# 2023-06-04 - Handout – DFS & BFS

## Q1. Island Perimeter

Link: https://leetcode.com/problems/island-perimeter/

You are given **row x col grid** representing a map where **grid[i][j] = 1** represents land and **grid[i][j] = 0** represents water.

Grid cells are connected **horizontally/vertically** (not diagonally). The grid is surrounded by water, and there is exactly one island (i.e., one or more connected land cells).

The island doesn't have "lakes", meaning the water inside isn't connected to the water around the island. One cell is a square with side length 1. The grid is rectangular, width and height don't exceed 100. Determine the perimeter of the island.

Example 1:
**Input**: grid = [[0,1,0,0],[1,1,1,0],[0,1,0,0],[1,1,0,0]]
**Output**: 16

Example 2:
**Input**: grid = [[1]]
**Output**: 4

Example 3:
**Input**: grid = [[1,0]]
**Output**: 4

Constraints:

- row == grid.length
- col == grid[i].length
- 1 <= row, col <= 100
- grid[i][j] is 0 or 1.
- There is exactly one island in grid.

## Q2. Shortest Bridge

Link: https://leetcode.com/problems/shortest-bridge/

You are given an **n x n** binary matrix **grid** where 1 represents land and 0 represents water. An island is a 4-directionally connected group of 1's not connected to any other 1's. There are exactly two islands in grid.

You may change 0's to 1's to connect the two islands to form one island.

Return the smallest number of 0's you must flip to connect the two islands. Return true if and only if it is possible to pick up and drop off all passengers for all the given trips.

Example 1:
**Input**: grid = [[0,1],[1,0]]
**Output**: 1

Example 2:
**Input**: grid = [[0,1,0],[0,0,0],[0,0,1]]
**Output**: 2

Example 3:
**Input**:[[1,1,1,1,1],[1,0,0,0,1],[1,0,1,0,1],[1,0,0,0,1],[1,1,1,1,1]]
**Output**: 1

Constraints:

- n == grid.length == grid[i].length
- 2 <= n <= 100
- grid[i][j] is either 0 or 1.
- There are exactly two islands in grid.

## Q3. Smallest String With Swaps

Link: https://leetcode.com/problems/smallest-string-with-swaps/

You are given a string **s**, and an array of pairs of indices in the string **pairs** where **pairs[i] = [a, b]** indicates 2 indices(0-indexed) of the string.

You can swap the characters at any pair of indices in the given pairs any number of times.

Return the lexicographically smallest string that s can be changed to after using the swaps.

Example 1:
**Input**: s = "dcab", pairs = [[0,3],[1,2]]
**Output**: "bacd"
**Explaination**:
Swap s[0] and s[3], s = "bcad"
Swap s[1] and s[2], s = "bacd"

Example 2:
**Input**: s = "dcab", pairs = [[0,3],[1,2],[0,2]]
**Output**: "abcd"
**Explaination**:
Swap s[0] and s[3], s = "bcad"
Swap s[0] and s[2], s = "acbd"
Swap s[1] and s[2], s = "abcd"

Example 3:
**Input**: s = "cba", pairs = [[0,1],[1,2]]
**Output**: "abc"
**Explaination**:
Swap s[0] and s[1], s = "bca"
Swap s[1] and s[2], s = "bac"
Swap s[0] and s[1], s = "abc"

Constraints:

- 1 <= s.length <= 10^5
- 0 <= pairs.length <= 10^5
- 0 <= pairs[i][0], pairs[i][1] < s.length
- s only contains lower case English letters.

## Q4. Longest Increasing Path in a Matrix

Link: https://leetcode.com/problems/longest-increasing-path-in-a-matrix/

Given an **m x n** integers **matrix**, return the length of the longest increasing path in **matrix**.

From each cell, you can either move in four directions: left, right, up, or down. You may not move diagonally or move outside the boundary (i.e., wrap-around is not allowed).

Example 1:

**Input**: matrix = [[9,9,4],[6,6,8],[2,1,1]]

**Output**: 4

**Explanation**: The longest increasing path is [1, 2, 6, 9].

Example 2:

**Input**: matrix = [[3,4,5],[3,2,6],[2,2,1]]

**Output**: 4

**Explanation**: The longest increasing path is [3, 4, 5, 6]. Moving diagonally is not allowed.