山东科技大学

SHANDONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

程序设计综合实践报告

不能抄袭，学弟学妹只能参考！！！

姓名	22 级学长	学号	202211070511
班级		项目名称	飞机大战游戏设计与实现
实习地点	软件工程实验室	实习时间	2023-6-26 至 2023-7-7
实习成绩		指导教师签字	

计算机科学与工程学院

2023 年 07 月 07 日

目　录

程序设计综合实践报告 1

1. 概述 3

2. 相关技术 4

2.1 血条绘制函数 4

2.2 分数显示 4

2.3 敌机类的实现（以蓝色小型敌机为例） 6

2.4 关卡类的实现 9

2.5 历史最高分的读取 10

2.6 双缓冲技术 11

2.7 设置定时器 12

2.8 获取矩形区域函数 12

3. 总体设计与详细设计 12

3.1 系统模块划分 12

3.2 主要功能模块 13

4. 编码实现 13

5. 实训中遇到的主要问题及解决方法 47

6. 实训分工及体会 47

1. 概述

本次项目是一个基于 MFC 进行 Windows 版本竖版飞机大战游戏的设计与实现。

主要实现功能：玩家对战机的控制（发射子弹、移动等）、敌机类的实现、关卡类的实现、历史最高分的记录、血包类的实现、BOSS 的实现。

程序要求：游戏可玩性高，玩家可持续玩十分钟以上，运行流畅，关卡难度安排合理，游戏屏幕上的对象数目合理、界面易于区分各种角色、素材搭配的视觉融合性好。

详细玩法与操作介绍：

（1）玩家负责操纵一架血量为 100 点的战机，通过 W、A、S、D 控制飞机的上、左、下、右移动；

（2）游戏共有三个关卡，每个关卡的敌机刷新频率会逐渐提高，敌机子弹的伤害也会逐渐提高，玩家每击杀一个普通敌机获得 1 分，击杀特殊敌机获得 10 分，游戏结束后返回本次游戏的分数以及是否超过历史最高分。

（3）游戏中共有以下四种特殊敌机：

第一种：蓝色小型敌机，飞机出现时左右摇摆，最后速度加快，向玩家方向发起冲刺。

第二种：蓝色大型敌机，飞机出现后投掷三枚不同方向的子弹。

第三种：灰绿色敌机，飞机出现后定时投掷追踪导弹，追踪导弹会在一定时间内对玩家战机追踪，这段时限结束后导弹继续向下冲刺。

第四种：红色战机，会向周围以圆形发射子弹。

2. 相关技术

2.1 血条绘制函数

采用 CPen 和 CBrush 的方法绘制，并且封装成一个函数。

完整实现：

```
void CPlaneGameView::drawBloodBar(int x, int y, int width, int height, int lineWidth,
COLORREF boardColor, COLORREF emptyColor, COLORREF fillColor, float percent)
{
    CDC* pDC= m_pMemDC;
    CPen penBloodBar;
    CBrush brushEmpty, brushFill;

    penBloodBar.CreatePen(PS_SOLID | PS_ENDCAP_ROUND, lineWidth, boardColor);
    brushEmpty.CreateSolidBrush(emptyColor);
    brushFill.CreateSolidBrush(fillColor);

    CPen* pOldPen = pDC->SelectObject(&penBloodBar);
    CBrush* pOldBrush = pDC->SelectObject(&brushEmpty);

    CRect rectEmpty(x, y, x + width, y + height);
    pDC->Rectangle(&rectEmpty);

    pDC->SelectObject(&brushFill);
    pDC->SelectObject(pOldPen);

    if (percent > 0) {
        CRect rectFill(x + 0.5 * lineWidth, y + lineWidth * 0.5, x + width * percent,
y + height - 0.5 * lineWidth);
        pDC->Rectangle(&rectFill);
    }

    pDC->SelectObject(pOldBrush);
}
```

2.2 分数显示

```
// 初始化 GDI+
Gdiplus::GdiplusStartupInput gdiplusStartupInput;
ULONG_PTR gdiplusToken;
Gdiplus::GdiplusStartup(&gdiplusToken, &gdiplusStartupInput, NULL);

CString strScore;
strScore.Format(_T("%d"), m_score);

// 加载数字图片
for (int i = 0; i < strScore.GetLength(); i++)
{
    wchar_t filePath[50];
    swprintf(filePath, L"res//sz//%c.png", strScore[i]);
    Gdiplus::Bitmap sz(filePath);

    // 计算图片位置
    int x = 20 + i * 50;
    int y = 25;

    // 绘制数字图片
    Gdiplus::Graphics graphics(pMemDC->m_hDC);
    graphics.DrawImage(&sz, x, y, 40, 60);
}
```

初始化 GDI+，创建了一个 GdiplusStartupInput 对象 gdiplusStartupInput 和一个 ULONG_PTR 类型的变量 gdiplusToken。然后调用 GdiplusStartup 函数初始化 GDI+，传入参数&gdiplusToken 和&gdiplusStartupInput。这段代码创建了一个 CString 对象 strScore 并使用 Format 函数将分数 m_score 转换为字符串格式，并存储到 strScore 中。这部分代码用于加载数字图片并绘制到设备上下文中。通过循环遍历 strScore 字符串的每个字符，创建一个 wchar_t 类型的数组 filePath 用于存储数字图片的路径。路径格式为 res//sz//%c.png，其中%c 会被替换为对应的数字字符。然后使用加载图片的路径创建一个 Gdiplus::Bitmap 对象 sz，表示数字图片。接下来，计算每张图片的绘制位置，通过变量 x 和 y 分别表示横坐标和纵坐标。最后，使用设备上下文对象 pMemDC->m_hDC 创建一个 Gdiplus::Graphics 对象 graphics，并调用 DrawImage 方法将数字图片绘制到指定的位置，并指定图片的宽度和高度为 40 和 60。这样，通过循环遍历分数字符串的每个字符并绘制对应的数字图片，就可以在设备上下文中显示出分数的数字图形。

2.3 敌机类的实现（以蓝色小型敌机为例）

这段代码是一个基于 MFC 的游戏程序中的敌人类（CEnemy2），提供了一些成员函数来实现敌人的移动、绘制以及发射子弹等功能。以下是每个函数的功能说明：构造函数 CEnemy2::CEnemy2(void)初始化敌人对象的位置、图像索引、方向、速度等属性。重载构造函数 CEnemy2::CEnemy2(int c)根据参数 c 初始化敌人对象的位置、图像索引、方向、速度等属性，用于控制敌人的冲锋方向。析构函数 CEnemy2::~CEnemy2(void)销毁敌人对象。静态成员函数 CEnemy2::LoadImage()加载游戏对象的图标对象，并创建一个 CImageList 对象来存储图标对象。成员函数 CEnemy2::Draw(CDC* pDC, BOOL bPause)绘制敌人对象，根据敌人的位置和状态绘制对应的图标。参数 pDC 是一个指向设备上下文的指针，用于绘制图标。参数 bPause 表示游戏是否暂停，如果为 TRUE，则敌人不会移动。成员函数 CEnemy2::Fired()判断敌人是否可以发射子弹，根据等待时间间隔进行判断。

```
//Enemy2.h
#pragma once
#include "gameobject.h"
#include "Enemy.h"
#define ENEMY_HEIGHT 90//敌机的高度
#define ENEMY_WIDTH 120//敌机的宽度
class CEnemy2 :public CEnemy
{
public:
    CEnemy2(void);
    CEnemy2(int c);
    ~CEnemy2(void);
    BOOL Draw(CDC* pDC, BOOL bPause);
    static BOOL LoadImage();
    int GetMontion() const{return m_nMotion;}
    //是否可以开火发射子弹
    BOOL Fired();
private:
    int m_xV;//水平方向速度
    int imgCnt;//敌机种类
    int m_freshFreaq;//刷新帧频率
public:
    int enhancedEnemyBlood = 80;//血量（相对于玩家100点血量）
    int m_chargeDirection;//最后冲锋方向
    static CImageList m_Images;
};
```

```cpp
//Enemy2.cpp
#include "StdAfx.h"
#include "Enemy2.h"
#include "resource.h"
CImageList CEnemy2::m_Images;
CEnemy2::CEnemy2(void)
{
    //随机确定X位置
    m_ptPos.x = (GAME_WIDTH / 2) - (ENEMY_WIDTH / 2)-rand()%200;
    //随机确定图像索引
    m_nImgIndex = 0;
    //根据图像索引确定方向
    m_nMotion = 1;
    m_ptPos.y = -ENEMY_HEIGHT;
    if (m_nImgIndex % 2 != 0)//如果是图像索引是偶数
    {
        m_nMotion = -1;
        m_ptPos.y = GAME_HEIGHT + ENEMY_HEIGHT;
    }
    m_freshFreaq = 0;
    //随机确定速度
    m_V = 3;
    m_xV = rand()%6+5;
    m_chargeDirection = 1;
    m_nWait = 0;
}
CEnemy2::CEnemy2(int c)
{
    //随机确定X位置
    m_ptPos.x = (GAME_WIDTH / 2) - (ENEMY_WIDTH / 2) - rand() % 200;
    //随机确定图像索引
    m_nImgIndex = 0;
    //根据图像索引确定方向
    m_nMotion = 1;
    m_ptPos.y = -ENEMY_HEIGHT;
    if (m_nImgIndex % 2 != 0)//如果是图像索引是偶数
    {
        m_nMotion = -1;
        m_ptPos.y = GAME_HEIGHT + ENEMY_HEIGHT;
    }
    m_freshFreaq = 0;
    //随机确定速度
    m_V = 3;
    m_xV = rand() % 6 + 5;
    m_nWait = 0;
    m_chargeDirection = c;
}
```

```
BOOL CEnemy2::LoadImage()
{   //加载游戏对象的图标对象
    CBitmap enemy1;
    if (!enemy1.LoadBitmap(IDB_ENHANCED_ENEMY))
        return FALSE;
    //创建CImageList对象
    if (!m_Images.Create(ENEMY_WIDTH, ENEMY_HEIGHT, ILC_COLOR24 | ILC_MASK, 5, 0))
        return FALSE;//cx,cy 图片的宽度
    //图像链表中加入对象对应的图标对象，之后直接通过该链表访问图标对象
    m_Images.Add(&enemy1, RGB(0, 0, 0));
}
BOOL CEnemy2::Draw(CDC* pDC, BOOL bPause)
{
    m_nWait++;
    if (m_nWait > 10)
        m_nWait = 0;

    if (!bPause)
    {
        m_freshFreaq++;
        if (m_freshFreaq < 40)m_ptPos.y = m_ptPos.y + m_nMotion * m_V;
        else if (m_freshFreaq >= 40 && m_freshFreaq <= 70)
        m_ptPos.x = m_ptPos.x + m_nMotion * m_xV;
        else if (m_freshFreaq > 70 && m_freshFreaq <= 100)
        m_ptPos.x = m_ptPos.x - m_nMotion * m_xV;
        else if (m_freshFreaq > 100 && m_freshFreaq <= 120)
        m_ptPos.x = m_ptPos.x + m_nMotion * m_xV;
        else if (m_freshFreaq > 120 && m_freshFreaq <= 140)
        m_ptPos.x = m_ptPos.x - m_nMotion * m_xV;
        else if (m_freshFreaq > 140 && m_freshFreaq <= 160)
        m_ptPos.x = m_ptPos.x + m_nMotion * m_xV;
        else if (m_freshFreaq > 140 && m_freshFreaq <= 180)
        m_ptPos.x = m_ptPos.x - m_nMotion * m_xV;
        else if(m_freshFreaq >180 && m_freshFreaq <= 240 )
        m_ptPos.y = m_ptPos.y + m_nMotion * m_V;
        else if (m_freshFreaq > 240 && m_freshFreaq <= 250)
        m_ptPos.x = m_ptPos.x + m_nMotion * m_xV;
        else if (m_freshFreaq > 250 && m_freshFreaq <= 260)
        m_ptPos.x = m_ptPos.x - m_nMotion * m_xV;
        else if (m_freshFreaq > 260 && m_freshFreaq <= 280)
        m_ptPos.y = m_ptPos.y + m_nMotion * m_V;
        else if(m_freshFreaq > 280 && m_freshFreaq <= 320)
        {
            m_ptPos.y = m_ptPos.y + m_nMotion * m_V * 1.5;
            m_ptPos.x = m_ptPos.x + m_nMotion * m_xV * m_chargeDirection;
        }
        else m_ptPos.y = m_ptPos.y + m_nMotion * m_V * 5;
    if (m_ptPos.y > GAME_HEIGHT + ENEMY_HEIGHT)
        return FALSE;
```

```
if (m_ptPos.y < -ENEMY_HEIGHT)
        return FALSE;
    m_Images.Draw(pDC, 1, m_ptPos, ILD_TRANSPARENT);
    return TRUE;
}
BOOL CEnemy2::Fired()
{
    if (m_nWait == 0)
        return TRUE;
    else
        return FALSE;
}
```

2.4 关卡类的实现

设置一个关卡类 Level.cpp，使得可以通过 gameLevel->enhanced2Freq 的方式访问一些 PlaneGameView.cpp 里面 Level 类型的属性，并且通过 setLevelData 函数调整关卡属性进而控制难度。

```
Level.h
#pragma once
class Level
{
public:
    int enhanced1Freq;int enhanced2Freq;int enhanced3Freq;
    int UFOFreq;int backgroundSpeed;int normalEnemyHarm;
    int ufoBallHarm;int CollideHarm;int enhancedBallHarm;
    int fullScore;int missleHarm;int BOSSFreq;
    wchar_t bg[50]= L"plane\\bg06.jpg";
public:
    void setLevelData(int enhanced1Freq_, int enhanced2Freq_, int enhanced3Freq_, int UFOFreq_,
int backgroundSpeed_,
        int normalEnemyHarm_, int ufoBallHarm_, int CollideHarm_, int enhancedBallHarm_,int
missleHarm_,int fullScore_, int BOSSFreq_);
    Level();
    ~Level();
    Level(int enhanced1Freq_, int enhanced2Freq_, int enhanced3Freq_, int UFOFreq_, int
backgroundSpeed_,
        int normalEnemyHarm_, int ufoBallHarm_, int CollideHarm_, int enhancedBallHarm_, int
missleHarm_, int fullScore_, int BOSSFreq_);
    wchar_t getBg()
    {
        return bg[50];
    }
};
```

2.5 历史最高分的读取

在项目文件夹目录下添加 record.txt 文档，通过 C++的文件操作，在每次游戏开始和结束时都进行更新历史最高分的操作。

```
void CPlaneGameView::ReadHighestScore()
{
    ifstream recordFile("record.txt");
    if (!recordFile)
    {
        // 文件打开失败，返回默认的最高分数
        return;
    }
    string line;
    while (getline(recordFile, line))
    {
        stringstream ss(line);
        string title;
        int score;
        if (ss >> title >> score && title == "最高分数")
        {
            m_highScore = score;
            break;
        }
    }
    recordFile.close();
}
int lastHighScore = m_highScore;
    if (m_IsWin)
    {
        if (m_score > lastHighScore)
        {
            std::ofstream recordFile("record.txt", std::ios::out);
            if (!recordFile)
            {
                CString str;
                str.Format(_T("存档文件损坏"));
                AfxMessageBox(str);
                return;
            }
            m_IsWin = false;
            m_highScore = m_score;
            recordFile << "最高分数 " << m_highScore << std::endl;
            recordFile.close();
        }
```

```
else
        {
            if (m_score == lastHighScore)
            {
                CString str;
                str.Format(_T("您的分数：%d，恭喜您超越了记录获得历史游戏最高分"),
m_score);
                pMemDC->SetBkMode(TRANSPARENT);
                pMemDC->SetTextColor(WHITE);
                pMemDC->TextOut(20, 550, str);
            }
            else
            {
                CString str,str1;
                str.Format(_T("您的分数：%d，很遗憾这一次没有创造历史"), m_score);
                str1.Format(_T("距离历史游戏最高分%d分仅差%d分"),m_highScore, m_highScore
- m_score);
                //AfxMessageBox(str);
                pMemDC->SetBkMode(TRANSPARENT);
                pMemDC->SetTextColor(WHITE);
                //pMemDC->SetTextAlign(TA_CENTER);
                pMemDC->TextOut(50, 550, str);
                pMemDC->TextOut(100, 570, str1);
                m_IsWin = false;
            }
        }
    }
```

2.6 双缓冲技术

关于双缓冲技术主要就是利用缓存的原理进行将所有的东西都先存在一个缓冲得虚拟的区域，然后再一次性的将所有的虚拟缓存中的东西都放入实在的存储器中。

```
//建立设备 DC
m_pDC = new CClientDC(this);

//建立内存 DC
m_pMemDC = new CDC;
m_pMemDC->CreateCompatibleDC(m_pDC);

//建立内存位图
m_pMemBitmap = new CBitmap;
m_pMemBitmap->CreateCompatibleBitmap(m_pDC,GAME_WIDTH,GAME_HEIGHT);

//将位图选入内存 DC
m_pMemDC->SelectObject(m_pMemBitmap);
```

2.7 设置定时器

定时器告诉 WINDOWS 一个时间间隔，然后 WINDOWS 以此时间间隔周期性触发程序。通常有两种方法来实现：发送 WM_TIMER 消息和调用应用程序定义的回调函数。

```
SetTimer(1,30,NULL); //设置每 30 毫秒刷新一次
```

2.8 获取矩形区域函数

```
CRect bRect = pBomb->GetRect();//获得矩形区域
CRect GetRect()
{
        return CRect(m_ptPos,CPoint(m_ptPos.x+10,m_ptPos.y+BOMB_HEIGHT));
}
static const int BOMB_HEIGHT = 20;//图片定义
```

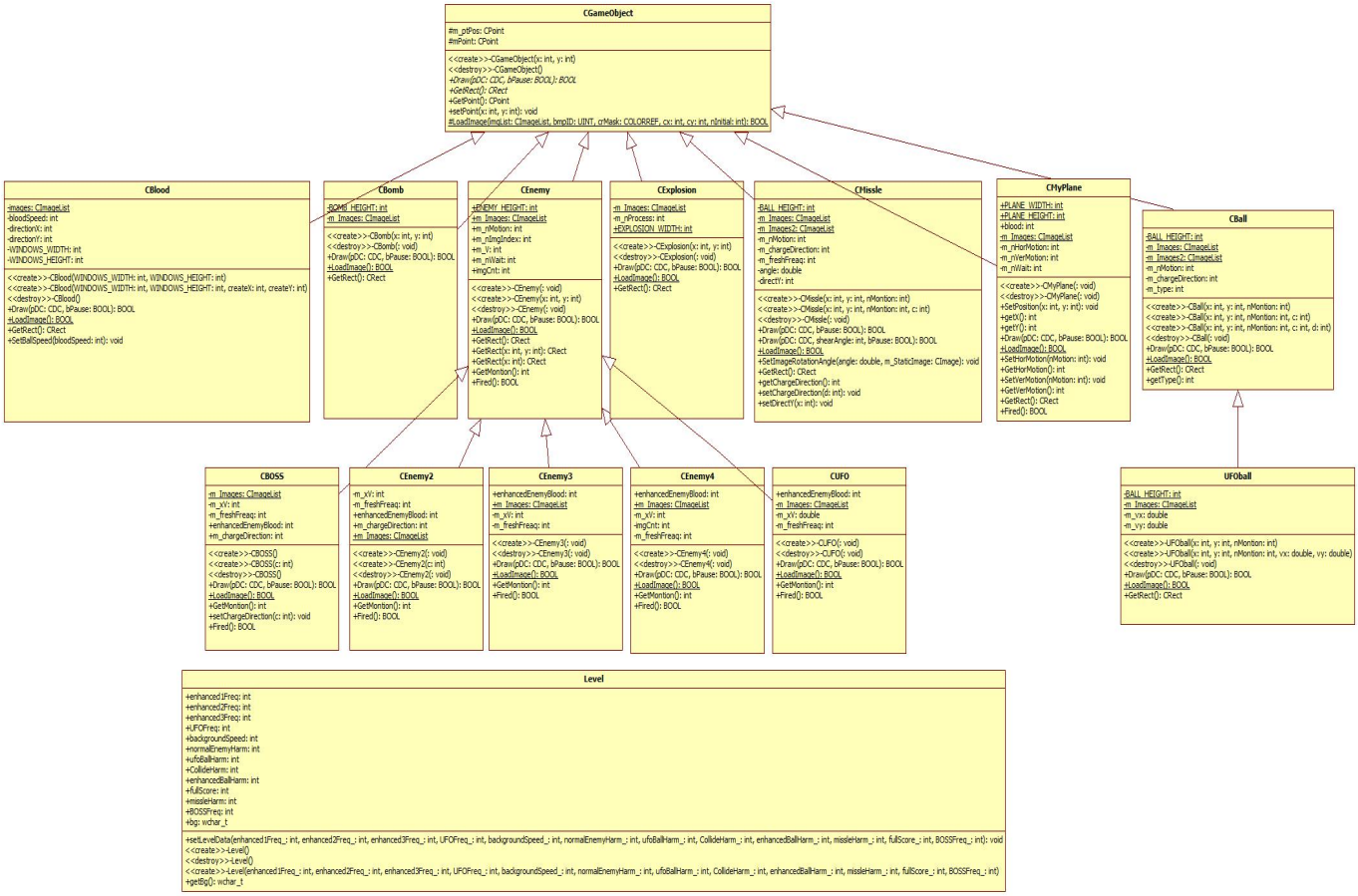
3. 总体设计与详细设计

3.1 系统模块划分

模块名称	功能简述
应用程序对象	游戏程序的加载、游戏对象的绘制、游戏规则的调用、玩家的键盘事件获取。
游戏对象	各个游戏对象的抽象父类
战机对象	战机类，继承于游戏对象父类
敌机对象	敌机类，继承游戏父类
加强敌机 1 对象	敌机 2 类，继承敌机类
加强敌机 2 对象	敌机 3 类，继承敌机类
加强敌机 3 对象	敌机 4 类，继承敌机类
加强敌机 4 对象	UFO 类，继承敌机类
BOSS 对象	BOSS 类，继承敌机类
血包对象	血包类，继承游戏对象父类
敌机子弹对象	CBall 类，继承游戏对象父类
战机子弹对象	CBomb 类，继承游戏对象父类
追踪导弹对象	Missile 类，继承游戏对象父类
关卡对象	关卡类

3.2 主要功能模块

UML 图如下所示：



4. 编码实现

// PlaneGameView.cpp : CPlaneGameView 类的实现

```
#include "stdafx.h"
#include "PlaneGame.h"
#include "gdiplus.h"
#include "PlaneGameDoc.h"
#include "PlaneGameView.h"
#include "MyPlane.h"
#include "Enemy.h"
#include "Enemy2.h"
#include "Enemy3.h"
#include "Enemy4.h"
#include "Bomb.h"
#include "Ball.h"
#include "missle.h"
#include "UFO.h"
#include "UFOball.h"
#include "Explosion.h"
```

```
#include "BOSS.h"
#include <atlimage.h>
#include <string>
#include <graphics.h>
#include <easyx.h>
#include <mmsystem.h>
#include "bloodAdd.h"
#include <fstream> //  文件读写操作
#include <string>   //  字符串处理
#include <ctime>     //  时间处理
#pragma comment(lib,"winmm.lib")
using namespace std;
#ifdef _DEBUG
#define new DEBUG_NEW
#endif
IMPLEMENT_DYNCREATE(CPlaneGameView, CView)
BEGIN_MESSAGE_MAP(CPlaneGameView, CView)
	//  标准打印命令
	ON_COMMAND(ID_FILE_PRINT, &CView::OnFilePrint)
	ON_COMMAND(ID_FILE_PRINT_DIRECT, &CView::OnFilePrint)
	ON_COMMAND(ID_FILE_PRINT_PREVIEW, &CView::OnFilePrintPreview)
	ON_WM_TIMER()
	ON_WM_PAINT()
	ON_WM_MOUSEMOVE()
END_MESSAGE_MAP()
// CPlaneGameView  构造/析构
void    playSound(const char* name) {
	static int index = 1;
	char cmd[512];

	if (index == 1) {
		sprintf_s(cmd, sizeof(cmd), "play %s-1", name);
		mciSendString(cmd, 0, 0, 0);
		sprintf_s(cmd, sizeof(cmd), "close %s-2", name);
		mciSendString(cmd, 0, 0, 0);
		sprintf_s(cmd, sizeof(cmd), "open %s alias %s-2", name, name);
		mciSendString(cmd, 0, 0, 0);
		index++;
	}
	else if (index == 2) {
		sprintf_s(cmd, sizeof(cmd), "play %s-2", name);
		mciSendString(cmd, 0, 0, 0);
		sprintf_s(cmd, sizeof(cmd), "close %s-1", name);
		mciSendString(cmd, 0, 0, 0);
```

```
            sprintf_s(cmd, sizeof(cmd), "open %s alias %s-1", name, name);
            mciSendString(cmd, 0, 0, 0);
            index = 1;
        }
    }
    void CPlaneGameView::ReadHighestScore()
    {
        ifstream recordFile("record.txt");
        if (!recordFile)
        {
            // 文件打开失败，返回默认的最高分数
            return;
        }
        string line;
        while (getline(recordFile, line))
        {
            stringstream ss(line);
            string title;
            int score;
            if (ss >> title >> score && title == "最高分数")
            {
                m_highScore = score;
                break;
            }
        }
        recordFile.close();
    }
    CPlaneGameView::CPlaneGameView():m_pMe(NULL),m_score(0),playerBlood(100),m_IsWin(f
    alse), m_isScoreWindowCreated(false), isBOSSDie(false),isBOSSExist(false)
    {
        // TODO: 在此处添加构造代码
        bg = new wchar_t[15];
        wcscpy(bg, L"plane\\bg00.png");
        gameLevel = new Level(30, 40, 50, 60, 1, 5, 3, 10, 5, 5, 100,60);
        ReadHighestScore();
    }
    CPlaneGameView::~CPlaneGameView()
    {
        delete m_pMe;
        delete m_pMemDC;
        delete m_pDC;
        delete m_pMemBitmap;
        delete[] bg;
    }
```

```
BOOL CPlaneGameView::PreCreateWindow(CREATESTRUCT& cs)
{
	playSound("plane\\3.mp3");
	return CView::PreCreateWindow(cs);
}
void CPlaneGameView::OnDraw(CDC* /*pDC*/)
{
	CPlaneGameDoc* pDoc = GetDocument();
	ASSERT_VALID(pDoc);
	if (!pDoc)
		return;
}
BOOL CPlaneGameView::OnPreparePrinting(CPrintInfo* pInfo)
{
	return DoPreparePrinting(pInfo);
}

void CPlaneGameView::OnBeginPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/)
{
}

void CPlaneGameView::OnEndPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/)
{

}
#ifdef _DEBUG
void CPlaneGameView::AssertValid() const
{
	CView::AssertValid();
}
void CPlaneGameView::Dump(CDumpContext& dc) const
{
	CView::Dump(dc);
}
CPlaneGameDoc* CPlaneGameView::GetDocument() const //  非调试版本是内联的
{
	ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CPlaneGameDoc)));
	return (CPlaneGameDoc*)m_pDocument;
}
#endif //_DEBUG
void CPlaneGameView::OnInitialUpdate()
{
	CView::OnInitialUpdate();
	// TODO:  在此添加专用代码和/或调用基类
```

```
	//初始化游戏
	InitGame();
}
void CPlaneGameView::StopGame()
{
	delete m_pMe;
	delete m_pMemDC;
	delete m_pDC;
	delete m_pMemBitmap;
}
void CPlaneGameView::Remove()
{
	m_ObjList[enEnhanced].RemoveAll();
	m_ObjList[enEnemy].RemoveAll();
	m_ObjList[enEnhanced2].RemoveAll();
	m_ObjList[enEnhanced3].RemoveAll();
	m_ObjList[enBall].RemoveAll();
	m_ObjList[enMissle].RemoveAll();
	m_ObjList[enBOSS].RemoveAll();
	isExist = FALSE;
	delete m_pMe;
	m_pMe = NULL;
	mciSendString("stop plane/3.mp3", 0, 0, 0);//关闭背景音乐
}
int CPlaneGameView::k = 1;
BOOL CPlaneGameView::InitGame()
{
	CRect rc;
	GetClientRect(rc);
	//产生随机数种子
	srand( (unsigned)time( NULL ) );
	//建立设备 DC
	m_pDC = new CClientDC(this);
	//建立内存 DC
	m_pMemDC = new CDC;
	m_pMemDC->CreateCompatibleDC(m_pDC);
	//建立内存位图
	m_pMemBitmap = new CBitmap;
	m_pMemBitmap->CreateCompatibleBitmap(m_pDC,GAME_WIDTH,GAME_HEIGHT);
	//将位图选入内存 DC
	m_pMemDC->SelectObject(m_pMemBitmap);
	CMyPlane::LoadImage();
	CEnemy::LoadImage();
	CBomb::LoadImage();
```

```
	CBall::LoadImage();
	CExplosion::LoadImage();
	CEnemy2::LoadImage();
	CEnemy3::LoadImage();
	CBlood::LoadImage();
	CMissle::LoadImage();
	CEnemy4::LoadImage();
	CUFO::LoadImage();
	UFOball::LoadImage();
	CBOSS::LoadImage();
	//产生主角(战机)
	m_pMe = new CMyPlane;
	//启动游戏
	SetTimer(1,30,NULL);
	mciSendString("play plane/3.mp3 repeat", 0, 0, 0);//开启背景音乐
	return TRUE;
}
void CPlaneGameView::drawBloodBar(int x, int y, int width, int height, int lineWidth, COLORREF boardColor, COLORREF emptyColor, COLORREF fillColor, float percent)
{
	CDC* pDC= m_pMemDC;
	CPen penBloodBar;
	CBrush brushEmpty, brushFill;
	penBloodBar.CreatePen(PS_SOLID | PS_ENDCAP_ROUND, lineWidth, boardColor);
	brushEmpty.CreateSolidBrush(emptyColor);
	brushFill.CreateSolidBrush(fillColor);
	CPen* pOldPen = pDC->SelectObject(&penBloodBar);
	CBrush* pOldBrush = pDC->SelectObject(&brushEmpty);
	CRect rectEmpty(x, y, x + width, y + height);
	pDC->Rectangle(&rectEmpty);
	pDC->SelectObject(&brushFill);
	pDC->SelectObject(pOldPen);
	if (percent > 0) {
		CRect rectFill(x + 0.5 * lineWidth, y + lineWidth * 0.5, x + width * percent, y + height - 0.5 * lineWidth);
		pDC->Rectangle(&rectFill);
	}
	pDC->SelectObject(pOldBrush);
}
void CPlaneGameView::UpdateFrame(CDC* pMemDC)
{
	const wchar_t overImg[15] = L"plane\\over.png";
	// 初始化 GDI+
	Gdiplus::GdiplusStartupInput gdiplusStartupInput;
```

```
ULONG_PTR gdiplusToken;
Gdiplus::GdiplusStartup(&gdiplusToken, &gdiplusStartupInput, NULL);
if ((m_score >= gameLevel->fullScore) )
{
    if (k==1&& (isBOSSDie == true))
    {
        m_ObjList[enEnhanced].RemoveAll();
        m_ObjList[enEnemy].RemoveAll();
        m_ObjList[enEnhanced2].RemoveAll();
        m_ObjList[enEnhanced3].RemoveAll();
        m_ObjList[enBall].RemoveAll();
        m_ObjList[enMissle].RemoveAll();
        m_ObjList[enUFO].RemoveAll();
        isExist = FALSE;
        if (GetKey(0x56))
        {
            //Sleep(1000);//效果不好
            m_score = 0;
            playerBlood = 100;
            isExist = TRUE;
            isBOSSDie = false;
            isBOSSExist = false;
            //产生主角(战机)
            m_pMe = new CMyPlane;
            mciSendString("play plane/3.mp3 repeat", 0, 0, 0);//关闭背景音乐
            gameLevel->setLevelData(30 - k * 5, 40 - k * 5, 50 - k * 5, 60 - k * 5, k, 5 + 2
* k, 3 + 2 * k, 10 + 2 * k, 5 + 2 * k, 8+2*k,100,100);
            wcscpy(bg, bg2);
            m_score = 0;
            k++;
            playerBlood = 100;
        }
    }
    else if (k==2&& (isBOSSDie == true))
    {
        m_ObjList[enEnhanced].RemoveAll();
        m_ObjList[enEnemy].RemoveAll();
        m_ObjList[enEnhanced2].RemoveAll();
        m_ObjList[enEnhanced3].RemoveAll();
        m_ObjList[enBall].RemoveAll();
        m_ObjList[enMissle].RemoveAll();
        isExist = FALSE;
        if (GetKey(0x56))
        {
```

```
                    m_score = 0;
                    playerBlood = 100;
                    isExist = TRUE;
                    isBOSSExist = false;
                    isBOSSDie = false;
                    //产生主角(战机)
                    m_pMe = new CMyPlane;
                    mciSendString("play plane/3.mp3 repeat", 0, 0, 0);//关闭背景音乐
                    gameLevel->setLevelData(30 - k * 5, 40 - k * 5, 50 - k * 5, 60 - k * 5, k, 5 + 2
* k, 3 + 2 * k, 10 + 2 * k, 5 + 2 * k, 8 + 2 * k, 100,100);
                    wcscpy(bg, bg3);
                    m_score = 0;
                    k++;
                    playerBlood = 100;
                }
            }
        }
        if (k == 3 && playerBlood <= 0)
        {
            m_IsWin = true;
        }
        Gdiplus::Bitmap skyBitmap(bg);

        // 绘制天空图片
        Gdiplus::Graphics graphics(pMemDC->m_hDC);
        graphics.DrawImage(&skyBitmap, bgX, bgY, GAME_WIDTH, GAME_HEIGHT * 2);

        //绘制我方战机
        if(m_pMe!=NULL)
        {
            m_pMe->Draw(m_pMemDC,FALSE);
        }
        else
        {

            Gdiplus::Bitmap over(overImg);

            // 绘制游戏失败图片
            Gdiplus::Graphics graphics(pMemDC->m_hDC);
            if (isBOSSExist!=true && k != LEVEL_NUM)
            {
                graphics.DrawImage(&over, bgX, GAME_HEIGHT / 2 - 100, 512, 170);
            }
            if (GetKey(0x43))
```

```
        {
            m_score = 0;
            playerBlood = 100;

            isExist = TRUE;

            isBOSSDie = false;
            isBOSSExist = false;
            //产生主角(战机)
            m_pMe = new CMyPlane;
            mciSendString("play plane/3.mp3 repeat", 0, 0, 0);//关闭背景音乐
        }
    }
    if (k != LEVEL_NUM&&m_score>=gameLevel->fullScore)
    {
        if (isBOSSDie == true)
        {
            CString levelDisplay = _T("按 V 键击杀更多敌机.....");
            pMemDC->SetBkMode(TRANSPARENT);
            pMemDC->SetTextColor(WHITE);
            pMemDC->TextOut(160, 550, levelDisplay);
            //isBOSSDie = false;
        }

    }
    //绘制 导弹、爆炸、敌机、子弹、强化敌机、血包
    for(int i=0;i<12;i++)
    {
        POSITION pos1,pos2;
        for( pos1 = m_ObjList[i].GetHeadPosition(); ( pos2 = pos1 ) != NULL; )
        {
            CGameObject* pObj = (CGameObject*)m_ObjList[i].GetNext( pos1 );
            if(!pObj->Draw(pMemDC,FALSE))
            {
                m_ObjList[i].RemoveAt(pos2);
                delete pObj;
            }
        }
    }
    // 绘制分数
    CString strScore;
    strScore.Format(_T("%d"), m_score);
    // 加载数字图片
    for (int i = 0; i < strScore.GetLength(); i++)
```

```
{
    wchar_t filePath[50];
    swprintf(filePath, L"res//sz//%c.png", strScore[i]);
    Gdiplus::Bitmap sz(filePath);

    // 计算图片位置
    int x = 20 + i * 50;
    int y = 25;

    // 绘制数字图片
    Gdiplus::Graphics graphics(pMemDC->m_hDC);
    graphics.DrawImage(&sz, x, y, 40, 60);
}
drawBloodBar(10, 10, 200, 10, 2, BLUE, DARKGRAY, RED, playerBlood / 100.0);
CString levelDisplay = _T("当前关卡：");
CString strLevel;
strLevel.Format(_T("%d"), k);
pMemDC->SetBkMode(TRANSPARENT);
//pMemDC->SetTextAlign(TA_CENTER);
pMemDC->SetTextColor(RED);
pMemDC->TextOut(GAME_WIDTH - 140, 20, levelDisplay);
pMemDC->TextOut(GAME_WIDTH - 50, 20, strLevel);
int lastHighScore = m_highScore;
if (m_IsWin)
{
    if (m_score > lastHighScore)
    {
        std::ofstream recordFile("record.txt", std::ios::out);
        if (!recordFile)
        {
            CString str;
            str.Format(_T("存档文件损坏"));
            AfxMessageBox(str);
            return;
        }
        m_IsWin = false;
        m_highScore = m_score;
        recordFile << "最高分数  " << m_highScore << std::endl;
        recordFile.close();
    }
    else
    {
        if (m_score == lastHighScore)
        {
```

```
                    CString str;
                    str.Format(_T("您的分数：%d，恭喜您超越了记录获得历史游戏最高分"),
m_score);
                    pMemDC->SetBkMode(TRANSPARENT);
                    pMemDC->SetTextColor(WHITE);
                    pMemDC->TextOut(20, 550, str);
                }
                else
                {
                    CString str,str1;
                    str.Format(_T("您的分数：%d，很遗憾这一次没有创造历史"), m_score);
                    str1.Format(_T("距离历史游戏最高分%d 分仅差%d 分"),m_highScore,
m_highScore - m_score);
                    //AfxMessageBox(str);
                    pMemDC->SetBkMode(TRANSPARENT);
                    pMemDC->SetTextColor(WHITE);
                    //pMemDC->SetTextAlign(TA_CENTER);
                    pMemDC->TextOut(50, 550, str);
                    pMemDC->TextOut(100, 570, str1);
                    m_IsWin = false;
                }

            }

        }

        CString tipsStr1, tipsStr2, tipsStr3, tipsStr4;
        tipsStr1.Format(_T("移动：W A S D 或鼠标右键拖动"));
        tipsStr2.Format(_T("发射子弹：空格或鼠标左键"));
        tipsStr3.Format(_T("加速：H"));
        tipsStr4.Format(_T("复活：C"));
        // 创建并选择一个新的字体对象
        CFont font;
        font.CreatePointFont(100, _T("楷体")); // 12 是字体大小，"Arial"是字体名称
        CFont* pOldFont = pMemDC->SelectObject(&font);
        // 绘制文本
        pMemDC->SetBkMode(TRANSPARENT);
        pMemDC->SetTextColor(WHITE);
        pMemDC->TextOut(10, 705, tipsStr1);
        pMemDC->TextOut(10, 720, tipsStr2);
        pMemDC->TextOut(10, 735, tipsStr3);
        pMemDC->TextOut(10, 750, tipsStr4);

        // 恢复原始字体
```

```
    pMemDC->SelectObject(pOldFont);


    //复制内存 DC 到设备 DC
    m_pDC->BitBlt(0, 0, GAME_WIDTH, GAME_HEIGHT, m_pMemDC, 0, 0, SRCCOPY);

    //m_pDC->BitBlt(0, 0, GAME_WIDTH, GAME_HEIGHT, &cdc, 0, 0, SRCCOPY);
}

void CPlaneGameView::OnMouseMove(UINT nFlags, CPoint point)
{
    // 更新飞机位置为鼠标位置
    m_planePosition = point;

    // 调用父类的鼠标移动事件处理函数
    CView::OnMouseMove(nFlags, point);
}
void CPlaneGameView::backgroundMove()
{
    if (bgY >= 0)
    {
        bgY = -GAME_HEIGHT;
    }
    else
    {
        bgY += gameLevel->backgroundSpeed;
    }

}
void CPlaneGameView::enemyCreate()
{
    if (m_pMe == NULL)
    {
        return;
    }

    static int nCreator = rand() % 30 + 10;

    //随机产生普通敌机
    if (nCreator <= 0)
    {
        nCreator = rand() % 30 + 5;
        m_ObjList[enEnemy].AddTail(new CEnemy);
        //m_ObjList[enEnhanced].AddTail(new CEnemy2);
```

```
        enemyCnt1++;
        enemyCnt2++;
        enemyCnt3++;
        enemyCnt4++;
        enemyCnt5++;

if (enemyCnt1 > gameLevel->enhanced1Freq)
{
        m_ObjList[enEnhanced].AddTail(new CEnemy2);
        enemyCnt1 = 0;
}

if (enemyCnt2 > gameLevel->enhanced2Freq)
{
        m_ObjList[enEnhanced2].AddTail(new CEnemy3);
        enemyCnt2 = 0;
}

if (enemyCnt3 > gameLevel->enhanced3Freq)
{
        enemyCnt3 = 0;
        m_ObjList[enEnhanced3].AddTail(new CEnemy4);
}

if (enemyCnt4 > gameLevel->UFOFreq)
{
        enemyCnt4 = 0;
        m_ObjList[enUFO].AddTail(new CUFO);
}
if (enemyCnt5 > gameLevel->BOSSFreq&&m_score>100)
{
        enemyCnt5 = 0;
        enemyCnt4 = 0;
        enemyCnt3 = 0;
        enemyCnt2 = 0;
        enemyCnt1 = 0;

        isBOSSDie = false;
        m_ObjList[enBOSS].AddTail(new CBOSS);
        m_ObjList[enEnhanced].RemoveAll();
        m_ObjList[enEnemy].RemoveAll();
        m_ObjList[enEnhanced2].RemoveAll();
        m_ObjList[enEnhanced3].RemoveAll();
```

```
                //m_ObjList[enBall].RemoveAll();
                //m_ObjList[enMissle].RemoveAll();
                m_ObjList[enUFO].RemoveAll();
            }
        }
        nCreator--;
        if (m_pMe == NULL)
            return;
}
void CPlaneGameView::playerShoot()
{
        if (m_pMe == NULL)
        {
            return;
        }
        //产生战机导弹
        if (GetKey(VK_SPACE) == 1 || GetKeyState(VK_LBUTTON) & 0x8000)//按下了空格键
        {
            //cplaySound("plane\\shoot.mp3");
            if (m_pMe != NULL && m_pMe->Fired())
            {
                CPoint pt = m_pMe->GetPoint();
                m_ObjList[enBomb].AddTail(new CBomb(pt.x + 10, pt.y + 10));
                m_ObjList[enBomb].AddTail(new CBomb(pt.x + 30, pt.y + 10));
                if (k >= 2)
                {
                    m_ObjList[enBomb].AddTail(new CBomb(pt.x + 20, pt.y + 10));
                }
                if (k >= 3)
                {

                }
            }
        }
}
void CPlaneGameView::enemyShoot()
{
        if (m_pMe == NULL)
        {
            return;
        }
        //加强版敌机 1 发射子弹
        for (POSITION ePos = m_ObjList[enEnhanced].GetHeadPosition(); ePos != NULL;)
        {
```

```
            CEnemy2* pEnemy = (CEnemy2*)m_ObjList[enEnhanced].GetNext(ePos);
            if (!pEnemy->Fired())
                continue;
            CPoint    ePt = pEnemy->GetPoint();

            BOOL by = FALSE;

            m_ObjList[enBall].AddTail(new CBall(ePt.x + 55, ePt.y + 30, pEnemy->GetMontion(),
0, 2));

        }

        //加强版敌机 2 发射子弹
        for (POSITION ePos = m_ObjList[enEnhanced2].GetHeadPosition(); ePos != NULL;)
        {
            CEnemy3* pEnemy = (CEnemy3*)m_ObjList[enEnhanced2].GetNext(ePos);
            if (!pEnemy->Fired())
                continue;
            CPoint    ePt = pEnemy->GetPoint();
            BOOL by = FALSE;
            m_ObjList[enBall].AddTail(new CBall(ePt.x + 55, ePt.y + 30, pEnemy->GetMontion(),
0, 1));
            m_ObjList[enBall].AddTail(new CBall(ePt.x + 65, ePt.y + 30, pEnemy->GetMontion(),
-1, 1));
            m_ObjList[enBall].AddTail(new CBall(ePt.x + 45, ePt.y + 30, pEnemy->GetMontion(),
1, 1));
        }

        //加强版敌机 3 发射子弹
        for (POSITION ePos = m_ObjList[enEnhanced3].GetHeadPosition(); ePos != NULL;)
        {
            CEnemy4* pEnemy = (CEnemy4*)m_ObjList[enEnhanced3].GetNext(ePos);
            if (!pEnemy->Fired())
                continue;
            CPoint    ePt = pEnemy->GetPoint();
            BOOL by = FALSE;
            m_ObjList[enMissle].AddTail(new      CMissle(ePt.x      +      55,      ePt.y      +      30,
pEnemy->GetMontion()));
            m_ObjList[enMissle].AddTail(new      CMissle(ePt.x      +      55,      ePt.y      +      30,
pEnemy->GetMontion()));
        }
        //UFO 发射子弹
        if (m_pMe != NULL)
        {
```

```
        for (POSITION ePos = m_ObjList[enUFO].GetHeadPosition(); ePos != NULL;)
        {
            CUFO* pEnemy = (CUFO*)m_ObjList[enUFO].GetNext(ePos);
            if (!pEnemy->Fired())
                continue;
            CPoint    ePt = pEnemy->GetPoint();

            BOOL by = FALSE;

            for (int i = 0; i < UFO_BALL_NUM; i++)
            {
                double angle = 2 * PI * i / UFO_BALL_NUM;
                double r = 4 ;
                if (r == 0)continue;
                m_ObjList[enUFOBall].AddTail(new  UFOball(ePt.x  +  60,  ePt.y  +  70,
pEnemy->GetMontion(), r * sin(angle), r * cos(angle)));
            }
        }
    }

    //BOSS 发射子弹
    for (POSITION ePos = m_ObjList[enBOSS].GetHeadPosition(); ePos != NULL;)
    {
        CBOSS* pEnemy = (CBOSS*)m_ObjList[enBOSS].GetNext(ePos);
        if (!pEnemy->Fired())
            continue;
        CPoint    ePt = pEnemy->GetPoint();

        BOOL by = FALSE;
        if (playerBlood <= 0)
        {
            m_ObjList[enBOSS].RemoveAll();
        }
        //BOSS 发动技能
        int randInt = rand() % 3;
        if (randInt == 0)
        {
            m_ObjList[enMissle].AddTail(new  CMissle(ePt.x  +  120,  ePt.y  +  200,
pEnemy->GetMontion()));
            m_ObjList[enMissle].AddTail(new  CMissle(ePt.x  +  240,  ePt.y  +  200,
pEnemy->GetMontion()));
        }
        else if (randInt == 1)
        {
```

```
                for (int i = 0; i < 30; i++)
                {
                    double angle = 2 * PI * i / UFO_BALL_NUM;
                    double r = 4 * angle;
                    if (r == 0)
                    {
                        continue;
                    }
                    m_ObjList[enUFOBall].AddTail(new UFOball(ePt.x + 180, ePt.y + 200,
pEnemy->GetMontion(), r * sin(angle), r * cos(angle)));
                }
            }
            else
            {
                m_ObjList[enBall].AddTail(new CBall(ePt.x + 100, ePt.y + 200,
pEnemy->GetMontion(), -2, 1));
                m_ObjList[enBall].AddTail(new CBall(ePt.x + 180, ePt.y + 200,
pEnemy->GetMontion(), 0, 1));
                m_ObjList[enBall].AddTail(new CBall(ePt.x + 260, ePt.y + 200,
pEnemy->GetMontion(), 2, 1));
                m_ObjList[enBall].AddTail(new CBall(ePt.x + 120, ePt.y + 250,
pEnemy->GetMontion(), -1, 2));
                m_ObjList[enBall].AddTail(new CBall(ePt.x + 180, ePt.y + 250,
pEnemy->GetMontion(), 0, 2));
                m_ObjList[enBall].AddTail(new CBall(ePt.x + 240, ePt.y + 250,
pEnemy->GetMontion(), 1, 2));
                m_ObjList[enBall].AddTail(new CBall(ePt.x + 180, ePt.y + 250,
pEnemy->GetMontion(), 0, 0));
                m_ObjList[enBall].AddTail(new CBall(ePt.x + 180, ePt.y + 250,
pEnemy->GetMontion(), 0, 0));
                m_ObjList[enBall].AddTail(new CBall(ePt.x + 180, ePt.y + 250,
pEnemy->GetMontion(), 0, 0));

            }

        }
        //追踪导弹与战机碰撞
        if (m_pMe != NULL)
        {
            POSITION bPos1 = NULL, bPos2 = NULL;
            CRect mRect = m_pMe->GetRect();
            for (bPos1 = m_ObjList[enMissle].GetHeadPosition(); (bPos2 = bPos1) != NULL;)
            {
                CMissle* pCEnemy = (CMissle*)m_ObjList[enMissle].GetNext(bPos1);
```

```
CRect bRect = pCEnemy->GetRect();
CRect tmpRect;

if (m_pMe->getX() < pCEnemy->GetPoint().x)
{
    pCEnemy->setChargeDirection(-5);
}
else if (m_pMe->getX() > pCEnemy->GetPoint().x && m_pMe->getX() < pCEnemy->GetPoint().x)
{
    pCEnemy->setChargeDirection(0);
}
else
{
    pCEnemy->setChargeDirection(5);
}

if (m_pMe->getY() <= pCEnemy->GetPoint().y - 100)
{
    pCEnemy->setDirectY(-1);
}
else if (m_pMe->getY() > pCEnemy->GetPoint().y)
{
    pCEnemy->setDirectY(1);
}

if (tmpRect.IntersectRect(&bRect, mRect))
{

    //删除子弹
    m_ObjList[enMissle].RemoveAt(bPos2);
    delete pCEnemy;

    //添加爆炸效果
    m_ObjList[enExplosion].AddTail(
        new CExplosion(mRect.left, mRect.top)
    );
    playSound("plane\\blast.mp3");
    playerBlood -= MISSLE_HARM;
    //删除战机
    if (playerBlood <= 0)
    {
        Remove();
        break;
```

```
                    }
                }
            }
        }
}
void CPlaneGameView::keyEvent()
{
    //检测四个方向键，移动战机
    if (m_pMe == NULL)
    {
        return;
    }
    for (int i = 0; i < 4; i++)
    {
        if (GetKey(0x48))
        {
            int nMeMotion = 0;
            m_pMe->SetVerMotion(0);
            m_pMe->SetHorMotion(0);

            nMeMotion = GetKey(0x57);//w
            if (nMeMotion == 1)
                m_pMe->SetVerMotion(2);

            nMeMotion = GetKey(0x53);
            if (nMeMotion == 1)
                m_pMe->SetVerMotion(-2);

            nMeMotion = GetKey(0x44);
            if (nMeMotion == 1)
                m_pMe->SetHorMotion(2);

            nMeMotion = GetKey(0x41);//a
            if (nMeMotion == 1)
                m_pMe->SetHorMotion(-2);
        }
        else
        {
            int nMeMotion = 0;
            m_pMe->SetVerMotion(0);
            m_pMe->SetHorMotion(0);

            nMeMotion = GetKey(0x57);
            if (nMeMotion == 1)
```

```
                m_pMe->SetVerMotion(1);

            nMeMotion = GetKey(0x53);
            if (nMeMotion == 1)
                m_pMe->SetVerMotion(-1);

            nMeMotion = GetKey(0x44);
            if (nMeMotion == 1)
                m_pMe->SetHorMotion(1);

            nMeMotion = GetKey(0x41);
            if (nMeMotion == 1)
                m_pMe->SetHorMotion(-1);
        }

    }
    if (GetKey(VK_RBUTTON))
        m_pMe->SetPosition(m_planePosition.x, m_planePosition.y);
}
void CPlaneGameView::planeCollideCheck()
{
    if (m_pMe == NULL)
    {
        return;
    }
    //敌机与战机碰撞
    POSITION bPos1 = NULL, bPos2 = NULL;
    CRect mRect = m_pMe->GetRect();
    for (bPos1 = m_ObjList[enEnemy].GetHeadPosition(); (bPos2 = bPos1) != NULL;)
    {
        CEnemy* pCEnemy = (CEnemy*)m_ObjList[enEnemy].GetNext(bPos1);
        CRect bRect = pCEnemy->GetRect();
        CRect tmpRect;

        if (tmpRect.IntersectRect(&bRect, mRect))
        {

            //删除子弹
            m_ObjList[enEnemy].RemoveAt(bPos2);
            delete pCEnemy;

            //添加爆炸效果
            m_ObjList[enExplosion].AddTail(
                new CExplosion(mRect.left, mRect.top)
```

```
                );
                playSound("plane\\blast.mp3");


                playerBlood -= gameLevel->CollideHarm;
                //删除战机
                if (playerBlood <= 0)
                {
                    Remove();
                    break;
                }
            }
        }


        //加强敌机 1 与战机碰撞
        for (bPos1 = m_ObjList[enEnhanced].GetHeadPosition(); (bPos2 = bPos1) != NULL;)
        {
            CEnemy2* pCEnemy = (CEnemy2*)m_ObjList[enEnhanced].GetNext(bPos1);
            CRect bRect = pCEnemy->GetRect();
            CRect tmpRect;

            if (m_pMe->getX() < pCEnemy->GetPoint().x)
            {
                pCEnemy->m_chargeDirection = -1;
            }
            else if (m_pMe->getX() > pCEnemy->GetPoint().x && m_pMe->getX() <
pCEnemy->GetPoint().x)
            {
                pCEnemy->m_chargeDirection = 0;
            }
            else
            {
                pCEnemy->m_chargeDirection = 2;
            }

            if (tmpRect.IntersectRect(&bRect, mRect))
            {

                //删除子弹
                m_ObjList[enEnhanced].RemoveAt(bPos2);
                delete pCEnemy;

                //添加爆炸效果
```

```
            m_ObjList[enExplosion].AddTail(
                new CExplosion(mRect.left, mRect.top)
            );
            playSound("plane\\blast.mp3");


            playerBlood -= gameLevel->CollideHarm;
            //删除战机
            if (playerBlood <= 0)
            {
                Remove();
                break;
            }
        }
    }

    //加强敌机 2 与战机碰撞
    for (bPos1 = m_ObjList[enEnhanced2].GetHeadPosition(); (bPos2 = bPos1) != NULL;)
    {
        CEnemy3* pCEnemy = (CEnemy3*)m_ObjList[enEnhanced2].GetNext(bPos1);
        CRect bRect = pCEnemy->GetRect();
        CRect tmpRect;

        if (tmpRect.IntersectRect(&bRect, mRect))
        {

            //删除子弹
            m_ObjList[enEnhanced2].RemoveAt(bPos2);
            delete pCEnemy;

            //添加爆炸效果
            m_ObjList[enExplosion].AddTail(
                new CExplosion(mRect.left, mRect.top)
            );
            playSound("plane\\blast.mp3");

            playerBlood -= gameLevel->CollideHarm;
            //删除战机
            if (playerBlood <= 0)
            {
                Remove();
                break;
            }
        }
```

```
}

//加强敌机 3 与战机碰撞
for (bPos1 = m_ObjList[enEnhanced3].GetHeadPosition(); (bPos2 = bPos1) != NULL;)
{
	CEnemy4* pCEnemy = (CEnemy4*)m_ObjList[enEnhanced3].GetNext(bPos1);
	CRect bRect = pCEnemy->GetRect();
	CRect tmpRect;

	if (tmpRect.IntersectRect(&bRect, mRect))
	{

		//删除子弹
		m_ObjList[enEnhanced3].RemoveAt(bPos2);
		delete pCEnemy;

		//添加爆炸效果
		m_ObjList[enExplosion].AddTail(
			new CExplosion(mRect.left, mRect.top)
		);
		playSound("plane\\blast.mp3");


		playerBlood -= gameLevel->CollideHarm;
		//删除战机
		if (playerBlood <= 0)
		{
			Remove();
			break;
		}
	}
}
//UFO 与战机碰撞
for (bPos1 = m_ObjList[enUFO].GetHeadPosition(); (bPos2 = bPos1) != NULL;)
{
	CUFO* pCEnemy = (CUFO*)m_ObjList[enUFO].GetNext(bPos1);
	CRect bRect = pCEnemy->GetRect();
	CRect tmpRect;

	if (tmpRect.IntersectRect(&bRect, mRect))
	{

		//删除子弹
		m_ObjList[enUFO].RemoveAt(bPos2);
```

```
            delete pCEnemy;

            //添加爆炸效果
            m_ObjList[enExplosion].AddTail(
                new CExplosion(mRect.left, mRect.top)
            );
            playSound("plane\\blast.mp3");


            playerBlood -= gameLevel->CollideHarm;
            //删除战机
            if (playerBlood <= 0)
            {
                Remove();
                break;
            }
        }
    }
    for (bPos1 = m_ObjList[enEnemy].GetHeadPosition(); (bPos2 = bPos1) != NULL;)
    {
        CBOSS* pCEnemy = (CBOSS*)m_ObjList[enBOSS].GetNext(bPos1);
        CRect bRect = pCEnemy->GetRect();
        CRect tmpRect;

        if (tmpRect.IntersectRect(&bRect, mRect))
        {
            playerBlood -= 80;
            pCEnemy->enhancedEnemyBlood -= 10;
            if (pCEnemy->enhancedEnemyBlood <= 0)
            {
                //删除子弹
                m_ObjList[enBOSS].RemoveAt(bPos2);
                delete pCEnemy;

                //添加爆炸效果
                m_ObjList[enExplosion].AddTail(
                    new CExplosion(mRect.left, mRect.top)
                );
                playSound("plane\\blast.mp3");
            }
            playerBlood -= gameLevel->CollideHarm;
            //删除战机
            if (playerBlood <= 0)
            {
```

```
                    Remove();
                    break;
                }
            }
        }
    }
    void CPlaneGameView::bulletCollideCheck()
    {
        if (m_pMe == NULL)
        {
            return;
        }
        //敌机子弹攻击战机
        POSITION bPos1 = NULL, bPos2 = NULL;
        CRect mRect = m_pMe->GetRect();
        for (bPos1 = m_ObjList[enBall].GetHeadPosition(); (bPos2 = bPos1) != NULL;)
        {
            CBall* pBall = (CBall*)m_ObjList[enBall].GetNext(bPos1);
            CRect bRect = pBall->GetRect();
            CRect tmpRect;
            if (tmpRect.IntersectRect(&bRect, mRect))
            {
                //删除子弹
                m_ObjList[enBall].RemoveAt(bPos2);
                delete pBall;

                //添加爆炸效果
                m_ObjList[enExplosion].AddTail(
                    new CExplosion(mRect.left, mRect.top)
                );
                playSound("plane\\blast.mp3");

                playerBlood -= gameLevel->normalEnemyHarm;
                //删除战机
                if (playerBlood <= 0)
                {
                    Remove();
                    break;
                }
            }
        }
        //战机导弹炸掉敌机
        POSITION mPos1 = NULL, mPos2 = NULL;
        for (mPos1 = m_ObjList[enBomb].GetHeadPosition(); (mPos2 = mPos1) != NULL;)
```

```
{
    CBomb* pBomb = (CBomb*)m_ObjList[enBomb].GetNext(mPos1);
    CRect bRect = pBomb->GetRect();

    POSITION ePos1 = NULL, ePos2 = NULL;
    for (ePos1 = m_ObjList[enEnemy].GetHeadPosition(); (ePos2 = ePos1) != NULL;)
    {
        CEnemy* pEnemy = (CEnemy*)m_ObjList[enEnemy].GetNext(ePos1);
        CRect mRect = pEnemy->GetRect();
        CRect tmpRect;
        if (tmpRect.IntersectRect(&bRect, mRect))
        {
            //添加爆炸效果
            m_ObjList[enExplosion].AddTail(
                new CExplosion(mRect.left, mRect.top)
            );
            playSound("plane\\blast.mp3");
            //删除导弹
            m_ObjList[enBomb].RemoveAt(mPos2);
            delete pBomb;
            //删除敌机
            m_ObjList[enEnemy].RemoveAt(ePos2);
            delete pEnemy;
            m_score++;
            break;

        }
    }
}

//战机导弹炸掉加强敌机 1
for (mPos1 = m_ObjList[enBomb].GetHeadPosition(); (mPos2 = mPos1) != NULL;)
{
    CBomb* pBomb = (CBomb*)m_ObjList[enBomb].GetNext(mPos1);
    CRect bRect = pBomb->GetRect();

    POSITION ePos1 = NULL, ePos2 = NULL;
    for (ePos1 = m_ObjList[enEnhanced].GetHeadPosition(); (ePos2 = ePos1) != NULL;)
    {
        CEnemy2* pEnemy = (CEnemy2*)m_ObjList[enEnhanced].GetNext(ePos1);
        CRect mRect = pEnemy->GetRect(1);
        CRect tmpRect;
        if (tmpRect.IntersectRect(&bRect, mRect))
        {
```

```
                    //删除导弹
                    m_ObjList[enBomb].RemoveAt(mPos2);
                    delete pBomb;

                    //添加爆炸效果
                    m_ObjList[enExplosion].AddTail(
                        new CExplosion(mRect.left, mRect.top)
                    );

                    pEnemy->enhancedEnemyBlood -= 1;

                    //删除敌机
                    if (pEnemy->enhancedEnemyBlood <= 0)
                    {
                        int x = pEnemy->GetPoint().x;
                        int y = pEnemy->GetPoint().y;

                        m_ObjList[enEnhanced].RemoveAt(ePos2);
                        delete pEnemy;
                        m_score += 10;

                        //添加爆炸效果
                        m_ObjList[enExplosion].AddTail(
                            new CExplosion(mRect.left, mRect.top)
                        );
                        m_ObjList[enBlood].AddTail(new CBlood(GAME_WIDTH,
GAME_HEIGHT,x,y));

                        playSound("plane\\blast.mp3");
                    }
                    break;

                }
            }
        }


        //战机导弹炸掉加强敌机 2
        for (mPos1 = m_ObjList[enBomb].GetHeadPosition(); (mPos2 = mPos1) != NULL;)
        {
            CBomb* pBomb = (CBomb*)m_ObjList[enBomb].GetNext(mPos1);
            CRect bRect = pBomb->GetRect();

            POSITION ePos1 = NULL, ePos2 = NULL;
```

```
        for (ePos1 = m_ObjList[enEnhanced2].GetHeadPosition(); (ePos2 = ePos1) != NULL;)
        {
            CEnemy3* pEnemy = (CEnemy3*)m_ObjList[enEnhanced2].GetNext(ePos1);
            CRect mRect = pEnemy->GetRect(1);
            CRect tmpRect;
            if (tmpRect.IntersectRect(&bRect, mRect))
            {
                //删除导弹
                m_ObjList[enBomb].RemoveAt(mPos2);
                delete pBomb;

                //添加爆炸效果
                m_ObjList[enExplosion].AddTail(
                    new CExplosion(mRect.left, mRect.top)
                );

                pEnemy->enhancedEnemyBlood -= 1;

                //删除敌机
                if (pEnemy->enhancedEnemyBlood <= 0)
                {
                    int x = pEnemy->GetPoint().x;
                    int y = pEnemy->GetPoint().y;

                    m_ObjList[enEnhanced2].RemoveAt(ePos2);
                    delete pEnemy;
                    m_score += 10;

                    //添加爆炸效果
                    m_ObjList[enExplosion].AddTail(
                        new CExplosion(mRect.left, mRect.top)
                    );
                    m_ObjList[enBlood].AddTail(new CBlood(GAME_WIDTH,
GAME_HEIGHT,x,y));

                    playSound("plane\\blast.mp3");

                }
                break;

            }
        }
    }
    //战机导弹炸掉加强敌机 3
```

```
for (mPos1 = m_ObjList[enBomb].GetHeadPosition(); (mPos2 = mPos1) != NULL;)
{
    CBomb* pBomb = (CBomb*)m_ObjList[enBomb].GetNext(mPos1);
    CRect bRect = pBomb->GetRect();

    POSITION ePos1 = NULL, ePos2 = NULL;
    for (ePos1 = m_ObjList[enEnhanced3].GetHeadPosition(); (ePos2 = ePos1) != NULL;)
    {
        CEnemy4* pEnemy = (CEnemy4*)m_ObjList[enEnhanced2].GetNext(ePos1);
        CRect mRect = pEnemy->GetRect(1);
        CRect tmpRect;
        if (tmpRect.IntersectRect(&bRect, mRect))
        {
            //删除导弹
            m_ObjList[enBomb].RemoveAt(mPos2);
            delete pBomb;

            //添加爆炸效果
            m_ObjList[enExplosion].AddTail(
                new CExplosion(mRect.left, mRect.top)
            );

            pEnemy->enhancedEnemyBlood -= 1;

            //删除敌机
            if (pEnemy->enhancedEnemyBlood <= 0)
            {
                int x = pEnemy->GetPoint().x;
                int y = pEnemy->GetPoint().y;

                m_ObjList[enEnhanced3].RemoveAt(ePos2);
                delete pEnemy;
                m_score += 10;

                //添加爆炸效果
                m_ObjList[enExplosion].AddTail(
                    new CExplosion(mRect.left, mRect.top)
                );
                m_ObjList[enBlood].AddTail(new CBlood(GAME_WIDTH,
GAME_HEIGHT,x,y));

                playSound("plane\\blast.mp3");

            }
```

```
                    break;

                }
            }
        }
        //战机导弹炸掉 UFO
        for (mPos1 = m_ObjList[enBomb].GetHeadPosition(); (mPos2 = mPos1) != NULL;)
        {
            CBomb* pBomb = (CBomb*)m_ObjList[enBomb].GetNext(mPos1);
            CRect bRect = pBomb->GetRect();

            POSITION ePos1 = NULL, ePos2 = NULL;
            for (ePos1 = m_ObjList[enUFO].GetHeadPosition(); (ePos2 = ePos1) != NULL;)
            {
                CUFO* pEnemy = (CUFO*)m_ObjList[enUFO].GetNext(ePos1);
                CRect mRect = pEnemy->GetRect();
                CRect tmpRect;
                if (tmpRect.IntersectRect(&bRect, mRect))
                {
                    //删除导弹
                    m_ObjList[enBomb].RemoveAt(mPos2);
                    delete pBomb;

                    //添加爆炸效果
                    m_ObjList[enExplosion].AddTail(
                        new CExplosion(mRect.left, mRect.top)
                    );
                    pEnemy->enhancedEnemyBlood -= 1;
                    //删除敌机
                    if (pEnemy->enhancedEnemyBlood <= 0)
                    {
                        int x = pEnemy->GetPoint().x;
                        int y = pEnemy->GetPoint().y;
                        m_ObjList[enUFO].RemoveAt(ePos2);
                        delete pEnemy;
                        m_score += 10;
                        //添加爆炸效果
                        m_ObjList[enExplosion].AddTail(
                            new CExplosion(mRect.left, mRect.top)
                        );
                        m_ObjList[enBlood].AddTail(new CBlood(GAME_WIDTH,
GAME_HEIGHT,x,y));
                        playSound("plane\\blast.mp3");
                    }
```

```
                    break;

                }
            }
        }
        //战机导弹炸掉 BOSS
        for (mPos1 = m_ObjList[enBomb].GetHeadPosition(); (mPos2 = mPos1) != NULL;)
        {
            CBomb* pBomb = (CBomb*)m_ObjList[enBomb].GetNext(mPos1);
            CRect bRect = pBomb->GetRect();

            POSITION ePos1 = NULL, ePos2 = NULL;
            for (ePos1 = m_ObjList[enBOSS].GetHeadPosition(); (ePos2 = ePos1) != NULL;)
            {
                CBOSS* pEnemy = (CBOSS*)m_ObjList[enBOSS].GetNext(ePos1);
                CRect mRect = pEnemy->GetRect(BOSS_WIDTH,BOSS_HEIGHT);
                CRect tmpRect;

                drawBloodBar(pEnemy->GetPoint().x+180,  pEnemy->GetPoint().y,  200,  10,  2,
BLUE, DARKGRAY, GREEN, pEnemy->enhancedEnemyBlood / 200.0);

                if (tmpRect.IntersectRect(&bRect, mRect))
                {
                    pEnemy->enhancedEnemyBlood -= 1;
                    //添加爆炸效果

                    m_ObjList[enExplosion].AddTail(
                        new CExplosion(mRect.left, mRect.top)
                    );
                    playSound("plane\\blast.mp3");

                    //删除导弹
                    m_ObjList[enBomb].RemoveAt(mPos2);
                    delete pBomb;

                    //删除敌机
                    if (pEnemy->enhancedEnemyBlood <= 0)
                    {
                        int x = pEnemy->GetPoint().x;
                        int y = pEnemy->GetPoint().y;
                        m_ObjList[enBOSS].RemoveAt(ePos2);
                        delete pEnemy;
                        m_score += 10;
```

```
                        //添加爆炸效果
                        m_ObjList[enExplosion].AddTail(
                            new CExplosion(mRect.left, mRect.top)
                        );
                        m_ObjList[enBlood].AddTail(new CBlood(GAME_WIDTH,
GAME_HEIGHT, x, y));
                        m_ObjList[enBlood].AddTail(new CBlood(GAME_WIDTH,
GAME_HEIGHT, x, y));
                        m_ObjList[enBlood].AddTail(new CBlood(GAME_WIDTH,
GAME_HEIGHT, x, y));

                        playSound("plane\\blast.mp3");

                        isBOSSDie = true;
                        isBOSSExist = false;
                    }
                    break;

                }
            }
        }
        //UFO 子弹攻击战机
        for (bPos1 = m_ObjList[enUFOBall].GetHeadPosition(); (bPos2 = bPos1) != NULL;)
        {
            UFOball* pBall = (UFOball*)m_ObjList[enUFOBall].GetNext(bPos1);
            CRect bRect = pBall->GetRect();
            CRect tmpRect;
            if (tmpRect.IntersectRect(&bRect, mRect))
            {
                //删除子弹
                m_ObjList[enUFOBall].RemoveAt(bPos2);
                delete pBall;

                //添加爆炸效果
                m_ObjList[enExplosion].AddTail(
                    new CExplosion(mRect.left, mRect.top)
                );
                playSound("plane\\blast.mp3");

                playerBlood -= gameLevel->ufoBallHarm;
                //删除战机
                if (playerBlood <= 0)
                {
                    // 清空敌机列表
```

```
                    m_ObjList[enEnhanced].RemoveAll();
                    m_ObjList[enEnemy].RemoveAll();
                    m_ObjList[enBOSS].RemoveAll();
                    isExist = FALSE;
                    delete m_pMe;
                    m_pMe = NULL;
                    mciSendString("stop plane/3.mp3", 0, 0, 0);//关闭背景音乐
                    break;
                }
            }
        }

}
void CPlaneGameView::propCollideCheck()
{
    if (m_pMe == NULL)
    {
        return;
    }
    //血包被吃掉
    POSITION bPos1 = NULL, bPos2 = NULL;
    CRect mRect = m_pMe->GetRect();
    for (bPos1 = m_ObjList[enBlood].GetHeadPosition(); (bPos2 = bPos1) != NULL;)
    {
        CBlood* pCEnemy = (CBlood*)m_ObjList[enBlood].GetNext(bPos1);
        CRect bRect = pCEnemy->GetRect();
        CRect tmpRect;

        if (tmpRect.IntersectRect(&bRect, mRect))
        {
            m_ObjList[enBlood].RemoveAt(bPos2);
            delete pCEnemy;
            //吃到血包回复血量
            if (playerBlood >= 80)
            {
                playerBlood = 100;
            }
            else
            {
                playerBlood += 20;
            }
        }
    }
}
```

```
void CPlaneGameView::debug()
{
	if (GetKey(0x58))//调试模式
	{
		isDebugMode = true;
	}
	if (GetKey(0x5A))//退出锁血调试模式
	{
		//lastScore = m_score;
		isDebugMode = false;
	}
	if (isDebugMode)
	{
		playerBlood = 100;
	}
	if (GetKey(0x42))
	{
		m_score += 100;
	}
}
void CPlaneGameView::OnTimer(UINT_PTR nIDEvent)
{
	UpdateFrame(m_pMemDC);//刷新游戏帧画面: 在内存 DC 上绘图
	backgroundMove();//背景图片滚动
	enemyCreate();//创造敌机
	playerShoot();//玩家发射子弹
	enemyShoot();//敌机发射子弹
	keyEvent();//玩家按键检测
	planeCollideCheck();//飞机与敌机碰撞检测
	bulletCollideCheck();//子弹与飞机碰撞检测
	propCollideCheck();//战机与道具碰撞检测
	debug();//进入调试模式
	CView::OnTimer(nIDEvent);
```

5. 实训中遇到的主要问题及解决方法

碰撞检测问题：敌人和玩家子弹之间的碰撞检测不准确，导致不正确的击中判定。解决方法：检查碰撞检测的逻辑和算法，确保正确判断碰撞。

图片加载问题：无法加载游戏资源中的图片，导致游戏对象无法正确显示。解决方法：检查资源文件路径是否正确，确保资源文件存在于指定的位置。同时，确保资源文件的格式与加载代码相匹配，并尝试重新加载图片资源。

输入响应问题：玩家操作飞机时，无法正确响应输入指令。解决方法：确保输入事件被正确监听和处理。检查键盘或鼠标事件的注册和绑定，以及相应的回调函数是否正确执行。如果使用了第三方输入库，可以查阅其文档和示例代码，确保正确使用和配置。

内存泄漏问题：长时间运行游戏后，内存占用逐渐增加，导致内存泄漏。解决方法：检查游戏中的对象创建和销毁过程，确保没有未释放的资源。

6. 实训分工及体会

本次项项目由我独立完成。实训提供了一个将课堂所学理论知识应用到实际项目中的机会。通过实际操作和实际问题解决，加深了我对理论知识的理解和掌握。在实训过程中，学我遇到了各种问题和挑战，例如调试 bug、优化代码、解决技术难题等。通过面对这些问题并寻找解决方法，我的问题解决能力和创新能力得到了锻炼和提高。实训有一定的时间限制，我需要在规定的时间内完成项目。因此，我需要合理安排时间，高效利用时间进行任务分配、进度安排等，培养了我的时间管理能力。通过实训，我积累了宝贵的项目实践经验，了解实际工作中遇到的问题和挑战，这样的经验对于我未来的就业或者进一步学习都具有重要的意义。总之，这次实训对我来说非常有意义。