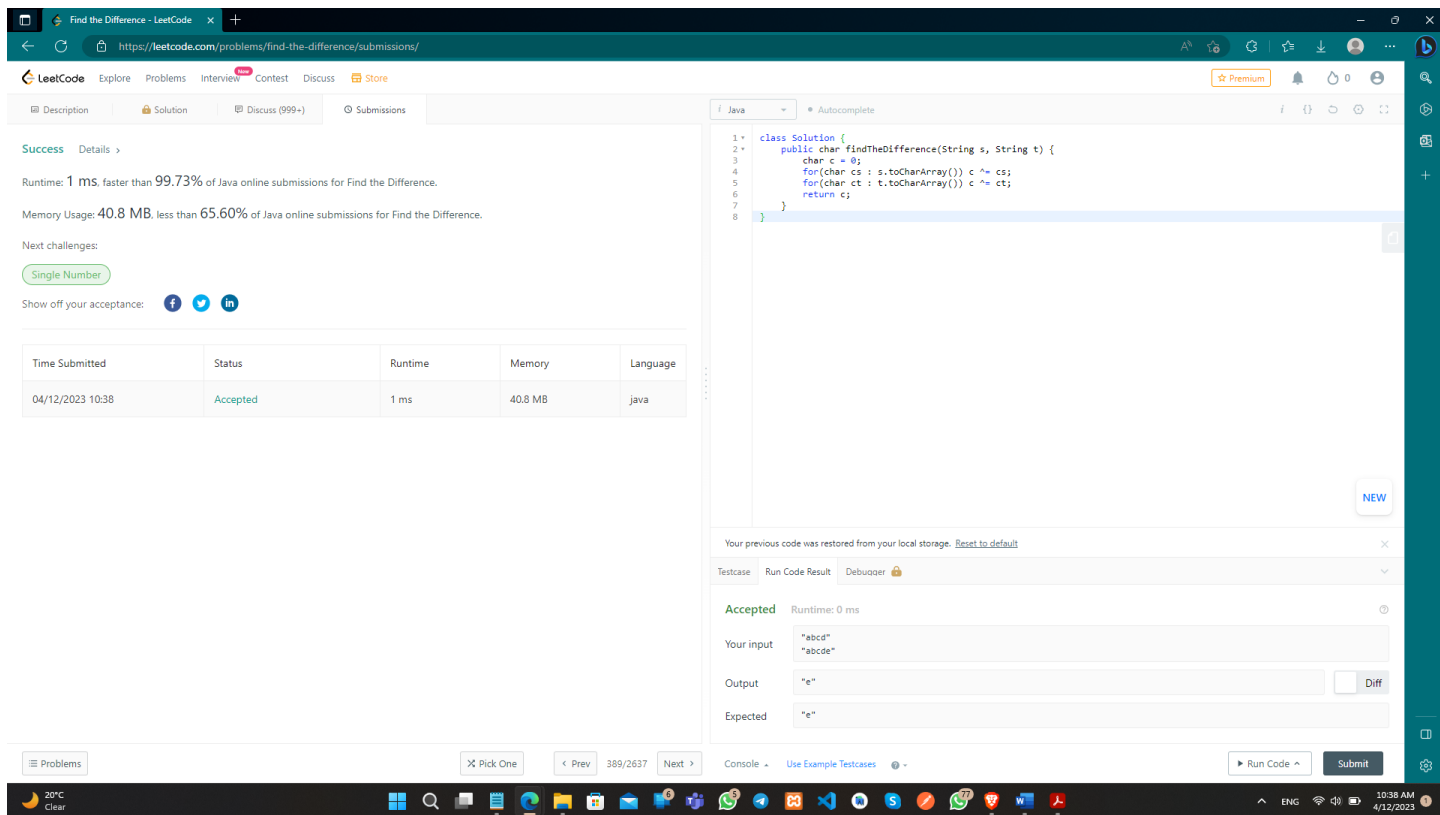
## 4. Code:

```
class Solution {
    public char findTheDifference(String s, String t) {
        char c = 0;
        for(char cs : s.toCharArray()) c ^= cs;
        for(char ct : t.toCharArray()) c ^= ct;
        return c;
    }
}
```

## 5. Result/Output/Writing Summary:



The screenshot shows the LeetCode interface for the problem "Find the Difference". The problem description states: "You are given two strings s and t. String t is generated by random shuffling string s and then add one more letter at a random position. Return the letter that was added to t." Example 1: Input: s = "abcd", t = "abcde", Output: "e". Example 2: Input: s = "", t = "y", Output: "y". Constraints: 0 <= s.length <= 1000, t.length == s.length + 1, s and t consist of lowercase English letters. The solution code in Java is shown in the editor, and the test results show "Accepted" with a runtime of 0 ms.



Find the Difference - LeetCode

https://leetcode.com/problems/find-the-difference/submissions/

LeetCode Explore Problems Interview Contest Discuss Store

Description Solution Discuss (999+) Submissions

Success Details

Runtime: 1 ms, faster than 99.73% of Java online submissions for Find the Difference.

Memory Usage: 40.8 MB, less than 65.60% of Java online submissions for Find the Difference.

Next challenges: Single Number

Show off your acceptance: f t in

Time Submitted	Status	Runtime	Memory	Language
04/12/2023 10:38	Accepted	1 ms	40.8 MB	java

```

1 class Solution {
2     public char findTheDifference(String s, String t) {
3         char c = 0;
4         for(char cs : s.toCharArray()) c ^= cs;
5         for(char ct : t.toCharArray()) c ^= ct;
6         return c;
7     }
8 }

```

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

Testcase Run Code Result Debugger

Accepted Runtime: 0 ms

Your input: "abcd", "abcde"

Output: "e"

Expected: "e"

Run Code Submit

20°C Clear 10:38 AM 4/12/2023

### Learning outcomes (What I have learnt):

- Learned the concept of Bipartite Graph.
- Learnt about Tree and finding the Extra Character in String.