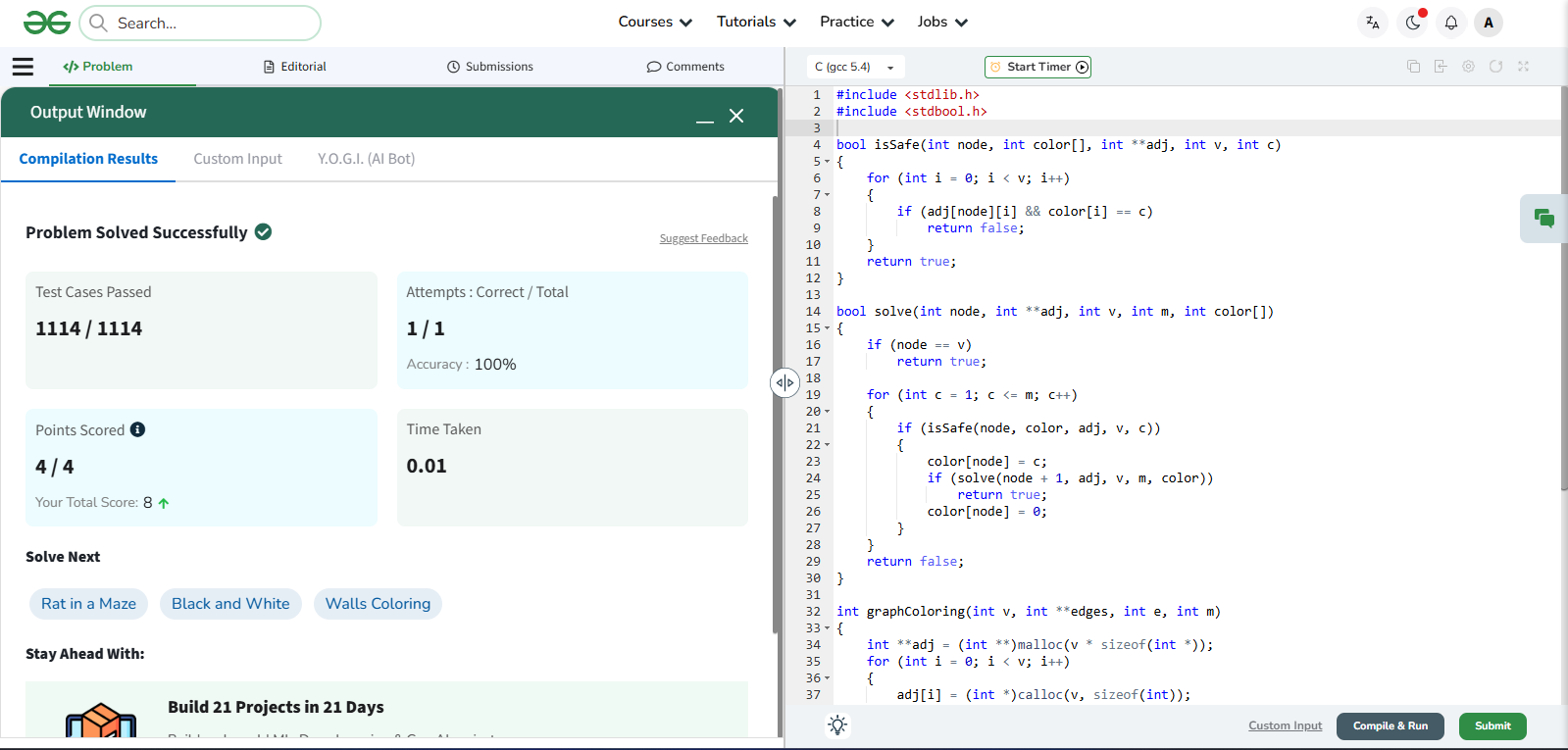
**DAA**

**Name – Chetna Deshmukh**

**Section – A4**

**Roll No. – 24**

**Practical 8 CP**



#include <stdlib.h>

#include <stdbool.h>

bool isSafe(int node, int color[], int \*\*adj, int v, int c)

{

for (int i = 0; i < v; i++)

{

if (adj[node][i] && color[i] == c)

return false;

}

return true;

}

bool solve(int node, int \*\*adj, int v, int m, int color[])

{

if (node == v)

return true;

for (int c = 1; c <= m; c++)

{

if (isSafe(node, color, adj, v, c))

{

color[node] = c;

if (solve(node + 1, adj, v, m, color))

return true;

color[node] = 0;

}

}

return false;

}

int graphColoring(int v, int \*\*edges, int e, int m)

{

int \*\*adj = (int \*\*)malloc(v \* sizeof(int \*));

for (int i = 0; i < v; i++)

{

adj[i] = (int \*)calloc(v, sizeof(int));

}

for (int i = 0; i < e; i++)

{

int u = edges[i][0];

int w = edges[i][1];

adj[u][w] = 1;

adj[w][u] = 1;

}

int \*color = (int \*)calloc(v, sizeof(int));

bool ans = solve(0, adj, v, m, color);

for (int i = 0; i < v; i++)

free(adj[i]);

free(adj);

free(color);

return ans ? 1 : 0;

}