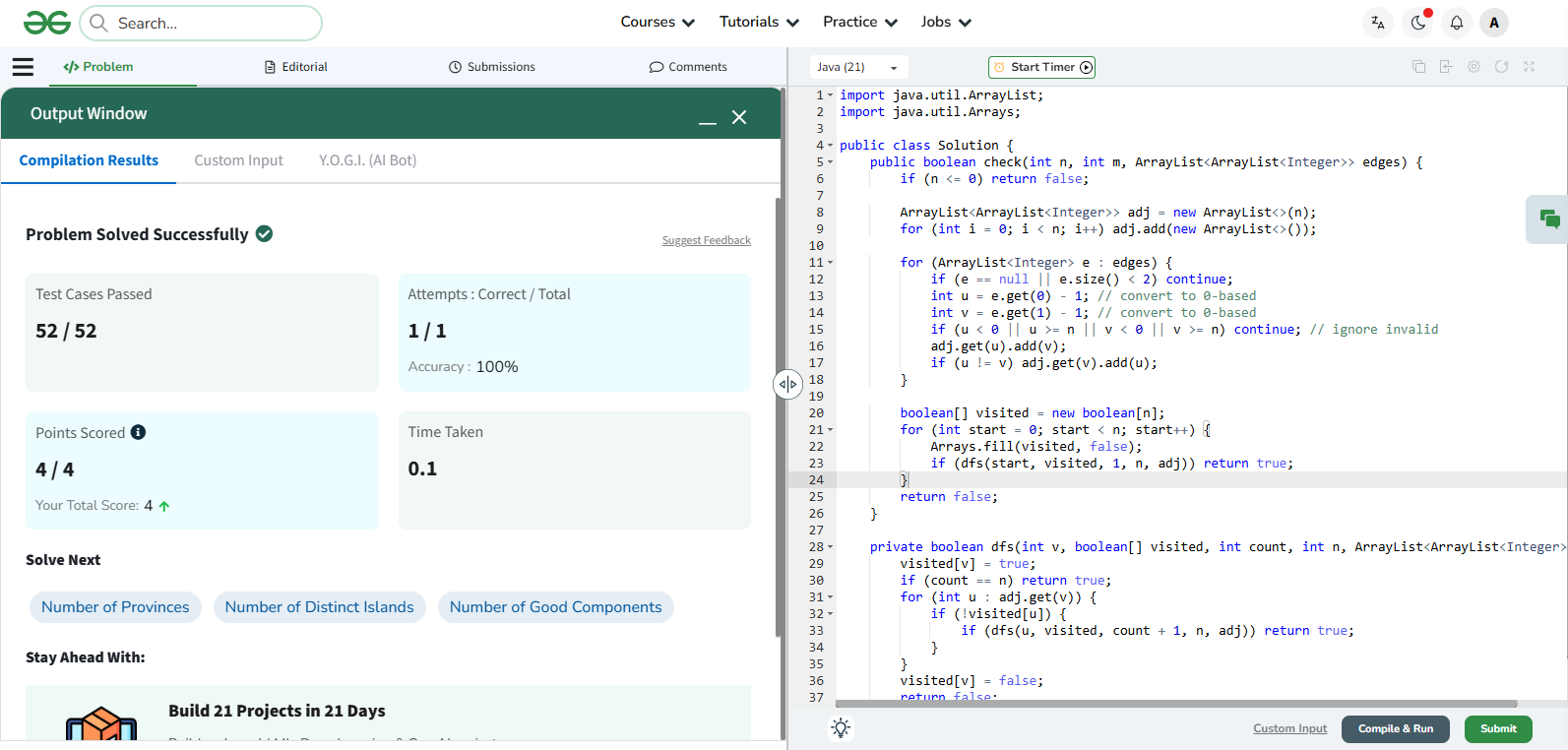
**DAA**

**Name – Chetna Deshmukh**

**Section – A4**

**Roll No. – 24**

**Practical 7 CP**

****

import java.util.ArrayList;

import java.util.Arrays;

public class Solution {

public boolean check(int n, int m, ArrayList<ArrayList<Integer>> edges) {

if (n <= 0) return false;

ArrayList<ArrayList<Integer>> adj = new ArrayList<>(n);

for (int i = 0; i < n; i++) adj.add(new ArrayList<>());

for (ArrayList<Integer> e : edges) {

if (e == null || e.size() < 2) continue;

int u = e.get(0) - 1; // convert to 0-based

int v = e.get(1) - 1; // convert to 0-based

if (u < 0 || u >= n || v < 0 || v >= n) continue; // ignore invalid

adj.get(u).add(v);

if (u != v) adj.get(v).add(u);

}

boolean[] visited = new boolean[n];

for (int start = 0; start < n; start++) {

Arrays.fill(visited, false);

if (dfs(start, visited, 1, n, adj)) return true;

}

return false;

}

private boolean dfs(int v, boolean[] visited, int count, int n, ArrayList<ArrayList<Integer>> adj) {

visited[v] = true;

if (count == n) return true;

for (int u : adj.get(v)) {

if (!visited[u]) {

if (dfs(u, visited, count + 1, n, adj)) return true;

}

}

visited[v] = false;

return false;

}

}