  74

**《Python程序设计》**

**课程设计报告**

　　　　　　　　　　（2018 —2019 学年第 一 学期）

题 目 贪吃蛇小游戏

学生姓名 李文鹏

专业班级 信管16-3

学生学号 311609030314

教师姓名 徐 文 鹏

成 绩**：**

评 语：

教师签名：

日期：

目录

[1、设计目的 2](#_Toc535309242)

[1.1 课程设计教学目的： 2](#_Toc535309243)

[1.2 课设选择题目：贪吃蛇小游戏 2](#_Toc535309244)

[2、设计任务和要求 3](#_Toc535309245)

[2.1 课程设计教学任务和要求 3](#_Toc535309246)

[2.2 本课程设计具体任务和要求 3](#_Toc535309247)

[3、总体设计 4](#_Toc535309248)

[3.1设计图纸 4](#_Toc535309249)

[3.2任务分配： 9](#_Toc535309250)

[王萌：（毒）苹果的创建、蛇吃到毒苹果的效果以及蛇的死亡判定。4、设计实现 9](#_Toc535309251)

[4.1图片展示 9](#_Toc535309252)

[4.2 实现结果评价：优秀。 13](#_Toc535309253)

[5、详细设计 14](#_Toc535309254)

[5.1 游戏开始界面 14](#_Toc535309255)

[5.2 游戏结束界面： 15](#_Toc535309256)

[5.3蛇的创建： 16](#_Toc535309257)

[5.4移动贪吃蛇 17](#_Toc535309258)

[5.5蛇的速度 18](#_Toc535309259)

[5.6 添加音效： 19](#_Toc535309260)

[5.7 幸运传送门： 21](#_Toc535309261)

[6、调试与测试 23](#_Toc535309262)

[一、蛇移动的bug。 23](#_Toc535309263)

[二、蛇移动的速度调整。 24](#_Toc535309264)

[7、设计总结 25](#_Toc535309265)

1、设计目的

## 1.1 课程设计教学目的：

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。

## 1.2 课设选择题目：贪吃蛇小游戏

1、初步想法是经典游戏贪吃蛇是一定要敲的，就像学代码，一定要敲一句“hello，world！”一样。

2、我们对于做一个贪吃蛇小游戏是相当有兴趣，而且决定投入精神、精力与热情于其中的。倘若我们做一个不感兴趣的项目，想必最终我们也就是浑水摸鱼，随便应付过去了，这样既浪费我们的时间与精力，又会使老师的教学目的事倍功半。

所以综上所述，在我们了解到有很多人也是做贪吃蛇这个项目的前提下，仍然固执的选择了做一个贪吃蛇小游戏。

# 2、设计任务和要求

## 2.1 课程设计教学任务和要求

本次课程设计的主要任务是以Python为开发语言完成一个100~300行左右规模的程序项目开发。

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。

由于课程设计项目具有一定的综合性，鼓励具有不同特长和不同能力的学生互相组队。项目小组自己推荐一名组长，实行“组长负责制”。组长组织组员进行项目选题、任务分配、方案确定、方案设计、系统调试测试，组员分工协作。小组成员开展项目讨论，互相支持，形成协作意识。

## 2.2 本课程设计具体任务和要求

（1）实现贪吃蛇的基本功能；

（2）可以通过按来控制游戏过程；

（3）利用pyinstaller工具对程序进行打包；

（4）游戏界面美观，对用户友好；

（5）游戏提示显然，使用户一目了然；

（6）始终以一个用户的心理体验和制作游戏，使用户游戏体验良好。

# 3、总体设计

## 3.1设计图纸

软件总体流程图分析

模块初始化

游戏运行主体

main函数

游戏进入界面

游戏结束界面

蛇死亡

按其余键

游戏结束

按其余键

按esc键

按esc键

按esc键或q键

按esc键

函数调用关系图

贪吃蛇

Main函数

主体运行函数

结束界面

画食物

画毒物

画蛇

开始界面

蛇移动

蛇速度控制

蛇死亡判定

食物随机生成

蛇吃食物判定

程序终止

画成绩

画网格

略

无边界

蛇撞墙

软件功能图

游戏进入界面

游戏界面

退出

游戏结束界面

游戏暂停界面

按esc键

按其余任意键

按P键

按esc键

按esc键或q键

用键盘控制蛇的移动

蛇加速减速

蛇吃苹果加分

蛇加速减速

蛇吃毒苹果变色，混乱

幸运传送门

蛇的设计

**[0]**

**[1]**

**[-1]**

**[2]**

**头**

**身体**



**一节身体：**

**{‘x’:X,‘y’:Y}**

**[**

**]**

（毒）苹果的设计



**{‘x’:X,‘y’:Y}**

蛇的移动设计

**添加**

**删除**

蛇的方向控制设计

**向上**

**新头**

**{x,y }**

**{x,y+1 }**

**向下**

**新头**

**{x,y-1}**

**{x,y}**

**向左**

**新头**

**{x,y}**

**{x-1,y}**

**向右**

**新头**

**{x+1,y}**

**{x,y}**

蛇吃果实

**头**

**果实**

## 3.2任务分配：

李文鹏：贪吃蛇的创造和移动以及游戏具体优化；

# 王萌：（毒）苹果的创建、蛇吃到毒苹果的效果以及蛇的死亡判定。4、设计实现

## 4.1图片展示

游戏启动界面：

（1）最终实现结果



4-1 游戏开始界面

（2）实现结果评价：优秀。

基本实现基本想法，界面美观大方。

游戏结束界面：

（1）最终实现效果



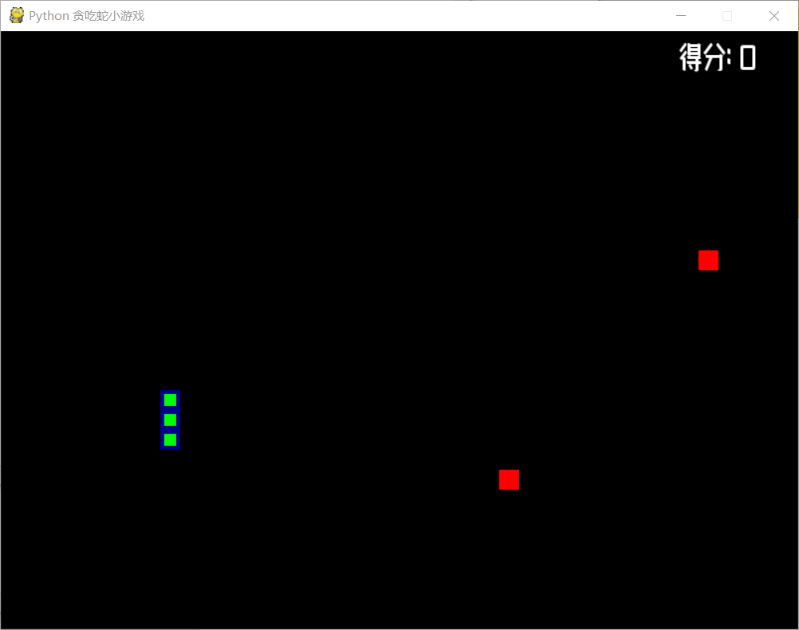
4-2游戏结束界面

（2）实现结果评价：优秀。

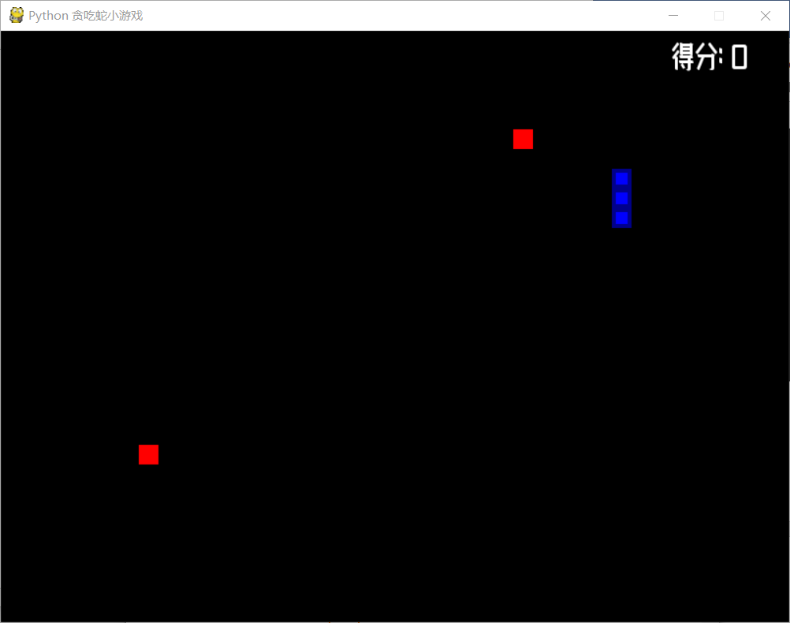
基本实现当初设想，遵循简约美的原则，颜色柔和不刺眼。

游戏界面

1. 最终实现效果



4-3 贪吃蛇吃到毒苹果



4-4 贪吃蛇游戏界面

## 4.2 实现结果评价：优秀。

本来有设想过，找张草地之类的图片拿来当背景，但仔细思考后，觉得有两点不妥之处：1、草地图案太繁琐，这与我们这款软件本身所贯彻的简约美原则相违背；2、草地颜色是绿色和中毒之后的蛇的颜色一样，会影响游戏观感。我最终还是没有改变，选择了黑色当底色，好处也有几点：1、黑色简约，但又不失去一丝神秘和冷酷；2、黑色对于任何鲜艳颜色都是很好的衬色，对游戏玩家的视觉感不会造成负面影响。所以，也许这个界面会让一些人感觉差强人意，但是，我还是蛮满意的，当然也还是有很大的进步空间。嗯，我觉得我超额完成自己的任务。

# 5、详细设计

## 5.1 游戏开始界面

代码：

def show\_start\_info(screen):

font = pygame.font.Font('ttf\myfont.ttf', 40)

tip = font.render('按任意键开始游戏~~~', True, (65, 105, 225))

gamestart = pygame.image.load('img\gamestart.jpg')

screen.blit(gamestart, (140, 80))

screen.blit(tip, (240, 420))

pygame.display.update()

while True: #键盘监听事件

for event in pygame.event.get(): # event handling loop

if event.type == QUIT:

terminate() #终止程序

elif event.type == KEYDOWN:

if (event.key == K\_ESCAPE): #终止程序

terminate() #终止程序

else:

return #结束此函数, 开始游戏

分析：用Font()方法引入字体，用render()将字写入帧；用load()方法加载图片，用blit()方法将文字和图片贴在界面上。用update()方法更新数据。然后监听事件，事件为QUIT或者是ESC，终止程序，退出游戏，如果是其他键，结束循环，进入游戏界面。

输入：所创建的屏幕；

输出：文字，图片出现在应当出现的位置。

## 5.2 游戏结束界面：

代码：

def show\_gameover\_info(screen):

luckDog[0] = 0

font = pygame.font.Font('ttf\myfont.ttf', 30)

tip = font.render('按Q或者ESC退出游戏, 按任意键重新开始游戏~', True, (65, 105, 225))

gamestart = pygame.image.load('img\gameover.png')

screen.fill(yellow)

screen.blit(gamestart, (150,0))

screen.blit(tip, (120, 450))

pygame.display.update()

while True: #键盘监听事件

for event in pygame.event.get(): # event handling loop

if event.type == QUIT:

terminate() #终止程序

elif event.type == KEYDOWN:

if event.key == K\_ESCAPE or event.key == K\_q: #终止程序

terminate() #终止程序

elif event.key != K\_SPACE:

return #结束此函数, 重新开始游戏

分析：与开始界面几乎一样。

## 5.3蛇的创建：

def draw\_snake(screen, snake\_coords):

for coord in snake\_coords:

x = coord['x'] \* cell\_size

y = coord['y'] \* cell\_size

if not poison:

wormSegmentRect = pygame.Rect(x, y, cell\_size, cell\_size)

pygame.draw.rect(screen, dark\_blue, wormSegmentRect)

wormInnerSegmentRect = pygame.Rect( #蛇身子里面的第二层亮蓝色

x + 4, y + 4, cell\_size - 8, cell\_size - 8)

pygame.draw.rect(screen, blue, wormInnerSegmentRect)

else:

wormSegmentRect = pygame.Rect(x, y, cell\_size, cell\_size)

pygame.draw.rect(screen, dark\_blue, wormSegmentRect)

wormInnerSegmentRect = pygame.Rect(

x + 4, y + 4, cell\_size - 8, cell\_size - 8)

pygame.draw.rect(screen, Green, wormInnerSegmentRect)

分析：先用FOR循环遍历获取存储蛇身的字典snake\_coords内存储的蛇身块的位置，根据这个位置调用Rect()方法绘制蛇。

输入：屏幕与存储蛇身的字典snake\_coords。用字典获取蛇身块的位置，再与屏幕对应知道这个位置到底在哪；

输出：绘制出一条蛇。

## 5.4移动贪吃蛇

代码：

def move\_snake(direction, snake\_coords):

if direction == UP:

newHead = {'x': snake\_coords[HEAD]['x'], 'y': snake\_coords[HEAD]['y'] - 1}

elif direction == DOWN:

newHead = {'x': snake\_coords[HEAD]['x'], 'y': snake\_coords[HEAD]['y'] + 1}

elif direction == LEFT:

newHead = {'x': snake\_coords[HEAD]['x'] - 1, 'y': snake\_coords[HEAD]['y']}

elif direction == RIGHT:

newHead = {'x': snake\_coords[HEAD]['x'] + 1, 'y': snake\_coords[HEAD]['y']}

snake\_coords.insert(0, newHead)

分析：蛇的移动，根据总体设计里，我对蛇移动的思考，其实蛇的移动的关键在于蛇的新头添加的位置。

输入：方向参数direction，蛇身snake\_coords。用snake\_coords获取蛇原本的头的位置，用direction确定这个蛇头应该怎么加。

输出：根据方向键控制蛇朝正确的方向移动。

## 5.5蛇的速度

代码：

def snake\_speed\_change(slow,carry,speed,snake\_coords):

if len(snake\_coords) <= 30:

speed = 10 + len(snake\_coords)\*(1/2)

else:

if speed > 10:

speed = 21 - len(snake\_coords)\*(1/2)

#print(len(snake\_coords))

if poison:

return speed/3

else:

if carry:#蛇加速

return 2\*speed

elif slow:#蛇减速

return speed/2

else:

return speed

分析：输入：代表变慢的信号量slow，代表冲刺的信号量carry，还有一个全局变量不用传入就能用的、代表中毒的信号量poison，然后就是分支语句判断蛇属于那种状况，然后返回相应输出。传入蛇身的字典列表，获取蛇的长度，并以此改变蛇的速度。

输出：如果蛇中毒，返回1/3的蛇本身应有速度；如果蛇处于变慢状态，返回1/2蛇速度；处于变快，返回2倍蛇速度；否则返回正常速度。

## 5.6 添加音效：

代码：

pygame.mixer.init()#初始化混音器模块的加载和播放

pygame.mixer.music.load("wav\snake\_bg.wav")

pygame.mixer.music.set\_volume(1)#设置音量

dieSound = pygame.mixer.Sound("wav\snake\_die\_01.wav")

dieSound.set\_volume(1)#设置音量

eatFoodSound = pygame.mixer.Sound("wav\snake\_eat\_food\_01.wav")

eatFoodSound.set\_volume(1)

eatNoFood = pygame.mixer.Sound("wav\snake\_eat\_nofood.wav")

eatNoFood.set\_volume(1)

分析：

一、背景音乐

由于背景音乐是直接加载到mixer.music里了，所以，类似于全局变量，可以直接使用，在游戏运行的时候这个对象直接调用play()方法即可

代码：(在running\_game函数中)

pygame.mixer.music.play() #开始播放

二、吃东西的音效

其实可以在你需要用的函数里再加载这个音乐，这样子就不用传参，但是，我乐意提前加载，所以，我就放到main函数里加载提前加载了，这样需要将你生成的这个变量，一直传参，传到你要调用play()的函数里。由于这个是吃东西的音效，所以，只有当蛇吃到水果的时侯才会发出声音，所以代码放到了蛇吃水果的判定函数中。

代码:(在snake\_is\_eat\_food函数中)

eatFoodSound.play()

三、其他音效：上面吃东西的音效添加方法一样，就不赘述了。

暂停功能

代码：

elif event.key == K\_p:

pause\_flag = 0

while True:

for event in pygame.event.get():

if event.type == QUIT:

pygame.quit()

exit()

if event.type == KEYDOWN:

if event.key == K\_p:

pause\_flag = 1

if pause\_flag == 1:

pygame.mixer.music.unpause()

break

pygame.mixer.music.pause()

elif event.key == K\_SPACE:

carry = not carry

分析：暂停功能关键在于无限循环，设置信号量pause\_flag开始为0，然后就是无限循环监听事件，音乐暂停；倘若监听到键盘事件K\_p的话，就将pause\_flag设置为1，然后加上一个条件判断语句，如果pause\_flag等于1，就使音乐重新播放，跳出无限循环，恢复游戏继续进行。

## 5.7 幸运传送门：

#蛇撞墙

def snake\_is\_dwang(snake\_coords):

tag = True

if snake\_coords[HEAD]['x'] == -1 or snake\_coords[HEAD]['x'] == map\_width or snake\_coords[HEAD]['y'] == -1 or \

snake\_coords[HEAD]['y'] == map\_height:

tag = False # 蛇碰壁啦

return tag

#无边界

def snake\_is\_running(snake\_coords):

if snake\_coords[HEAD]['x'] == -1:

newHead = {'x':map\_width-1, 'y': snake\_coords[HEAD]['y']}

elif snake\_coords[HEAD]['x'] == map\_width:

newHead = {'x':0, 'y': snake\_coords[HEAD]['y']}

elif snake\_coords[HEAD]['y'] == -1:

newHead = {'x': snake\_coords[HEAD]['x'], 'y': map\_height-1}

elif snake\_coords[HEAD]['y'] == map\_height:

newHead = {'x': snake\_coords[HEAD]['x'], 'y': 0}

snake\_coords.insert(0, newHead)

分析：幸运传送门机制就是，当蛇撞墙之后，不一定会死，可能会有一定概率遇到“幸运传送门”，当蛇遇到幸运传送门的时候，蛇将会从屏幕的另外一边以之前的方向出来，而且得分加10分。但是其实根本就没有一个传送门。

我们初步时候是判断，蛇碰壁之后直接死亡，这个是根据蛇头的坐标是否超过屏幕边缘来判断的，如果蛇碰到屏幕边缘，就返回一个参数ret，在游戏运行主函数内有一个条件判断语句，当ret值为真的时候，就break，跳出游戏运行的无限循环，控制游戏进入到游戏结束界面。然后我对此做了优化。

我先使游戏是无边界的，做法是：由四个条件语句，判断蛇头是从什么位置超出了屏幕的边缘，例如，蛇头从屏幕左边超出屏幕边缘，就设置蛇的新头的坐标为{map\_width-1，y},然后插入新头，就和蛇的移动的思想一样。然后我写了一个函数是判断蛇撞墙，蛇撞墙的话，就返回参数ret=True，在游戏运行主体函数内，我写了一个条件判断语句，假如蛇撞墙的话，就按照蛇撞墙函数添加蛇头，否则就按照蛇移动函数还添加蛇头。

# 6、调试与测试

## 一、蛇移动的bug。

初步的贪吃蛇控制方向的代码为：

for event in pygame.event.get():

if event.type == QUIT:

terminate()

elif event.type == KEYDOWN:

if (event.key == K\_LEFT or event.key == K\_a) and direction != LEFT:

direction = RIGHT

elif (event.key == K\_RIGHT or event.key == K\_d) and direction != RIGHT:

direction = LEFT

elif (event.key == K\_UP or event.key == K\_w) and direction != UP:

direction = DOWN

elif (event.key == K\_DOWN or event.key == K\_s) and direction != DOWN:

direction = UP

elif event.key == K\_ESCAPE:

terminate()

我们设定初始方向：direction = RIGHT

这样同时按up(w) 和left(a)键会出现蛇咬到自己的bug

这是由于这一段代码是放到一个while True里面，同时使用了get()方法获得指令。这样，在每一轮时，游戏可以同时接受多个按键指令，这样会导致上面后半部分的 “direction != 方向”代码失效。

所以我们在后面改成了使用poll()方法获取指令，这样在while True的循环的每一轮里，最多只能同时获得一个指令，避免了自己回头咬自己的bug。

## 二、蛇移动的速度调整。

经历：

一、 我们设置了一个全局变量snake\_speed,并在蛇每次吃到水果的时候引用全局变量，改变值为加一，并在游戏结束时重置蛇的速度为15.

缺点：1、过快的速度增加，游戏难度倍增；2、蛇身加长本身也是难度加长，同时还速度加快，这样设计非人类；3、代码本身与我们从开始所坚持的模块化设计思想有冲突，这样设计使我们的结构变得不那么明朗。

二、 我们增加了一个snake\_speed\_change函数，在函数中控制蛇的速度，有了第一次的经验，我们先将蛇的初始速度定为5，再根据蛇的长度来确定是否再给蛇加速。

我们设定在蛇身(len(snake\_coords))<=35时，蛇的速度会每三格加一(speed = 5 +len(snake\_coords)//3),同时在蛇的速度大于5的前提下，蛇身超过35后会每三格减一(speed =15 -

len(snake\_coords)//3),最后速度维持在5

缺点：1、初玩感觉还行，但时间一长就感觉很难受，是速度为5的时候，刷新频率太低，蛇的移动不流畅所致，经多次验证，fps最低为10才看着勉强可以，同时fps在20左右，蛇就已经跑的很快了，所以得出结论fps尽量是在（10-25）之间，玩家最有游戏体验。2、速度公式还是有问题，但由于，我不是主弄这个的，我就试着随便调了调。

# 7、设计总结

经历这次课设，我的收获良多，主要是以下几点：

1. 关于程序的整体思考问题。我打这个贪吃蛇的代码的时候，开始仔仔细细的思考了关于这个贪吃蛇游戏的实现问题。当然，开始我肯定是去网上看了看别人贪吃蛇的代码，然后，我就考虑了几个问题：（1）这个蛇是怎么创建的；（2）这个蛇是怎么移动的；（3）这个蛇是怎么吃东西的。在我对这几个问题做出答案的同时，这个贪吃蛇游戏的本质也在我的面前暴露，这让我在后面的打代码变得尤为简单。
2. 关于程序的整体设计问题。在做这个游戏的时候我是找了很多张图片的，游戏从开始的花花绿绿，到最后的简约。我觉得游戏也许从开始不一定要定下他的整体基调，但是在后面是一定要统一他的整体的风格的，无论是颜色、字体、或是图片的选择，只有风格统一，你这个游戏才显得高大上，而不是像个伪劣次品。
3. 关于程序的用户体验问题。在我没写完一步代码，我都会成为我的程序的第一任玩家，但是，可能是因为我自己敲的代码，然后往往都感受不出来一些瑕疵。所以，在制作这款游戏过程中，我是找了几位同学帮忙试玩的，他们也提出了很多宝贵的意见。我想，一个软件，在做好他的“硬件”的同时，一定要抓好用户体验感，也就是软件的“服务”。