Game of Life

Prompt

Conway's Game of Life is a zero-player game defined as a grid of cells, each having a state that corresponds to either "dead" or "alive." The cells follow the following rules:

- Any live cell with two or three live neighbours survives.

- Any dead cell with three live neighbours becomes a live cell.

- All other live cells die in the next generation. Similarly, all other dead cells stay dead.

Your task, given a starting condition for the game, is to compute the next generation of cells and find how many cells are alive.

Input Format

- The first line contains an integer n, denoting the size of the grid.

- The next n lines contain n space-separated values each, either 0 or 1.

Output Format

Your output must contain a single integer denoting how many cells are alive in the next generation.

Sample Input/Output

Input	Output
8	
0 1 0 1 0 0 0 1	
0 1 1 1 1 1 1 1	
0 1 0 1 1 0 1 1	
1 1 0 0 1 1 0 0	33
1 0 1 1 1 0 0 0	
0 1 1 0 0 1 0 0	
0 1 0 1 1 1 0 0	
1 0 0 1 1 0 0 0	

Sample Program

```python
n=int(input())

grid=[]

for i in range(n):
  grid.append(list(map(int, input().strip().split())))
```

```python
newGrid = grid.copy()

def safeCell(i, j):
    try:
        assert i>-1 and j>-1 # prevent back-indexes
        return grid[i][j]
    except:
        return 0

for i in range(n):
    for j in range(n):
        total = 0
        cells_to_check=[
            [i-1, j-1],
            [i-1, j],
            [i-1, j+1],
            [i, j-1],
            [i, j+1],
            [i+1, j-1],
            [i+1, j],
            [i+1, j+1],
        ]
        for cell in cells_to_check:
            total+=safeCell(cell[0], cell[1])

        if grid[i][j]  == 1:
            if (total < 2) or (total > 3):
                newGrid[i][j] = 0
        else:
            if total == 3:
                newGrid[i][j] = 1

print(sum([sum(i) for i in newGrid]))
```