

Experiment No. - 9

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: 02/05/2023

Subject Code: 20CSP-351

1. Aim/Overview of the practical:

Q.1 All Path from Source to Target.

<https://leetcode.com/problems/all-paths-from-source-to-target/>

2. Apparatus / Simulator Used:

- Windows 7 or above
- Google Chrome

3. Objective:

- To understand the concept of Backtracking.
- To implement the concept of All path from source to target.

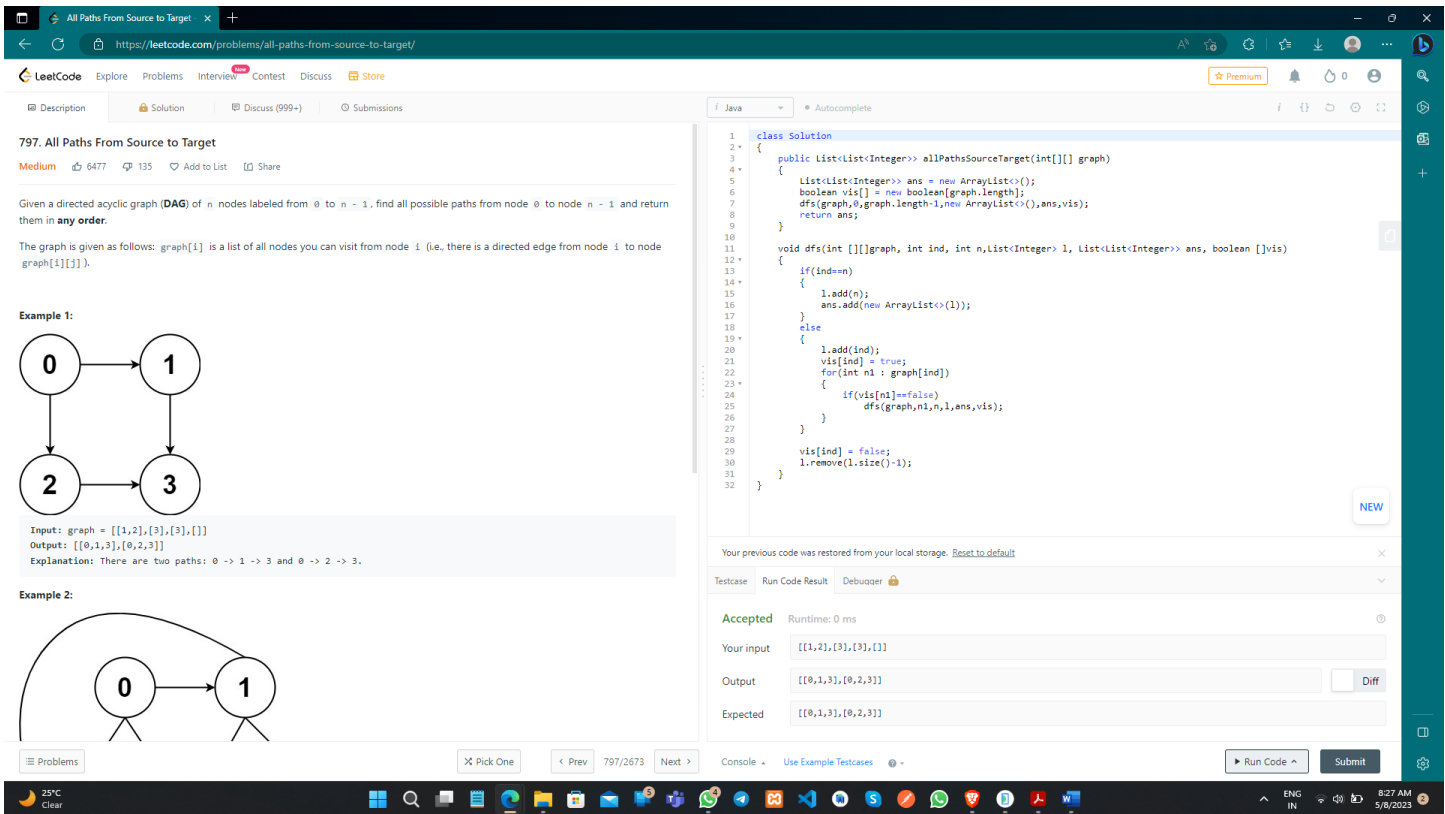
4. Code:

```
class Solution{
    public List<List<Integer>> allPathsSourceTarget(int[][] graph) {
        List<List<Integer>> ans = new ArrayList<>();
        boolean vis[] = new boolean[graph.length];
        dfs(graph,0,graph.length-1,new ArrayList<>(),ans,vis);
        return ans;
    }

    void dfs(int [][]graph, int ind, int n,List<Integer> l, List<List<Integer>> ans, boolean []vis){
        if(ind==n){
            l.add(n);
            ans.add(new ArrayList<>(l));
        }
        Else{
            l.add(ind);
            vis[ind] = true;
            for(int n1 : graph[ind]){
                if(vis[n1]==false)
                    dfs(graph,n1,n,l,ans,vis);
            }
        }

        vis[ind] = false;
        l.remove(l.size()-1);
    }
}
```

5. Result/Output/Writing Summary:



797. All Paths From Source to Target

Medium 6477 135 Add to List Share

Given a directed acyclic graph (DAG) of n nodes labeled from 0 to $n - 1$, find all possible paths from node 0 to node $n - 1$ and return them in **any order**.

The graph is given as follows: `graph[i]` is a list of all nodes you can visit from node i (i.e., there is a directed edge from node i to node `graph[i][j]`).

Example 1:

```

graph LR
    0((0)) --> 1((1))
    0((0)) --> 2((2))
    1((1)) --> 3((3))
    2((2)) --> 3((3))

```

Input: `graph = [[1,2],[3],[3],[]]`
Output: `[[0,1,3],[0,2,3]]`
Explanation: There are two paths: $0 \rightarrow 1 \rightarrow 3$ and $0 \rightarrow 2 \rightarrow 3$.

Example 2:

```

graph LR
    0((0)) --> 1((1))
    1((1)) --> 1((1))

```

Solution (Java):

```

class Solution {
    public List<List<Integer>> allPathsSourceTarget(int[][] graph) {
        List<List<Integer>> ans = new ArrayList<>();
        boolean vis[] = new boolean[graph.length];
        dfs(graph, 0, graph.length - 1, new ArrayList<>(), ans, vis);
        return ans;
    }

    void dfs(int[][] graph, int ind, int n, List<Integer> l, List<List<Integer>> ans, boolean vis[]) {
        if (ind == n) {
            l.add(n);
            ans.add(new ArrayList<>());
        } else {
            l.add(ind);
            vis[ind] = true;
            for (int n1 : graph[ind]) {
                if (vis[n1] == false) {
                    dfs(graph, n1, n, l, ans, vis);
                }
            }
            vis[ind] = false;
            l.remove(l.size() - 1);
        }
    }
}

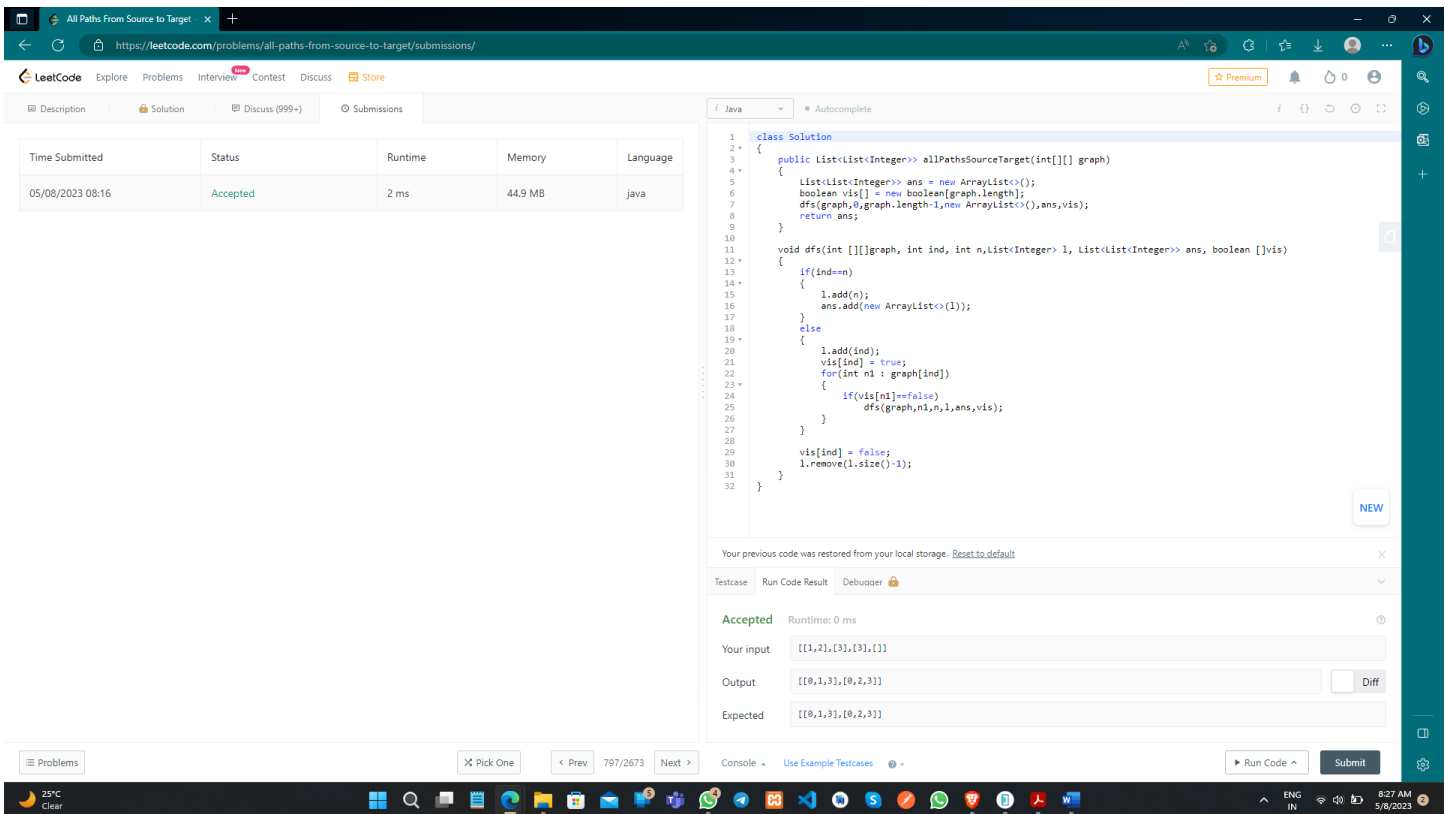
```

Accepted Runtime: 0 ms

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

Output: `[[0,1,3],[0,2,3]]`

Expected: `[[0,1,3],[0,2,3]]`



Submissions

Time Submitted	Status	Runtime	Memory	Language
05/08/2023 08:16	Accepted	2 ms	44.9 MB	java

Solution (Java):

```

class Solution {
    public List<List<Integer>> allPathsSourceTarget(int[][] graph) {
        List<List<Integer>> ans = new ArrayList<>();
        boolean vis[] = new boolean[graph.length];
        dfs(graph, 0, graph.length - 1, new ArrayList<>(), ans, vis);
        return ans;
    }

    void dfs(int[][] graph, int ind, int n, List<Integer> l, List<List<Integer>> ans, boolean vis[]) {
        if (ind == n) {
            l.add(n);
            ans.add(new ArrayList<>());
        } else {
            l.add(ind);
            vis[ind] = true;
            for (int n1 : graph[ind]) {
                if (vis[n1] == false) {
                    dfs(graph, n1, n, l, ans, vis);
                }
            }
            vis[ind] = false;
            l.remove(l.size() - 1);
        }
    }
}

```

Accepted Runtime: 0 ms

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

Output: `[[0,1,3],[0,2,3]]`

Expected: `[[0,1,3],[0,2,3]]`

1. Aim/Overview of the practical:

Q.2 Subsets.

<https://leetcode.com/problems/subsets/>

2. Apparatus / Simulator Used:

- Windows 7 or above
- Google Chrome

3. Objective:

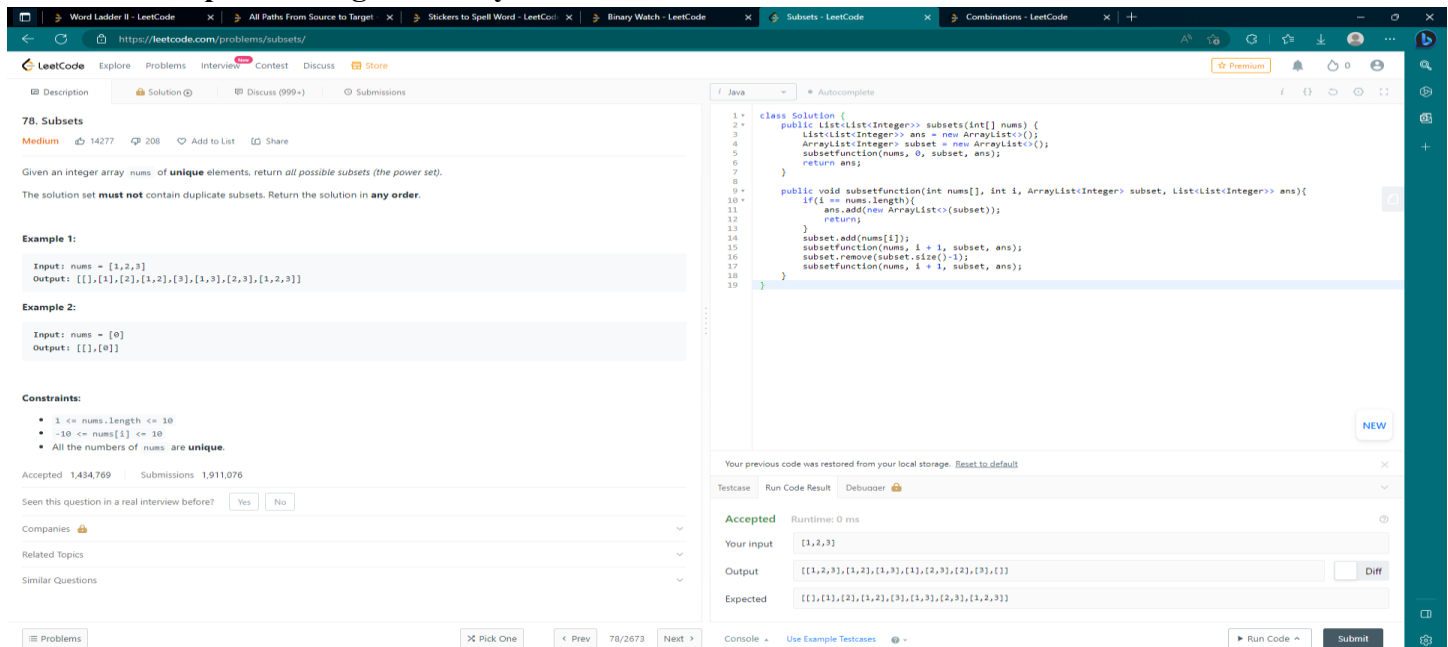
- To understand the concept of Backtracking.
- To implement the concept of Subsets.

4. Code:

```
class Solution {
    public List<List<Integer>> subsets(int[] nums) {
        List<List<Integer>> ans = new ArrayList<>();
        ArrayList<Integer> subset = new ArrayList<>();
        subsetfunction(nums, 0, subset, ans);
        return ans;
    }

    public void subsetfunction(int nums[], int i, ArrayList<Integer> subset, List<List<Integer>> ans){
        if(i == nums.length){
            ans.add(new ArrayList<>(subset));
            return;
        }
        subset.add(nums[i]);
        subsetfunction(nums, i + 1, subset, ans);
        subset.remove(subset.size()-1);
        subsetfunction(nums, i + 1, subset, ans);
    }
}
```

5. Result/Output/Writing Summary:



78. Subsets
Medium 14277 208 Add to List Share

Given an integer array `nums` of **unique** elements, return *all possible subsets (the power set)*.
The solution set **must not** contain duplicate subsets. Return the solution in **any order**.

Example 1:
Input: `nums = [1,2,3]`
Output: `[[[]],[1],[2],[3],[1,2],[1,3],[2,3],[1,2,3]]`


Example 2:
Input: `nums = [0]`
Output: `[[[]],[0]]`


Constraints:


- $1 \leq \text{nums.length} \leq 10$
- $-10 \leq \text{nums}[i] \leq 10$
- All the numbers of `nums` are **unique**.

Accepted 1,434,769 Submissions 1,911,076

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

Companies 

Related Topics 

Similar Questions 

```
class Solution {
    public List<List<Integer>> subsets(int[] nums) {
        List<List<Integer>> ans = new ArrayList<>();
        ArrayList<Integer> subset = new ArrayList<>();
        subsetfunction(nums, 0, subset, ans);
        return ans;
    }

    public void subsetfunction(int nums[], int i, ArrayList<Integer> subset, List<List<Integer>> ans){
        if(i == nums.length){
            ans.add(new ArrayList<>(subset));
            return;
        }
        subset.add(nums[i]);
        subsetfunction(nums, i + 1, subset, ans);
        subset.remove(subset.size()-1);
        subsetfunction(nums, i + 1, subset, ans);
    }
}
```

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

Testcase Run Code Result Debugger

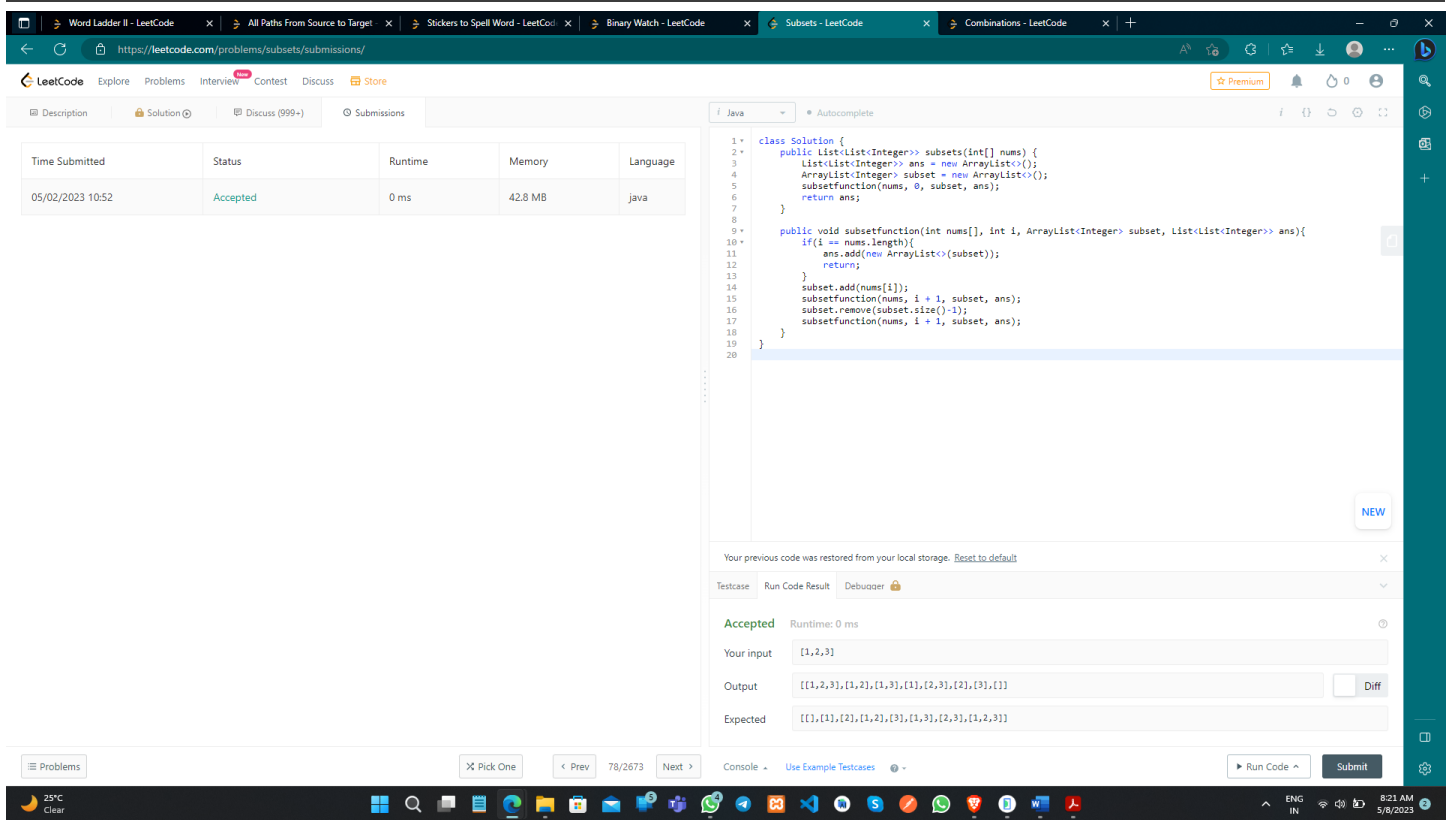
Accepted Runtime: 0 ms

Your input `[1,2,3]`

Output `[[[]],[1],[2],[3],[1,2],[1,3],[2,3],[1,2,3]]`

Expected `[[[]],[1],[2],[3],[1,3],[2,3],[1,2,3]]`

Console Use Example Testcases



Word Ladder II - LeetCode | All Paths From Source to Target | Stickers to Spell Word - LeetCode | Binary Watch - LeetCode | Subsets - LeetCode | Combinations - LeetCode

https://leetcode.com/problems/subsets/submissions/

LeetCode Explore Problems Interview Contest Discuss Store

Description Solution Discuss (999+) Submissions

Time Submitted	Status	Runtime	Memory	Language
05/02/2023 10:52	Accepted	0 ms	42.8 MB	java

```

1 class Solution {
2     public List<List<Integer>> subsets(int[] nums) {
3         List<List<Integer>> ans = new ArrayList<>();
4         ArrayList<Integer> subset = new ArrayList<>();
5         subsetFunction(nums, 0, subset, ans);
6         return ans;
7     }
8
9     public void subsetFunction(int nums[], int i, ArrayList<Integer> subset, List<List<Integer>> ans){
10         if(i == nums.length){
11             ans.add(new ArrayList<>(subset));
12             return;
13         }
14         subset.add(nums[i]);
15         subsetFunction(nums, i + 1, subset, ans);
16         subset.remove(subset.size() - 1);
17         subsetFunction(nums, i + 1, subset, ans);
18     }
19 }
20

```

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

Testcase Run Code Result Debuzzer

Accepted Runtime: 0 ms

Your input [1, 2, 3]

Output [[1, 2, 3], [1, 2], [1, 3], [1], [2, 3], [2], [3], []] Diff

Expected [[1], [1, 2], [1, 2, 3], [1, 3], [2, 3], [1, 2, 3]]

Problems Pick One < Prev 78/2673 Next > Console Use Example Testcases Run Code Submit

25°C Clear ENG IN 8:21 AM 5/8/2023

Learning outcomes (What I have learnt):

- Learned the concept of Backtracking Algorithm such as DFS and so on.
- Learnt about All Path from Source to Target & Subsets.