Problem Link:

<https://leetcode.com/problems/map-of-highest-peak/?envType=daily-question&envId=2025-01-22>

Solution:

class Solution {

public:

vector<vector<int>> highestPeak(vector<vector<int>>& isWater) {

int m = isWater.size(), n = isWater[0].size();

vector<vector<int>> height(m, vector<int>(n, -1));

vector<pair<int, int>> directions = {{-1, 0}, {1, 0}, {0, -1}, {0, 1}};

queue<pair<int, int>> q;

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

{

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

{

if(isWater[i][j] == 1)

{

height[i][j] = 0;

q.push({i, j});

}

}

}

while(!q.empty())

{

auto [x, y] = q.front();

q.pop();

for(auto [dx, dy] : directions)

{

int nx = x + dx, ny = y + dy;

if(nx >= 0 && nx < m && ny >= 0 && ny < n && height[nx][ny] == -1)

{

height[nx][ny] = height[x][y] + 1;

q.push({nx, ny});

}

}

}

return height;

}

};