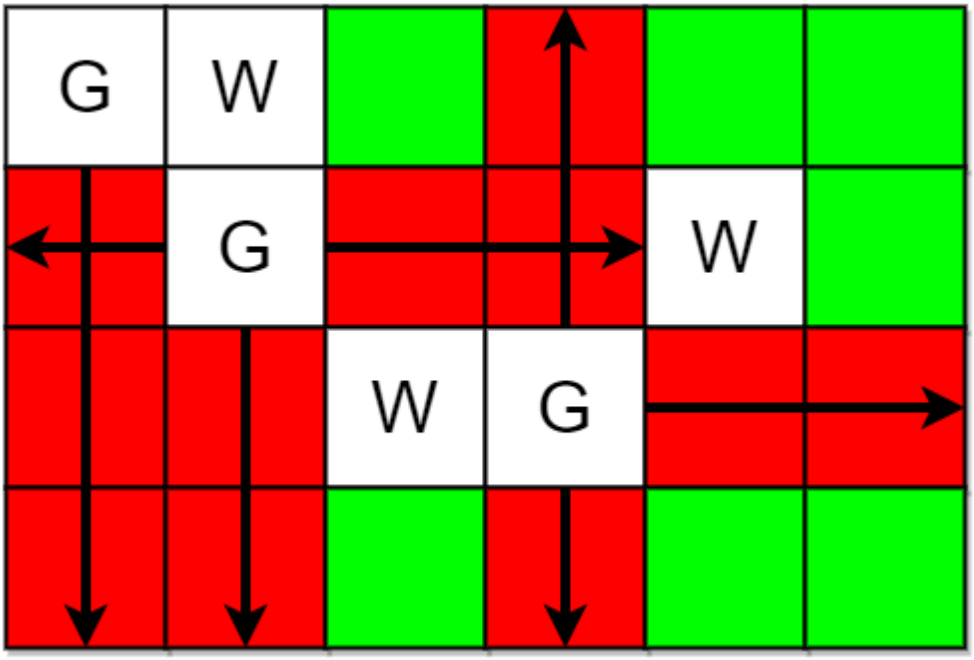
2257. Count Unguarded Cells in the Grid



This problem is about counting the number of "unguarded" cells in a grid where some cells contain guards, some contain walls, and the guards "watch" cells in the four cardinal directions until blocked by either another guard or a wall. Let's break down my solution step-by-step:

1. **Setup the Grid**:
   - `grid` is a 2D list (a grid) of size `m` x `n`, initialized to 0. Each cell in `grid` will represent a cell's status:
     - `0` means an unguarded cell,
     - `1` means a cell with a guard,
     - `2` means a cell with a wall,
     - `3` means a cell guarded by a guard.
2. **Place Guards and Walls**:

- The code loops over the list of guards and walls and places them in the `grid`:
    - Guards are marked with `1` in the respective cells,
    - Walls are marked with `2` in their cells.

3. **Mark Guarded Cells**:
    - The function `mark_guarded` takes a guard's position `(r, c)` and marks cells in the four directions (down, up, right, left) as guarded (set to `3`) until a wall or another guard is encountered.
        - Each direction is checked one at a time:
            - **Downward** (`for row in range(r+1, m)`): goes down from `(r, c)` until it finds a wall or guard, marking cells as `3`.
            - **Upward** (`for row in reversed(range(0, r))`): goes up from `(r, c)`, marking cells as guarded.
            - **Rightward** (`for col in range(c+1, n)`): goes right until blocked.
            - **Leftward** (`for col in reversed(range(0, c))`): goes left until blocked.
4. **Mark All Guarded Cells**:
    - After defining `mark_guarded`, the code loops through all guard positions and calls `mark_guarded` for each guard, updating the grid with all guarded cells.
5. **Count Unguarded Cells**:
    - Finally, the code counts all cells that remain `0` (ungarded) by looping through each row of `grid`. For each cell equal to `0`, it increments `res` (the result count).