

## PRACTICAL 11

### 11.1 Implement DFS of Graph.

<https://practice.geeksforgeeks.org/problems/depth-first-traversal-for-a-graph/1>

CODE :

```
class Solution {
private:
    void dfs(int node,vector<int> adj[],vector<int> &visited,vector<int> &dfs1)
    {
        visited[node] = 1;
        dfs1.push_back(node);
        for(auto it:adj[node])
        {
            if(visited[it] == 0)
            {
                dfs(it,adj,visited,dfs1);
            }
        }
    }
public:
    // Function to return a list containing the DFS traversal of the graph.
    vector<int> dfsOfGraph(int V, vector<int> adj[]) {
        // Code here
        vector<int> visited(V,0);
        vector<int> dfs1;
        dfs(0,adj,visited,dfs1);
        return dfs1;
    }
};
```

OUTPUT:

The screenshot displays a coding environment with two main panels. The left panel, titled 'Output Window', shows the 'Compilation Results' tab with a green checkmark indicating 'Problem Solved Successfully'. It also displays performance metrics: 'Test Cases Passed: 1120 / 1120', 'Attempts: Correct / Total: 1 / 1', 'Accuracy: 100%', 'Points Scored: 0 / 2', and 'Time Taken: 0.07'. The right panel shows the code editor with the same C++ code for DFS traversal. The code is numbered from 1 to 31, and the output window is partially visible behind it.

## 11.2 Implement BFS of Graph.

<https://practice.geeksforgeeks.org/problems/bfs-traversal-of-graph/1>

CODE:

```
class Solution {
public:
    // Function to return Breadth First Traversal of given graph.
    vector<int> bfsOfGraph(int V, vector<int> adj[]) {
        // Code here
        //int n=adj.size();
        int vis[V]={0};
        vis[0]=1;
        queue<int>q;
        q.push(0);
        vector<int>bfs;
        while(!q.empty()){
            int node=q.front();
            q.pop();
            bfs.push_back(node);
            for(auto it:adj[node]){
                if(!vis[it]){
                    q.push(it);
                    vis[it]=1;
                }
            }
        }
        return bfs;
    }
};
```

OUTPUT:

The screenshot displays a coding platform interface with the following components:

- Problem Header:** Shows the problem name "DFS of Graph" and a hint "Mother Vertex".
- Compilation Results:** Indicates "Problem Solved Successfully" with a green checkmark and a "Suggest Feedback" link.
- Test Cases Passed:** Displays "1113 / 1113" in a large font.
- Attempts:** Shows "Correct / Total" as "1 / 1" and "Accuracy: 100%".
- Points Scored:** Displays "2 / 2" in a large font.
- Time Taken:** Shows "0.05".
- Your Total Score:** Displays "7" with a green upward arrow.
- Solve Next:** Provides buttons for "DFS of Graph", "Mother Vertex", and "Print adjacency list".
- Code Editor:** Shows the C++ code for the BFS of Graph problem, with line numbers 1 through 30. The code is identical to the one provided in the previous block.
- Output Window:** Is currently empty.