**《Python程序设计》**

**课程设计报告**

　　　　　　　　　　（2018 —2019 学年第 一 学期）

题 目 弹球小游戏

学生姓名 冯 孝 勇

专业班级 信管16-3

学生学号 311609030313

教师姓名 徐 文 鹏

成 绩**：**

评 语：

教师签名：

日期：

目录

[一、设计目的 2](#_Toc535314676)

[二、设计任务和要求 2](#_Toc535314677)

[三、总体设计 3](#_Toc535314678)

[1.任务分配情况： 3](#_Toc535314679)

[2.程序流程图： 4](#_Toc535314680)

[3.功能模块图 4](#_Toc535314681)

[四、设计实现 5](#_Toc535314682)

[1.最终实现结果： 5](#_Toc535314683)

[2.实现结果评价： 7](#_Toc535314684)

[五、详细设计 7](#_Toc535314685)

[1.输入输出分析 7](#_Toc535314686)

[2.程序流程图 8](#_Toc535314687)

[3.算法的说明 8](#_Toc535314688)

[六、调试与测试 13](#_Toc535314689)

[1. 调试方法 13](#_Toc535314690)

[七、设计总结 15](#_Toc535314691)

# 一、设计目的

课程设计的目的：本课程设计是本专业的一门重要实践性教学环节。在学习了专业基础课和《Python程序设计》课程的基础上，本课程设计旨在加深对Python程序设计的认识，对Python语言及其语言生态有一个进一步的掌握和应用，学会运用Python标准库及外接相关库来解决实际问题的基本能力，培养和提高学生分析问题、解决问题的能力，尤其是提高学生使用Python为开发语言来进行问题描述、交流与思考的能力，为毕业设计和以后的工程实践打下良好的基础。

选择“弹球小游戏”课设的目的：弹球小游戏是一款很有趣味的小游戏，其设计过程有一定的技巧，它既可以加深对python程序设计及其语言生态的认识和进一步掌握，也可以锻炼思维的敏捷性、缜密性和操作的灵活性。其中主要运用到pygame游戏开发工具，pygame是python计算生态里一个开发游戏很好的模块，本次课设通过使用pygame库，致力于设计出一个美观、用户体验良好的“弹球小游戏”。

# 二、设计任务和要求

课程设计的基本要求是：在课程设计的各个阶段严格、规范地完成相关的文档，例如在初期按时完成设计目的、设计要求、总体设计，在后期完成详细设计、调试与测试、设计总结等。要求能完成所选题目的主要功能，程序运行健壮、正确，代码要有详细注释，可读性好；所写文档结构合理、内容完整、叙述清晰。更高要求是：有创意、系统界面美观。

本课程设计具体任务和要求：我们小组的想法是通过pygame库来设计一个美观、用户体验良好的弹球小游戏。在设计中，用户体验第一，要求美观大方，以用户体验为主要出发点，所以在这款游戏里我们选取了优美的图片、音效、以及场景。用户可以通过左右移动挡板反弹小球，小球碰到四周或挡板会改变运动方向，如果挡板没有接到小球，游戏结束，并且返回游戏所用时间，此时用户还可以选择是否重新开始游戏。

# 三、总体设计

## 1.任务分配情况：

“弹球小游戏”代码框架、程序调试和代码整理汇总由小组成员共同完成。

冯孝勇主要负责GameStart函数模块和窗口初始化的编写；

高若云主要负责GameResult函数模块和主函数调用的编写。

## 2.程序流程图：

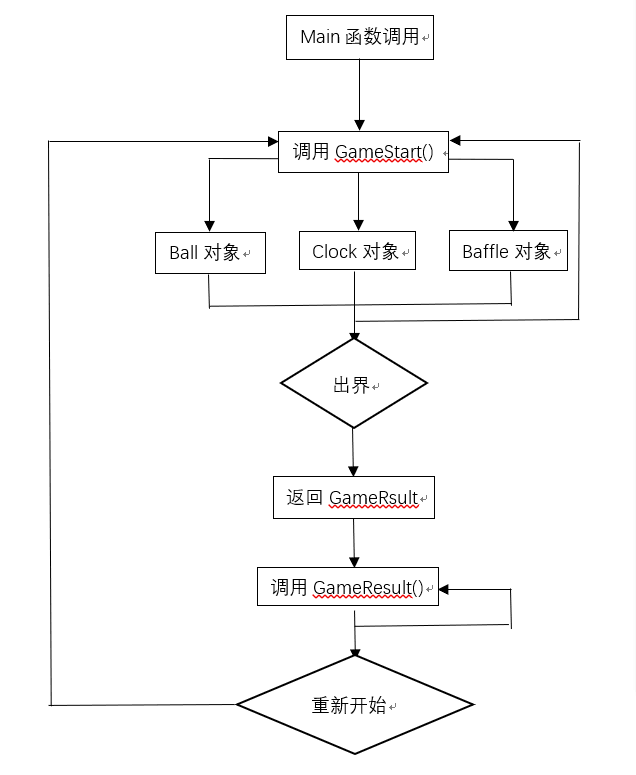


图3-1程序流程图

3.功能模块图：“弹球小游戏”主要包括背景初始化模块、游戏开始模块、游戏结果模块和主函数模块。

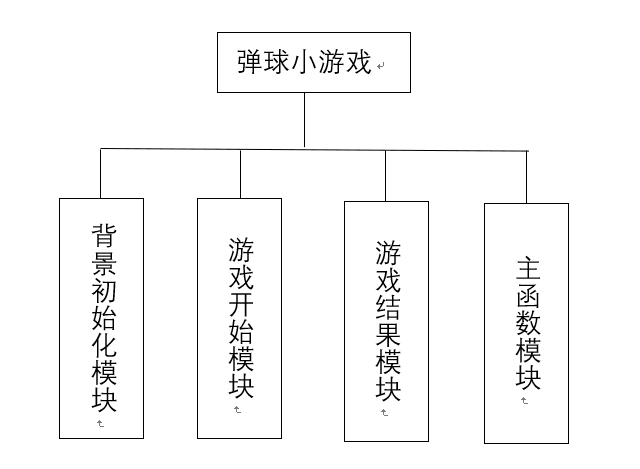


图3-2功能模块图

# 四、设计实现

## 1.最终实现结果：

（1）游戏的启动界面



图4-1启动界面

（2）主要功能运行界面

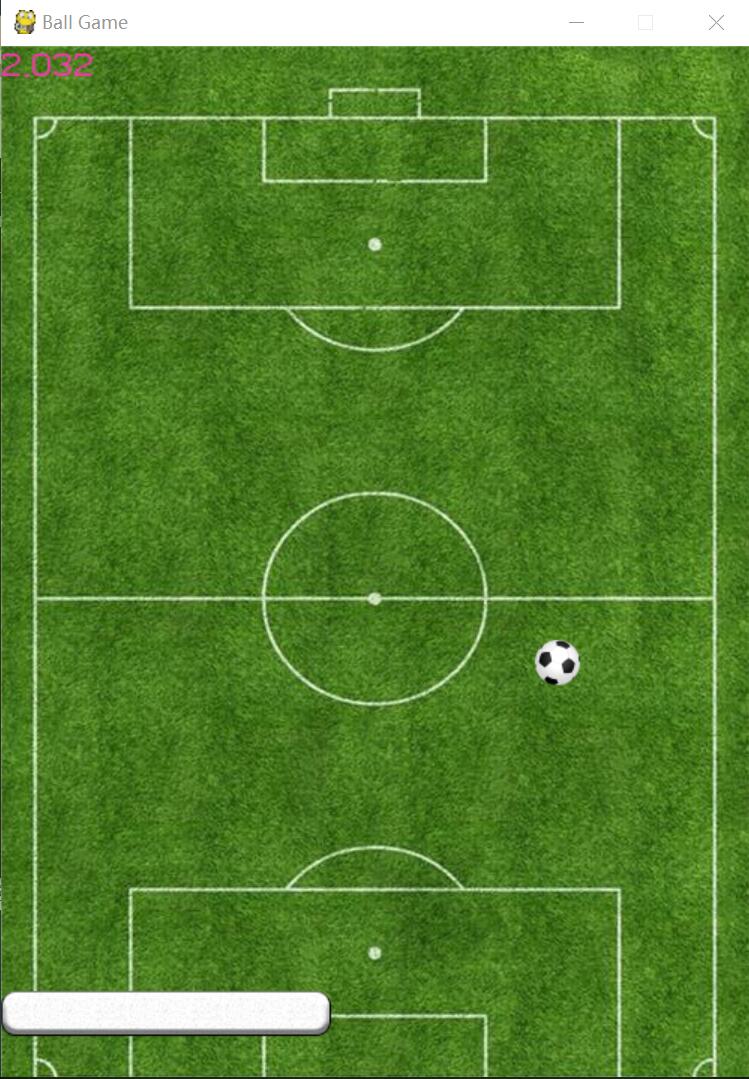


图4-2游戏运行界面

（3）游戏结束界面

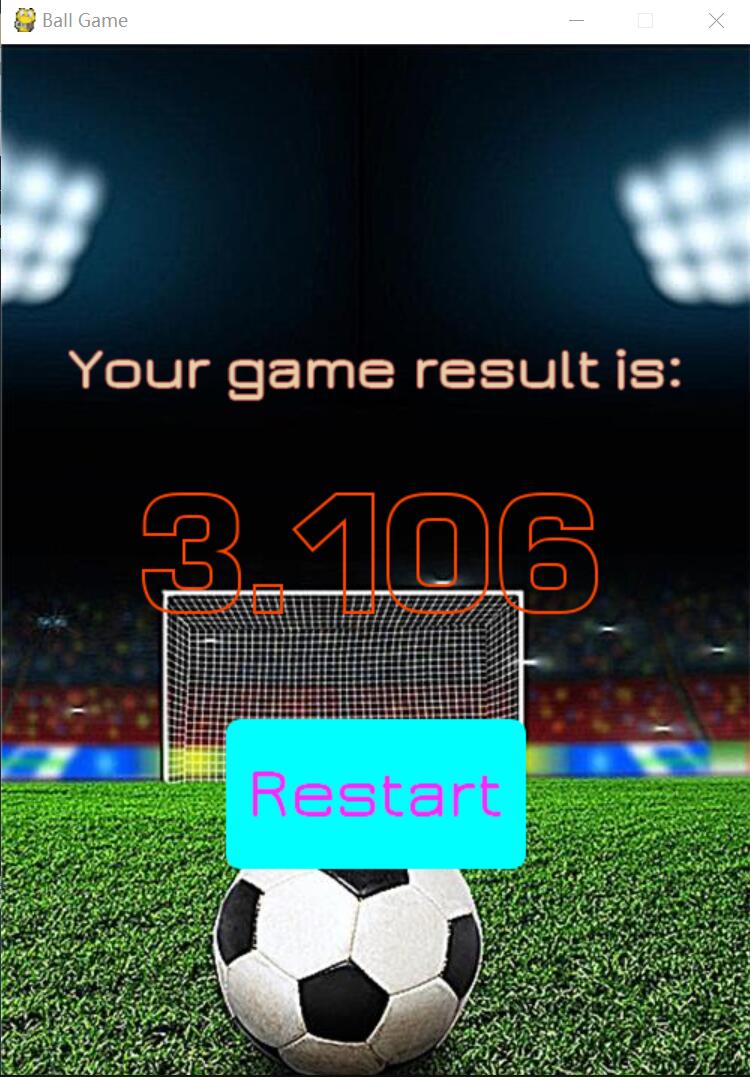


图4-3结束界面

## 2.实现结果评价：

通过设计结果与预期目标比较，我们小组基本完成了最初的想法：通过pygame库来设计一个美观、用户体验良好的弹球小游戏。这款游戏里我们选取了优美的图片、音效、以及场景。用户可以通过左右移动挡板反弹小球，小球碰到四周或挡板会改变运动方向，如果挡板没有接到小球，游戏结束，并且返回游戏所用时间，此时用户还可以选择是否重新开始游戏。

100%完成自己的任务，完成了预期的设想，利用pygame库设计设计了一个美观、用户体验良好的弹球小游戏。

# 五、详细设计

我负责的部分是游戏结果开始界面、界面背景选图、游戏场景设定、球的位置与挡板的位置判断、边界的判断、球与挡板初始速度的设置、球初始位置的设置、游戏初始化、GameStart函数，通过球与挡板的为位置来判断游戏是否结束，如果结束则循环结束函数结束并且返回时间字符串。

## 1.输入输出分析

初始开始界面输入：窗口的宽和高、以及背景图片的加载；输出：一个宽和高分别为500像素和720像素的窗口。

GameStart函数输入：小球图像的加载、挡板图片的加载、小球的初始位置、挡板的初始位置、小球的X方向和Y方向的速度、时间对象Clock的创建、挡板和小球位置的判断、球和边界的判断、挡板和边界的判断。

GameStart函数输入：捕捉一个时间对象，然后转换成字符串输出。

GameStart函数分析：通过改变小球的X方向和Y方向的速度可以改变小球的运动角度，即发球角度；通过时间对象可以时刻获得一个时间然后与速度的乘积得到小球改变的X方向和Y方向的变化位移，然后通过不断刷新界面实现小球不断运动的情景；小球与左边界碰撞改变X方向的速度为其相反数，小球与右边界碰撞改变X方向的速度为其相反数 ，小球与上边界碰撞改变Y方向的速度为其相反数，小球与下边界碰撞改变Y方向的速度为其相反数，如果小球的Y值大于了挡板的Y方向的Y值则GameStart函数结束。

## 2.程序流程图

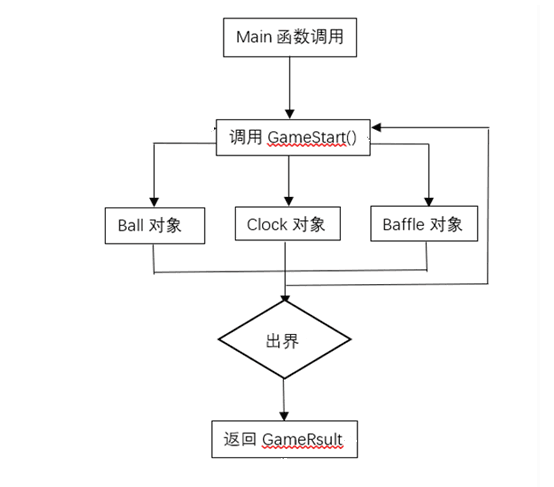


图5-1GameStart函数流程图

## 3.算法的说明

界面初始化：

pygame.init()

# 创建窗口

Screen = pygame.display.set\_mode((500,720), 0, 32)

pygame.display.set\_caption("Ball Game")

定义GameStart函数：

def GameStart():

# 游戏背景Surface对象

Background = pygame.image.load('GameBackground.jpg').convert()

# 挡板Surface对象

Baffle = pygame.image.load('Baffle.png').convert\_alpha()

# 球Surface对象

Ball = pygame.image.load('Ball.png').convert\_alpha()

# 挡板位置信息

BaffleX = 140

BaffleY = 600

BaffleSpeed = 1000

BaffleXSpeed = BaffleSpeed

BaffleYSpeed = BaffleSpeed

BaffleMove = {K\_LEFT: 0, K\_RIGHT: 0, K\_UP: 0, K\_DOWN: 0}

# 球位置信息

BallX = 235

BallY = 0

BallSpeed = 1000.

BallXSpeed = BallSpeed

BallYSpeed = BallSpeed

# 帧率控制Clock对象

FPSClock = pygame.time.Clock()

# 时间显示Clock对象

ProgramRunClock = pygame.time.get\_ticks()

# 时间显示Font对象

RunTimeFont = pygame.font.Font('Jura-DemiBold.ttf', 24)

# 游戏结果

GameResult = ''

while True:

# 接收信息处理

for event in pygame.event.get():

if event.type == QUIT:

exit()

if event.type == KEYDOWN:

if event.key in BaffleMove:

BaffleMove[event.key] = 1

elif event.type == KEYUP:

if event.key in BaffleMove:

BaffleMove[event.key] = 0

# 绘制背景

Screen.blit(Background, (0, 0))

RunTimeStr = str((pygame.time.get\_ticks() - ProgramRunClock) / 1000.0)

# print(RunTimeStr)

# 使用render方法显示时间字体

RunTimeSurface = RunTimeFont.render(RunTimeStr, True, (255, 52, 179))

# 显示时间

Screen.blit(RunTimeSurface, (0, 0))

# 距上次调用clock对象时间

SecondTimePassed = FPSClock.tick(60) / 1000.0

# 绘制球

Screen.blit(Ball, (BallX, BallY))

BallX += BallXSpeed \* SecondTimePassed

BallY += BallYSpeed \* SecondTimePassed

# 判断球边界条件

if BallX > 500 - Ball.get\_width():

BallXSpeed = -BallXSpeed

BallX = 500 - Ball.get\_width()

elif BallX < 0:

BallXSpeed = -BallXSpeed

BallX = 0

if BallY > 720 - Ball.get\_width():

BallYSpeed = -BallYSpeed

BallY = 720 - Ball.get\_width()

elif BallY < 0:

BallYSpeed = -BallYSpeed

BallY = 0

# 定位挡板移动后坐标

BaffleX -= BaffleMove[K\_LEFT] \* BaffleXSpeed \* SecondTimePassed

BaffleX += BaffleMove[K\_RIGHT] \* BaffleXSpeed \* SecondTimePassed

BaffleY -= BaffleMove[K\_UP] \* BaffleYSpeed \* SecondTimePassed

BaffleY += BaffleMove[K\_DOWN] \* BaffleYSpeed \* SecondTimePassed

# 判断挡板边界条件

if BaffleX > 500 - Baffle.get\_width():

BaffleX = 500 - Baffle.get\_width()

elif BaffleX < 0:

BaffleX = 0

if BaffleY > 720 - 45 - Baffle.get\_height():

BaffleY = 720 - 45 - Baffle.get\_height()

elif BaffleY < 720 - Baffle.get\_height() \* 3:

BaffleY = 720 - Baffle.get\_height() \* 3

# 绘制挡板

Screen.blit(Baffle, (BaffleX, BaffleY))

# 判断球碰撞挡板条件

# 挡板左上角

if BallX == BaffleX - Ball.get\_width() and BallY == BaffleY - Ball.get\_height():

BallXSpeed = -BallXSpeed

BallYSpeed = -BallYSpeed

# 挡板左下角

elif BallX == BaffleX - Ball.get\_width() and BallY == BaffleY + Baffle.get\_height():

BallXSpeed = -BallXSpeed

BallYSpeed = -BallYSpeed

elif BallX == BaffleX + Baffle.get\_width() and BallY == BaffleY - Ball.get\_height():

BallXSpeed = -BallXSpeed

BallYSpeed = -BallYSpeed

# 挡板右下角

elif BallX == BaffleX + Baffle.get\_width() and BallY == BaffleY + Baffle.get\_height():

BallXSpeed = -BallXSpeed

BallYSpeed = -BallYSpeed

# 挡板上表面

elif BallX > BaffleX and BallX < BaffleX + Baffle.get\_width() and BallY > BaffleY - Ball.get\_height() and BallY < BaffleY:

BallYSpeed = -BallYSpeed

BallY = BaffleY - Ball.get\_height()

# 挡板下表面

elif BallX > BaffleX and BallX <BaffleX+Baffle.get\_width() and BallY < BaffleY + Baffle.get\_height() and BallY > BaffleY:

BallYSpeed = -BallYSpeed

BallY = BaffleY + Baffle.get\_height()

# 挡板左侧面

elif BallY > BaffleY and BallY<BaffleY+Baffle.get\_height() and BallX > BaffleX - Ball.get\_width() and BallX < BaffleX:

BallXSpeed = -BallXSpeed

BallX = BaffleX

# 挡板右侧面

elif BallY > BaffleY and BallY<BaffleY+Baffle.get\_height() and BallX > BaffleX + Baffle.get\_width() - Ball.get\_width() and BallX < BaffleX + Baffle.get\_width():

BallXSpeed = -BallXSpeed

BallX = BaffleX + Baffle.get\_width()

if BallY > 720 - 45:

GameResult = RunTimeStr

return GameResult

# 刷新显示

pygame.display.update()

# 六、调试与测试

## 1. 调试方法

初始化窗口调试阶段：通过调整窗口的宽和高来设置一个比较美观的初始化窗口。

GameStart函数调整阶段：不断调整球的位置、球的大小、球的初始速度，包括X和Y方向的速度、球与边界的碰撞判断、碰撞之后球速度发生的改变、挡板的初始位置、挡板的移动速度、挡板与边界的判断。

2.测试结果

初始化界面设计：

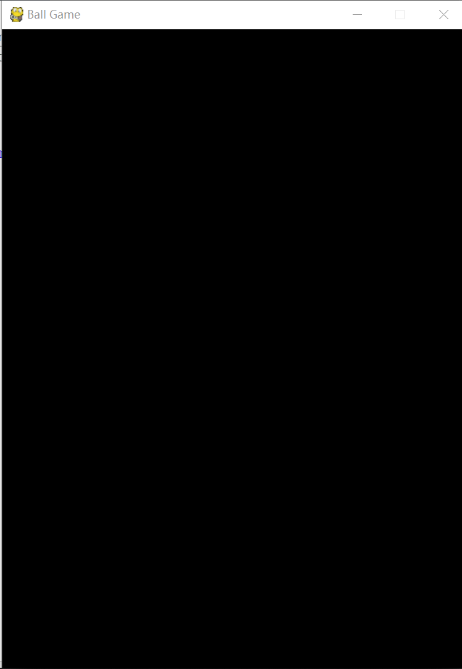


图6-1初始化界面设计

import pygame

pygame.init()

# 创建窗口

Screen = pygame.display.set\_mode((500,720), 0, 32)

pygame.display.set\_caption("Ball Game")

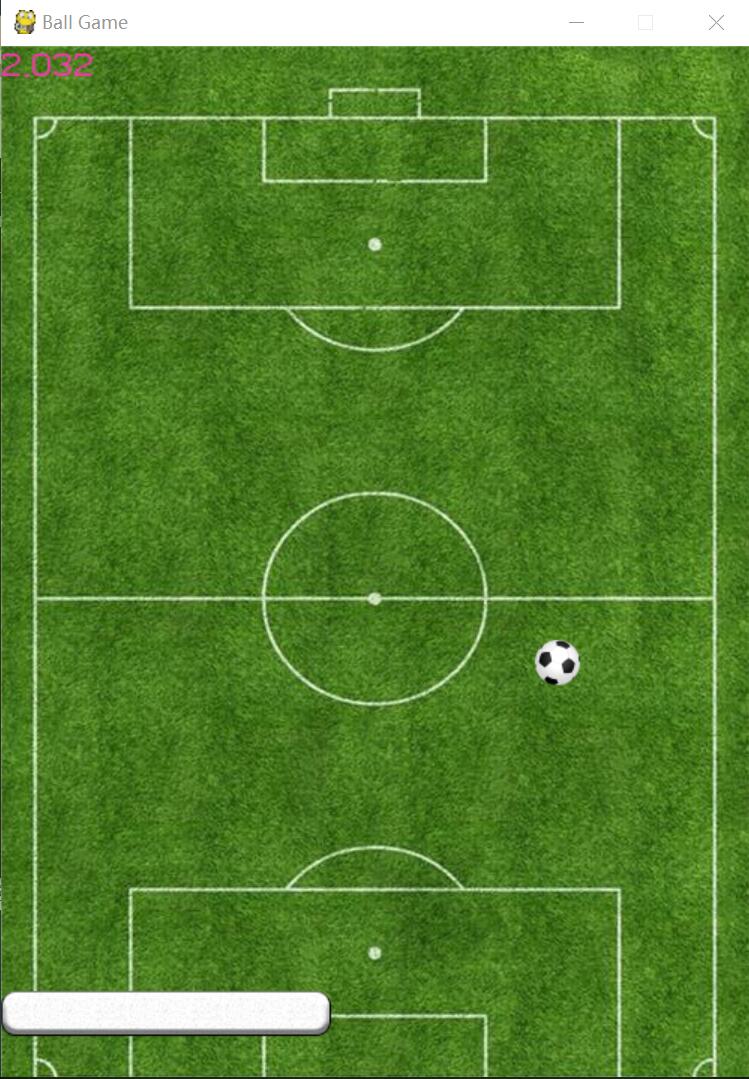


图6-2函数调用后

def GameStart():

# 游戏背景Surface对象

Background = pygame.image.load('GameBackground.jpg').convert()

# 挡板Surface对象

Baffle = pygame.image.load('Baffle.png').convert\_alpha()

# 球Surface对象

Ball = pygame.image.load('Ball.png').convert\_alpha()

while True:

# 接收信息处理

for event in pygame.event.get():

if event.type == QUIT:

exit()

if event.type == KEYDOWN:

if event.key in BaffleMove:

BaffleMove[event.key] = 1

elif event.type == KEYUP:

if event.key in BaffleMove:

BaffleMove[event.key] = 0

# 绘制背景

Screen.blit(Background, (0, 0))

RunTimeStr = str((pygame.time.get\_ticks() - ProgramRunClock) / 1000.0)

# print(RunTimeStr)

# 使用render方法显示时间字体

RunTimeSurface = RunTimeFont.render(RunTimeStr, True, (255, 52, 179))

# 显示时间

Screen.blit(RunTimeSurface, (0, 0))

# 距上次调用clock对象时间

SecondTimePassed = FPSClock.tick(60) / 1000.0

# 绘制球

Screen.blit(Ball, (BallX, BallY))

# 七、设计总结

尚未解决的问题：①游戏中球和挡板、边界碰撞之后的音效没有加入。②挡板的大小不能随意切换。

设计完成过程中的体会：通过此次课程设计,使我更加扎实的掌握了有关方面的知识，在设计过程中虽然遇到了一些问题,但经过一次又一次的思考,一遍又一遍的检查终于找出了原因所在,也暴露出了前期我在这方面的知识欠缺和经验不足。实践出真知,通过亲自动手制作,使我们掌握的知识不再是纸上谈兵。我们虽然只有小组两个人，但是我们的努力比大多数同学都要都多，我们是第一次接触pygame，看了许多的教程与案例，最后通过我们的努力做出了我们的课程设计，按时完成了任务。