****

**2021—2022学年第3学期**

**《计算机综合实训》实习报告**

项目名称：太空战机

**班级： 理科实验班2102班**

**学号： \_\_\_2109060202\_\_\_\_\_\_\_\_\_**

**姓名： XX**

|  |  |
| --- | --- |
| **评价指标和评分结果** | 总分：0 |
| 1、项目有创新，有技术含量；系统功能丰富，工作量大；独立完成（20分） |  |
| 2、需求分析充分合理，设计方案完整；关键技术运用合理；图表规范（30） |  |
| 3、代码质量高；代码包完整、可运行（30分） |  |
| 4、实习报告文档结构完整，逻辑性好，语言文字规范，排版良好（10分） |  |
| 5、PPT结构清晰，设计精美；短视频剪辑良好，配音效果好（10分） |  |
| **评语：** | |

# 说 明

1. 《计算机综合实训》作为《程序设计基础》、《高级程序设计》等的后续课程，旨在进行程序设计能力的综合训练，以提高问题分析求解能力、创新能力和团队协作能力。所有参加本课程的学生都必须参与程序设计和课程报告撰写工作。
2. 《计算机综合实训》结束后，需要提交**课程报告（电子版）、程序代码包、短视频、答辩PPT**。未提交相关资料者不能参加成绩评定。
3. 课程报告要求使用MS Word，严格按照本报告模板撰写（**保留首页、扉页、正文的页面和页脚**）。文档结构完整、格式规范、条理清晰、论述准确、语言通顺、文字正确。字数不少于5000字。
4. **文档撰写不符合要求，或者抄袭他人作品，记零分**。

**太空战机**

**一、项目描述**

基于所学程序设计课的基础上，在程序设计实训期间，以小组的形式基于Funcode开发平台设计开发一款游戏。

游戏的主要场景是：在外星球上，存在许多不同型号的外形战机，玩家需要通过键盘上的WSAD键控制己方战机，消灭外星球上的邪恶战机。具体要求如下：

1、游戏初始界面如图1所示：



图1 游戏初始界面

2、玩家按下空格键，“空格开始”字样消失，接着，从边界右侧随机位置会持续出现恒定速度的敌方战机，并且战机会随机上下浮动，并已一恒定时间发射子弹攻击我方；

3、玩家用键盘上WASD键控制战机向上、左、下、右方向浮动。按下空格键时，战机就会发射子弹消灭对方；

4、玩家的战机飞到屏幕边界时，不能继续冲出边界。

**二、程序功能分析**

功能分析：根据游戏要求，可将游戏分为五个功能模块，即游戏的初始化、战机的飞动、战机发射子弹、战机攻击与被攻击，系统累算积分。

结合Funcode平台提供的功能，作如下技术分析：

1、在Funcode里每个游戏被称作一个工程，因此首先要创建一个工程，并导入相应的太空战机场景；

2、游戏中的每个图形对象被称作一个精灵，在不考虑特效与音乐等前提下，因此要为本项目精灵己方战机、敌方战机和子弹创建3个精灵即可；

3、要让敌方战机随机出现需要用到平台提供的函数 RandomRange(,)并设置相应的参数；

4、要是一个精灵游到边界后再返回来，则要设置游戏的世界边界，并且当鱼游到边界时“精灵与世界边界碰撞”的事件，要利用边界碰撞函数dSetSpriteWorldLimit()，当鱼返回时要水平翻转鱼精灵图片；

5、本项目会产生许多精灵，难以寻找与管理，因此使用链表方法与OOP结合的方式，可以有效地找到精灵们发生碰撞在什么地方，由谁发生碰撞，以实现对其的管理；

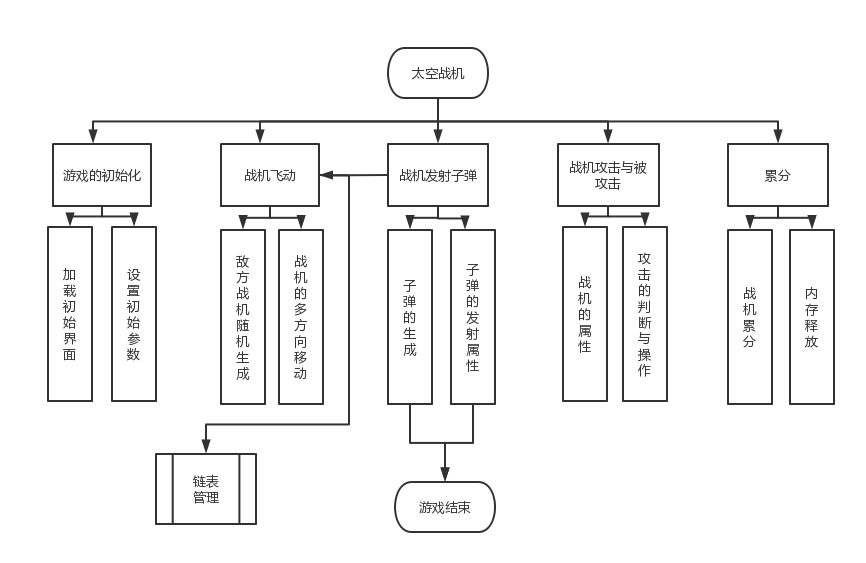
6、由于游戏需要统计分数来体现玩家的实力，所以程序需要 SetTextValue( )函数来显示分数，同时需要用文件流输入输出的方法来统计分数;

7、要用键盘上WASD键控制飞机向上、左、下、右方向游动，则会触发到“键盘按下”事件和“键盘释放”事件，并调用响应时间处理函数dOnKeyDown()和dOnKeyUp()；

8、要使子弹命中敌机和战机坠毁有特效，需要调用CEffect类中的PlayEffect（ ， ， ）函数得以实现。

**三、程序设计**

根据任务功能分析，设计如下游戏功能框架：



**1、主控模块**

主控模块控制游戏的主循环，此模块被不停的调用，引擎每刷新一次屏幕，此模块就被调用一次。主要用来处理游戏的开始、进行和结束等各种状态。

**2、初始化模块**

主要是实现对游戏的初始化，将项目中的指针指向空指针，包括加载初始场景，设置全局变量初值等。每局游戏开始前进行初始化清空上一局数据，消除对新一局游戏数据的干扰，玩家按下空格键，游戏开始。

**3、战机的飞动模块**

主要是实现己方战机的多方向移动和敌机的浮动产生，玩家通过WSAD键进行操控战机的方向躲避危险，同时按下空格键进行开火消灭敌机。

**4、子弹模块**

主要是实现子弹变量的产生与回收，同时设置其属性包括类别，伤害，时间等参数，用以攻击战机。

**5、攻击模块**

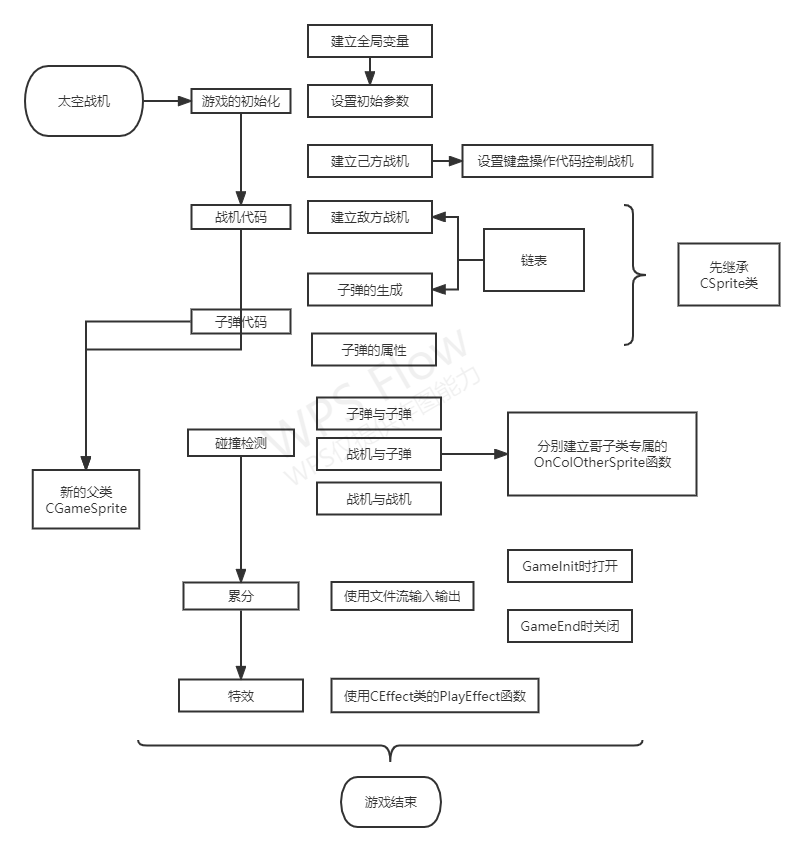
主要是实现己方战机攻击敌机与躲避敌机攻击，双方都有初始血量与伤害，己方通过操作消灭敌机。

1. **积分模块**

主要实现玩家在消灭敌机后获得积分，不断刷新纪录，获得快乐。

**四、系统实现与核心代码描述**

**1、游戏代码的整体设计**



**2、游戏实现细节**

**（1）新建工程并导入“AirPlane”模板**

**（2）游戏初始化，设置全局变量初值：**

float g\_fScreenLeft = 0.f; // 屏幕左边界值

float g\_fScreenRight = 0.f; // 右

float g\_fScreenTop = 0.f; // 上

float g\_fScreenBottom = 0.f; // 下

**（3）己方战机战机的产生与飞动**

CMyFighter.h

#ifndef CMYFIGHTER\_H

#define CMYFIGHTER\_H

#include <CommonClass.h>

#include "CGameSprite.h"

class CMyFighter : public CGameSprite

{

public:

float m\_fBulletCreateTime;

CMyFighter(const char\* szName);

virtual ~CMyFighter();

float GetVelocityLeft() { return m\_fVelocityLeft; }

void SetVelocityLeft( float val) { m\_fVelocityLeft = val; }

float GetVelocityRight() { return m\_fVelocityRight; }

void SetVelocityRight( float val) { m\_fVelocityRight = val; }

float GetVelocityUp() { return m\_fVelocityUp; }

void SetVelocityUp( float val) { m\_fVelocityUp = val; }

float GetVelocityDown() { return m\_fVelocityDown; }

void SetVelocityDown( float val) { m\_fVelocityDown = val; }

void OnMove(bool bKeyDown,int iKey);

void OnFile(float fDeltaTime);

bool GetCanFile() { return m\_bCanFile; }

void SetCanFile(bool val) { m\_bCanFile = val; }

bool IsMyFighter(){return true;}

void OnColOtherSprite(CGameSprite\* pOther);

protected:

private:

float m\_fVelocityLeft;

float m\_fVelocityRight;

float m\_fVelocityUp;

float m\_fVelocityDown;

bool m\_bCanFile;

};

#endif // CMYFIGHTER\_H

CMyFighter.cpp

#include "CMyFighter.h"

#include "LessonX.h"

#include "CBullet.h"

#include "CGameSprite.h"

CMyFighter::CMyFighter(const char\* szName): CGameSprite(szName)

{

//ctor

m\_fVelocityLeft =0.f;

m\_fVelocityRight =0.f;

m\_fVelocityUp =0.f;

m\_fVelocityDown =0.f;

m\_fBulletCreateTime =0.3;

}

CMyFighter::~CMyFighter()

{

//dtor

}

void CMyFighter::OnMove(bool bKeyDown,int iKey)

{

if (bKeyDown)

{

switch(iKey)

{

case KEY\_A:

m\_fVelocityLeft=30.f;

break;

case KEY\_D:

m\_fVelocityRight=30.f;

break;

case KEY\_W:

m\_fVelocityDown=15.f;

break;

case KEY\_S:

m\_fVelocityUp=15.f;

break;

}

}

else

{

switch(iKey)

{

case KEY\_A:

m\_fVelocityLeft=0.f;

break;

case KEY\_D:

m\_fVelocityRight=0.f;

break;

case KEY\_W:

m\_fVelocityDown=0.f;

break;

case KEY\_S:

m\_fVelocityUp=0.f;

break;

}

}

float fVelX=m\_fVelocityRight-m\_fVelocityLeft;

float fVelY=m\_fVelocityUp-m\_fVelocityDown;

SetSpriteLinearVelocity(fVelX,fVelY);

}

**（4）敌方战机的产生与飞动**

CEnemyFighter.h

#ifndef CENEMYFIGHTER\_H

#define CENEMYFIGHTER\_H

#include <CommonClass.h>

#include "CGameSprite.h"

class CEnemyFighter : public CGameSprite

{

public:

CEnemyFighter(const char\* szName);

virtual ~CEnemyFighter();

static float GetCreateTime() { return m\_fCreateTime; }

static int Getm\_iCreatedSpriteCount() { return m\_iCreatedSpriteCount; }

void static ClassTick ( float fDeltaTime );

void LoopTick(float fDeltatime);

void OnColOtherSprite(CGameSprite\* pOther);

protected:

private:

static float m\_fCreateTime;

static int m\_iCreatedSpriteCount;

float m\_fCanFileAfterCreated;

float m\_fBulletCreateTime;

float m\_fFloatTime;

bool m\_bFloatUp;

};

#endif // CENEMYFIGHTER\_H

CEnemyFighter.cpp

#include "CEnemyFighter.h"

#include "stdio.h"

#include "CommonClass.h"

#include "LessonX.h"

#include "CGameSprite.h"

CEnemyFighter::CEnemyFighter(const char\* szName): CGameSprite(szName)

{

//ctor

m\_fCanFileAfterCreated =3.f;

m\_fBulletCreateTime =1.f;

m\_fFloatTime =0.f;

m\_bFloatUp =true;

}

CEnemyFighter::~CEnemyFighter()

{

//dtor

}

void CEnemyFighter::ClassTick ( float fDeltaTime )

{

m\_fCreateTime-=fDeltaTime ;

if (m\_fCreateTime<=0.f)

{

m\_fCreateTime =(float)CSystem::RandomRange(5,10);

int iPosBase = CSystem::RandomRange((int)CSystem::GetScreenTop()+10,(int)CSystem::GetScreenBottom()-10);

char szName[MAX\_NAME\_LEN];

sprintf(szName,"HorizontalSprite\_%d", m\_iCreatedSpriteCount);

m\_iCreatedSpriteCount++;

CEnemyFighter\* pSprite = new CEnemyFighter (szName );

pSprite->CloneSprite ("HorizontalSprite\_Template");

g\_GameMain.AddSprite(pSprite);

int iRandom = CSystem::RandomRange (iPosBase-10, iPosBase+10);

float fPosX =(int)CSystem::GetScreenRight()+20.f;

pSprite->SetSpritePosition(fPosX ,(float)iRandom );

pSprite->SetSpriteLinearVelocityX(-10.f);

pSprite->SetSpriteWorldLimit(WORLD\_LIMIT\_NULL,CSystem::GetScreenLeft()-10.f, CSystem::GetScreenTop(), CSystem::GetScreenRight()+200.f, CSystem::GetScreenBottom());

pSprite->SetSpriteCollisionActive(true,true);

pSprite->SetHp(300);

pSprite->SetScore(100);

pSprite->SetType(true);

}

}

**（5）子弹精灵的属性**

CBullet.h

#ifndef CBULLET\_H

#define CBULLET\_H

#include <CommonClass.h>

#include "CGameSprite.h"

class CBullet : public CGameSprite

{

public:

CBullet(const int iType,const char\* szName);

virtual ~CBullet();

bool IsMyBullet();

void OnColOtherSprite(CGameSprite\* pOther);

protected:

private:

int m\_iType;

};

#endif // CBULLET\_H

CBullet.cpp

#include "CBullet.h"

#include "CGameSprite.h"

#include "LessonX.h"

CBullet::CBullet(const int iType,const char\* szName):CGameSprite(szName)

{

//ctor

m\_iType=iType;

}

CBullet::~CBullet()

{

//dtor

}

bool CBullet::IsMyBullet()

{

m\_iType=GetType();

if (m\_iType==0)

return true;

else

return false;

}

void CBullet::OnColOtherSprite(CGameSprite\* pOther)

{

if (NULL==pOther)

{

return ;

}

if (IsMyBullet())

{

if (pOther->IsMyFighter())

return ;

SetHp(GetHp()- pOther->GetDamage());

}

else

{

if (pOther->IsMyFighter()||pOther->IsMyBullet())

{

SetHp(GetHp()- pOther->GetDamage());

}

}

}

**（6）战机发射子弹**

敌方战机

void CEnemyFighter::LoopTick(float fDeltaTime)

{

m\_fCanFileAfterCreated-=fDeltaTime;

if (m\_fCanFileAfterCreated<=0.f)

{

m\_fBulletCreateTime-=fDeltaTime;

if (m\_fBulletCreateTime<=0.f)

{

m\_fBulletCreateTime=1.f;

g\_GameMain.CreateBullet(1,GetSpritePositionX(),GetSpritePositionY());

}

}

if (m\_bFloatUp)

{

m\_fFloatTime+=fDeltaTime;

if (m\_fFloatTime>=1.f)

{

m\_bFloatUp=false;

}

float fPosY=GetSpritePositionY();

fPosY+=6.f\*fDeltaTime;

SetSpritePositionY(fPosY);

}

else

{

m\_fFloatTime-=fDeltaTime;

if (m\_fFloatTime<=0.f)

{

m\_bFloatUp=true;

}

float fPosY=GetSpritePositionY();

fPosY-=6.f\*fDeltaTime;

SetSpritePositionY(fPosY);

}

}

己方战机

void CMyFighter::OnFile(float fDeltaTime)

{

m\_fBulletCreateTime-=fDeltaTime;

if (m\_fBulletCreateTime<=0.f&&m\_bCanFile)

{

m\_fBulletCreateTime=0.3f;

g\_GameMain.CreateBullet(0,GetSpritePositionX(),GetSpritePositionY());

}

}

void CGameMain::CreateBullet(int iType,const float fPosX,const float fPosY)

{

char szName[MAX\_NAME\_LEN];

sprintf(szName,"Bullet1\_%d",m\_iCreatedSpriteCount);

m\_iCreatedSpriteCount++;

CBullet\* pBullet =new CBullet(iType,szName);

pBullet->CloneSprite("Bullet1\_Template");

pBullet->SetSpritePosition(fPosX,fPosY);

pBullet->SetSpriteLinearVelocityX(60);

pBullet->SetSpriteWorldLimit(WORLD\_LIMIT\_NULL,CSystem::GetScreenLeft()-10.f,CSystem::GetScreenTop(),CSystem::GetScreenRight()+200.f,CSystem::GetScreenBottom());

if (1==iType)

{

pBullet->SetSpriteLinearVelocityX(-30);

}

else

{

pBullet->SetSpriteFlipX(true);

pBullet->SetSpriteLinearVelocityX(60);

}

pBullet->SetScore(100);

pBullet->SetHp(10);

pBullet->SetDamage(100);

pBullet->SetType(iType);

pBullet->SetSpriteCollisionActive(true,true);

AddSprite(pBullet);

}

**（8）战机、子弹间的碰撞**

CGameSprite\* CGameMain::GetSprite(const int ilndex)

{

return m\_SpriteList.GetSprite(ilndex);

}

CGameSprite\* CGameMain::GetSprite(const char\* szName)

{

return m\_SpriteList.GetSprite(szName);

}

CMyFighter\* CGameMain::GetMyFighter()

{

if (NULL!=m\_pMyFighter)

{

return m\_pMyFighter;

}

else return NULL;

}

bool CGameMain::IsMyFighter(const char\* szName)

{

return (strcmp(m\_pMyFighter->GetName(),szName)==0);

}

void CGameMain::OnSpriteColSprite(const char\* szSrcName,const char\* szTarName)

{

if (2!=GetGameState())

return ;

CGameSprite\* pSrcSprite=IsMyFighter(szSrcName)?m\_pMyFighter:GetSprite(szSrcName);

CGameSprite\* pTarSprite=IsMyFighter(szTarName)?m\_pMyFighter:GetSprite(szTarName);

if (NULL==pSrcSprite||NULL==pTarSprite)

return ;

pSrcSprite->OnColOtherSprite(pTarSprite);

pTarSprite->OnColOtherSprite(pSrcSprite);

if (!pSrcSprite->IsMyFighter())

{

if (pSrcSprite->IsDead())

m\_SpriteList.DeleteSprite(pSrcSprite,true);

}

if (!pTarSprite->IsMyFighter())

{

if (pTarSprite->IsDead())

m\_SpriteList.DeleteSprite(pTarSprite,true);

}

}

bool CGameMain::IsGameLost()

{

return (m\_pMyFighter?m\_pMyFighter->IsDead():false);

}

void CGameMain::OnKeyDown(const int iKey, const int iAltPress, const int iShiftPress, const int iCtrlPress)

{

if (KEY\_SPACE ==iKey&&0==GetGameState())

{

SetGameState(1);

}

if (2==GetGameState())

{

m\_pMyFighter->OnMove(true,iKey);

if (KEY\_SPACE==iKey&&NULL!=m\_pMyFighter)

{

m\_pMyFighter->SetCanFile(true);

}

}

}

void CMyFighter::OnColOtherSprite(CGameSprite\* pOther)

{

if (NULL==pOther)

return ;

if (pOther->IsMyBullet())

return ;

SetHp(GetHp()-pOther->GetDamage());

if (GetHp()<=200)

{

SetSpriteColorGreen(0);

SetSpriteColorBlue(0);

}

else if (GetHp()<=500)

{

SetSpriteColorGreen(128);

SetSpriteColorBlue(128);

}

else

{

SetSpriteColorGreen(255);

SetSpriteColorBlue(255);

}

g\_GameMain.GetMyFighter()->SetScore(g\_GameMain.GetMyFighter()->GetScore()+pOther->GetScore());

}

void CEnemyFighter::OnColOtherSprite(CGameSprite\* pOther)

{

if (NULL==pOther)

return ;

if (pOther->IsMyFighter()||pOther->IsMyBullet())

{

SetHp(GetHp()- pOther->GetDamage());

if (IsDead())

{

g\_GameMain.GetMyFighter()->SetScore(g\_GameMain.GetMyFighter()->GetScore()+GetScore());

}

}

}

**（9）游戏累分**

void CGameMain::GameInit()

{

m\_pBeginSprite->SetSpriteVisible(false);

m\_pCurScoreText->SetTextValue(0);

m\_pMaxScoreText->SetTextValue(0);

m\_pMyFighter=NULL;

if (NULL==m\_pMyFighter)

{

m\_pMyFighter=new CMyFighter("ControlSprite");

m\_pMyFighter->SetSpriteVisible(true);

m\_pMyFighter->SetSpriteColorGreen(255);

m\_pMyFighter->SetSpriteColorBlue(255);

m\_pMyFighter->SetSpriteWorldLimit(WORLD\_LIMIT\_STICKY,CSystem::GetScreenLeft()-10.f,CSystem::GetScreenTop(),CSystem::GetScreenRight(),CSystem::GetScreenBottom());

}

m\_pMyFighter->SetHp(500);

m\_pMyFighter->SetScore(0);

m\_pMyFighter->SetCanFile(true);

fstream ScoreFile("Score.dat",fstream::in|fstream::binary);

if (ScoreFile.is\_open())

{

ScoreFile>>m\_iMaxScore;

ScoreFile.close();

}

m\_pMaxScoreText->SetTextValue(m\_iMaxScore);

}

//=============================================================================

//

// 每局游戏进行中

void CGameMain::GameRun(float fDeltaTime)

{

if (m\_pMyFighter)

{

m\_pMyFighter->OnFile(fDeltaTime);

}

CEnemyFighter::ClassTick(fDeltaTime);

int iListSize=m\_SpriteList.GetListSize();

for (int iLoop=0;iLoop<iListSize;iLoop++)

{

CGameSprite\* pSprite=m\_SpriteList.GetSprite(iLoop);

if (pSprite!=NULL&&(strstr(pSprite->GetName(),"HorizontalSprite")!=NULL))

{

((CEnemyFighter\*)pSprite)->LoopTick(fDeltaTime);

}

}

m\_pCurScoreText->SetTextValue(m\_pMyFighter->GetScore());

}

//=============================================================================

// 本局游戏结束

void CGameMain::GameEnd()

{

m\_pBeginSprite->SetSpriteVisible(true);

m\_SpriteList.DeleteAllSprite(true);

m\_pMyFighter->SetSpriteVisible(false);

if (m\_iMaxScore<GetMyFighter()->GetScore())

{

m\_iMaxScore=GetMyFighter()->GetScore();

fstream ScoreFile("Score.dat",fstream::out|fstream::binary);

if (ScoreFile.is\_open())

{

ScoreFile<<m\_iMaxScore;

ScoreFile.close();

}

}

delete m\_pMyFighter;

m\_pMyFighter=NULL;

SetGameState(0);

}

**（10）主体代码**

void CGameMain::GameMainLoop(float fDeltaTime)

{

switch(GetGameState())

{

// 初始化游戏，清空上一局相关数据

case 1:

{

GameInit();

SetGameState(2); // 初始化之后，将游戏状态设置为进行中

}

break;

// 游戏进行中，处理各种游戏逻辑

case 2:

{

// TODO 修改此处游戏循环条件，完成正确游戏逻辑

if (!IsGameLost())

{

GameRun(fDeltaTime);

}

else // 游戏结束。调用游戏结算函数，并把游戏状态修改为结束状态

{

SetGameState(0);

GameEnd();

}

}

break;

// 游戏结束/等待按空格键开始

case 0:

default:

break;

};

}

**五、系统的功能扩展（创新点）**

**1、通过精灵之间的碰撞实现了消灭敌机的功能**

实现方法：调用精灵碰撞函数，并设置两个碰撞的精灵在同一层上，当发生碰撞时，设置某个精灵消失，代码如下：

void CGameMain::OnSpriteColSprite(const char\* szSrcName,const char\* szTarName)

{

if (2!=GetGameState())

return ;

CGameSprite\* pSrcSprite=IsMyFighter(szSrcName)?m\_pMyFighter:GetSprite(szSrcName);

CGameSprite\* pTarSprite=IsMyFighter(szTarName)?m\_pMyFighter:GetSprite(szTarName);

if (NULL==pSrcSprite||NULL==pTarSprite)

return ;

pSrcSprite->OnColOtherSprite(pTarSprite);

pTarSprite->OnColOtherSprite(pSrcSprite);

if (!pSrcSprite->IsMyFighter())

{

if (pSrcSprite->IsDead())

m\_SpriteList.DeleteSprite(pSrcSprite,true);

}

if (!pTarSprite->IsMyFighter())

{

if (pTarSprite->IsDead())

m\_SpriteList.DeleteSprite(pTarSprite,true);

}

}

**2、通过链表实现众多精灵的管理**

实现方法：建立链表类，设置其一个指针，调用自定义的增添、删减、查找函数完成管理项目

实现代码：

void CGameMain::OnSpriteColSprite(const char\* szSrcName,const char\* szTarName)

{

if (2!=GetGameState())

return ;

CGameSprite\* pSrcSprite=IsMyFighter(szSrcName)?m\_pMyFighter:GetSprite(szSrcName);

CGameSprite\* pTarSprite=IsMyFighter(szTarName)?m\_pMyFighter:GetSprite(szTarName);

if (NULL==pSrcSprite||NULL==pTarSprite)

return ;

pSrcSprite->OnColOtherSprite(pTarSprite);

pTarSprite->OnColOtherSprite(pSrcSprite);

if (!pSrcSprite->IsMyFighter())

{

if (pSrcSprite->IsDead())

m\_SpriteList.DeleteSprite(pSrcSprite,true);

}

if (!pTarSprite->IsMyFighter())

{

if (pTarSprite->IsDead())

m\_SpriteList.DeleteSprite(pTarSprite,true);

}

}

#include "CSpriteList.h"

#include "CGameSprite.h"

CSpriteList::CSpriteList(const char\* szName)

{

//ctor

}

CSpriteList::CSpriteList()

{

}

CSpriteList::~CSpriteList()

{

//dtor

}

int m\_iListSize=0;

SpriteStruct\* m\_pListHeader=NULL;

CGameSprite\* CSpriteList::GetSprite(const char\*szName)

{

SpriteStruct\* pPtr=m\_pListHeader;

while (NULL!=pPtr)

{

if (strcmp(pPtr->pSprite->GetName(),szName)==0)

return pPtr->pSprite;

pPtr=pPtr->pNext;

}

return NULL;

}

CGameSprite\* CSpriteList::GetSprite(const int ilndex)

{

int iLoop=0;

SpriteStruct\* pPtr=m\_pListHeader;

while (NULL!=pPtr)

{

if (iLoop==ilndex)

return pPtr->pSprite;

iLoop++;

pPtr=pPtr->pNext;

}

return NULL;

}

SpriteStruct\*CSpriteList::AddSprite(CGameSprite\* pSprite)

{

if (NULL==pSprite)

return NULL;

SpriteStruct\* pPtr=new SpriteStruct;

pPtr->pSprite=pSprite;

pPtr->pNext=NULL;

pPtr->pPrev=NULL;

if (NULL==m\_pListHeader)

{

m\_pListHeader=pPtr;

}

else

{

SpriteStruct\* pTemp=m\_pListHeader;

while (NULL!=pTemp->pNext)

{

pTemp=pTemp->pNext;

}

pPtr->pPrev=pTemp;

pTemp->pNext=pPtr;

}

m\_iListSize++;

return pPtr;

}

void CSpriteList::DeleteSprite(const char\* szName,bool bDeletelmage)

{

SpriteStruct\* pPtr=NULL;

for (pPtr=m\_pListHeader;NULL!=pPtr;pPtr=pPtr->pNext)

{

if (strcmp(szName,pPtr->pSprite->GetName())==0)

{

if (NULL!=pPtr->pNext)

{

pPtr->pNext->pPrev=pPtr->pPrev;

}

if (NULL!=pPtr->pPrev)

{

pPtr->pPrev->pNext=pPtr->pNext;

}

if (pPtr==m\_pListHeader)

{

m\_pListHeader=m\_pListHeader->pNext;

}

if (bDeletelmage)

{

pPtr->pSprite->DeleteSprite();

}

delete pPtr;

m\_iListSize--;

return ;

}

}

}

void CSpriteList::DeleteSprite(CGameSprite\* pSprite,bool bDeletelmage)

{

SpriteStruct\* pPtr=NULL;

for (pPtr=m\_pListHeader;NULL!=pPtr;pPtr=pPtr->pNext)

{

if (pPtr->pSprite==pSprite)

{

if (NULL!=pPtr->pNext)

{

pPtr->pNext->pPrev=pPtr->pPrev;

}

if (NULL!=pPtr->pPrev)

{

pPtr->pPrev->pNext=pPtr->pNext;

}

if (pPtr==m\_pListHeader)

{

m\_pListHeader=m\_pListHeader->pNext;

}

if (bDeletelmage)

{

pPtr->pSprite->DeleteSprite();

}

delete pPtr;

m\_iListSize--;

return ;

}

}

}

void CSpriteList::DeleteAllSprite(bool bDeletelmage)

{

SpriteStruct\* pPtr=NULL;

SpriteStruct\* pPtrhNext=m\_pListHeader;

while (NULL!=pPtrhNext)

{

pPtr=pPtrhNext;

pPtrhNext=pPtrhNext->pNext;

if (bDeletelmage)

pPtr->pSprite->DeleteSprite();

delete pPtr;

}

m\_pListHeader=NULL;

m\_iListSize=0;

}

**3、实现了文件流输入输出显示分数**

实现方法：利用fstream建立文件流运算符 ScoreFile，进行输入输出

实现代码：

fstream ScoreFile("Score.dat",fstream::in|fstream::binary);

if (ScoreFile.is\_open())

{

ScoreFile>>m\_iMaxScore;

ScoreFile.close();

}

m\_pMaxScoreText->SetTextValue(m\_iMaxScore);

if (m\_iMaxScore<GetMyFighter()->GetScore())

{

m\_iMaxScore=GetMyFighter()->GetScore();

fstream ScoreFile("Score.dat",fstream::out|fstream::binary);

if (ScoreFile.is\_open())

{

ScoreFile<<m\_iMaxScore;

ScoreFile.close();

}

}

**4、实现了特效显示**

实现方法：利用CEffect类中的PlayEffect函数实现

实现代码：

g\_GameMain.m\_penemyExplode->PlayEffect(pOther->GetSpritePositionX(),pOther->GetSpritePositionY(),0.f);

g\_GameMain.m\_pSmallExplosion->PlayEffect(pOther->GetSpritePositionX(),pOther->GetSpritePositionY(),0.f);

g\_GameMain.m\_pPlayerExplode->PlayEffect(pOther->GetSpritePositionX(),pOther->GetSpritePositionY(),0.f);

**六、遇到的问题与解决方法**

**1、程序运行结果显示许多函数没有被声明**

原因分析：没有包含其他类的头文件

解决方法：直接在cpp文件首行添加

**2、敌机出现一直下行，没有上浮**

原因分析：在进行判断时

float fPosY=GetSpritePositionY();

fPosY-=6.f\*fDeltaTime;

SetSpritePositionY(fPosY); -=写成了 +=

解决方法：将符号纠正

**3、按下空格键程序直接退出问题**

原因分析：创建链表对象时，定义了一个指针而非直接的对象 CSpriteList\* m\_SpriteList;

解决方法：将\*号去掉 CSpriteList m\_SpriteList;

**4、运行程序特别卡的问题**

原因分析：敌机子弹发射函数参数不合理，子弹属性参数未初始化

解决方法：合理设置参数，初始化子弹属性

m\_fCreateTime =(float)CSystem::RandomRange(5,10);

m\_fCanFileAfterCreated =3.f;

m\_fBulletCreateTime =1.f;

m\_fFloatTime =0.f;

m\_bFloatUp =true;

**5、程序运行一会后界面中间出现不连续问题**

原因分析：出现了内存泄漏或者子弹产生边界出现了漏洞

解决方法：更改代码

pSprite->SetSpriteWorldLimit(WORLD\_LIMIT\_KULL, ...) 为

pSprite->SetSpriteWorldLimit(WORLD\_LIMIT\_NULL, ...) 即可

**6、敌机不消失问题**

原因分析：在main.cpp文件中偶然删去一行代码

解决方法：添加代码

g\_GameMain.OnSpriteColSprite(szSrcName,szTarName);

**7、最高分无法刷新问题**

原因分析：错将指针先delete指向NULL，在放入函数中判断导致判断失效

delete m\_pMyFighter;

m\_pMyFighter=NULL;

if (m\_iMaxScore<GetMyFighter()->GetScore())

{

m\_iMaxScore=GetMyFighter()->GetScore();

fstream ScoreFile("Score.dat",fstream::out|fstream::binary);

if (ScoreFile.is\_open())

{

ScoreFile<<m\_iMaxScore;

ScoreFile.close();

}

}

解决方法：调换上述代码顺序

if (m\_iMaxScore<GetMyFighter()->GetScore())

{

m\_iMaxScore=GetMyFighter()->GetScore();

fstream ScoreFile("Score.dat",fstream::out|fstream::binary);

if (ScoreFile.is\_open())

{

ScoreFile<<m\_iMaxScore;

ScoreFile.close();

}

}

delete m\_pMyFighter;

m\_pMyFighter=NULL;

**七、程序运行情况描述**

游戏初始界面如图2所示。



图2 游戏初始界面

当按空格键游戏开始后，会从右边陆续出现敌机并且发射子弹，如图3所示。



图3 游戏开始，遭遇敌机

己方战机发现敌机，立即躲避子弹，并主动还击，如图4所示。



图4 战机躲避子弹

通过键盘控制飞机的浮动，飞机原来从左下方通过玩家操控，变为向上方移动，效果如图5所示。

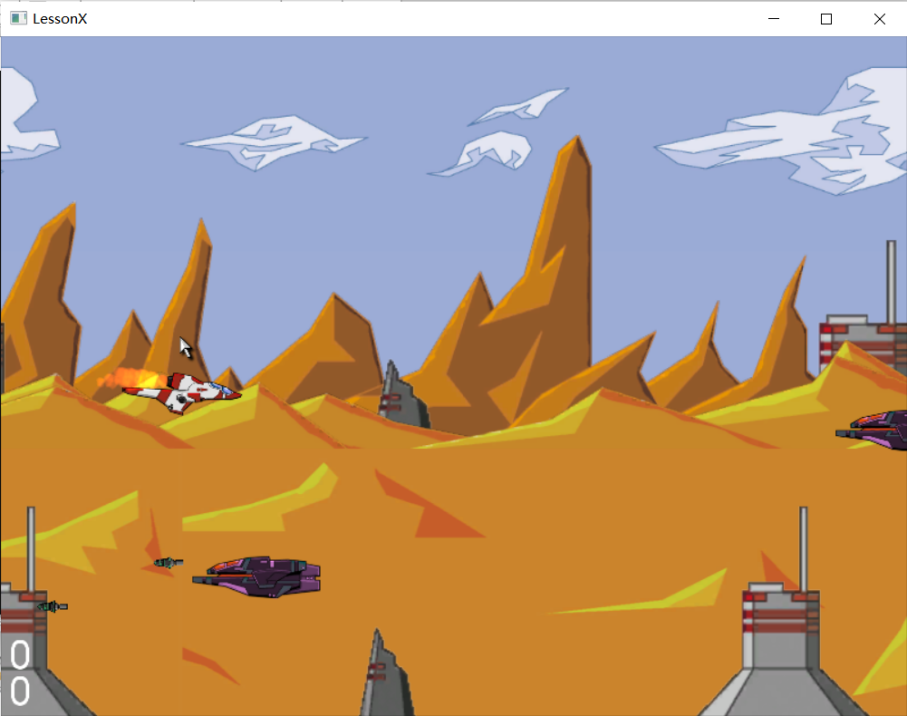


图5 键盘控制战机移动

战机在枪零弹雨与飞翔，玩家通过按下空格键开火击败敌机，并获得积分，效果如图6所示。



图6 战机发射子弹攻击敌机，敌机消失，玩家获得积分

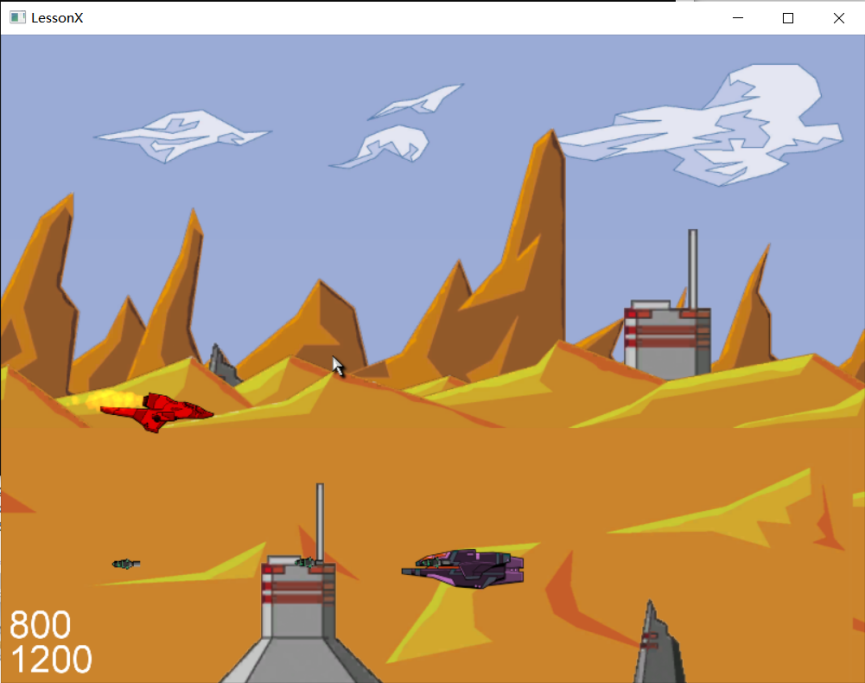


图7 战机被敌机攻击受损，颜色变红

**八、总结与反思**

通过运行测试，游戏实现了系统所要求的飞机的自由移动·、按下空格键开火、敌机被消灭、键盘控制等功能，并且还实现了敌机出现位置与时间可以控制的扩展功能。并且在实现项目的过程中，发现问题，分析问题，解决问题的方法也将有助于我以后的学习。