

“程序设计基础”

课程设计报告

**设计题目 生命游戏**

**姓名 戴宇豪**

**学号 2019216072**

**专业 计算机科学与技术**

**班级 19-5**

**完成日期 2021/1/1**

## （一） 需求和规格说明

### 1.1题目

编程实现生命游戏，使用图形界面完成。

游戏规则如下：

① 如果一个细胞周围有 3 个细胞为生（一个细胞周围共有 8 个细胞），则该细胞为生（即该细胞若原先为死，则转为生，若原先为生，则保持不变）。

② 如果一个细胞周围有 2 个细胞为生，则该细胞的生死状态保持不变；

③ 在其它情况下，该细胞为死（即该细胞若原先为生，则转为死，若原先

为死，则保持不变设定图像中每个像素的初始状态后依据上述的游戏规则演绎生命的变化，由于初始状态和迭代次数不同，将会得到令人叹服的优美图案）。

### 1.2 主要类型

在演化过程中，随着初始状态的不同（即“生”格在初始状态下的个数和位置的分布），演化过程中的状态也会不同。然而，随着进化次数的增加，最终的进化结果可归纳为以下四种类型：

① 静止型：从初始状态开始，经过一定的时间演化，细胞空间趋于空间稳定的形态，其中空间稳定是指每个细胞处于固定状态，不随时间变化

②周期型：经过一定的运行周期后，细胞空间趋于一系列简单的稳定模式或周期模式

③混沌型：经过一段时间后，混沌细胞机制不再具有分形结构的特征

④复杂型：出现复杂的局部结构，或局部混沌，其中一些会继续蔓延。

## （二） 设计

### 2.1初始化

 构建一个二维数组，通过二重循环将所有元素用**Ture**来表示状态“生”，用**False**来表示状态“死”，将每个细胞现态和下个状态设置为“死”；

### 2.2遍历与更新

 对数组中的每个元素cells[i][j]，再次通过二重循环统计其邻居（即cells[i−1][j−1]、cells[i−1][j]、

cells[i−1][j+1]、cells[i][j−1]、cells[i][j+1]、cells[i+1][j−1]、cells[i+1][j]、cells[i+1][j+1]）中数值为**Ture**的元素个数，然后按照规定的法则确定下一时刻该元素的值；根据上面遍历的结果和生命游戏的规则更新下一个细胞空间的细胞，使成为“新的一代”。

### 2.3 可视化

利用EasyX中的函数将细胞地图显示并更新，提供一系列的操作函数来改变细胞状态

### 2.4详细设计展示

表 2.4‑1 初始化中的变量

|  |  |  |  |
| --- | --- | --- | --- |
| 变量 | 变量类型 | 默认值 | 解释 |
| cells[j][i] | cell（自定义的结构类型） | False | j行i列的细胞的状态 |

表 2.4‑2 遍历更新用到的变量

|  |  |  |  |
| --- | --- | --- | --- |
| 变量 | 变量类型 | 默认值 | 解释 |
| cells[j][i] | cell（自定义的结构类型） | False | j行i列的细胞的状态 |
| flag | int | 0 | 一个细胞周围的活细胞数 |

表 2.4‑3 可视化用到的变量与函数

|  |  |  |  |
| --- | --- | --- | --- |
| 函数名称 | 函数参数列表 | 返回值 | 实现功能 |
| Welcome | void | void | 加载游戏欢迎界面 |
| RanCells | void | void | 生成随机的细胞地图 |
| EditCells | MOUSEMSG | void | 收集鼠标信息绘制细胞地图 |
| DelButton | char | void | 处理键盘的信息执行相应操作 |
| SurCellsNum | void | int | 统计目前存活的细胞数 |

表 2.4‑4 宏

|  |  |  |
| --- | --- | --- |
| 宏 | 默认值 | 解释 |
| BOXSIZE | 20 | 细胞边长 |
| BOXNUM | 40 | 每行或每列的细胞数目 |
| GAMELENGTH | (BOXSIZE\*BOXNUM) | 游戏框长度 |
| INFORWIDTH | 150 | 信息框宽度 |
| INFORHEIGHT | GAMELENGTH | 信息框高度 |
| GAP | 10 | 缝隙像素 |
| WNDWIDTH | (GAP\*3+INFORWIDTH+GAMELENGTH) | 窗口宽度 |
| WNDHEIGHT | (GAMELENGTH+GAP\*2) | 窗口高度 |

## （三） 用户手册

根据侧边栏的提示信息查看规则并且进行相关操作

R(r)：随机细胞

O(O)：清空细胞

S(s)：单步演化

D(d)：持续演化

P(p)：停止演化

Q(q)：游戏规则

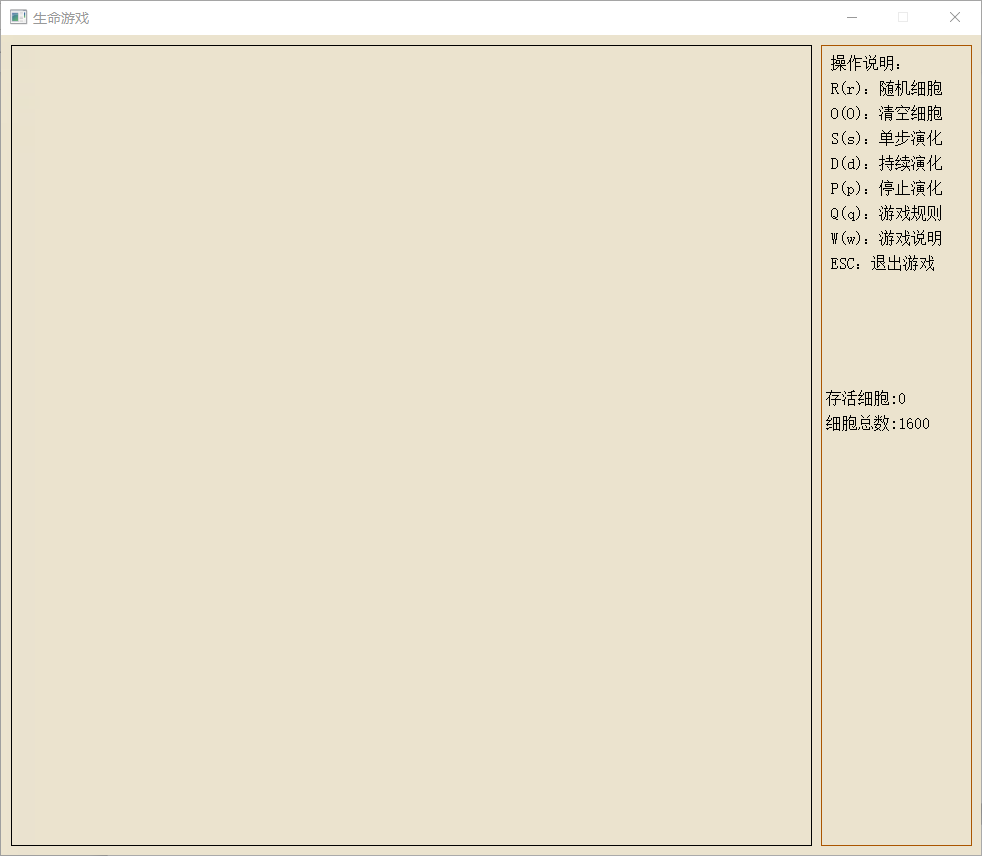
W(w)：游戏说明

ESC：退出游戏

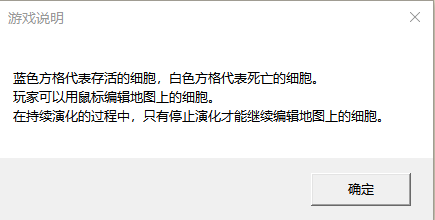
## 调试及测试



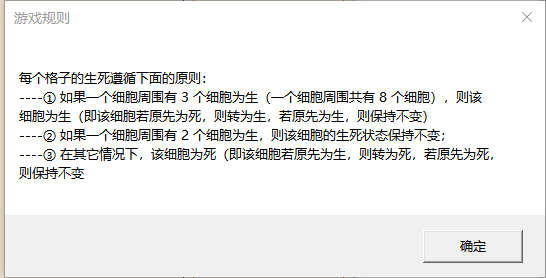
(初始页面)



(主页面)

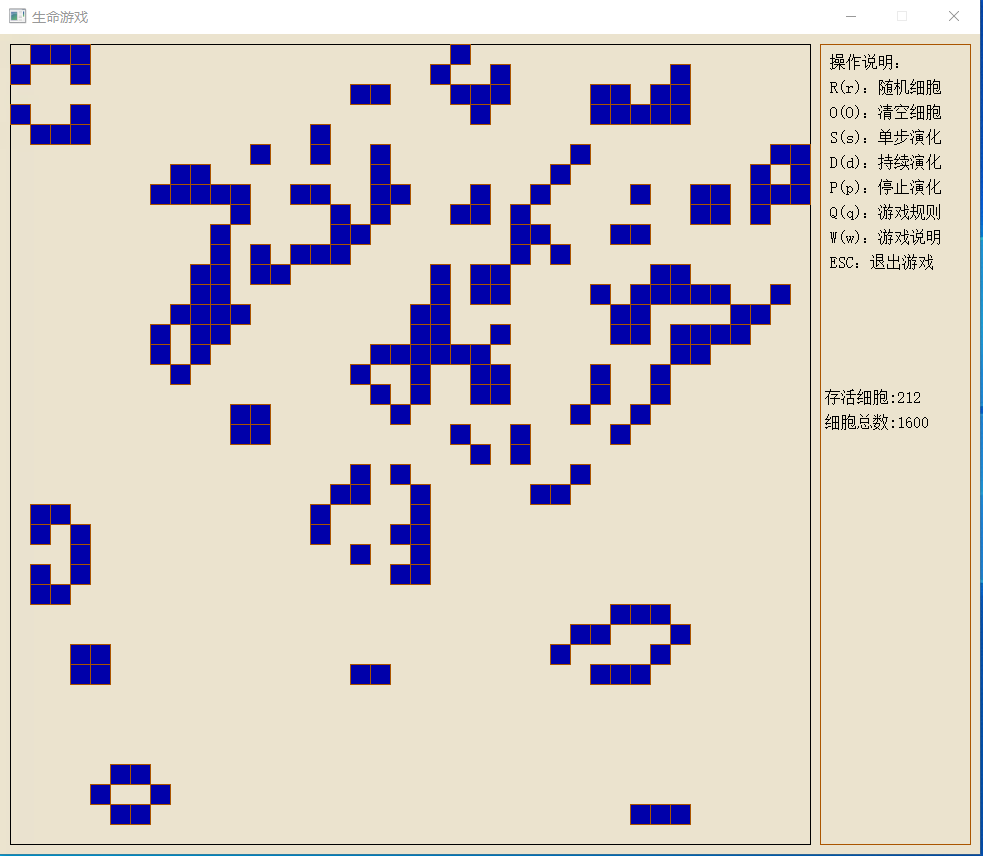


(Q：游戏说明)



(W:游戏规则)

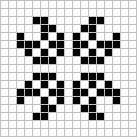
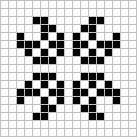
#### （五）运行实例：



### （六）进一步改进

可增加特定图形模块实现，增强玩家的体验感和游戏的震撼性

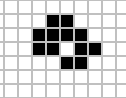
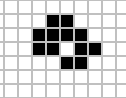
"脉冲星"：它的周期为3，看起来像一颗周期爆发的星星。



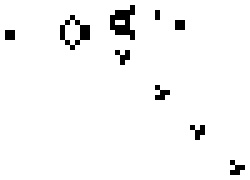
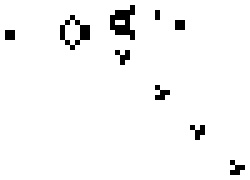
“滑翔者”：每4个回合“它”会向右下角走一格。虽然细胞早就是不同的细胞了，但它能保持原本的形态。



“轻量级飞船”：它的周期是4，每2个回合会向右边走一格。



“滑翔者枪”：它会不停地释放出一个又一个滑翔者。

“繁殖者”：它会向右行进，留下一个接一个的“滑翔者枪”。动图最后一帧定格时用三种颜色区分了繁殖者本体、滑翔者枪和它们打出来的滑翔者。

### （七）心得体会

在这次的课程设计中，我第一次了解和学习了程序可视化，通过网上自学，掌握了EasyX的一些方法，能够实现人机交互，放音乐，可视化这些都让我激动和开心。但在程序设计方面，我对c++的掌握还不够所以程序主要是以c语言的代码编写，同时代码的整洁性，规范性还不够好。我在这次课设中学到的最重要的一点就是要把复杂的东西简单化，当面对陌生的大程序而言，规划往往比马上动手编码更为重要，没有弄清楚清晰的规划，清楚的规则之前切记急于上机编写，不能写一步看一步，尽可能的降低编程时不确定性。在编写代码的过程中，不要总是急于往下写，要及时添加注释，测试数据，并且不断思考代码的可读性如缩进风格，变量的命名等看看哪些地方能够更好地优化。

### （八）对课程设计的建议

希望有更多的指导和学习

### （九）附录⎯⎯源程序

#include <stdio.h>

#include <graphics.h>

#include <conio.h>

#include <time.h>

#pragma comment(lib,"winmm.lib")

#define BOXSIZE 20 //细胞边长

#define BOXNUM 40 //每行或每列的细胞数目

#define GAMELENGTH (BOXSIZE\*BOXNUM) //游戏框长度

#define INFORWIDTH 150 //信息框宽度

#define INFORHEIGHT GAMELENGTH //信息框高度

#define GAP 10 //缝隙像素

#define WNDWIDTH (GAP\*3+INFORWIDTH+GAMELENGTH) //窗口宽度

#define WNDHEIGHT (GAMELENGTH+GAP\*2) //窗口高度

struct cell

{

bool life; //细胞当前生存状态(false为死亡，true为存活)

bool nextlife; //细胞下一步生存状态

}cells[BOXNUM][BOXNUM]; //地图

//游戏规则

char rule[] = {

"每个格子的生死遵循下面的原则：\n\

----① 如果一个细胞周围有 3 个细胞为生（一个细胞周围共有 8 个细胞），则该\n\

细胞为生（即该细胞若原先为死，则转为生，若原先为生，则保持不变）\n\

----② 如果一个细胞周围有 2 个细胞为生，则该细胞的生死状态保持不变；\n\

----③ 在其它情况下，该细胞为死（即该细胞若原先为生，则转为死，若原先为死，则保持不变" };

//游戏说明

char intro[] = {

"蓝色方格代表存活的细胞，白色方格代表死亡的细胞。\n\

玩家可以用鼠标编辑地图上的细胞。\n\

在持续演化的过程中，只有停止演化才能继续编辑地图上的细胞。"};

bool isnext=false; //是否在持续演化过程中

void Welcome(); //游戏欢迎界面(包括资源的加载)

void InitCells(); //初始化细胞

void RanCells(); //随机细胞

void next(); //细胞进行下一步的演化

void Draw(); //画界面

void EditCells(MOUSEMSG); //编辑细胞

void DelButton(char); //处理按键

int SurCellsNum(); //统计目前存活细胞数

IMAGE help;

int main()

{

Welcome();

InitCells();

Draw();

while (1)

{

if (MouseHit())

{

EditCells(GetMouseMsg());

}

if (kbhit())

{

DelButton(getch());

}

}

return 0;

}

void Welcome()

{

initgraph(WNDWIDTH,WNDHEIGHT);

loadimage(&help, \_T("help.png"), 1500, 900);

mciSendString("open ../res/bgm.mp3 alias BGM", 0, 0, 0);

mciSendString("play BGM repeat", 0, 0, 0);

setbkcolor(WHITE);

cleardevice();

putimage(0, 0, &help);

settextcolor(BROWN);

setbkmode(0);

settextstyle(100,80,"楷体");

outtextxy(150,300,"生命游戏");

settextcolor(BLACK);

settextstyle(60, 40, "楷体");

outtextxy(300, 600, "按任意键进入游戏");

settextstyle(50, 30, "楷体");

outtextxy(300, 700, "学号：2019216072");

settextstyle(16, 8, "宋体");

getch();//按任意键进入游戏

cleardevice();

}

void InitCells()

{

for (int j = 0; j < BOXNUM; j++)

{

for (int i = 0; i < BOXNUM; i++)

{

cells[j][i].life = false;

cells[j][i].nextlife = false;

}

}

}

void RanCells()

{

InitCells();

srand((unsigned)time(NULL));

int count = (int)((BOXNUM\*BOXNUM)\*(((rand() % 21) + 10) / 100.0));//控制随机生成的活细胞数量保持在合理水平

for (int k = 0; k < count; k++)

{

int x = rand() % BOXNUM;

int y = rand() % BOXNUM;

cells[y][x].life = true;

}

}

void next()

{

for (int j = 0; j < BOXNUM; j++)//遍历所有细胞

{

for (int i = 0; i < BOXNUM; i++)

{

int flag = 0; //一个细胞周围的活细胞数

for (int m = j - 1; m <= j + 1; m++)//遍历（m，n）细胞的周围9个细胞

{

for (int n = i - 1; n <= i + 1; n++)

{

if (m >= 0 && m <= BOXNUM - 1 && n >= 0 && n <= BOXNUM - 1)//防止越界

{

if (m != j || n != i)//（排除自身）

{

if (cells[m][n].life == true)

{

flag++;

}

}

}

}

}

if (cells[j][i].life == false)//如果细胞为死亡状态

{

if (flag == 3)//如果周围有3个细胞存活,那就存活

{

cells[j][i].nextlife = true;

}

else

{

cells[j][i].nextlife = false;

}

}

else//如果细胞为存活状态

{

if (flag > 3||flag<2)//如果周围存活的细胞大于3或小于2个，那就死亡

{

cells[j][i].nextlife = false;

}

else

{

cells[j][i].nextlife =true;

}

}

}

}

for (int j = 0; j < BOXNUM; j++)//更新细胞地图

{

for (int i = 0; i < BOXNUM; i++)

{

cells[j][i].life = cells[j][i].nextlife;

}

}

}

void Draw()

{

cleardevice();

putimage(0, 0, &help);

setlinecolor(BLACK);//画框

rectangle(GAP, GAP, GAP + GAMELENGTH, GAP + GAMELENGTH);

setlinecolor(BROWN);

rectangle(GAP\*2+GAMELENGTH,GAP,GAP\*2+GAMELENGTH+INFORWIDTH,GAP+INFORHEIGHT);

setfillcolor(BLUE);//画细胞

for (int j = 0; j < BOXNUM; j++)

{

for (int i = 0; i < BOXNUM; i++)

{

if (cells[j][i].life == true)//蓝色是活细胞，白色是死细胞

{

fillrectangle(GAP + i\*BOXSIZE, GAP + j\*BOXSIZE, GAP + (i + 1)\*BOXSIZE, GAP + (j+1)\*BOXSIZE);

}

}

}

char str[50];//侧边栏

outtextxy(GAP \* 2 + GAMELENGTH + 10, GAP + 10, "操作说明：");

outtextxy(GAP \* 2 + GAMELENGTH + 10, GAP + 35, "R(r)：随机细胞");

outtextxy(GAP \* 2 + GAMELENGTH + 10, GAP + 60, "O(O)：清空细胞");

outtextxy(GAP \* 2 + GAMELENGTH + 10, GAP + 85, "S(s)：单步演化");

outtextxy(GAP \* 2 + GAMELENGTH + 10, GAP + 110, "D(d)：持续演化");

outtextxy(GAP \* 2 + GAMELENGTH + 10, GAP + 135, "P(p)：停止演化");

outtextxy(GAP \* 2 + GAMELENGTH + 10, GAP + 160, "Q(q)：游戏规则");

outtextxy(GAP \* 2 + GAMELENGTH + 10, GAP + 185, "W(w)：游戏说明");

outtextxy(GAP \* 2 + GAMELENGTH + 10, GAP + 210, "ESC：退出游戏");

sprintf(str, "存活细胞:%d", SurCellsNum());

outtextxy(GAP \* 2 + GAMELENGTH + 5, GAP + 345, str);

sprintf(str,"细胞总数:%d",BOXNUM\*BOXNUM);

outtextxy(GAP \* 2 + GAMELENGTH + 5, GAP + 370, str);

if (isnext == true)

{

outtextxy(GAP \* 2 + GAMELENGTH + 5, GAP + 280, "正在持续演化…");

}

}

void EditCells(MOUSEMSG m)

{

if (m.uMsg == WM\_LBUTTONDOWN&&m.x >= GAP&&m.x <= GAP + GAMELENGTH&&m.y >= GAP&&m.y <= GAP + GAMELENGTH)

{

int x = (m.x - GAP) / BOXSIZE;

int y = (m.y - GAP) / BOXSIZE;

cells[y][x].life = !cells[y][x].life;

Draw();

}

}

void DelButton(char con)

{

HWND hwnd = GetHWnd();

int ID;

switch (con)

{

case 'o'://清空细胞

case 'O':

ID=MessageBox(hwnd,"确定要清空细胞吗？","",MB\_YESNO);

if (ID == IDYES)

{

InitCells();

Draw();

}

break;

case 'r'://随机细胞

case 'R':

ID = MessageBox(hwnd, "确定要重新随机细胞吗？", "", MB\_YESNO);

if (ID == IDYES)

{

RanCells();

Draw();

}

break;

case 's'://单步演化

case 'S':next(); Draw();

break;

case 'd'://开始持续演化

case 'D':

isnext = true;

while (1)

{

next();

Draw();

Sleep(100);

if (kbhit())

{

char con = getch();

if (con == 'p' || con == 'P')//结束不断演化

{

isnext = false;

Draw();

break;

}

}

}

break;

case 'Q'://游戏规则

case 'q':

MessageBox(hwnd,rule,"游戏规则",MB\_OK);

break;

case 'W'://游戏说明

case 'w':

MessageBox(hwnd, intro, "游戏说明", MB\_OK);

break;

case 27://退出游戏

ID = MessageBox(hwnd, "确定要退出游戏吗？", "", MB\_YESNO);

if (ID == IDYES)

{

exit(0);

}

break;

}

}

int SurCellsNum()

{

int count = 0;

for (int j = 0; j < BOXNUM; j++)

{

for (int i = 0; i < BOXNUM; i++)

{

if (cells[j][i].life == true)

{

count++;

}

}

}

return count;

}