1. **需求和规格说明**
2. ***题目说明***：

我们可以把计算机中的宇宙想象成是一堆方格子构成的封闭空间，尺寸为 N的空间就有 N\*N 个格子。而每一个格子都可以看成是一个生命体，每个生命都 有生和死两种状态，如果该格子生就显示蓝色，死则显示白色。每一个格子旁边都有邻居格子存在，如果我们把 3\*3 的 9个格子构成的正方形看成一个基本单位的话，那么这个正方形中心的格子的邻居就是它旁边的8个格子。

***2.规则说明：***

每个格子的生死遵循下面的原则：

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。）

④再满足以上条件下，改变不同的初始条件和迭代次数将会得到不同的生命图案。

1. **设计**

首先根据生命游戏的需求，先定义出了全局变量cells二维数组，并为了将数组以平面图形的形式打印出来，我设计了show函数用于显示初始状态和更新后的状态，利用sleep来控制更新时间。为了控制初始化的随机性，将利用rand（）函数以及循环赋值用于start（）函数的设计。之后就是游戏的核心update（更新）函数的设计。利用temp二维数组暂存状态判定后的情况，再将更新后的数组复制给cells数组。

为了强化游戏体验，在游戏中加入了隐藏光标的函数，让光标不出现在屏幕上以免损伤游戏体验。在main函数中，设计了一个简易的菜单，便于玩家选择具体的操作。

在第一次版本更新中，添加了调速菜单，增加了菜单之间的返回功能。优化了用户在选择时的体验，增设了出错时的容错机制。

***游戏使用全局变量：***

*1.High 25;*

*2.Width 25;*

*3.sleeptime 200;*

***游戏使用函数：***

*1.int main（void）;*

*2.void HideCursor(void);*

*3.void goto\_xy(int x,int y);*

*4.void start(void);*

*5.void show(int sleeptime);*

*6.void update(void);*

*7.int sleep（int i）;*

具体参考下表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表一 函数和变量定义 | | | | |
| 函数 | 函数类型 | 函数变量 | 变量类型 | 描述 |
| Main() | int | i | int | 条件 |
| HideCursor() | void | 无 |  |  |
| goto\_xy() | void | 无 |  |  |
| Start() | void | i, j | int | 循环 |
| Show(sleeptime) | void | i, j | int | 循环 |
| Update() | void | i, j  NeighborCells  Temp[High][Width] | Int  Int  int | 循环  判断  暂存 |
| Sleep(int i) | Int | sleeptime | int | 睡眠时间 |

1. **用户手册**

用户在开始菜单可以选择1.开始游戏，0.退出游戏，开始后系统随机生成初始细胞，随后开始进行迭代。玩家可以进行观察具体的情况。

1. **调试及测试**

输入1，程序开始运行

按任意键退出

输入2，进入调速页面

输入1,2,3 可调控相应的速度。

输入0，返回主菜单。

输入0，程序退出

输入其他字符，页面刷新，再来一次。

1. **进一步改进**

1.增加了隐藏鼠标的设计，优化了游戏体验。

2.优化了显示界面，让图形界面更加美观。

3.添加了修改刷新时间的函数，完善了游戏界面。

4.修改了字体颜色。

5.修复了一些常见的小bug。

1. **心得体会**

生命游戏的核心是显示和判断。在设计游戏时，如何将每一代的情况输出到显示屏上，是一个问题。另外，在更新时，如何基于上一代推演也困扰我许久。最终这些问题都得以解决，还一定程度上取决于我采用了模块化的函数式的设计。这样让功能模块分开，让单一设计更加简单。通过此次课程设计，让我意识到了真实的程序设计与简单的小程序有很大的差别。

1. **对课程设计的建议**

希望可以将题目的衡量标准更加具体些。可以添加适当的加分项目，同时为题目的具体要求提出一些标准。

1. **附录——源程序**

/\*-----------------------------------------------

Filename: LifeGame.C

Purpose:To Start a LifeGame

Date: 2019.12.02

Author: Li Haoran

----------------------------------------------\*/

#include<stdio.h>

#include<windows.h>

#include<time.h>

#include<conio.h>

#include<conio.h>

#define High 25 //游戏画面大小

#define Width 25

//定义全局变量

int cells[High][Width]; //细胞的生1死0

int sleeptime = 200;

void HideCursor(void); //隐藏光标

void goto\_xy(int x,int y); //光标移动到(x,y)位置

void start(void); //数据初始化

void show(int sleeptime); //显示游戏的界面

void update(void); //更新控制函数

int sleep(int i); //控制刷新时间的函数

int main(void)

{

int i, j;

char key;

//开始界面

system("color A");

the:

goto\_xy(50,10);

printf(" 生命游戏\t\n");

goto\_xy(45,11);

printf(" 1.开始游戏");

goto\_xy(45,12);

printf(" 2.设置速度\n");

goto\_xy(45,13);

printf(" 0.退出游戏");

goto\_xy(45,14);

printf("请输入(1/2/0):\t");

scanf("%d", &i);

getchar();

system("CLS");

//系统判断

switch(i)

{

case 1:

{

start();

while(kbhit()==0)

{

printf("按任意键结束......\n");

show(sleeptime);

update();

}

system("pause");

return 0;

}

case 2:

{

the2:

system("CLS");

system("color A");

goto\_xy(50,10);

printf("1.慢速\n");

goto\_xy(50,11);

printf("2.系统\n");

goto\_xy(50,12);

printf("3.快速\n");

goto\_xy(50,13);

printf("0.返回主菜单\n");

goto\_xy(50,14);

printf("请输入(1/2/3/0):\t");

scanf("%d", &i);

if(i == 1 ||i == 2||i == 3)

{

sleeptime = sleep(i);

goto\_xy(50,16);

printf("设置成功!\n");

system("pause");

system("CLS");

goto the;

}

else if(i == 0)

{

system("CLS");

goto the;

}

else

{

goto\_xy(50,16);

printf("输入错误，请在输入一遍！\n");

system("pause");

system("CLS");

goto the2;

}

}

case 0:

{

the3:

system("CLS");

goto\_xy(50,10);

printf("确定要退出了吗？(y / n)\n");

goto\_xy(50,11);

printf("y退出，n返回主菜单。\n");

key = getchar();

if(key == 'y'||key == 'Y')

exit(0);

else if(key == 'n'||key == 'N')

{

system("CLS");

goto the;

}

else

{

goto\_xy(50,10);

printf("输入错误，请在输入一遍！\n");

system("pause");

system("CLS");

goto the3;

}

}

default:

{

goto\_xy(50,10);

printf("输入错误，请在输入一遍！\n");

system("pause");

system("CLS");

goto the;

}

}

return 0;

}

void HideCursor(void) //隐藏光标

{

CONSOLE\_CURSOR\_INFO cursor\_info = {1, 0};

SetConsoleCursorInfo(GetStdHandle(STD\_OUTPUT\_HANDLE), &cursor\_info);

}

void goto\_xy(int x,int y) //光标移动到(x,y)位置

{

HANDLE handle = GetStdHandle(STD\_OUTPUT\_HANDLE);

COORD pos;

pos.X = x;

pos.Y = y;

SetConsoleCursorPosition(handle,pos);

}

void start(void) //数据初始化

{

srand(time(NULL));

int i,j;

for(i=0;i<High;i++)

{

for(j=0;j<Width;j++)

{

cells[i][j]=(rand() % 2);//用rand()函数使细胞的初始状态随机

}

}

HideCursor();

}

void show(int sleeptime) //显示游戏的界面

{

system("color 2");

goto\_xy(0,0);

int i,j;

for(i=1;i<High;i++)

{

for(j=1;j<Width;j++)

{

if(cells[i][j]==1)

printf("■");

else

printf("□");

}

printf("\n");

}

Sleep(sleeptime); //控制刷新速度

}

void update(void) //更新控制函数

{

int i,j;

int NeighborCells;

int temp[High][Width];

//判断细胞生死

for(i=1;i<High;i++)

{

for(j=1;j<Width;j++)

{ //相邻细胞个数是NeighborCells

NeighborCells=(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]);

//判断状态

if(NeighborCells==3) //周围有3个活细胞时，该细胞生

temp[i][j]=1;

else if(NeighborCells==2) //周围有2个活细胞时，该细胞状态不变

temp[i][j]=cells[i][j];

else

temp[i][j]=0;

}

}

//转换为静态的情况

for(i=1;i<High;i++)

for(j=1;j<Width;j++)

cells[i][j]=temp[i][j];

}

int sleep(int i)

{

if(i == 1)

return 400;

else if(i == 2)

return 200;

else if(i == 3)

return 50;

}