DAY 7

Aryan **22BCS10190** **KPIT-901(A)**

Find Center of Star Graph

```cpp
#include <iostream>
#include <vector>
using namespace std;

int findCenter(vector<vector<int>>& edges) {
    return (edges[0][0] == edges[1][0] || edges[0][0] == edges[1][1]) ? edges[0][0] : edges[0][1];
}

int main() {
    vector<vector<int>> edges = {{1, 2}, {2, 3}, {4, 2}};
    cout << findCenter(edges) << endl;
    return 0;
}
```

```
C:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1\" && g++ day7ques1.cpp -o day7ques1 && "c
:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1\"day7ques1
2

c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1>
```

Find if Path Exists in Graph

```cpp
#include <iostream>
#include <vector>
#include <queue>
using namespace std;

bool validPath(int n, vector<vector<int>>& edges, int source, int destination) {
    vector<vector<int>> graph(n);
    for (auto& edge : edges) {
        graph[edge[0]].push_back(edge[1]);
        graph[edge[1]].push_back(edge[0]);
    }
    vector<bool> visited(n, false);
    queue<int> q;
    q.push(source);
    visited[source] = true;
    while (!q.empty()) {
        int node = q.front(); q.pop();
        if (node == destination) return true;
        for (int neighbor : graph[node]) {
            if (!visited[neighbor]) {
                visited[neighbor] = true;
                q.push(neighbor);
            }
        }
    }
    return false;
}

int main() {
    int n = 6;
    vector<vector<int>> edges = {{0, 1}, {0, 2}, {3, 5}, {5, 4}, {4, 3}};
    int source = 0, destination = 5;
    cout << (validPath(n, edges, source, destination) ? "true" : "false") << endl;
    return 0;
}
```

```
Active code page: 65001

C:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1\" && g++ day7ques2.cpp -o day7ques2 && "c
:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1\"day7ques2
false

c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1>
```

01 Matrix

```cpp
#include <iostream>
#include <vector>
#include <queue>
#include <climits> // For INT_MAX
using namespace std;

vector<vector<int>> updateMatrix(vector<vector<int>>& mat) {
    int rows = mat.size(), cols = mat[0].size();
    vector<vector<int>> dist(rows, vector<int>(cols, INT_MAX));
    queue<pair<int, int>> q;

    for (int i = 0; i < rows; ++i) {
        for (int j = 0; j < cols; ++j) {
            if (mat[i][j] == 0) {
                dist[i][j] = 0;
                q.push(make_pair(i, j));
            }
        }
    }

    vector<int> dirs = {-1, 0, 1, 0, -1};
    while (!q.empty()) {
        pair<int, int> front = q.front();
        q.pop();
        int x = front.first, y = front.second;

        for (int k = 0; k < 4; ++k) {
            int nx = x + dirs[k], ny = y + dirs[k + 1];
            if (nx >= 0 && ny >= 0 && nx < rows && ny < cols && dist[nx][ny] > dist[x][y] + 1) {
                dist[nx][ny] = dist[x][y] + 1;
                q.push(make_pair(nx, ny));
            }
        }
    }

    return dist;
}

int main() {
    vector<vector<int>> mat = {{0, 0, 0}, {0, 1, 0}, {1, 1, 1}};
    vector<vector<int>> result = updateMatrix(mat);
    for (const auto& row : result) {
        for (int val : row) cout << val << " ";
        cout << endl;
    }
    return 0;
}
```

```
Active code page: 65001

C:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1\" && g++ day7ques3.cpp -o day7ques3 && "c
:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1\"day7ques3
0 0 0
0 1 0
1 2 1

c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1>
```

Accounts Merge

day7ques4.cpp > ...

```cpp
#include <iostream>
#include <vector>
#include <string>
#include <unordered_map>
#include <unordered_set>
#include <queue>
#include <algorithm>
using namespace std;

vector<vector<string>> accountsMerge(vector<vector<string>>& accounts) {
    unordered_map<string, unordered_set<string>> graph;
    unordered_map<string, string> emailToName;

    for (auto& account : accounts) {
        string name = account[0];
        for (int i = 1; i < account.size(); ++i) {
            emailToName[account[i]] = name;
            if (i > 1) {
                graph[account[i]].insert(account[i - 1]);
                graph[account[i - 1]].insert(account[i]);
            }
        }
    }

    unordered_set<string> visited;
    vector<vector<string>> result;

    for (auto& pair : emailToName) {
        string email = pair.first;
        string name = pair.second;
        if (visited.count(email)) continue;

        vector<string> merged;
        queue<string> q;
        q.push(email);
        visited.insert(email);
        while (!q.empty()) {
            string node = q.front();
            q.pop();
            merged.push_back(node);
            for (const string& neighbor : graph[node]) {
                if (!visited.count(neighbor)) {
                    visited.insert(neighbor);
                    q.push(neighbor);
                }
            }
        }
        sort(merged.begin(), merged.end());
        merged.insert(merged.begin(), name);
        result.push_back(merged);
    }

    return result;
}
```

```cpp
int main() {
    vector<vector<string>> accounts = {
        {"John", "johnsmith@mail.com", "john00@mail.com"},
        {"John", "johnnybravo@mail.com"},
        {"John", "johnsmith@mail.com", "john_newyork@mail.com"},
        {"Mary", "mary@mail.com"}
    };
    vector<vector<string>> result = accountsMerge(accounts);
    for (auto& acc : result) {
        for (string& str : acc) cout << str << " ";
        cout << endl;
    }
    return 0;
}
```

```
Active code page: 65001

C:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1\" && g++ day7ques4.cpp -o day7ques4 && "c
:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1\"day7ques4
Mary mary@mail.com
John john00@mail.com john_newyork@mail.com johnsmith@mail.com
John johnnybravo@mail.com

c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1>
```

Network Delay Time

day7ques5.cpp > ...

```cpp
#include <iostream>
#include <vector>
#include <queue>
#include <algorithm>
#include <climits>
using namespace std;

int networkDelayTime(vector<vector<int>>& times, int n, int k) {
    vector<vector<pair<int, int>>> graph(n + 1);
    for (auto& time : times)
        graph[time[0]].emplace_back(time[1], time[2]);

    vector<int> dist(n + 1, INT_MAX);
    dist[k] = 0;

    priority_queue<pair<int, int>, vector<pair<int, int>>, greater<>> pq;
    pq.push(make_pair(0, k));

    while (!pq.empty()) {
        pair<int, int> front = pq.top();
        pq.pop();
        int time = front.first;
        int node = front.second;

        if (time > dist[node]) continue;

        for (auto& edge : graph[node]) {
            int neighbor = edge.first;
            int weight = edge.second;
            if (dist[neighbor] > dist[node] + weight) {
                dist[neighbor] = dist[node] + weight;
                pq.push(make_pair(dist[neighbor], neighbor));
            }
        }
    }

    int maxDist = *max_element(dist.begin() + 1, dist.end());
    return maxDist == INT_MAX ? -1 : maxDist;
}

int main() {
    vector<vector<int>> times = {{2, 1, 1}, {2, 3, 1}, {3, 4, 1}};
    int n = 4, k = 2;
    cout << networkDelayTime(times, n, k) << endl;
    return 0;
}

```

```
Active code page: 65001

C:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1\" && g++ day7ques5.cpp -o day7ques5 && "c
:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1\"day7ques5
2

c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1>
```

All Paths From Source to Target

day7ques6.cpp > main()

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;

void dfs(int i, int j, vector<vector<int>>& grid, int& area) {
    if (i < 0 || j < 0 || i >= grid.size() || j >= grid[0].size() || grid[i][j] == 0) {
        return;
    }
    grid[i][j] = 0;
    area++;
    dfs(i + 1, j, grid, area);
    dfs(i - 1, j, grid, area);
    dfs(i, j + 1, grid, area);
    dfs(i, j - 1, grid, area);
}

int maxAreaOfIsland(vector<vector<int>>& grid) {
    int maxArea = 0;
    for (int i = 0; i < grid.size(); i++) {
        for (int j = 0; j < grid[0].size(); j++) {
            if (grid[i][j] == 1) {
                int area = 0;
                dfs(i, j, grid, area);
                maxArea = max(maxArea, area);
            }
        }
    }
    return maxArea;
}

int main() {
    vector<vector<int>> grid = {{0, 1, 0, 0, 0},
                                {1, 1, 1, 0, 0},
                                {0, 1, 0, 0, 1},
                                {0, 0, 0, 1, 1}};
    cout << maxAreaOfIsland(grid) << endl;
    return 0;
}
```

```
Active code page: 65001

C:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1\" && g++ day7ques6.cpp -o day7ques6 && "c
:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1\"day7ques6
5

c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1>
```

Max Area of Island

day7ques7.cpp > ...

```cpp
#include <iostream>
#include <vector>
using namespace std;

void dfs(int node, vector<vector<int>>& graph, vector<int>& path, vector<vector<int>>& result) {
    path.push_back(node);
    if (node == graph.size() - 1) {
        result.push_back(path);
    } else {
        for (int neighbor : graph[node]) {
            dfs(neighbor, graph, path, result);
        }
    }
    path.pop_back();
}

vector<vector<int>> allPathsSourceTarget(vector<vector<int>>& graph) {
    vector<vector<int>> result;
    vector<int> path;
    dfs(0, graph, path, result);
    return result;
}

int main() {
    vector<vector<int>> graph = {{1, 2}, {3}, {3}, {}};
    vector<vector<int>> result = allPathsSourceTarget(graph);
    for (const auto& path : result) {
        for (int node : path) {
            cout << node << " ";
        }
        cout << endl;
    }
    return 0;
}
```

```
Active code page: 65001

C:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1\" && g++ day7ques7.cpp -o day7ques7 && "c
:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1\"day7ques7
0 1 3
0 2 3

c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\pratice1>
```