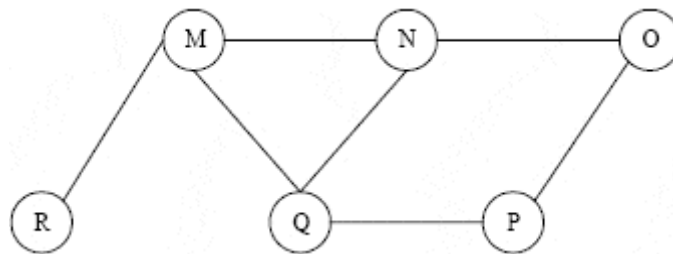


NSF IMPEL assignment and quiz

Created by Andy(Xiang-Yu) Cui

Quiz:

P1: The Breadth First Search algorithm has been implemented using the queue data structure. One possible order of visiting the nodes of the following graph is ?



- A: M -> N -> O -> P -> Q -> R
- B: N -> Q -> M -> P -> O -> R
- C: Q -> M -> N -> P -> R -> O
- D: Q -> M -> N -> P -> O -> R

P2: The data structure used for Breadth First Search Algorithm is:

- A: STACK
- B: QUEUE
- C: BOTH STACK & QUEUE
- D: NONE OF THESE

Assignment:

There is a 2D grid world build by string 0 and 1 , which 0 is water, and 1 is land. Let's we calculate the number of the island. So the island always around by water, meanwhile each island can only be formed by horizontally and/or vertically adjacent land connections. In additional, we assume the grid is surrounded by water on all four sides.

Example 1:

Input: grid = [

["1", "1", "1", "1", "0"],

["1", "1", "0", "1", "0"],

["1", "1", "0", "0", "0"],

["0", "0", "0", "0", "0"],

]

Output: 1

Example 2:

Input: grid = [

["1", "1", "0", "0", "0"],

["1", "1", "0", "0", "0"],

["0", "0", "1", "0", "0"],

["0", "0", "0", "1", "1"],

]

Output: 3

Formal Solutions:

Quiz:

- P1: C
- P2: B

Assignment:

```
1 def num_islands(grid: List[List[str]]) -> int:
2     if grid is None or len(grid) == 0:
3         return 0
4     result = 0
5     row = len(grid)
6     col = len(grid[0])
7     queue = []
8
9     for i in range(0, row):
10        for j in range(0, col):
11            if grid[i][j] == '1':
12                result += 1
13                queue.append([i, j])
14                grid[i][j] = '0'
15                while len(queue) > 0:
16                    current = queue.pop()
17                    x = current[0]
18                    y = current[1]
19                    if x - 1 >= 0 and grid[x - 1][y] == '1':
20                        queue.append([x - 1, y])
21                        grid[x - 1][y] = '0'
22                    if y - 1 >= 0 and grid[x][y - 1] == '1':
23                        queue.append([x, y - 1])
24                        grid[x][y - 1] = '0'
25                    if x + 1 < row and grid[x + 1][y] == '1':
26                        queue.append([x + 1, y])
27                        grid[x + 1][y] = '0'
28                    if y + 1 < col and grid[x][y + 1] == '1':
29                        queue.append([x, y + 1])
30                        grid[x][y + 1] = '0'
31        return result
```