```
import random
import copy


GRID_WIDTH = 10
GRID_HEIGHT = 10
MAX_GENERATIONS = 20


def initialize_grid(width, height):
    return [[random.randint(0, 1) for _ in range(width)] for _ in range(height)]


def count_live_neighbors(grid, i, j):
    live_neighbors = 0
    directions = [(-1, -1), (-1, 0), (-1, 1),
                  ( 0, -1),          ( 0, 1),
                  ( 1, -1), ( 1, 0), ( 1, 1)]

    for dx, dy in directions:
        x = (i + dx) % len(grid)
        y = (j + dy) % len(grid[0])
        live_neighbors += grid[x][y]

    return live_neighbors


def apply_rules(grid, i, j):
    live_neighbors = count_live_neighbors(grid, i, j)
    if grid[i][j] == 1:
        return 1 if live_neighbors == 2 or live_neighbors == 3 else 0
    else:
        return 1 if live_neighbors == 3 else 0


def update_grid(grid):
    new_grid = copy.deepcopy(grid)
    for i in range(len(grid)):
        for j in range(len(grid[0])):
            new_grid[i][j] = apply_rules(grid, i, j)
    return new_grid


def display_grid(grid):
    for row in grid:
        print(' '.join(str(cell) for cell in row))
    print("\n" + "="*20 + "\n")


def count_alive_cells(grid):
    return sum(sum(row) for row in grid)


def game_of_life(grid_width, grid_height, max_generations):
    grid = initialize_grid(grid_width, grid_height)
```

```
    print("Initial Grid:")
    display_grid(grid)

    for generation in range(max_generations):
        print(f"Generation {generation + 1}:")
        grid = update_grid(grid)
        # display_grid(grid)
        alive_cells = count_alive_cells(grid)
        print(f"Number of alive cells: {alive_cells}")


if __name__ == "__main__":
    game_of_life(GRID_WIDTH, GRID_HEIGHT, MAX_GENERATIONS)
```

```
Initial Grid:
0 0 0 1 0 0 0 0 1 0
1 0 1 1 1 0 1 0 1 1
0 0 0 1 0 1 1 0 0 0
1 1 0 1 0 1 1 0 1 1
0 1 1 1 1 1 0 1 1 1
1 0 0 0 0 0 0 0 0 1
1 1 0 1 0 0 1 0 1 0
0 1 0 1 0 0 1 0 1 0
0 0 0 0 0 0 1 0 0 1
1 1 1 1 1 1 0 0 0 0

====================

Generation 1:
Number of alive cells: 28
Generation 2:
Number of alive cells: 27
Generation 3:
Number of alive cells: 26
Generation 4:
Number of alive cells: 18
Generation 5:
Number of alive cells: 19
Generation 6:
Number of alive cells: 25
Generation 7:
Number of alive cells: 21
Generation 8:
Number of alive cells: 20
Generation 9:
Number of alive cells: 23
Generation 10:
Number of alive cells: 29
Generation 11:
Number of alive cells: 21
Generation 12:
Number of alive cells: 28
Generation 13:
Number of alive cells: 19
Generation 14:
Number of alive cells: 23
Generation 15:
Number of alive cells: 17
Generation 16:
```

```
Number of alive cells: 22
Generation 17:
Number of alive cells: 13
Generation 18:
Number of alive cells: 19
Generation 19:
Number of alive cells: 17
Generation 20:
Number of alive cells: 20
```

Start coding or generate with AI.