**《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 方 留 龙

专业班级 信管1602班

学生学号 311609030223

教师姓名 徐 文 鹏

成 绩**：**

评 语：

教师签名：

日期：

**目录**

1 设计目的1

1.1 课程设计目的1

1.2 本课程设计具体目的1

2 设计任务和要求2

2.1 课程设计教学任务和要求2

2.2 本课程设计具体任务和要求2

3 总体设计4

3.1 游戏设计任务分配4

3.2 游戏设计功能图4

3.3 游戏设计流程图4

4 设计实现6

4.1 最终实现结果6

4.2 实现结果评价7

5 详细设计8

6 调试与测试16

7 设计总结19

7.1 尚未解决的问题19

7.2 课程设计体会19

附录20

**1 设计目的**

* 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. **课程设计具体目的**

贪吃蛇游戏是一款经典的小游戏，深受大众喜爱。当我们学习了python程序设计课程后，了解到使用python第三方库pygame可以实现这款游戏，但经查阅资料，我们发现使用python内置的tkinter库，采用面向对象的方法也可以实现这款游戏。在此基础上，我们小组讨论决定做贪吃蛇小游戏，使用面向对象方法来实现这款游戏，以此来完成我们的课程设计。鉴于python面向对象我们没有学习，希望挑战一下自己，在团队合作的基础上尝试设计出这款小游戏。一来挑战自己、接受新知；二来自己体验自己做出的游戏，会有一丝小小的成就感；最后就是希望顺利完成这次课程设计。

2 设计任务和要求

2.1 课程设计教学任务和要求

本课程设计主要任务是以Python为开发语言完成一个100~300行左右规模的程序项目开发。设计参考题目见附件1。

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

使用python面向对象方法，结合tkinter库、random库和tkinter.messagebox弹窗库，实现贪吃蛇小游戏。具体的功能设计任务和游戏规则要求如下：

（1）功能设计任务

玩家运行应用程序，进入游戏界面，游戏直接开始。游戏界面由四部分组成，包括得分栏（Score:分值）、游戏操作框、小蛇、食物。初始化分数为0，小蛇的位置、颜色和大小初始化，食物的位置随机出现，初始颜色固定。之后，玩家通过控制方向键或鼠标使贪吃小蛇在屏幕上不停的移动，寻找食物。小蛇的移动速度正常情况下始终保持不变，通过连续点击前进键（小蛇当前前进方向）使小蛇行进加速。在小蛇移动的过程中，每当吃到食物后，游戏界面会出现新食物，食物分为3种类型：绿色代表普通食物，吃了得分并增加小蛇的长度；红色为特殊食物1，吃了得分并减短小蛇的长度；金色为特殊食物2，吃了得分并回到小蛇的最初长度状态；特殊食物3为变色食物，小蛇吃了变色食物会根据食物颜色随机改变自身的颜色。小蛇吃的食物越多，得分就越高。如果小蛇在移动的过程中碰到了自身（蛇不可以向自己反方向走），则游戏结束，弹出弹窗，显示得分，玩家点击确认，可直接进入下一局游戏。每次游戏结束，进入新一局游戏，本局游戏已被初始化。

（2）游戏规则要求

点击运行进入游戏界面，显示得分栏、小蛇、食物。

食物随机出现、小蛇根据初始化位置和方向移动。

玩家点击方向键，小蛇根据玩家的按下的方向键进行移动。

当小蛇头部触及食物时，得分框数据更新，分数加1，小蛇吞掉食物，下一个食物随机出现。

当小蛇吃到绿色食物（普通食物）时，小蛇长度增加1个格；当蛇吃到红色食物（特殊食物1）时，长度减少3个格；当小蛇吃到黄色食物时（特殊食物2），变回蛇的初始长度；当蛇吃到变色食物（特殊食物3）时，返回碰到食物时食物的颜色。

当蛇头触碰到自己身体时游戏结束，弹出弹窗显示得分，点击“确定”，初始化游戏，游戏重新开始。

3 总体设计

3.1 游戏设计任务分配

郭盼盼：总体功能方案分析设计、游戏整体界面设计实现（包括贪吃蛇移动界面，小蛇得分栏、小蛇和食物的显示）、后期整合完善。

樊璟琳：总体功能方案分析设计、游戏规则功能的实现（实现小蛇吃到不同颜色食物后的反应）、后期整合完善。

方留龙：总体功能方案分析设计、游戏结束功能的实现（游戏结束规则功能的实现）、后期整合完善。

3.2 游戏设计功能图



图3-1 贪吃蛇游戏功能图

3.3 游戏设计流程图

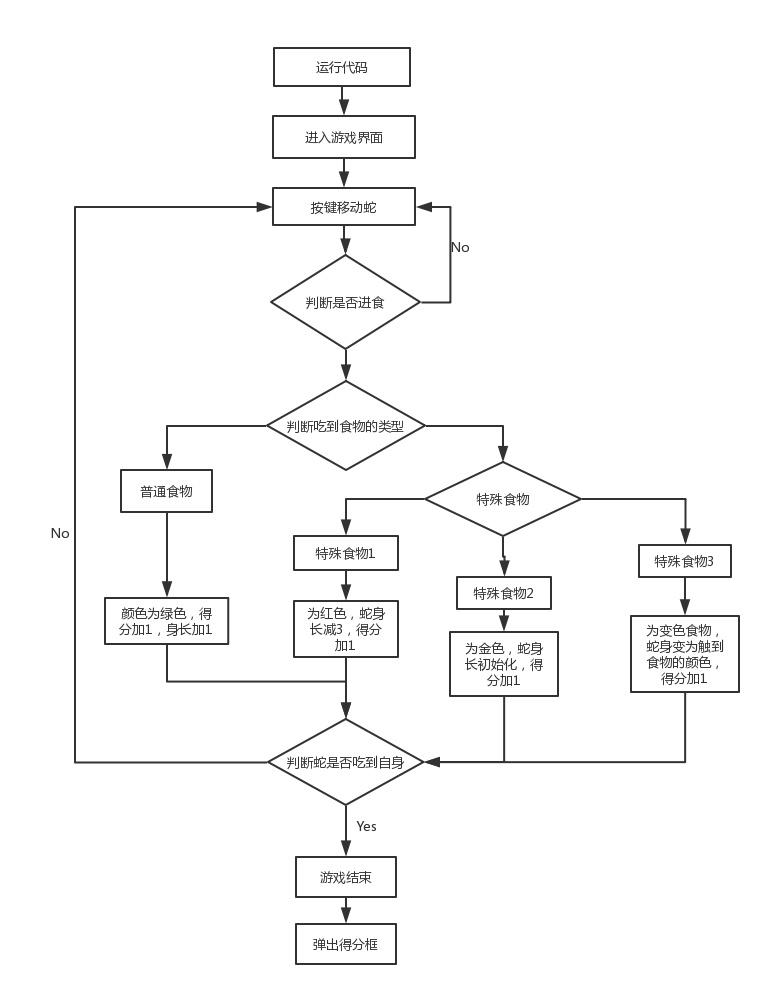


图3-2 贪吃蛇游戏流程图

4 设计实现

4.1 最终实现结果

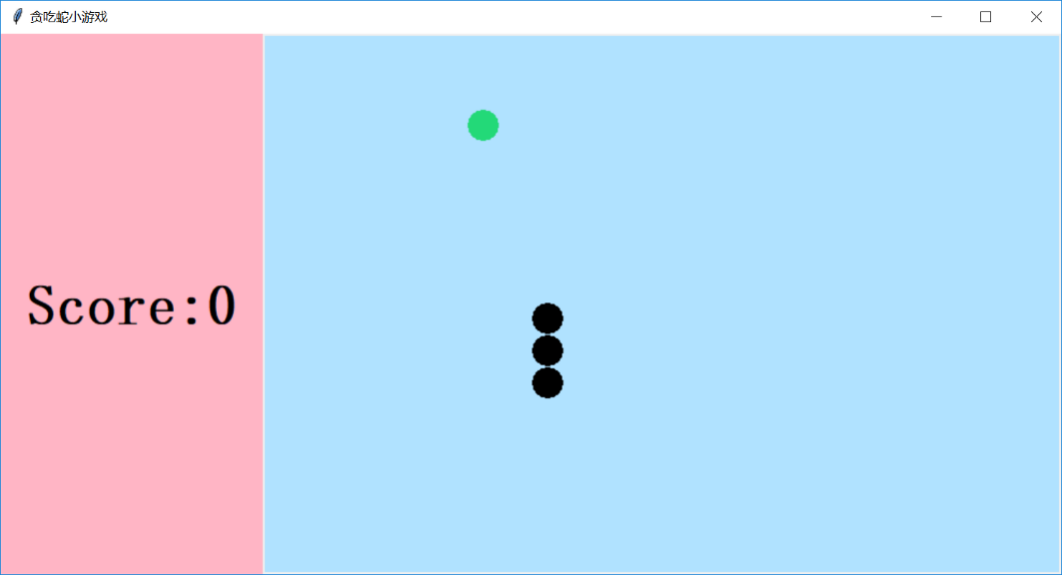


图4-1 玩家首次进入游戏界面

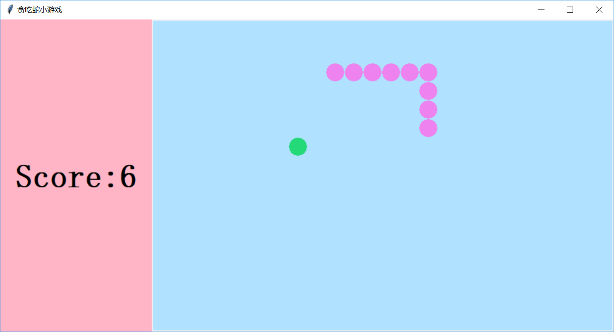
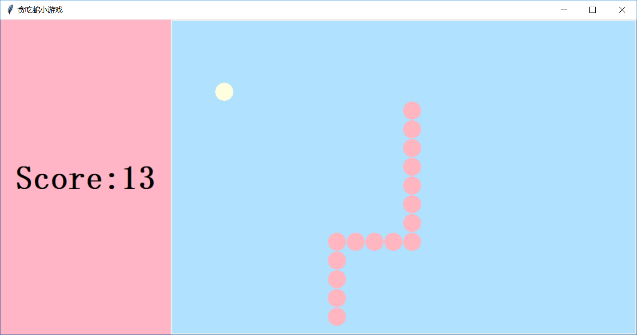
 

图4-2 小蛇吃到变色食物，身体变色图

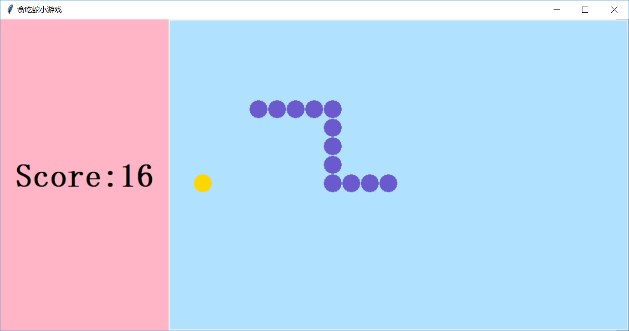
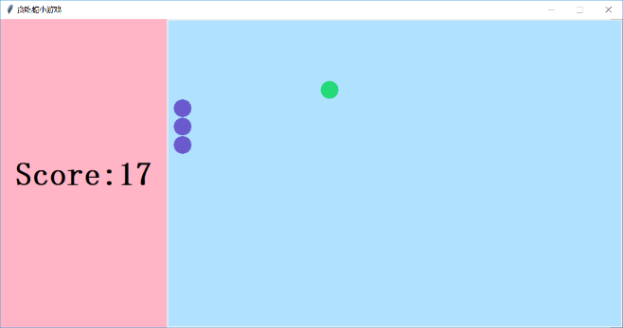
 

图4-3 小蛇吃到金色食物，体长被初始化图

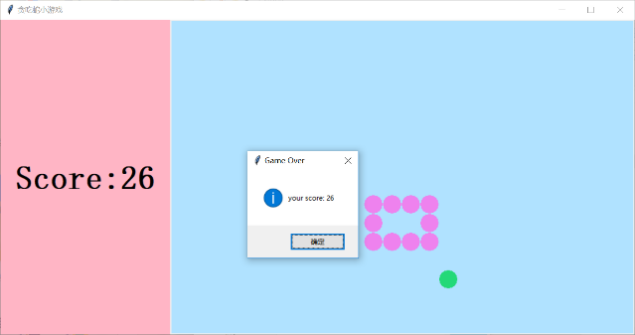
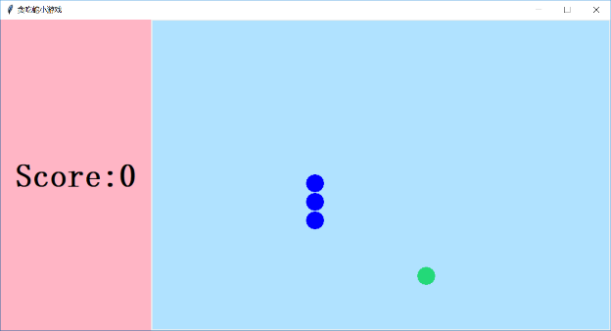
 

图4-4 游戏结束，点击“确定”进入新一轮图

4.2 实现结果评价

程序实现了最初的设计想法和要求，游戏整体界面比较清新美观，功能没有很高大上，但基本功能都已实现，游戏操作界面功能完备，尤其在食物方面，分出了普通食物和特殊食物，小蛇吃到不同事物进行不同反应，这使得游戏更有趣味性。小蛇碰到自身，游戏结束，显示得分，点击“确定”，进行下一轮游戏。游戏开始，游戏运行，游戏结束后开始新一轮游戏功能都已实现。

总体来说，我们认为我们小组的python课程设计“贪吃蛇小游戏”实现了最初的设计想法和要求，可以达到“良好”的成绩，希望也可以得到老师的认可。

5 详细设计

此次设计自己主要负责游戏规则定义模块，当然整体规则还需要小组成员共同讨论决定，现在我对自己所参与设计部分尽享详细介绍。

（1）\_\_init\_\_（）函数

如下代码所示，开始时我与小组成员共同商议游戏中必要参数的设定，定义了\_\_init\_\_（）函数，来定义初始化的必要参数。如开始时食物的颜色，变色食物的颜色。初始化得分栏。初始化蛇的速度为（300）等

def \_\_init\_\_(self, master): #采用\_\_init\_\_()构造方法对进行初始化

Frame.\_\_init\_\_(self, master) #调用用\_\_init\_\_()构造方法进行初始化

self.grid = Grid(master)

self.snake = Snake(self.grid)

self.food = Food(self.grid)

self.gameover = False #设置游戏结束的初始状态为False

self.score = 0 #设置游戏的初始得分为0

self.status = ['run', 'stop'] #设置蛇的状态有run和stop

self.speed = 300 #设置蛇的初始速度为300

self.grid.canvas.bind\_all("<KeyRelease>", self.key\_release)

self.display\_food() #调用初始化食物的显示颜色

#用于设置变色食物

self.color\_c = ("#FFB6C1","#6A5ACD","#0000FF","#F0FFF0","#FFFFE0","#F0F8FF","#EE82EE","#000000","#5FA8D9","#32CD32")

self.i = 0

#界面左侧显示分数

self.m = StringVar()

self.ft1 = ('Fixdsys', 40, "bold")

self.m1 = Message(master, textvariable=self.m, aspect=5000, font=self.ft1, bg="#FFB5C5")

self.m1.pack(side=LEFT, fill=Y)

self.m.set("Score:"+str(self.score))

（2）initial（）函数

定义initial（）函数，用于当游戏结束时初始化游戏，游戏结束时，显示总得分：

def initial(self):#定义一个initial()方法，用于游戏重新开始进行游戏

self.gameover = False#初始化游戏结束时游戏重新开始初始的状态

self.score = 0#初始化游戏结束时游戏重新开始初始时游戏的得分

self.m.set("Score:"+str(self.score))

self.snake.initial()#调用上面的initial()方法回到游戏重新开始时蛇的状态

（3）display\_food()函数

定义display\_food()函数，用来定义事物的类型及颜色。游戏刚开始时，定义食物类型为1，并定义食物的颜色。其他三种类型的食物分别赋予不同的颜色。使用randint()生成随机数，以此来控制不同类型的食物出现的概率。普通类型的食物出现的概率稍大，特殊类型的食物出现的概率稍小。以此来增加游戏的可玩性，吸引游戏客户。此外食物在蛇体内出现时，调用前部分的函数set\_pos()，再次随机生成食物位置.

def display\_food(self):

self.food.color = "#23D978" #初始化食物的颜色为绿色

self.food.type = 1 #初始化食物的类型为第一个类型

if randint(0, 20) == 5:#判断出现的（0,20）范围内出现随机数是否为5，对随机数为5的数字设置其食物颜色和类型

self.food.color = "#FFD700"

self.food.type = 3

while (self.food.pos in self.snake.body):#判断蛇是否吃到食物，若吃到食物，该类食物随机出现

self.food.set\_pos()

self.food.display()

elif randint(0, 4) == 2:#判断出现的（0,4）范围内出现随机数是否为2，对随机数为2的数字设置其食物颜色和类型

self.food.color = "#EE82EE"

self.food.type = 4

while (self.food.pos in self.snake.body): #判断蛇是否吃到食物，若吃到食物，该类食物随机出现

self.food.set\_pos()

self.food.display()

elif len(self.snake.body) > 10 and randint(0, 16) == 1:#当蛇的长度>10个格子，且随机数为（0,16）设置其食物颜色和类型

self.food.color = "#FF0000"

self.food.type = 2

while (self.food.pos in self.snake.body):#判断蛇是否吃到食物，若吃到食物，该类食物随机出现

self.food.set\_pos()

self.food.display()

else:#若以上均不满足，当蛇吃到食物时，新食物随机出现

while (self.food.pos in self.snake.body):

self.food.set\_pos()

self.food.display()

（4）key\_release()函数

定义一个函数key\_release(),在该函数中调用蛇的移动函数move(),监听键盘事件，根据键盘按下的上下左右来移动蛇位置，同时在该函数内部规定了蛇不可以向自己的反方向行走。

def key\_release(self, event):#定义一个key\_release()方法，用于监听键盘事件

key = event.keysym #获取一个键盘事件

key\_dict = {"Up": "Down", "Down": "Up", "Left": "Right", "Right": "Left"} #定义一个字典，后续进行蛇是否向自己的反方向走进行判断

if key in key\_dict and not key == key\_dict[self.snake.direction]: #判断蛇是否是向自己的反方向走，如果不是则蛇前进

self.snake.direction = key

self.move()

（5）run(self)函数

定义一个run()方法来判断游戏进程。首先判断游戏是否结束，如果游戏结束，则显示游戏结束界面，跳出弹窗，显示得分，如果点击了确定按钮，则调用initial函数，使游戏初始化。其次，判断食物的类型，判断类型是否为4，如果是，则调用color\_c()函数，将食物颜色进行切换。若游戏未结束，则调用move()函数，蛇继续移动。

def run(self):#定义一个run()方法判断游戏进程

if not self.status[0] == 'stop':#首先判断游戏是否暂停

if self.gameover == True:#如果游戏结束，则显示得分

message = tkinter.messagebox.showinfo("Game Over", "your score: %d" % self.score)

if message == 'ok':#点击确定按钮，则调用initial函数游戏重新开始，初始化游戏。

self.initial()

if self.food.type == 4:#判断食物类型是否为4,是的话执行以下操作，令变色食物闪烁

color = self.color\_c[self.i]

self.i = (self.i+1)%10#调用color\_c(包含颜色)进行食物颜色闪烁

self.food.color = color

self.food.display()

self.move(color)

else:#如果游戏不结束、食物类型不为4，调用move()方法

self.move()

self.after(self.speed, self.run)

流程图如下：

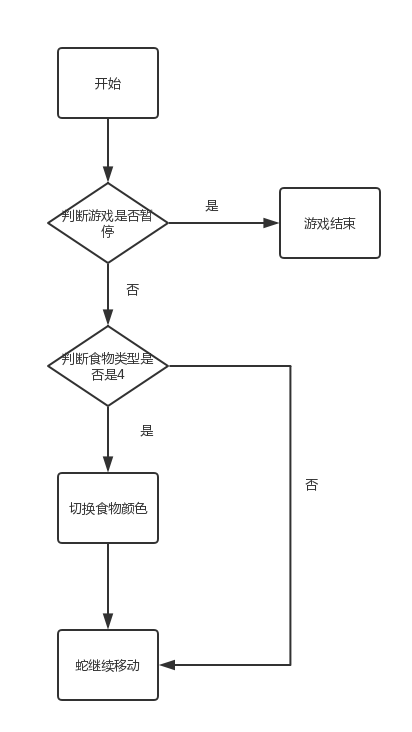


图5-1

（6）move()函数

定义一个move()函数，来对蛇的移动进行控制，按下键盘上的“上下左右”，方向键，蛇向不同的方向移动。计算蛇头的移动位置，如果蛇头没有触碰到自己的身体，则游戏继续，若蛇头触碰到自己的身体，则游戏结束。如果蛇吃到了食物，则蛇根据吃到的食物类型，调用不同的函数，做出不同的变换。如吃到普通食物，则蛇的身体加一，分数加一；如果蛇吃到食物类型为二，则调用cut\_down函数，身体长度减三，得分加一；如果蛇吃到食物类型为四，则调用change\_color函数，蛇身体颜色改变，得分加一。同时右边得分栏根据蛇是否吃到食物不断地刷新。该部分代码如下：

def move(self, color="#EE82EE"):

# 计算蛇下一次移动的点

head = self.snake.body[0]

if self.snake.direction == 'Up':#如果蛇向上移动，执行以下操作

if head[1] - 1 < 0:

new = (head[0], 16)

else:

new = (head[0], head[1] - 1)

elif self.snake.direction == 'Down':#如果蛇向上移动，执行以下操作

new = (head[0], (head[1] + 1) % 16)

elif self.snake.direction == 'Left':#如果蛇向左移动，执行下面的操作

if head[0] - 1 < 0:

new = (24, head[1])

else:

new = (head[0] - 1, head[1])

else: #如果蛇不符合上述3个移动方向，执行下面的一行代码

new = ((head[0] + 1) % 24, head[1])

if new in self.snake.body: #撞到自己，游戏结束

self.gameover=True

elif new == self.food.pos:#如果吃到食物

if self.food.type == 1: #如果吃到的食物类型是1，则调用add函数，蛇向一个指定点移动，并增加自身函数

self.snake.add(new)

elif self.food.type == 2: #如果吃到的食物类型是2，则调用cut\_down函数，剪短自身长度

self.snake.cut\_down(new)

elif self.food.type == 4: #如果吃到的食物类型是4，则调用change函数，改变自身颜色

self.snake.change\_color(new, color)

else: #如果没有吃到上述类型的食物，执行下面的方法

self.snake.init(new) #蛇吃到了黄色食物则回到原始长度。

self.display\_food() #蛇吃到了食物，则调用display\_food函数，随机生成食物

self.score = self.score+1 #只要蛇吃到了食物，不管什么食物，则分数自动加一

self.m.set("Score:" + str(self.score))#更新分数显示区域的分数显示。

else:#什么都没撞到，继续前进

self.snake.move(new)

流程图如下:

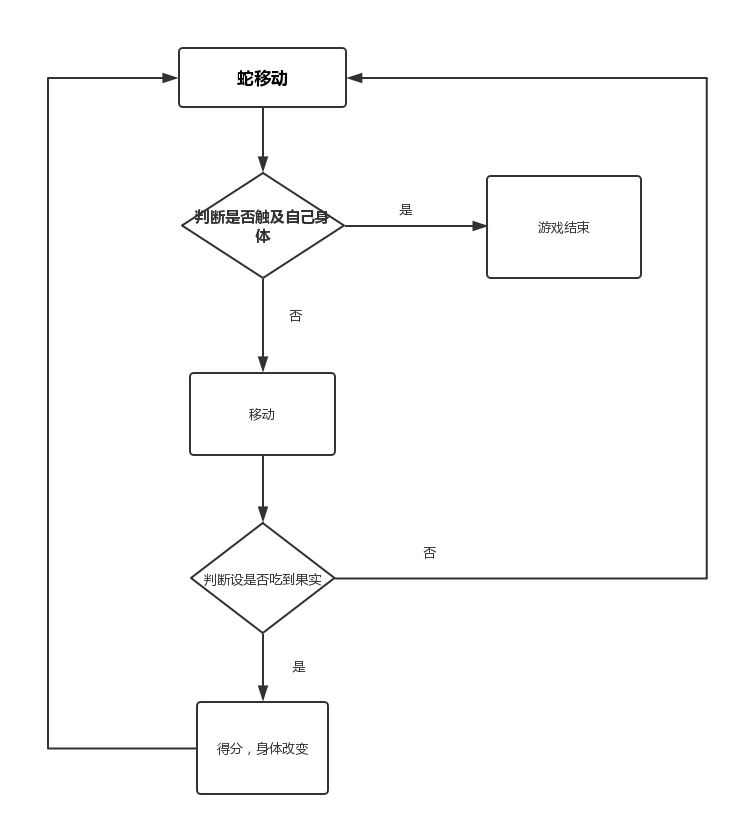


图5-2

6 调试与测试

在此次设计中，程序的调试并不是自己独立完成，而是与团队成员一起完成的，大家一起找错误，优化界面。

蛇的形状和食物形状的调试代码如下：

def draw(self, pos, color): #定义一个draw函数，作用是绘制小蛇的行进路径、形状实物的形状

#pos位置坐标参数

x = pos[0] \* self.grid\_width + self.offset

y = pos[1] \* self.grid\_width + self.offset

#outline属性要与网格的背景色（self.bg）相同，更加美观

self.canvas.create\_oval(x, y, x + self.grid\_width, y + self.grid\_width, fill=color, outline=self.bg)

#绘制一个圆形((a,b,c,d),值为左上角和右下角的坐标)，路径及小蛇、实物的形状，轮廓outline属性设置为网格的背景色

刚开始时，设计的蛇食物的形状都是矩形，后期考虑整体界面的美观，将其改成了圆形，方法是将draw()函数中的self.canvas.create\_rectangle改为self.canvas.create\_oval即可，调试完成的蛇形状和果实形状如下：

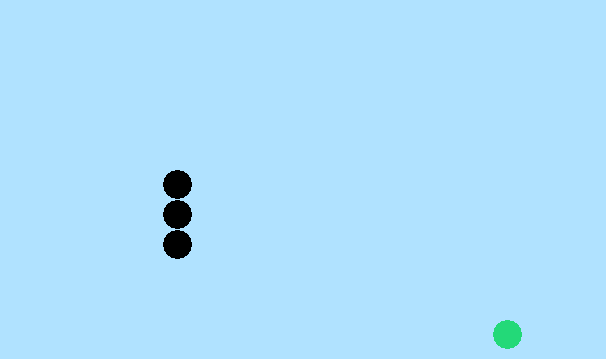


图6-1

食物出现概率的调试代码如下：

def display\_food(self):

self.food.color = "#23D978"#初始化食物的颜色为绿色

self.food.type = 1 #初始化食物的类型为第一个类型

if randint(0, 20) == 5:#判断出现的（0,20）范围内出现随机数是否为5，对随机数为5的数字设置其食物颜色和类型

self.food.color = "#FFD700"

self.food.type = 3

while (self.food.pos in self.snake.body):#判断蛇是否吃到食物，若吃到食物，该类食物随机出现

self.food.set\_pos()

self.food.display()

elif randint(0, 4) == 2:#判断出现的（0,4）范围内出现随机数是否为2，对随机数为2的数字设置其食物颜色和类型

self.food.color = "#EE82EE"

self.food.type = 4

while (self.food.pos in self.snake.body):#判断蛇是否吃到食物，若吃到食物，该类食物随机出现

self.food.set\_pos()

self.food.display()

elif len(self.snake.body) > 10 and randint(0, 16) == 1: #当蛇的长度>10个格子，且随机数为（0,16）设置其食物颜色和类型

self.food.color = "#FF0000"

self.food.type = 2

while (self.food.pos in self.snake.body):#判断蛇是否吃到食物，若吃到食物，该类食物随机出现

self.food.set\_pos()

self.food.display()

else:#若以上均不满足，当蛇吃到食物时，新食物随机出现

while (self.food.pos in self.snake.body):

self.food.set\_pos()

self.food.display()

游戏做完整后，团队成员对游戏进行了测试，发现四种类型的食物，出现的概率差别太大，有时候出现了十几次普通食物，还没有出现一次特殊食物。于是我们将特殊食物出现的概率增大，方法是修改randint()的范围，以此来增大特殊食物出现的概率。

游戏面板的调试是樊璟琳同学独自完成，在此不做过多介绍。

7 设计总结

7.1 尚未解决的问题

此次虽然将贪吃蛇小游戏完整的做了出来。但是对于整个游戏来说，还有很多不足需要完善，如游戏开始界面本小组并未进行设计，还有原本的想法（“蛇头触及墙壁，则游戏结束”）由于时间原因，也并未在游戏中体现。对于我个人，至今还有一个问题困扰着我，就是计算蛇头，下一步移动位置的问题。由于开始自己使用的pygame库进行调用编写的，所以并未考虑，蛇自身的太多问题，如当蛇头触及自身是否要游戏结束。后台经过团队商量，为了完善游戏，决定将这一部分加入，但是团队内部并没有良好的解决方法，只好查阅资料，终于在一个“大神”的博客下面，找到了方法。但是后期团队内部对该方法并不是很理解，直到现在，还有部分搞不懂。后期，团队成员要请教一些“高手”，将此部分“吃透”，在以后的学习中，自己要着重培养自己面对问题，解决问题的能力，训练自己的编程思维。

7.2 课程设计体会

很高兴能与樊璟琳、郭盼盼同学一起完成python课程设计，同时也很感谢老师这一学期来对学生的教诲。在这次课设过程中，自己学会了很多东西。以前自己做项目，都是个人单独开发，效率低，进展慢。这次课设让我真正体会到了团队协作的重要性。使我体会到了完成一件事情的喜悦。此次课设对我个人来说成长是非常快的，在这个过程中，所需要综合知识多，不仅要熟悉老师所讲内容，还要自学关于python开发的内容，要仔细考虑每一个环节的可行性和必要性。从一开始的毫无头绪，到最后把想法付诸于实践，并完整的将整个小项目做出来。这个完整的过程真正激发了我对编程的兴趣。再次感谢小组成员，樊璟琳、郭盼盼同学在设计过程中对我的帮助，感谢老师这半学期来的谆谆教诲，将学生从一个对编程的懵懂状态，带到理解编程，学会编程，并主动编程的状态中来。

**附录：**

from tkinter import \* #导入tkinter库，是python的内置库，导入即可使用

from random import randint #导入random模块中的randint函数，用于生成随机整数

import tkinter.messagebox #导入弹窗库，用于信息提示

class Grid(object): #定义一个网格（Gird）类，用于设置游戏总界面，蛇轨迹绘画等

#初始化函数，\_\_init\_\_()方法,被称为类的构造函数或初始化方法，当创建了类的实例时就会调用该方法

def \_\_init\_\_(self, master=None,height=16, width=24, offset=10, grid\_width=30, grid\_height=30,bg="#B0E2FF"): #采用\_\_init\_\_()构造方法初始化Gird类的宽和高、背景颜色(浅蓝)

self.height = height #把参数height赋值给对象变量height，值为16，用于设置canvas纵坐标中网格的个数

self.width = width #把参数width 赋值给对象变量width ，值为24，用于设置canvas横坐标中网格的个数

self.offset = offset #把参数offset 赋值给对象变量offset，为允许发生的偏移量

self.grid\_width = grid\_width #把参数grid\_width赋值给对象变量grid\_width，单个网格的宽度为50

self.grid\_height=grid\_height #把参数grid\_height赋值给对象变量grid\_height，单个网格的高度为50

self.bg = bg #把参数bg赋值给对象变量bg，设置背景色

self.canvas = Canvas(master, width=self.width\*self.grid\_width+2\*self.offset, height=self.height\*self.grid\_height+ #设置画布(蛇活动范围)的大小和背景颜色

2\*self.offset, bg=self.bg)

self.canvas.pack(side=RIGHT,fill=Y) #用pack()函数对画布进行布局，side=RIGHT设置得分栏位于canvas的右侧，fill=Y填充Y方向的空间

def draw(self, pos, color): #定义一个draw函数，作用是绘制小蛇的行进路径、形状实物的形状

#pos位置坐标参数

x = pos[0] \* self.grid\_width + self.offset

y = pos[1] \* self.grid\_width + self.offset

#outline属性要与网格的背景色（self.bg）相同，更加美观

self.canvas.create\_oval(x, y, x + self.grid\_width, y +self.grid\_width, fill=color, outline=self.bg)

#绘制一个圆形((a,b,c,d),值为左上角和右下角的坐标)，路径及小蛇、实物的形状，轮廓outline属性设置为网格的背景色

class Food(object): #定义一个食物类，用于设置食物的各种参数

def \_\_init\_\_(self, grid,color="#23D978"): #采用\_\_init\_\_()构造方法定义一个食物

self.grid = grid #引用grid类为食物网格

self.color = color #食物的颜色

self.set\_pos() #食物的位置

self.type = 1 #第一次游戏开始时食物的类型

def set\_pos(self): #定义食物出现位置的set\_pos()方法

x = randint(0, self.grid.width - 1) #调用randint随机出现食物横坐标的位置

y = randint(0, self.grid.height - 1) #调用randint随机出现食物纵坐标的位置

self.pos = (x, y) #返回食物出现的位置坐标

def display(self): #定义display方法显示食物和蛇

self.grid.draw(self.pos, self.color)

class Snake(object): #定义一个蛇类，用于定义蛇的各种参数

def \_\_init\_\_(self, grid, color = "#000000"): #采用\_\_init\_\_()构造方法对蛇进行初始化

self.grid = grid #引用grid类为蛇的网格

self.color = color #蛇的初始化颜色

self.body = [(8, 11), (8, 12), (8, 13)] #蛇的初始化长度

self.direction = "Up" #定义蛇的初始行进方向向上

for i in self.body: #用for循环遍历蛇

self.grid.draw(i, self.color)

#定义initial()方法用于游戏重新开始时初始化贪吃蛇的位置

def initial(self):

while not len(self.body) == 0: #进行while循环，知道蛇的长度为0，退出while循环

pop = self.body.pop()

self.grid.draw(pop, self.grid.bg)

self.body = [(8, 11), (8, 12), (8, 13)] #初始化蛇的大小

self.direction = "Up" #游戏重新开始初始化蛇的方向

self.color = "blue" #游戏重新开始初始化蛇的颜色

for i in self.body: #用for循环遍历蛇

self.grid.draw(i, self.color)

#蛇向一个指定点移动

def move(self, new): #定义move()方法使蛇发生移动

self.body.insert(0, new)

pop = self.body.pop()

self.grid.draw(pop, self.grid.bg)

self.grid.draw(new, self.color)

#蛇向一个指定点移动，并增加长度

def add(self ,new): #定义一个add()方法,用于增加蛇的长度

self.body.insert(0, new) #更新后蛇的长度

self.grid.draw(new, self.color) #显示更新后的蛇

#蛇吃到了特殊食物1，减短自身的长度

def cut\_down(self,new): #定义一个cut\_down()方法，用于减短蛇的长度

self.body.insert(0, new) #更新后蛇的长度

self.grid.draw(new, self.color) #显示更新后的蛇

for i in range(0,3): #遍历for循环，将蛇的长度减少3个格

pop = self.body.pop()

self.grid.draw(pop, self.grid.bg)

#蛇吃到了特殊食物2，回到最初长度

def init(self, new): #定义init()方法用于返回蛇最初的长度

self.body.insert(0, new)

self.grid.draw(new, self.color)

while len(self.body) > 3: #判断蛇的长度，如果大于3个格，则用while循环返回蛇的初始长度

pop = self.body.pop()

self.grid.draw(pop, self.grid.bg)

#蛇吃到了特殊食物3，改变了自身的颜色

def change\_color(self, new, color): #定义change\_color()方法，改变蛇的颜色

self.color = color #蛇改变为特殊食物的颜色

self.body.insert(0, new) #显示更新后的蛇

for item in self.body: #遍历蛇的身体，全部改变为改变之后的颜色

self.grid.draw(item, self.color)

class SnakeGame(Frame): #定义一个SnakeGame类，Frame框架，用于游戏规则的设定和进行游戏控制

def \_\_init\_\_(self, master): #采用\_\_init\_\_()构造方法对SnakeGame进行初始化

Frame.\_\_init\_\_(self, master) #调用用\_\_init\_\_()构造方法对框架进行初始化

self.grid = Grid(master)

self.snake = Snake(self.grid)

self.food = Food(self.grid)

self.gameover = False #设置游戏结束的初始状态为False

self.score = 0 #设置游戏的初始得分为0

self.status = ['run', 'stop'] #设置蛇的状态有run和stop

self.speed = 300 #设置蛇的初始速度为300

self.grid.canvas.bind\_all("<KeyRelease>", self.key\_release)

self.display\_food() #调用初始化食物的显示颜色

#用于设置变色食物

self.color\_c = ("#FFB6C1","#6A5ACD","#0000FF","#F0FFF0","#FFFFE0","#F0F8FF","#EE82EE","#000000","#5FA8D9","#32CD32")

self.i = 0

#界面左侧显示分数

self.m = StringVar()

self.ft1 = ('Fixdsys', 40, "bold")

self.m1 = Message(master, textvariable=self.m, aspect=5000, font=self.ft1, bg="#FFB5C5")

self.m1.pack(side=LEFT, fill=Y)

self.m.set("Score:"+str(self.score))

#这个方法用于游戏重新开始时初始化游戏

def initial(self): #定义一个initial()方法，用于游戏重新开始进行游戏

self.gameover = False #初始化游戏结束时游戏重新开始初始的状态

self.score = 0 #初始化游戏结束时游戏重新开始初始时游戏的得分

self.m.set("Score:"+str(self.score))

self.snake.initial() #调用上面的initial()方法回到游戏重新开始时蛇的状态

#type1:普通食物 type2:减少3 type3:回到最初状态 type4:吃了会变色

def display\_food(self): #定义display\_food()，用于显示食物的颜色、位置和类型

self.food.color = "#23D978" #初始化食物的颜色为绿色

self.food.type = 1 #初始化食物的类型为第一个类型

if randint(0, 20) == 5: #判断出现的（0,20）范围内出现随机数是否为5，对随机数为5的数字设置其食物颜色和类型

self.food.color = "#FFD700"

self.food.type = 3

while (self.food.pos in self.snake.body): #判断蛇是否吃到食物，若吃到食物，该类食物随机出现

self.food.set\_pos()

self.food.display()

elif randint(0, 4) == 2: #判断出现的（0,4）范围内出现随机数是否为2，对随机数为2的数字设置其食物颜色和类型

self.food.color = "#EE82EE"

self.food.type = 4

while (self.food.pos in self.snake.body): #判断蛇是否吃到食物，若吃到食物，该类食物随机出现

self.food.set\_pos()

self.food.display()

elif len(self.snake.body) > 10 and randint(0, 16) == 1: #当蛇的长度>10个格子，且随机数为（0,16）设置其食物颜色和类型

self.food.color = "#FF0000"

self.food.type = 2

while (self.food.pos in self.snake.body): #判断蛇是否吃到食物，若吃到食物，该类食物随机出现

self.food.set\_pos()

self.food.display()

else: #若以上均不满足，当蛇吃到食物时，新食物随机出现

while (self.food.pos in self.snake.body):

self.food.set\_pos()

self.food.display()

def key\_release(self, event): #定义一个key\_release()方法，用于监听键盘事件

key = event.keysym #获取一个键盘事件

key\_dict = {"Up": "Down", "Down": "Up", "Left": "Right", "Right": "Left"} #定义一个字典，后续进行蛇是否向自己的反方向走进行判断

if key in key\_dict and not key == key\_dict[self.snake.direction]: #判断蛇是否是向自己的反方向走，如果不是则蛇前进

self.snake.direction = key

self.move()

def run(self): #定义一个run()方法判断游戏进程

#首先判断游戏是否暂停

if not self.status[0] == 'stop':

#判断游戏是否结束

if self.gameover == True: #如果游戏结束，则显示界面Game Over,your score是多少分

message = tkinter.messagebox.showinfo("Game Over", "your score: %d" % self.score)

if message == 'ok': #点击确定按钮，则调用initial函数游戏重新开始，初始化游戏。

self.initial()

if self.food.type == 4: #判断食物类型是否为4,是的话执行以下操作，令变色食物闪烁

color = self.color\_c[self.i]

self.i = (self.i+1)%10 #调用color\_c(包含颜色)进行食物颜色闪烁

self.food.color = color

self.food.display()

self.move(color)

else: #如果游戏不结束、食物类型不为4，调用move()方法

self.move()

self.after(self.speed, self.run)

def move(self, color="#EE82EE"): #定义move()方法，用与判断蛇的移动方向

# 计算蛇下一次移动的点

head = self.snake.body[0]

if self.snake.direction == 'Up': #如果蛇向上移动，执行以下操作

if head[1] - 1 < 0:

new = (head[0], 16)

else:

new = (head[0], head[1] - 1)

elif self.snake.direction == 'Down': #如果蛇向上移动，执行以下操作

new = (head[0], (head[1] + 1) % 16)

elif self.snake.direction == 'Left': #如果蛇向左移动，执行下面的操作

if head[0] - 1 < 0:

new = (24, head[1])

else:

new = (head[0] - 1, head[1])

else: #如果蛇不符合上述3个移动方向，执行下面的一行代码

new = ((head[0] + 1) % 24, head[1])

#撞到自己，设置游戏结束的标志位，等待下一循环

if new in self.snake.body:

self.gameover=True

#如果吃到食物

elif new == self.food.pos:

if self.food.type == 1: #如果吃到的食物类型是1，则调用add函数，蛇向一个指定点移动，并增加自身函数

self.snake.add(new)

elif self.food.type == 2: #如果吃到的食物类型是2，则调用cut\_down函数，剪短自身长度

self.snake.cut\_down(new)

elif self.food.type == 4: #如果吃到的食物类型是4，则调用change函数，改变自身颜色

self.snake.change\_color(new, color)

else: #如果没有吃到上述类型的食物，执行下面的方法

self.snake.init(new) #蛇吃到了黄色食物则回到原始长度。

self.display\_food() #蛇吃到了食物，则调用display\_food函数，随机生成食物

self.score = self.score+1 #只要蛇吃到了食物，不管什么食物，则分数自动加一

self.m.set("Score:" + str(self.score)) #更新分数显示区域的分数显示。

#什么都没撞到，继续前进

else:

self.snake.move(new)

if \_\_name\_\_ == '\_\_main\_\_': #\_\_name\_\_就是标识模块的名字的一个系统变量,通过if判断这样就可以执行“\_\_mian\_\_

root = Tk() #创建一个窗口

root.title("贪吃蛇小游戏")

snakegame = SnakeGame(root)

snakegame.run() #游戏开始

snakegame.mainloop() #进入消息循环