

## DAY 7

Pawan Kumar

22BCS10224

KPIT-901(A)

### Find Center of Star Graph

```
day/ques1.cpp / ...
1 #include <iostream>
2 #include <vector>
3 using namespace std;
4
5 int findCenter(vector<vector<int>>& edges) {
6     return (edges[0][0] == edges[1][0] || edges[0][0] == edges[1][1] ? edges[0][0] : edges[0][1]);
7 }
8
9 int main() {
10     vector<vector<int>> edges = {{1, 2}, {2, 3}, {4, 2}};
11     cout << findCenter(edges) << endl;
12     return 0;
13 }
14
```

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

### Find if Path Exists in Graph

```
day/ques2.cpp / ...
1 #include <iostream>
2 #include <vector>
3 #include <queue>
4 using namespace std;
5
6 bool validPath(int n, vector<vector<int>>& edges, int source, int destination) {
7     vector<vector<int>> graph(n);
8     for (auto& edge : edges) {
9         graph[edge[0]].push_back(edge[1]);
10        graph[edge[1]].push_back(edge[0]);
11    }
12    vector<bool> visited(n, false);
13    queue<int> q;
14    q.push(source);
15    visited[source] = true;
16    while (!q.empty()) {
17        int node = q.front(); q.pop();
18        if (node == destination) return true;
19        for (int neighbor : graph[node]) {
20            if (!visited[neighbor]) {
21                visited[neighbor] = true;
22                q.push(neighbor);
23            }
24        }
25    }
26    return false;
27 }
28
29 int main() {
30     int n = 6;
31     vector<vector<int>> edges = {{0, 1}, {0, 2}, {3, 5}, {5, 4}, {4, 3}};
32     int source = 0, destination = 5;
33     cout << (validPath(n, edges, source, destination) ? "true" : "false") << endl;
34     return 0;
35 }
36
```

```
Active code page: 65001
C:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\praticel>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\praticel\" && g++ day7ques2.cpp -o day7ques2 && "c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\praticel\"day7ques2
false
C:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\praticel>
```

# 01 Matrix

```
1 // day7ques3.cpp
2 #include <iostream>
3 #include <vector>
4 #include <queue>
5 #include <limits> // For INT_MAX
6 using namespace std;
7
8 vector<vector<int>> updateMatrix(vector<vector<int>>& mat) {
9     int rows = mat.size(), cols = mat[0].size();
10    vector<vector<int>> dist(rows, vector<int>(cols, INT_MAX));
11    queue<pair<int, int>> q;
12
13    for (int i = 0; i < rows; ++i) {
14        for (int j = 0; j < cols; ++j) {
15            if (mat[i][j] == 0) {
16                dist[i][j] = 0;
17                q.push(make_pair(i, j));
18            }
19        }
20    }
21
22    vector<int> dirs = {-1, 0, 1, 0, -1};
23    while (!q.empty()) {
24        pair<int, int> front = q.front();
25        q.pop();
26        int x = front.first, y = front.second;
27
28        for (int k = 0; k < 4; ++k) {
29            int nx = x + dirs[k], ny = y + dirs[k+1];
30            if (nx >= 0 && ny >= 0 && nx < rows && ny < cols && dist[nx][ny] > dist[x][y] + 1) {
31                dist[nx][ny] = dist[x][y] + 1;
32                q.push(make_pair(nx, ny));
33            }
34        }
35    }
36
37    return dist;
38 }
39
40 int main() {
41    vector<vector<int>> mat = {{0, 0, 0}, {0, 1, 0}, {1, 1, 1}};
42    vector<vector<int>> result = updateMatrix(mat);
43    for (const auto& row : result) {
44        for (int val : row) cout << val << " ";
45        cout << endl;
46    }
47    return 0;
48 }
```

Active code page: 65001

```
c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\praticel>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\praticel\" && g++ day7ques3.cpp -o day7ques3 && "c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\praticel\"day7ques3
```

```
0 0 0
0 1 0
1 2 1
```

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

# Accounts Merge

```
1 #include <iostream>
2 #include <vector>
3 #include <string>
4 #include <unordered_map>
5 #include <unordered_set>
6 #include <queue>
7 #include <algorithm>
8 using namespace std;
9
10 vector<vector<string>> accountsMerge(vector<vector<string>>& accounts) {
11     unordered_map<string, unordered_set<string>> graph;
12     unordered_map<string, string> emailToName;
13
14     for (auto& account : accounts) {
15         string name = account[0];
16         for (int i = 1; i < account.size(); ++i) {
17             emailToName[account[i]] = name;
18             if (i > 1) {
19                 graph[account[i]].insert(account[i - 1]);
20                 graph[account[i - 1]].insert(account[i]);
21             }
22         }
23     }
24
25     unordered_set<string> visited;
26     vector<vector<string>> result;
27
28     for (auto& pair : emailToName) {
29         string email = pair.first;
30         string name = pair.second;
31         if (visited.count(email)) continue;
32
33         vector<string> merged;
34         queue<string> q;
35         q.push(email);
36         visited.insert(email);
37         while (!q.empty()) {
38             string node = q.front();
39             q.pop();
40             merged.push_back(node);
41             for (const string& neighbor : graph[node]) {
42                 if (!visited.count(neighbor)) {
43                     visited.insert(neighbor);
44                     q.push(neighbor);
45                 }
46             }
47         }
48         sort(merged.begin(), merged.end());
49         merged.insert(merged.begin(), name);
50         result.push_back(merged);
51     }
52
53     return result;
54 }
55
```

```
56 int main() {
57     vector<vector<string>> accounts = {
58         {"John", "johnsmith@mail.com", "john00@mail.com"},
59         {"John", "johnnybravo@mail.com"},
60         {"John", "johnsmith@mail.com", "john_newyork@mail.com"},
61         {"Mary", "mary@mail.com"}
62     };
63     vector<vector<string>> result = accountsMerge(accounts);
64     for (auto& acc : result) {
65         for (string& str : acc) cout << str << " ";
66         cout << endl;
67     }
68     return 0;
69 }
70
```

Active code page: 65001

```
C:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\praticel>cd "c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\praticel\" && g++ day/ques4.cpp -o day/ques4 && "c:\Users\Gaurav Kumar\OneDrive\Desktop\5th sem\codes\New folder\praticel\day/ques4
```

```
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\praticel>
```

# Network Delay Time

```
day7ques5.cpp > ...
1  #include <iostream>
2  #include <vector>
3  #include <queue>
4  #include <algorithm>
5  #include <limits>
6  using namespace std;
7
8  int networkDelayTime(vector<vector<int>>> times, int n, int k) {
9      vector<vector<pair<int, int>>> graph(n + 1);
10     for (auto& time : times)
11         graph[time[0]].emplace_back(time[1], time[2]);
12
13     vector<int> dist(n + 1, INT_MAX);
14     dist[k] = 0;
15
16     priority_queue<pair<int, int>, vector<pair<int, int>>, greater<>> pq;
17     pq.push(make_pair(0, k));
18
19     while (!pq.empty()) {
20         pair<int, int> front = pq.top();
21         pq.pop();
22         int time = front.first;
23         int node = front.second;
24
25         if (time > dist[node]) continue;
26
27         for (auto& edge : graph[node]) {
28             int neighbor = edge.first;
29             int weight = edge.second;
30             if (dist[neighbor] > dist[node] + weight) {
31                 dist[neighbor] = dist[node] + weight;
32                 pq.push(make_pair(dist[neighbor], neighbor));
33             }
34         }
35     }
36
37     int maxDist = *max_element(dist.begin() + 1, dist.end());
38     return maxDist == INT_MAX ? -1 : maxDist;
39 }
40
41 int main() {
42     vector<vector<int>>> times = {{2, 1, 1}, {2, 3, 1}, {3, 4, 1}};
43     int n = 4, k = 2;
44     cout << networkDelayTime(times, n, k) << endl;
45     return 0;
46 }
47
```

Active code page: 65001

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

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

## All Paths From Source to Target

```
day7ques6.cpp > main()
1  #include <iostream>
2  #include <vector>
3  #include <algorithm>
4  using namespace std;
5
6  void dfs(int i, int j, vector<vector<int>>& grid, int& area) {
7      if (i < 0 || j < 0 || i >= grid.size() || j >= grid[0].size() || grid[i][j] == 0) {
8          return;
9      }
10     grid[i][j] = 0;
11     area++;
12     dfs(i + 1, j, grid, area);
13     dfs(i - 1, j, grid, area);
14     dfs(i, j + 1, grid, area);
15     dfs(i, j - 1, grid, area);
16 }
17
18 int maxAreaOfIsland(vector<vector<int>>& grid) {
19     int maxArea = 0;
20     for (int i = 0; i < grid.size(); i++) {
21         for (int j = 0; j < grid[0].size(); j++) {
22             if (grid[i][j] == 1) {
23                 int area = 0;
24                 dfs(i, j, grid, area);
25                 maxArea = max(maxArea, area);
26             }
27         }
28     }
29     return maxArea;
30 }
31
32 int main() {
33     vector<vector<int>> grid = {{0, 1, 0, 0, 0},
34                                {1, 1, 1, 0, 0},
35                                {0, 1, 0, 0, 1},
36                                {0, 0, 0, 1, 1}};
37     cout << maxAreaOfIsland(grid) << endl;
38     return 0;
39 }
```

Active code page: 65801

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

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

## Max Area of Island

```
day7ques7.cpp > ...
1  #include <iostream>
2  #include <vector>
3  using namespace std;
4
5  void dfs(int node, vector<vector<int>>& graph, vector<int>& path, vector<vector<int>>& result) {
6      path.push_back(node);
7      if (node == graph.size() - 1) {
8          result.push_back(path);
9      } else {
10         for (int neighbor : graph[node]) {
11             dfs(neighbor, graph, path, result);
12         }
13     }
14     path.pop_back();
15 }
16
17 vector<vector<int>> allPathsSourceTarget(vector<vector<int>>& graph) {
18     vector<vector<int>> result;
19     vector<int> path;
20     dfs(0, graph, path, result);
21     return result;
22 }
23
24 int main() {
25     vector<vector<int>> graph = {{1, 2}, {3}, {3}, {}};
26     vector<vector<int>> result = allPathsSourceTarget(graph);
27     for (const auto& path : result) {
28         for (int node : path) {
29             cout << node << " ";
30         }
31         cout << endl;
32     }
33     return 0;
34 }
35
```

Active code page: 65001

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

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