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

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

#define SIZE 6
const int board[SIZE][SIZE] = {
    { 0 },
    { 0, 1, 1, 0, 0, 0 },
    { 0, 1, 1, 0, 0, 0 },
    { 0, 0, 0, 1, 1, 0 },
    { 0, 0, 0, 1, 1, 0 },
    { 0 }
};

//const int board[SIZE][SIZE] = {
//    [1][1] = 1, [1][2] = 1,
//    [2][1] = 1, [2][2] = 1,
//    [3][3] = 1, [3][4] = 1,
//    [4][3] = 1, [4][4] = 1
//};

int main() {
  // extended board
  int old_board[SIZE + 2][SIZE + 2] = { 0 };

  for (int row = 1; row <= SIZE; row++) {
    for (int col = 1; col <= SIZE; col++) {
      old_board[row][col] = board[row - 1][col - 1];
    }
  }

  // print the original board
  for (int row = 1; row <= SIZE; row++) {
    for (int col = 1; col <= SIZE; col++) {
      printf("%c ", old_board[row][col] ? '*' : ' ');
    }
    printf("\n");
  }
  system("clear");  // clear the screen/terminal

  int new_board[SIZE + 2][SIZE + 2] = { 0 };

  for (int round = 1; round < 10; round++) {
    for (int row = 1; row <= SIZE; row++) {
      for (int col = 1; col <= SIZE; col++) {
        // count the number of neighbours of old_board[row][col]
        int neighbours =
            old_board[row - 1][col - 1] +
```

```c
                old_board[row - 1][col] +
                old_board[row - 1][col + 1] +
                old_board[row][col - 1] +
                old_board[row][col + 1] +
                old_board[row + 1][col - 1] +
                old_board[row + 1][col] +
                old_board[row + 1][col + 1];

        // evaluate the new board
        if (old_board[row][col]) {  // old_board[row][col] is alive
          new_board[row][col] = (neighbours == 2 || neighbours == 3);
        } else {  // old_board[row][col] is dead
          new_board[row][col] = (neighbours == 3);
        }
      }
    }

    // print the new board
    for (int row = 1; row <= SIZE; row++) {
      for (int col = 1; col <= SIZE; col++) {
        printf("%c ", new_board[row][col] ? '*' : ' ');
      }
      printf("\n");
    }

    // sleep for a while
    // Linux: #include <unistd.h>
    sleep(1);
    // Windows: #include <windows.h>: Sleep(ms)
    // Sleep(1000);

    // clear the screen
    // Linux: #include <stdlib.h>
    system("clear");
    // Windows: #include <stdlib.h> system("clr);
    // system("clr");

    // start the next round
    for (int row = 1; row <= SIZE; row++) {
      for (int col = 1; col <= SIZE; col++) {
        old_board[row][col] = new_board[row][col];
      }
    }
  }

  return 0;
}
```