# Day 7

| | |
|---|---|
| Done | ☑ |
| Topic | Floyd Warshall Algorithm<br>Max Area Of Island 8-Directional |
| Languages | Java |
| Difficulty | ⭐⭐⭐⭐ |
| Date Started | @March 7, 2023 11:35 PM |
| Date Completed | @March 8, 2023 4:00 AM |
| Related to Progress (Days) | Your Progress |

## What I Learned Today

• Floyd Warshal Algorithm And Its Implementation

• Max Area Of Island **8-Directional**

## Key Concepts

### Floyd Warshall Algorithm

The problem is to find the shortest distances between every pair of vertices in a given edge-weighted directed graph. The graph is represented as an adjacency matrix of size n*n. Matrix[i][j] denotes the weight of the edge from i to j. If Matrix[i][j]=-1, it means there is no edge from i to j.

Example :

### Quick Links

Tutorial

Documentation

Tutorial

LeetCode Documentation

💡 Input Format: matrix[][] = { {0, 2, -1, -1},  {1, 0, 3, -1},{-1, -1, 0, -1},{3, 5, 4, 0} }
Result:0 2 5 -1 1 0 3 -1 -1 -1 0 -1 3 5 4 0
Explanation: In this example, the final matrix is storing the shortest distances. For example, matrix[i][j] is storing the shortest distance from node i to j.

### Maximum Area Of Island - 8-Directional

Approach #1: Depth-First Search (Recursive)

Intuition and Algorithm :

We want to know the area of each connected shape in the grid, then take the maximum of these. If we are on a land square and explore every square connected to it 4-directionally (and recursively squares connected to those squares, and so on), then the total number of squares explored will be the area of that connected shape. To ensure we don't count squares in a shape more than once, let's use seen to keep track of squares we haven't visited before. It will also prevent us from counting the same shape more than once.

## Code Snippets

```
import java.util.*;


//User function template for JAVA

class Solution {
    public void shortest_distance(int[][] matrix) {
        int n = matrix.length;
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
```

```
                if (matrix[i][j] == -1) {
                    matrix[i][j] = (int)(1e9);
                }
                if (i == j) matrix[i][j] = 0;
            }
        }

        for (int k = 0; k < n; k++) {
            for (int i = 0; i < n; i++) {
                for (int j = 0; j < n; j++) {
                    matrix[i][j] = Math.min(matrix[i][j],
 matrix[i][k] + matrix[k][j]);
                }
            }
        }

        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                if (matrix[i][j] == (int)(1e9)) {
                    matrix[i][j] = -1;
                }
            }
        }
    }
}

public class tUf {
    public static void main(String[] args) {
        int V = 4;
        int[][] matrix = new int[V][V];

        for (int i = 0; i < V; i++) {
            for (int j = 0; j < V; j++) {
                matrix[i][j] = -1;
            }
        }

        matrix[0][1] = 2;
        matrix[1][0] = 1;
        matrix[1][2] = 3;
        matrix[3][0] = 3;
        matrix[3][1] = 5;
        matrix[3][2] = 4;

        Solution obj = new Solution();
        obj.shortest_distance(matrix);

        for (int i = 0; i < V; i++) {
            for (int j = 0; j < V; j++) {
                System.out.print(matrix[i][j] + " ");
            }
            System.out.println("");
        }
    }
}
```

```
class Solution {
    int[][] grid;
    boolean[][] seen;

    public int area(int r, int c) {
        if (r < 0 || r >= grid.length || c < 0 || c >= grid[0].length ||
                seen[r][c] || grid[r][c] == 0)
            return 0;
        seen[r][c] = true;
        return (1 + area(r+1, c) + area(r-1, c)
                  + area(r, c-1) + area(r, c+1));
    }

    public int maxAreaOfIsland(int[][] grid) {
        this.grid = grid;
        seen = new boolean[grid.length][grid[0].length];
        int ans = 0;
        for (int r = 0; r < grid.length; r++) {
            for (int c = 0; c < grid[0].length; c++) {
                ans = Math.max(ans, area(r, c));
            }
        }
        return ans;
    }
}
```

## Challenges Experienced

The major challenge i experienced in traversing Diagonally also in Maximum Area of Island Problem. It took me about an hour to figure it out.

## Resources Used

Youtube, TakeUForward, GeeksForGeeks