#include<stdio.h>

#include<graphics.h>//graphics.h

#include<windows.h>

#include<math.h>

#include<time.h>

#include<stdlib.h>

//1:35: 1：50

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

#define NUM 13

/\*

1.把程序里面的元素用数据类型表示

//两个坐标桌面窗口的坐标，图片的坐标

单个烟花的大小240\*240

2.初始化

\*/

//烟花

struct Fire

{

int x, y;//烟花在窗口里面爆炸的坐标

int r;//烟花爆炸的当前半径

int max\_r;//烟花爆炸最大半径

int cen\_x, cen\_y;//图片坐标系中的中心位置

int xy[240][240];//240\*240的像素数组，图片的像素点保存在这里

bool show;//烟花是否开始爆炸 表示可不可以爆炸

bool draw;//开始绘制 表示是否爆炸

DWORD t1, t2, dt;//单位时间 帧数 爆炸的速度

//unsigned long = DWORD 4字节

}fire[NUM];

IMAGE flower, stepimg;

void load()

{

//IMAGE flower,stepimg;

loadimage(&flower,L"./flower.jpg",3120,240);//3120 240（高度） 13张

//工作区 默认工作区是窗口

//flower.jpg这个图片中有13朵烟花

//getimage(&stepimg,);

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

{

SetWorkingImage(&flower);//在flower图片上，裁图片

getimage(&stepimg, i \* 240, 0, 240, 240);//裁剪第一朵烟花 获取

SetWorkingImage(&stepimg);//小图片搞事情

for (int a = 0; a < 240; a++)

{

for (int b = 0; b < 240; b++)

{

fire[i].xy[a][b] = getpixel(a, b);//获取小图片上的每一个像素，存放在xy数组

}

}

}

SetWorkingImage(NULL);//工作区回到窗口

//return 0;

}

void init(int i)

{

fire[i].x = rand() % 1200;

fire[i].y = rand() % 800;

fire[i].r = 0;

fire[i].max\_r = 120;//图片的宽度240 240

fire[i].cen\_x = 120;

fire[i].cen\_y = 120;

//fire[i].xy[240][240];

fire[i].show = true; //\*\*\*\*\*\*\*\*\*\*\*

fire[i].draw = false;

fire[i].t1 = GetTickCount();//获取系统的时间

//fire[i].t2;

fire[i].dt = 5;//爆炸的间隔 5毫秒

fire[i].xy[240][240];

}

//最核心的地方

void draw(DWORD\* pmem )//显存指针

{

//逻辑

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

{

fire[i].t2 = GetTickCount();//GetTickCount:获取系统的时间

//“detat = t2 - t1”

if (fire[i].t2 - fire[i].t1 >= fire[i].dt && fire[i].show == true )

{

//r不能超过max\_r

if (fire[i].r < fire[i].max\_r)

{

fire[i].r++;//半径不断变大

fire[i].draw = true;//可以绘制

}

//万一超过 爆炸结束

if (fire[i].r >= fire[i].max\_r - 1)

{

fire[i].draw = false;

init(i);//爆炸结束给内存初始化，等待下一次爆炸

}

//重置时间 否则永远t2-t1不为5（只有一次为5）

fire[i].t1 = fire[i].t2;

}

//如果是可以绘制的状态

if (fire[i].draw = true)

{

// 绘制不是贴图 而是输出像素点

// 像素点 就是颜色（颜色：32位的数据）

// 00000000 00000000 00000000 00000000

// a透明度 red green blue

for (double a = 0; a <= 6.28; a += 0.01)

{

//x1 y1 图片坐标系的坐标

int x1 = fire[i].cen\_x + fire[i].r \* cos(a);

int y1 = fire[i].cen\_y - fire[i].r \* sin(a);

//编程坐标系 与数学不一样

//以半径r得到圆弧上的每一个x和y

if (x1 > 0 && x1 < 240 && y1 > 0 && y1 < 240)

{

//分开32位数据 颜色 00000000 00000000 00000000

//点不能超出图片的该范围以外

int b = fire[i].xy[x1][y1]&0xff;//blue

int g = (fire[i].xy[x1][y1]>>8)&0xff;//green

int r = (fire[i].xy[x1][y1] >> 16);//red

// 像素转移 小图片->工作区

int xx = fire[i].x + fire[i].r \* cos(a);

int yy = fire[i].y - fire[i].r \* sin(a);

//xx yy 窗口坐标系的坐标

if (r > 0x20 && g > 0x20 && b > 0x20 && xx > 0 && xx < 1200 && yy > 0 && yy < 800)

{

//如果这个点颜色太深（黑） 就不输出

pmem[yy \* 1200 + xx] = BGR(fire[i].xy[x1][y1]);

// 移像素点 图片 =》 窗口

}

}

}

//628个圆弧上的点

fire[i].draw = false;

}

}

}

int main()

{

//initgraph创建了一个窗口 1200像素 800像素

initgraph(1200, 800);

//初始化13个烟花

mciSendString(L"open ./bk1.mp3 alias music", 0, 0, 0);

mciSendString(L"play music", 0, 0, 0);

DWORD\* pmem = GetImageBuffer();//获取显存指针

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

{

init(i);

}

load();

BeginBatchDraw();//显存显示

while (1)

{

draw(pmem);

FlushBatchDraw();

//getchar();

//return 0;

}

}