DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

Experiment-6

Student Name: Anshuman Singh **UID: 20BCS2665**
Branch: CSE **Section/Group: 902/A**
Semester: 6th **Date of Performance: 06-04-2023**
Subject Name: Competitive Coding II **Subject Code: 20CSP-351**

Aim: **To demonstrate the concept of Graph.**

Problem1: Find if Path Exists in Graph

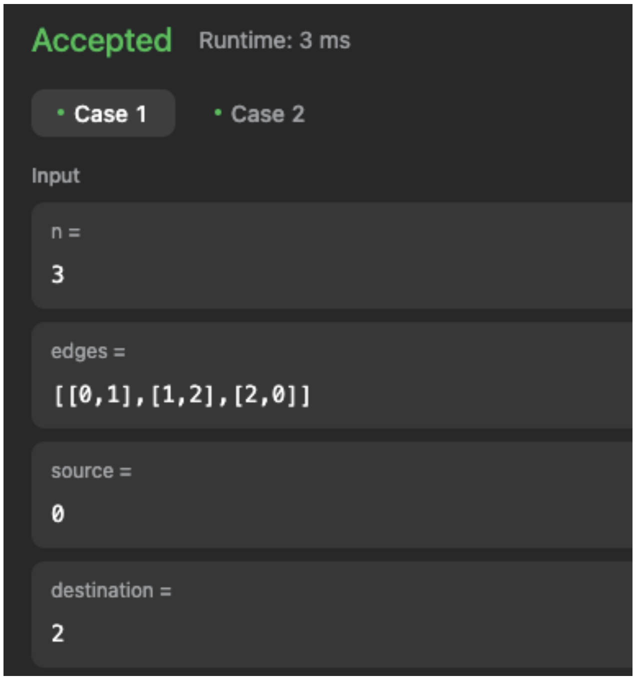
There is a bi-directional graph with n vertices, where each vertex is labeled from 0 to n - 1 (inclusive). The edges in the graph are represented as a 2D integer array edges, where each edges[i] = [ui, vi] denotes a bi-directional edge between vertex ui and vertex vi. Every vertex pair is connected by at most one edge, and no vertex has an edge to itself.
You want to determine if there is a valid path that exists from vertex source to vertex destination.
Given edges and the integers n, source, and destination, return true if there is a valid path from source to destination, or false otherwise.

Code:-

```
class Solution {
public:
int ans=0;
void dfs(int node,vector<int>&vis,vector<int>adj[],int d){
if(vis[node]==1){
return;
}
if(node==d){
ans=1;
return;
}
vis[node]=1;
for(auto i:adj[node]){
if(!vis[i]){
dfs(i,vis,adj,d);
}
}
}
bool validPath(int n, vector<vector<int>>& edges, int source, int destination) {
vector<int>vis(n+1,0);
vector<int>adj[n];
for(auto i:edges){
adj[i[0]].push_back(i[1]);
adj[i[1]].push_back(i[0]);
}
dfs(source,vis,adj,destination);

return ans;
}
};
```

Output:-



Accepted Runtime: 3 ms

• Case 1 • Case 2

Input

n =
3

edges =
[[0,1],[1,2],[2,0]]

source =
0

destination =
2

<u>Probem2:</u> Predict the winner

You are given an integer array nums. Two players are playing a game with this array: player 1 and player 2.

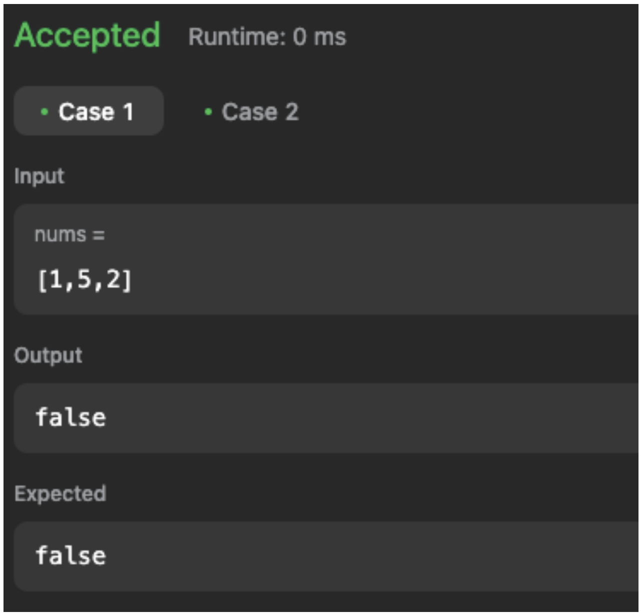
Player 1 and player 2 take turns, with player 1 starting first. Both players start the game with a score of 0. At each turn, the player takes one of the numbers from either end of the array (i.e., nums[0] or nums[nums.length - 1]) which reduces the size of the array by 1. The player adds the chosen number to their score. The game ends when there are no more elements in the array.

Return true if Player 1 can win the game. If the scores of both players are equal, then player 1 is still the winner, and you should also return true. You may assume that both players are playing optimally.

Code:-

```
class Solution {
public:
    bool PredictTheWinner(vector<int>& nums) {
        int n = nums.size();
        if(n % 2 == 0) return true;
        // dp[i][j] = the maximum score that the first player can get given the nums are nums[i..j]
        vector<vector<int>> dp(n, vector<int>(n));
        // sums[i] = sum of the first i numbers
        vector<int> sums(n + 1);
        for(int i = 0; i < n; ++i){
            sums[i + 1] = sums[i] + nums[i];
            dp[i][i] = nums[i];
        }
        for(int step = 1; step < n; ++step){
            for(int i = 0; i + step < n; ++i){
                int j = i + step;
                dp[i][j] = sums[j + 1] - sums[i] - min(dp[i + 1][j], dp[i][j - 1]);
            }
        }
        return 2 * dp[0][n - 1] >= sums[n];
    }
};
```

Output:-



Accepted Runtime: 0 ms

Input

nums =

```
[1,5,2]
```

Output

```
false
```

Expected

```
false
```