```c

//*********************************************************
//
// ライフゲームのサンプル
//
// Edition History
//
// 変更日付    Rev.  変更者       変更内容
// ----------  ----  ----------  ------------------------------------------------
// 2016/04/26  1.0   M.Hoshi     新規作成
// 2020/04/14  1.1   M.Hoshi     画面制御をエスケープシーケンスに変更
//
//*********************************************************************

#include <stdio.h>  // printf()で使用
#include <stdlib.h> // srand()で使用
#include <string.h> // memset()で使用

#define WORLD_H 38  // 縦の幅
#define WORLD_W 76  // 横の幅

// プロトタイプ宣言
int init_map(char map[WORLD_H][WORLD_W], char *inbuf);
int init_file(char map[WORLD_H][WORLD_W], char *inbuf);
void print_map(char map[WORLD_H][WORLD_W]);
void next_gen(char map[WORLD_H][WORLD_W]);
int get_lifeCnt(char cell[WORLD_H][WORLD_W], int h_pos, int w_pos);


//*******************************************************
//
// メイン
//
//*******************************************************

int main(void) {

    char map[WORLD_H][WORLD_W]; // マップテーブル
    char inbuf[256];            // 文字列の入力バッファ
    int gen_No;                 // 世代番号
    int result = 0;

    memset(map, 0, sizeof(map));    // マップを0クリアする
    printf("¥x1b[2J");              // 画面クリア

    printf("ファイル名か乱数の種の入力(0で終了) : ");
    scanf_s("%s", inbuf, sizeof(inbuf));
    if (inbuf[0] >= '0' && inbuf[0] <= '9') {   // 先頭文字が数字なら乱数で初期化
        result = init_map(map, inbuf);  // 乱数モードでマップの初期化
    }
    else {
        result = init_file(map, inbuf); // ファイルモードでマップの初期化
    }

    if (result == 0) {      // エラーがなければ、

        for (gen_No = 0; gen_No < 1000; gen_No++) {

            printf("¥x1b[2;1H");            //カーソル位置を、高さ1行目、横1行目に移動
            printf("世代 = %d¥n", gen_No);  //世代の表示
            print_map(map); // マップの表示
            next_gen(map);  // 次世代に進める
        }
    }
    system("pause");        // ポーズをする
    return 0;
}


//*******************************************************
//
// 乱数によるマップの初期化
//
//*******************************************************

```

```c
int init_map(char map[WORLD_H][WORLD_W], char *inbuf) {

    int h_pos, w_pos;
    int seed;
    int randval;

    // 乱数モードで初期化
    seed = atoi(inbuf); // 乱数の種を数値(10進数)に変換
    srand(seed);        // 乱数の種を設定
    for (h_pos = 0; h_pos < WORLD_H; h_pos++) {
        for (w_pos = 0; w_pos < WORLD_W; w_pos++) {
            randval = rand() % 10;
            if (randval < 7)
                map[h_pos][w_pos] = 0;
            else
                map[h_pos][w_pos] = 1;
        }
    }
    return 0;
}


//*****************************************************
//
// ファイルを読込んでのマップの初期化
//
//*****************************************************

int init_file(char map[WORLD_H][WORLD_W], char *inbuf) {

    FILE *fp;
    char buf[128];
    int h_pos, w_pos;
    int length;     // 1行の文字数
    int errflag;

    // ファイルからパターンを読み込んで、初期パターンとしてセット
    if ((fopen_s(&fp, inbuf, "r")) == 0) {              // テキストモードでファイルのオープン

        h_pos = 0;      // 行の初期化
        while (fgets(buf, sizeof(buf), fp) != NULL) {   // 1行の読み込み
            if ((length = strlen(buf)) > WORLD_W) {     // 文字列が最大幅を超えたら、
                length = WORLD_W;                       // 最大幅にする
            }
            for (w_pos = 0; w_pos < length; w_pos++) {  // 横方向のループ
                map[h_pos][w_pos] = (buf[w_pos] == '1') ? 1 : 0;
            }
            h_pos++;
            if (h_pos > WORLD_H) {  // 最大行数を超えたら終了
                break;
            }
        }
        fclose(fp);     // ファイルのクローズ
        errflag = 0;    // エラーなしを返す
    }
    else {
        printf("ファイルがオープンできません¥n");
        errflag = 1;    // エラーを返す
    }
    return errflag;
}


//*****************************************************
//
// マップの表示
//
//*****************************************************

void print_map(char map[WORLD_H][WORLD_W]) {

    int h_pos,w_pos;

    for (h_pos = 0; h_pos < WORLD_H; h_pos++) {
        for (w_pos = 0; w_pos < WORLD_W; w_pos++) {
```

```
151             if (map[h_pos][w_pos] > 0)
152                 printf("@");
153             else
154                 printf(".");
155         }
156         printf("¥n");   // 改行
157     }
158 }
159
160
161 //*************************************************
162 //
163 // 世代を進める
164 //
165 //*************************************************
166
167 void next_gen(char map[WORLD_H][WORLD_W]) {
168
169     char tmpmap[WORLD_H][WORLD_W];
170     int h_pos, w_pos;
171     int lifeCnt;        // 生存数
172
173     memcpy(tmpmap, map, sizeof tmpmap); // マップのコピー
174
175     for (h_pos = 0; h_pos < WORLD_H; h_pos++) {
176         for (w_pos = 0; w_pos < WORLD_W; w_pos++) {
177
178             lifeCnt = get_lifeCnt(tmpmap, h_pos, w_pos);    // 生存数を求める
179
180             if (tmpmap[h_pos][w_pos] == 0 && lifeCnt == 3)
181                 map[h_pos][w_pos] = 1;      // 誕生
182             if (tmpmap[h_pos][w_pos] == 1 && (lifeCnt <= 1 || lifeCnt >= 4))
183                 map[h_pos][w_pos] = 0;      // 死滅
184         } // for w
185     } // for h
186 }
187
188
189 //*************************************************
190 //
191 // 自分の周り8セルの生存数を取得
192 //
193 //*************************************************
194
195 int get_lifeCnt(char cell[WORLD_H][WORLD_W], int h_pos, int w_pos) {
196
197     int liveCnt = 0;    // 生存数を初期化
198
199     if (h_pos > 0 && w_pos > 0)
200         liveCnt += cell[h_pos - 1][w_pos - 1];  // 左上
201
202     if (w_pos > 0)
203         liveCnt += cell[h_pos][w_pos - 1];      // 左
204
205     if (h_pos < WORLD_H - 1 && w_pos>0)
206         liveCnt += cell[h_pos + 1][w_pos - 1];  // 左下
207
208     if (h_pos < WORLD_H - 1)
209         liveCnt += cell[h_pos + 1][w_pos];      // 真下
210
211     if (h_pos < WORLD_H - 1 && w_pos < WORLD_W - 1)
212         liveCnt += cell[h_pos + 1][w_pos + 1];  // 右下
213
214     if (w_pos < WORLD_W - 1)
215         liveCnt += cell[h_pos][w_pos + 1];      // 右
216
217     if (h_pos > 0 && w_pos < WORLD_W - 1)
218         liveCnt += cell[h_pos - 1][w_pos + 1];  // 右上
219
220     if (h_pos > 0)
221         liveCnt += cell[h_pos - 1][w_pos];      // 真上
222
223     return liveCnt; // 生存数を返す
224 }
225
```