

```
1 //
2 // Created by hfwei on 2023/10/19.
3 //
4
5 #include <stdio.h>
6 #include <stdlib.h>
7 #include <time.h>
8 #include <unistd.h>
9
10 // Define grid dimensions
11 #define ROWS 20
12 #define COLS 40
13
14 // Function to initialize the grid randomly
15 void initializeGrid(int grid[ROWS][COLS]) {
16     for (int i = 0; i < ROWS; i++) {
17         for (int j = 0; j < COLS; j++) {
18             grid[i][j] = rand() % 2; // 0 (dead) or 1 (alive)
19         }
20     }
21 }
22
23 // Function to print the grid
24 void printGrid(int grid[ROWS][COLS]) {
25     for (int i = 0; i < ROWS; i++) {
26         for (int j = 0; j < COLS; j++) {
27             if (grid[i][j] == 1) {
28                 printf("#"); // Alive cell
29             } else {
30                 printf(" "); // Dead cell
31             }
32         }
33         printf("\n");
34     }
35     printf("\n");
36 }
37
38 // Function to update the grid for the next generation
39 void updateGrid(int grid[ROWS][COLS]) {
40     int newGrid[ROWS][COLS];
41
42     for (int i = 0; i < ROWS; i++) {
43         for (int j = 0; j < COLS; j++) {
44             int neighbors = 0;
45
46             // Count neighbors
47             for (int x = -1; x <= 1; x++) {
48                 for (int y = -1; y <= 1; y++) {
49                     if (x == 0 && y == 0) { continue; } // Skip the current
cell
50                     int newX = i + x;
51                     int newY = j + y;
52
```

```

53         if (newX >= 0 && newX < ROWS && newY >= 0 && newY < COLS) {
54             neighbors += grid[newX][newY];
55         }
56     }
57 }
58
59 // Apply Game of Life rules
60 if (grid[i][j] == 1) {
61     newGrid[i][j] = (neighbors == 2 || neighbors == 3) ? 1 : 0;
62 } else {
63     newGrid[i][j] = (neighbors == 3) ? 1 : 0;
64 }
65 }
66 }
67
68 // Update the grid
69 for (int i = 0; i < ROWS; i++) {
70     for (int j = 0; j < COLS; j++) {
71         grid[i][j] = newGrid[i][j];
72     }
73 }
74 }
75
76 int main() {
77     int grid[ROWS][COLS];
78
79     // Seed the random number generator with the current time
80     srand(time(NULL));
81
82     // Initialize the grid
83     initializeGrid(grid);
84
85     // Number of generations
86     int generations = 50;
87
88     for (int gen = 0; gen < generations; gen++) {
89         system("clear"); // Use "clear" on Unix-based systems (Linux,
macOS)
90         printf("Generation %d:\n", gen);
91         printGrid(grid);
92         updateGrid(grid);
93         sleep(1); // Sleep for 100ms
94     }
95
96     return 0;
97 }

```