Problem Link:

<https://leetcode.com/problems/swim-in-rising-water/?envType=daily-question&envId=2025-10-06>

Solution:

class Solution {

public:

int swimInWater(vector<vector<int>>& grid) {

int n = grid.size();

vector<vector<bool>> v(n, vector<bool>(n, false));

priority\_queue<tuple<int, int, int>, vector<tuple<int, int, int>>, greater<>> pq;

pq.emplace(grid[0][0], 0, 0);

v[0][0] = true;

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

while(!pq.empty())

{

auto [t, x, y] = pq.top();

pq.pop();

if(x == n - 1 && y == n - 1)

return t;

for(int d = 0; d < 4; ++d)

{

int nx = x + dirs[d];

int ny = y + dirs[d + 1];

if(nx >= 0 && ny >= 0 && nx < n && ny < n && !v[nx][ny])

{

v[nx][ny] = true;

pq.emplace(max(t, grid[nx][ny]), nx, ny);

}

}

}

return -1;

}

};