**贪吃蛇游戏**

# 课程设计内容概述

## 开发环境和开发工具说明

使用基于Python语言的pygame库进行开发，编译环境是Pycharm。

## 课程设计的任务

开发一个贪吃蛇游戏，通过四个方向键来上下左右控制蛇的运动方向，寻找吃的东西，每吃一口蛇的身子会长一点，身子越长玩的难度就越大，当碰墙，咬到自己的身体或者咬自己的尾巴时，游戏结束。

## 程序运行顺序说明

先安装pygame, time, sys, random等库，接着在pycharm中打开snake.py运行。

# 系统设计方案

主要需要包括界面初始化，键盘按键检测，蛇的移动，吃果子检测，死亡检测等部分。



初始化部分：初始化界面

键盘按键检测部分与蛇的移动部分：检测到相应方向键按下，蛇做出相应方向的移动。

吃果子检测部分：如果有果子被吃，则随机出现新的果子，并且蛇长大。

死亡检测部分：如果蛇碰到了墙或者自己，则游戏结束，否则游戏继续。

# 程序主要代码分析

键盘按键检测部分：

# 判断键盘事件

if event.key == K\_RIGHT or event.key == ord('d'):

changeDirection = 'right'

if event.key == K\_LEFT or event.key == ord('a'):

changeDirection = 'left'

if event.key == K\_UP or event.key == ord('w'):

changeDirection = 'up'

if event.key == K\_DOWN or event.key == ord('s'):

changeDirection = 'down'

蛇的变化部分：

# 根据方向移动蛇头的坐标

if direction == 'right':

snakePosition[0] += 20

if direction == 'left':

snakePosition[0] -= 20

if direction == 'up':

snakePosition[1] -= 20

if direction == 'down':

snakePosition[1] += 20

# 增加蛇的长度

snakeSegments.insert(0,list(snakePosition))

吃果子检测部分：

# 判断是否吃掉了果子

if snakePosition[0] == raspberryPosition[0] and snakePosition[1] == raspberryPosition[1]:

raspberrySpawned = 0

else:

snakeSegments.pop()

# 如果吃掉果子，则重新生成果子

if raspberrySpawned == 0:

x = random.randrange(1,32)

y = random.randrange(1,24)

raspberryPosition = [int(x\*20),int(y\*20)]

raspberrySpawned = 1

# 程序调试中出现的问题及解决办法

程序运行出现界面后，遇到蛇的大小一直是一个方块大小的问题，吃了果子之后也一直是那么大，并没有变大。推测这可能是增加蛇的长度的语句出错了。

在snakeSegments.insert(0, snakePosition)中，.insert中第一个参数是要插入的元素的位置，第二个参数是要插入的元素，这里表示在最开始的位置插入坐标。snakePosition只是一个坐标组合，当要插入元素时，需要先将它强制转换成为list，才能再添加元素。将语句改为snakeSegments.insert(0, list(snakePosition))后，蛇就会正常变大，程序就正常了，和预想的一样了。

在蛇碰到墙，游戏结束后，会报错pygame.error: font not initialized。是字体没有初始化的意思，通过百度查找这个错误，有网友说是缺少对pygame的初始化，也就是缺少pygame.init()。将这句话加于程序运行的最开始，作为主函数里执行的第一条语句，再次运行程序，在游戏结束的时候就不会报错了。

# 主要参考资料

Python和pygame游戏开发指南

pygame教程

# 程序及解释

#!/usr/bin/env python

import pygame,sys,time,random

from pygame.locals import \*

# 定义颜色变量

redColour = pygame.Color(255,0,0)

blackColour = pygame.Color(0,0,0)

whiteColour = pygame.Color(255,255,255)

greyColour = pygame.Color(150,150,150)

# 定义gameOver函数

def gameOver(playSurface):

gameOverFont = pygame.font.Font('arial.ttf',72)

gameOverSurf = gameOverFont.render('Game Over', True, greyColour)

gameOverRect = gameOverSurf.get\_rect()

gameOverRect.midtop = (320, 10)

playSurface.blit(gameOverSurf, gameOverRect)

pygame.display.flip()

time.sleep(5)

pygame.quit()

sys.exit()

# 定义main函数

def main():

# 初始化pygame

pygame.init()

fpsClock = pygame.time.Clock()

# 创建pygame显示层

playSurface = pygame.display.set\_mode((640,480))

pygame.display.set\_caption('Raspberry Snake')

# 初始化变量

snakePosition = [100,100]

snakeSegments = [[100,100],[80,100],[60,100]]

raspberryPosition = [300,300]

raspberrySpawned = 1

direction = 'right'

changeDirection = direction

while True:

# 检测例如按键等pygame事件

for event in pygame.event.get():

if event.type == QUIT:

pygame.quit()

sys.exit()

elif event.type == KEYDOWN:

# 判断键盘事件

if event.key == K\_RIGHT or event.key == ord('d'):

changeDirection = 'right'

if event.key == K\_LEFT or event.key == ord('a'):

changeDirection = 'left'

if event.key == K\_UP or event.key == ord('w'):

changeDirection = 'up'

if event.key == K\_DOWN or event.key == ord('s'):

changeDirection = 'down'

if event.key == K\_ESCAPE:

pygame.event.post(pygame.event.Event(QUIT))

# 判断是否输入了反方向

if changeDirection == 'right' and not direction == 'left':

direction = changeDirection

if changeDirection == 'left' and not direction == 'right':

direction = changeDirection

if changeDirection == 'up' and not direction == 'down':

direction = changeDirection

if changeDirection == 'down' and not direction == 'up':

direction = changeDirection

# 根据方向移动蛇头的坐标

if direction == 'right':

snakePosition[0] += 20

if direction == 'left':

snakePosition[0] -= 20

if direction == 'up':

snakePosition[1] -= 20

if direction == 'down':

snakePosition[1] += 20

# 增加蛇的长度

snakeSegments.insert(0, list(snakePosition))

# 判断是否吃掉了果子

if snakePosition[0] == raspberryPosition[0] and snakePosition[1] == raspberryPosition[1]:

raspberrySpawned = 0

else:

snakeSegments.pop()

# 如果吃掉果子，则重新生成果子

if raspberrySpawned == 0:

x = random.randrange(1,32)

y = random.randrange(1,24)

raspberryPosition = [int(x\*20),int(y\*20)]

raspberrySpawned = 1

# 绘制pygame显示层

playSurface.fill(blackColour)

for position in snakeSegments:

pygame.draw.rect(playSurface,whiteColour,Rect(position[0],position[1],20,20))

pygame.draw.rect(playSurface,redColour,Rect(raspberryPosition[0],

raspberryPosition[1],20,20))

# 刷新pygame显示层

pygame.display.flip()

# 判断是否死亡

if snakePosition[0] > 620 or snakePosition[0] < 0:

gameOver(playSurface)

if snakePosition[1] > 460 or snakePosition[1] < 0:

for snakeBody in snakeSegments[1:]:

if snakePosition[0] == snakeBody[0] and snakePosition[1] == snakeBody[1]:

gameOver(playSurface)

# 控制游戏速度

fpsClock.tick(5)

if \_\_name\_\_ == "\_\_main\_\_":

main()