GAME OF DEATH

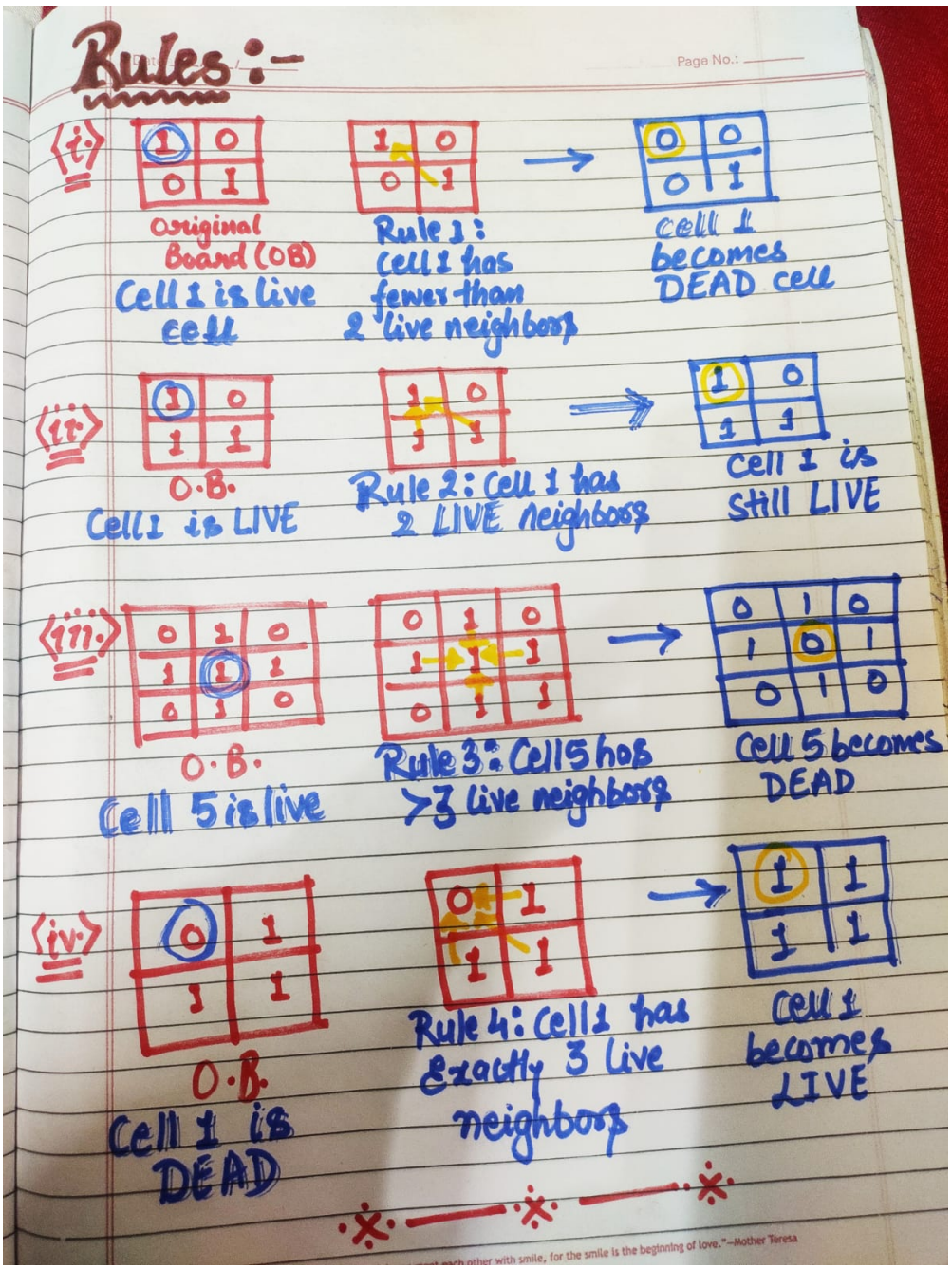
 # PROBLEM STATEMENT :

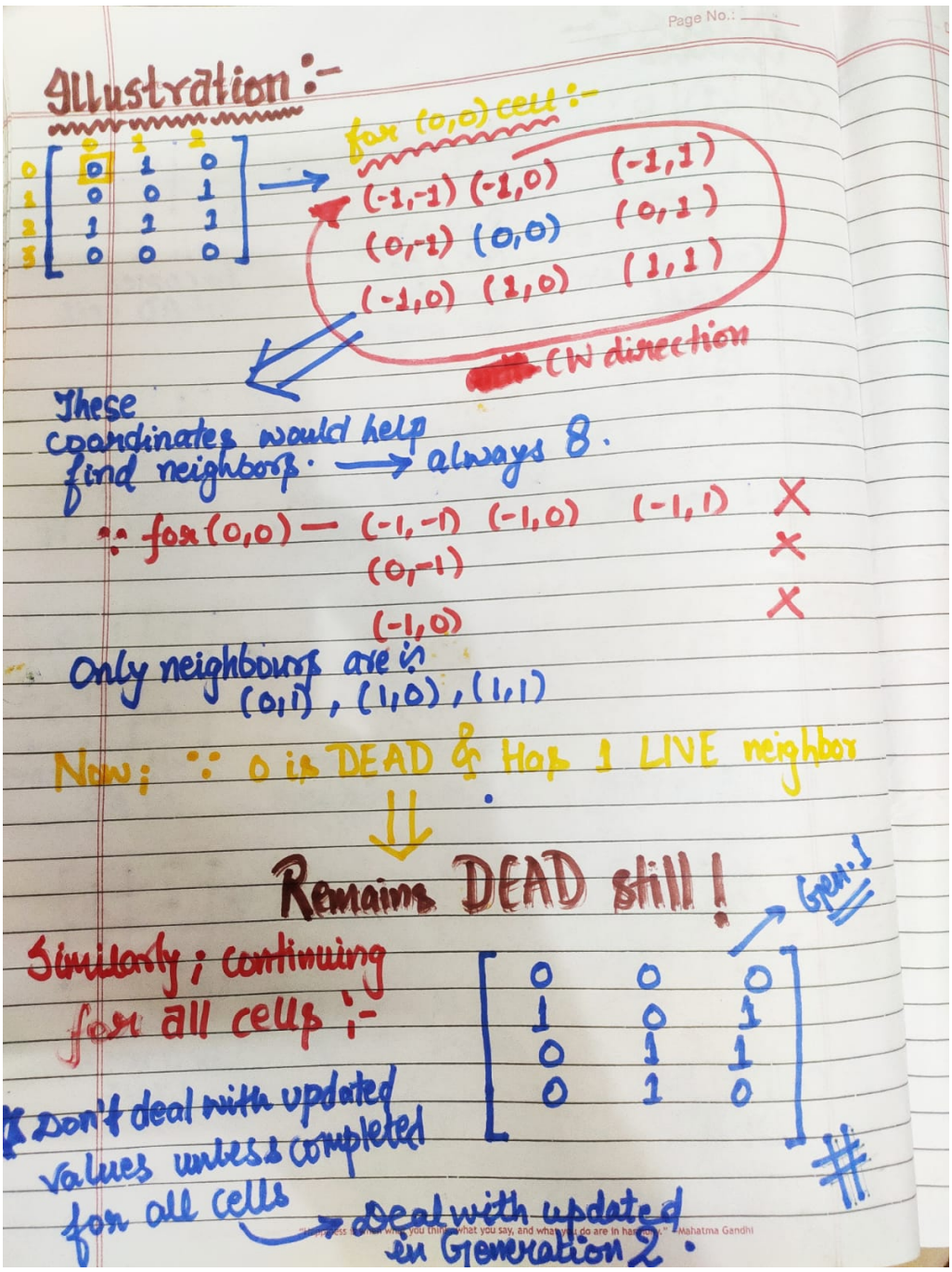
Implement an algorithm that produces the next move in the game of death. Basically given a two dimensional array it will have either values 1 (live cell) or 0 (dead cell).

1. A Live cell will live only if it has two or three live neighbors All other cases die.
2. A dead cell with exactly three live neighbors will live otherwise no change, dead.

Transform the array by using above two rules.

 # PICTORIAL REPRESENTATION :





 # ALGORITHM :

1. Make a copy of the original board which will remain unchanged throughout the process.
2. Iterate the cells of the Board one by one.
3. While computing the results of the rules, use the copy board and apply the result in the original board.

CODE :

```
package GameOfDeath;

public class GameOfDeath {

        public static void main(String[] args) {
            int[][] cellsData={
                        {0,1,0},
                        {0,0,1},
                        {1,1,1},
                        {0,0,0}
                    };

                    System.out.println("Before");
                    printData(cellsData);
                    gameOfLife(cellsData);
                    System.out.println("\n\nAfter");
                    printData(cellsData);
        }


        private static void printData(int[][] cellsData){

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

            }

        public static void gameOfLife(int[][] board) {

            int height = board.length;
      int width = board[0].length;
      int[][] next = new int[height][width];
      int[][] directions = {{-1, -1}, {-1, 0}, {-1, 1}, {0, 1}, {1, 1}, {1, 0}, {1, -1}, {0, -1}};

      for (int i = 0; i < board.length; i++) {
          for (int j = 0; j < board[0].length; j++) {
              int liveCellsCount = 0;
              //count all its live cells

              for (int[] dir : directions) {
                  int x = i + dir[0];
```

```
                int y = j + dir[1];
                if (x >= 0 && y >= 0 && x < height && y < width && board[x][y] == 1) {
                    liveCellsCount++;
                }
            }

            if (board[i][j] == 1) {
                if (liveCellsCount <= 3 && liveCellsCount >= 2) {
                    next[i][j] = 1;
                }
            } else if (board[i][j] == 0) {
                if (liveCellsCount == 3) {
                    next[i][j] = 1;
                }
            }
        }
    }

    for (int i = 0; i < board.length; i++) {
        for (int j = 0; j < board[0].length; j++) {
            board[i][j] = next[i][j];
        }
    }
}

}
```

 # OUTPUT :