```c
//
// Created by hfwei on 2023/10/19.
//

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <unistd.h>

// Define grid dimensions
#define ROWS 20
#define COLS 40

// Function to initialize the grid randomly
void initializeGrid(int grid[ROWS][COLS]) {
  for (int i = 0; i < ROWS; i++) {
    for (int j = 0; j < COLS; j++) {
      grid[i][j] = rand() % 2;  // 0 (dead) or 1 (alive)
    }
  }
}

// Function to print the grid
void printGrid(int grid[ROWS][COLS]) {
  for (int i = 0; i < ROWS; i++) {
    for (int j = 0; j < COLS; j++) {
      if (grid[i][j] == 1) {
        printf("#");  // Alive cell
      } else {
        printf(" ");  // Dead cell
      }
    }
    printf("\n");
  }
  printf("\n");
}

// Function to update the grid for the next generation
void updateGrid(int grid[ROWS][COLS]) {
  int newGrid[ROWS][COLS];

  for (int i = 0; i < ROWS; i++) {
    for (int j = 0; j < COLS; j++) {
      int neighbors = 0;

      // Count neighbors
      for (int x = -1; x <= 1; x++) {
        for (int y = -1; y <= 1; y++) {
          if (x == 0 && y == 0) { continue; }  // Skip the current
cell
          int newX = i + x;
          int newY = j + y;

          if (newX >= 0 && newX < ROWS && newY >= 0 && newY < COLS) {
            neighbors += grid[newX][newY];
          }
        }
      }

      // Apply Game of Life rules
      if (grid[i][j] == 1) {
        newGrid[i][j] = (neighbors == 2 || neighbors == 3) ? 1 : 0;
      } else {
        newGrid[i][j] = (neighbors == 3) ? 1 : 0;
      }
    }
  }

  // Update the grid
  for (int i = 0; i < ROWS; i++) {
    for (int j = 0; j < COLS; j++) {
      grid[i][j] = newGrid[i][j];
    }
  }
}

int main() {
  int grid[ROWS][COLS];

  // Seed the random number generator with the current time
  srand(time(NULL));

  // Initialize the grid
  initializeGrid(grid);

  // Number of generations
  int generations = 50;

  for (int gen = 0; gen < generations; gen++) {
    system("clear");  // Use "clear" on Unix-based systems (Linux, macOS)
    printf("Generation %d:\n", gen);
    printGrid(grid);
    updateGrid(grid);
    sleep(1);  // Sleep for 100ms
  }

  return 0;
}
```