

FLOYD-WARSHALL ALGORITHM:

```
#include <iostream>

#include <vector>

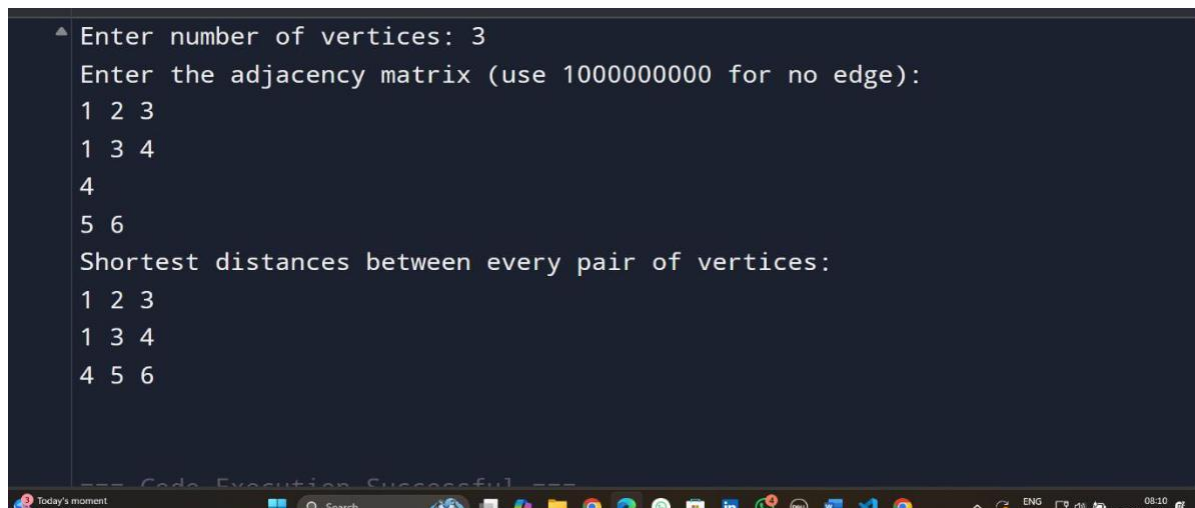
using namespace std;

const int INF = 1e9;

void floydWarshall(vector<vector<int>>& dist, int V)
{
    for (int k = 0; k < V; ++k) {
        for (int i = 0; i < V; ++i) {
            for (int j = 0; j < V; ++j) {
                if (dist[i][k] < INF && dist[k][j] < INF)
                    dist[i][j] = min(dist[i][j], dist[i][k] + dist[k][j]);
            }
        }
    }
}

void printDistanceMatrix(const vector<vector<int>>& dist)
{
    int V = dist.size();
    for (int i = 0; i < V; ++i) {
        for (int j = 0; j < V; ++j) {
            if (dist[i][j] == INF)
                cout << "INF ";
            else
                cout << dist[i][j] << " ";
        }
        cout << "\n";
    }
}
```

```
int main() {  
    int V;  
    cin >> V;  
    vector<vector<int>> dist(V, vector<int>(V));  
    for (int i = 0; i < V; ++i)  
        for (int j = 0; j < V; ++j)  
            cin >> dist[i][j];  
    floydWarshall(dist, V);  
    printDistanceMatrix(dist);  
    return 0;  
}
```



The screenshot shows a terminal window with the following text:

```
Enter number of vertices: 3  
Enter the adjacency matrix (use 1000000000 for no edge):  
1 2 3  
1 3 4  
4  
5 6  
Shortest distances between every pair of vertices:  
1 2 3  
1 3 4  
4 5 6
```

At the bottom of the terminal window, the text "=== Code Execution Successful ===" is visible. The Windows taskbar is also visible at the bottom of the screen.

BFS IMPLEMENTATION:

```
#include <iostream>

#include <vector>

#include <queue>

using namespace std;

void bfs(int start, vector<vector<int>>& adj, vector<bool>& visited)
{
    queue<int> q;
    q.push(start);
    visited[start] = true;

    while (!q.empty()) {
        int node = q.front();
        q.pop();
        cout << node << " ";

        for (int neighbor : adj[node]) {
            if (!visited[neighbor]) {
                visited[neighbor] = true;
                q.push(neighbor);
            }
        }
    }
}

int main() {
    int V, E;
    cin >> V >> E;
    vector<vector<int>> adj(V);
    for (int i = 0; i < E; ++i) {
        int u, v;
```

```
cin >> u >> v;  
adj[u].push_back(v);  
adj[v].push_back(u);  
}
```

```
vector<bool> visited(V, false);
```

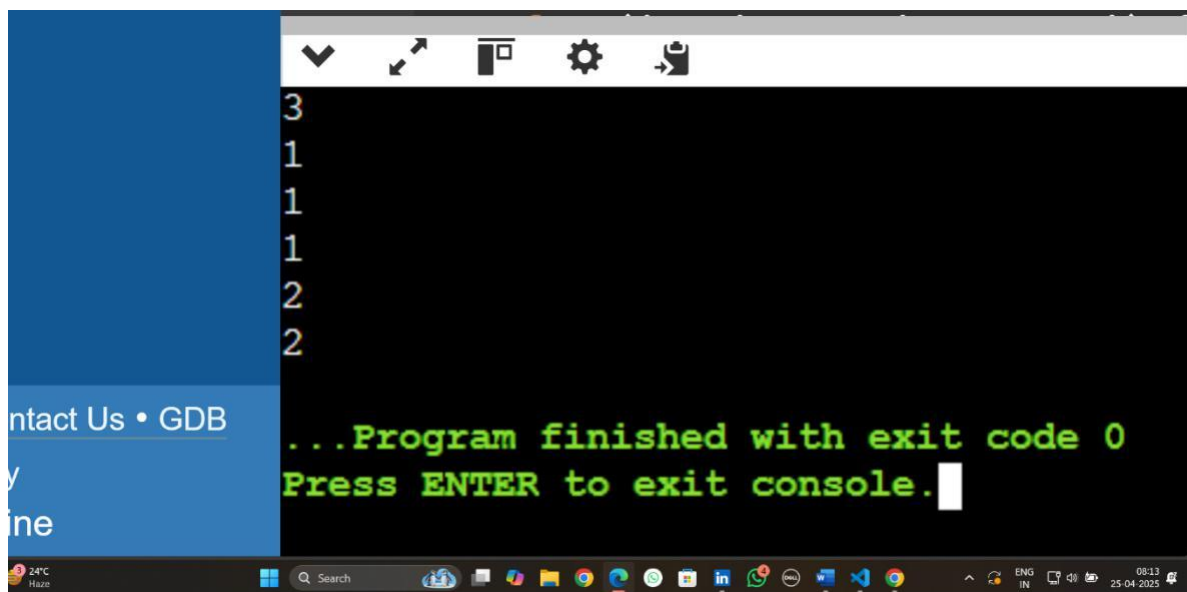
```
int start;
```

```
cin >> start;
```

```
bfs(start, adj, visited);
```

```
return 0;
```

```
}
```



The screenshot shows a GDB console window with a dark background. On the left, there is a blue sidebar with the text "Contact Us • GDB" and "y" and "ine" visible. The main console area displays the following output in green text: "3", "1", "1", "1", "2", "2", and "...Program finished with exit code 0". Below this, it says "Press ENTER to exit console." with a white cursor. The Windows taskbar is visible at the bottom, showing the search bar, task view, and various application icons. The system tray on the right shows the date and time as "08:13 25 04 2025".

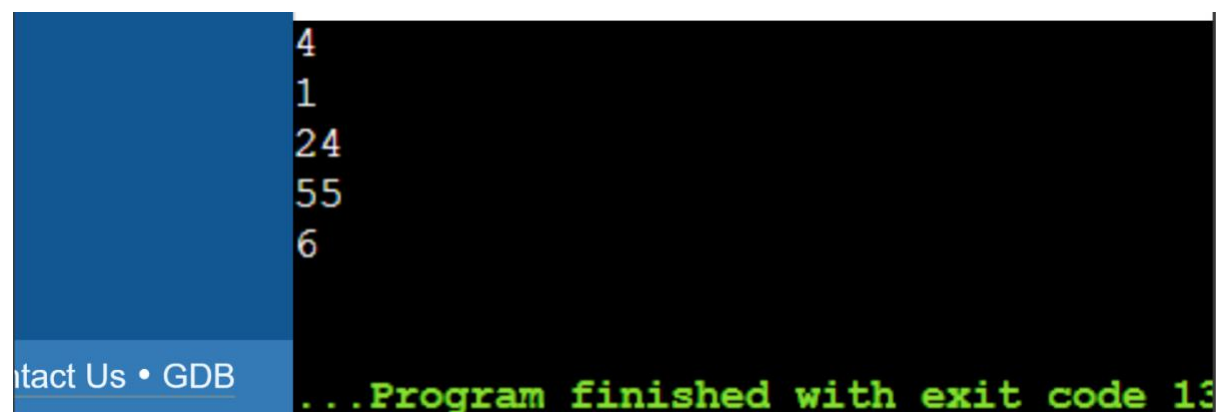
DFS IMPLEMENTATION:

```
#include <iostream>
#include <vector>

using namespace std;
void dfs(int node, vector<vector<int>>& adj, vector<bool>& visited)
{
    visited[node] = true;
    cout << node << " ";
    for (int neighbor : adj[node]) {
        if (!visited[neighbor])
            dfs(neighbor, adj, visited);
    }
}

int main() {
    int V, E;
    cin >> V >> E;
    vector<vector<int>> adj(V);
    for (int i = 0; i < E; ++i) {
        int u, v;
        cin >> u >> v;
        adj[u].push_back(v);
        adj[v].push_back(u); // Remove for directed graph
    }

    vector<bool> visited(V, false);
    int start;
    cin >> start;
    dfs(start, adj, visited);
    return 0;
}
```



```
4
1
24
55
6
...Program finished with exit code 13
```

N QUEENS :

```
#include <iostream>

#include <vector>

using namespace std;

void printSolution(const vector<string>& board) {
    for (const string& row : board)
        cout << row << "\n";
    cout << "\n";
}

bool isSafe(int row, int col, const vector<string>& board, int n)
{
    for (int i = 0; i < row; ++i)
        if (board[i][col] == 'Q') return false;

    for (int i = row - 1, j = col - 1; i >= 0 && j >= 0; --i, --j)
        if (board[i][j] == 'Q') return false;

    for (int i = row - 1, j = col + 1; i >= 0 && j < n; --i, ++j)
        if (board[i][j] == 'Q') return false;
    return true;
}

void solve(int row, vector<string>& board, int n) {
    if (row == n) {
        printSolution(board);
        return;
    }

    for (int col = 0; col < n; ++col) {
        if (isSafe(row, col, board, n)) {
            board[row][col] = 'Q';
            solve(row + 1, board, n);
            board[row][col] = '.';
        }
    }
}
```

```
int main() {  
    int n;  
  
    cin >> n;  
  
    vector<string> board(n, string(n, '.'));  
  
    solve(0, board, n);  
  
    return 0;  
}
```



TRAVELLING SALESMAN PROBLEM:

```
#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

int main() {

    int n;

    cin >> n;

    vector<vector<int>>> dist(n, vector<int>(n));

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

        for (int j = 0; j < n; ++j)

            cin >> dist[i][j];

    vector<int> cities;

    for (int i = 1; i < n; ++i)

        cities.push_back(i);

    int minCost = INT_MAX;

    do {

        int cost = 0;

        int curr = 0;

        for (int i = 0; i < cities.size(); ++i) {

            cost += dist[curr][cities[i]];

            curr = cities[i];

        }

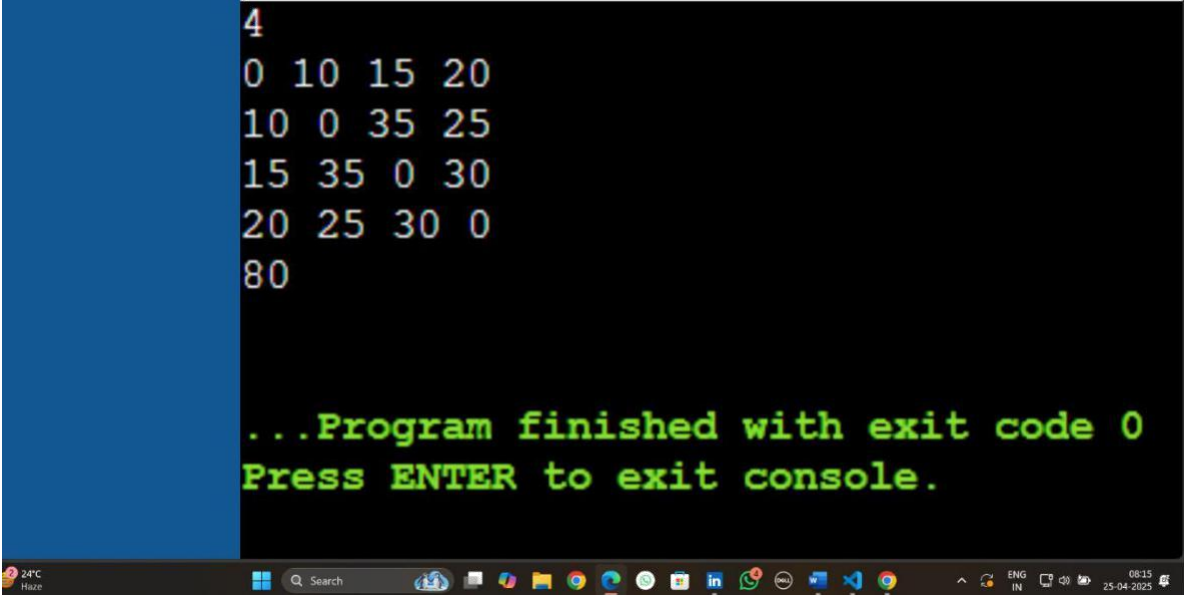
        cost += dist[curr][0];

        minCost = min(minCost, cost);

    } while (next_permutation(cities.begin(), cities.end()));
```



```
cout << minCost << endl;  
return 0;  
}
```



The screenshot shows a console window with a black background and white text. On the left, there is a solid blue vertical bar. The output of the program is as follows:

```
4  
0 10 15 20  
10 0 35 25  
15 35 0 30  
20 25 30 0  
80  
  
...Program finished with exit code 0  
Press ENTER to exit console.
```

At the bottom of the window, a Windows taskbar is visible, showing the Start button, a search bar, and several application icons. The system tray on the right indicates the temperature is 24°C, the language is ENG IN, and the date and time are 08:15 on 25-04-2023.

