**16."The Game of Life, also known simply as Life, is a cellular automaton devised by the British mathematician John Horton Conway in 1970." The board is made up of an m x n grid of cells, where each cell has an initial state: live (represented by a 1) or dead (represented by a 0). Each cell interacts with its eight neighbors (horizontal, vertical, diagonal) using the following four rules**

**Any live cell with fewer than two live neighbors dies as if caused by under-population.**

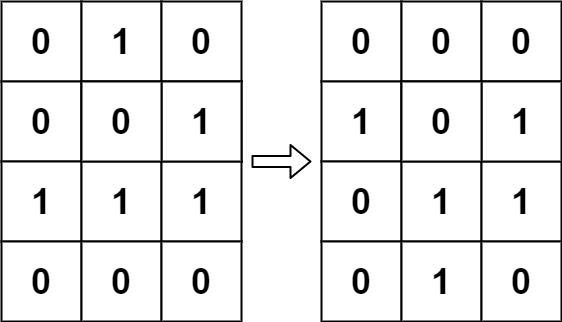
**Any live cell with two or three live neighbors lives on to the next generation.**

**Any live cell with more than three live neighbors dies, as if by over-population.**

**Any dead cell with exactly three live neighbors becomes a live cell, as if by reproduction.**

**The next state is created by applying the above rules simultaneously to every cell in the current state, where births and deaths occur simultaneously. Given the current state of the m x n grid board, return *the next state*.**

**Example 1:**

****

**Input: board = [[0,1,0],[0,0,1],[1,1,1],[0,0,0]]**

**Output: [[0,0,0],[1,0,1],[0,1,1],[0,1,0]]**

**Aim:**

To compute the next state of a given m x n grid where each cell is either alive (1) or dead (0), based on four biological rules applied simultaneously to all cells.

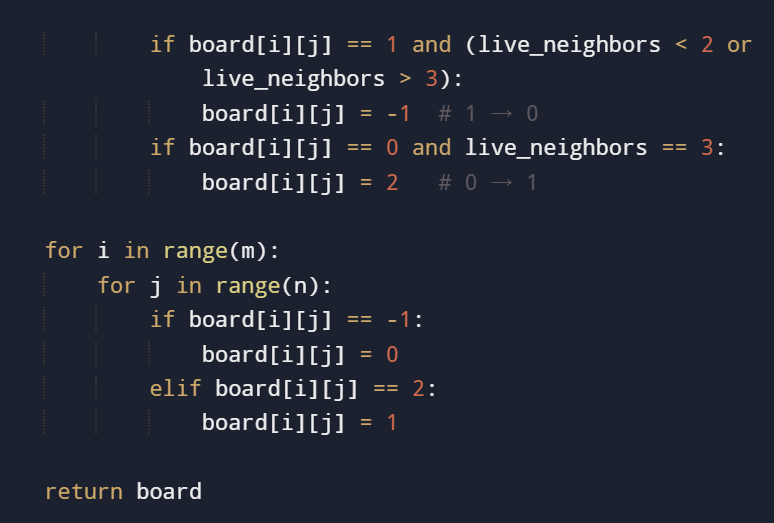
Rules:

1. Any live cell with fewer than 2 live neighbors dies (underpopulation).
2. Any live cell with 2 or 3 live neighbors lives on.
3. Any live cell with more than 3 live neighbors dies (overpopulation).
4. Any dead cell with exactly 3 live neighbors becomes alive (reproduction).

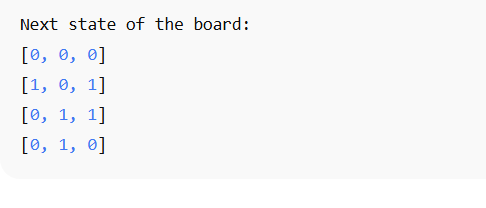
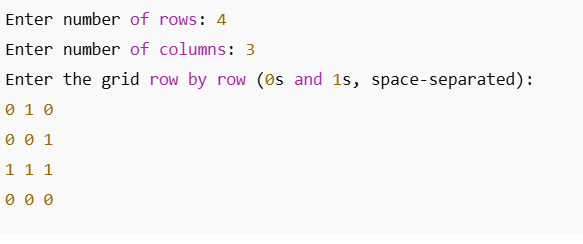
**Algorithm:**

1. Traverse each cell in the board.
2. For each cell, count live neighbors (all 8 directions).
3. Apply the rules, but store state changes in-place using encoding:
   * 1 → -1: live becomes dead
   * 0 → 2: dead becomes live
4. After scanning all cells, finalize state:
   * Change -1 to 0 (dead)
   * Change 2 to 1 (live)

**CODE:**



**Input and output:**



Result : the given program is executed successfully and output is verified

**Performance Analysis:**

* **Time Complexity:** O(m×n) *O* (*m*×*n*)
* **Space Complexity:** O(1)*O*(1) *(in-place using state encoding)*