Python吃豆豆实验报告

1. 选题

依然是吃豆豆小游戏。

本次游戏主要实现的功能特性有：

1. 四只怪兽对吃豆人坐标的追踪功能。
2. 吃豆人嘴巴的动态开合效果。
3. 吃豆人的方向通过键盘上下左右键控制。
4. 吃豆人、怪兽走过的地方图像的擦除。
5. 所有物体图像均使用像素作画，未贴图
6. 方案

本次设计分为五个文件：

|  |  |
| --- | --- |
| 文件名 | 实现功能 |
| Beans.py  对豆子的处理文件 | 1. 定义了初始路径Path，表示可走的路。   Path=[ ] |
| 2．定义一个豆子的类class SmallBeans，包括函数： |
| （1）def \_\_init\_\_(self,radius,beancolor,covercolor)  初始化函数。  self为类本身的一个对象，用于调用类的变量、函数等。  Beancolor表示豆子的颜色，covercolor表示用于覆盖的颜色。 |
| （2）def DrawSmallBeans(self,screen)  画豆子的函数。  遍历一个31\*23的二维列表，使用pygame库中的pygame.draw.circle()函数画圆。 |
| （3）def Cover(self,screen,x,y,width,hight)  使用pygame.draw.rect（）涂掉多余的豆子 |
| （4）def Coverbeans(self,screen,x0,y0,eatenbeans)  覆盖吃豆人走过的豆子。  x0,y0表示吃豆人之前的坐标，  返回值eatenbeans用来记录被吃掉的豆子数。 |
| 3．设置吃豆人吃完豆子，游戏结束的函数  def PrintBeans(screen,x0,y0)  用smallbeans.Cover（）函数涂掉多余的豆子。  numeatenbeans=smallbeans.Coverbeans（）表示被覆盖掉的豆子数。  当豆子计数器numeatenbeans等于272，玩家胜利  调用pygame.quit()  exit()  关闭窗口，结束游戏。 |
| Ghost.py  对怪兽的处理文件 | 1．定义怪兽的颜色GhostColor及怪兽的路径GhostPath |
| 1. 定义一个怪兽类 class Ghost，包括函数： |
| （1）def \_\_init\_\_(self,i,j,pointlist,color,directghost)  初始化函数。  i，j用来遍历列表，color表示怪兽颜色，directghost表示怪兽方向。 |
| （2）def DrawGhost(self,screen)  画怪兽。  调用了pygame.draw.polygon(…)画怪兽外部形状，再调用pygame.draw.circle（…）画怪兽的眼睛，根据怪兽移动方向不同，眼睛的朝向也不同。 |
| （3）def GetPosition(self)  根据i，j值得到怪兽坐标 |
| （4）def CoverGhost(self,a0,b0,screen)  覆盖怪兽已经走过的地方  用到pygame.draw.rect（…）函数 |
| （5）def PossibleTurn(self,c,d)  怪兽下一步可走的方向  根据怪兽的路径确定怪兽下一步可能的方向 |
| （6）def Move(self,directghost)  怪兽的移动函数  根据怪兽移动方向确定self.x及self.y相应变化，并返回坐标值。 |
| Puzzle.py  显示迷宫地图  画图文件 | 1．定义全局变量Up、Down、Left、Right四个列表 |
| 2．创建一个长方形类class Rectangle  进行初始化，用于在相应位置调用pygame.draw.rect()画无边框矩形。 |
| 3．创建一个圆形类class Circle:  进行初始化，用于在相应位置用pygame.draw.circle()画无边框圆形 |
| 4．再创建一个圆形类class Circle1  进行初始化，在相应位置调用pygame.draw.circle()画无边框圆形 |
| 5．将迷宫地图显示在窗口中  def PrintPuzzle(screen)  计算迷宫障碍物应该出现的位置，分67个部分，每个部分调用上面的三个类的函数来画矩形或圆形。 |
| Pac.py  对吃豆人的处理文件 | 1．导入Puzzle.py文件  用到了Up、Down、Left、Right四个列表。 |
| 2．创建一个吃豆人的类class Pac，包括函数： |
| （1）def \_\_init\_\_(self,x,y,direct)  初始化函数。 |
| （2）def CreatePacMan(self,screen,i)  创建吃豆人，实现嘴巴的动态效果。  当i=1，画嘴张大的图像，当i=0，画嘴变小的图像。  这里用到一个变量i，结合主函数中的调用，用一个if else语句体实现：当i<=1时，嘴张大，当i=0时，嘴变小。执行完这些后，i+=1。 |
| （3）def Move(self)  吃豆人的移动。  根据吃豆人下一步移动的方向是否在表示上下左右是墙壁的四个列表中来判断是否应该移动，并根据方向判断向何处移动。 |
| （4）def GetPosition(self)  得到吃豆人当前的坐标 |
| （5）def Turn(self)  键盘响应函数，按下上下左右键控制吃豆人移动方向。  调用了pygame.key.get\_pressed()函数 |
| （6）def CoverPac(self,x0,y0,screen)  覆盖吃豆人走过的地方。  x0，y0表示吃豆人之前的坐标  在此处调用pygame.draw.rect()用黑色覆盖 |
| 3．def PrintPacman(screen,pac,x,y,direct,i)  吃豆人的方向direct由吃豆人的类中的Turn函数决定。  并将吃豆人移动函数Move的返回值赋给吃豆人当前坐标x，y，实现吃豆人坐标随时更新。 |
| Pacman.py  主函数  调用各部分文件，完成游戏功能。 | 1．导入Beans.py、Ghost.py、Puzzle.py、Pac.py文件  x，y表示吃豆人当前坐标  x0，y0表示吃豆人之前一步的坐标  a1，b1表示怪兽1当前坐标  a01，b01表示怪兽1之前一步的坐标  a2，b2，a02，b02……以此类推  【注】吃豆人的坐标像素点为1，怪兽坐标的像素点为20。 |
| 2．初始化  - 设置吃豆人、怪兽的初始坐标及方向；  - 调用pygame.init()进行游戏初始化；  - 设置窗口大小；  - 调用pygame.event.get()函数捕捉事件，进行游戏是否结束以及是否按下键盘的判断。 |
| 3．实现怪兽对吃豆人坐标的追踪  设置循环，怪兽每走一步就比较一下怪兽与吃豆人横纵坐标的大小，并改变怪兽方向以实现追踪。每走一步都要调用怪兽的类中的函数进行相应处理（移动、覆盖、坐标改变等。）  【注】因四只怪兽的追踪路径相同，故此处设置当两只怪兽的坐标重合时，编号靠后的怪兽坐标变为（1,1）（移动到左上角），这样就避免了两只怪兽一直重合下去。 |
| 4．调用函数画迷宫地图、画豆子、画怪兽、画吃豆人等 |
| 5．判断吃豆人与怪兽是否相遇  根据计算两者的横纵坐标是否相等来判断是否相遇，若两者相遇，则使用pygame.quit()  exit()  终止游戏，关闭窗口 |
| 6． 调用pygame.display.update()更新画面 |

1. 关键技术

**大致关键技术都跟C语言版吃豆豆差不多，以下只叙述新增的两个功能。**

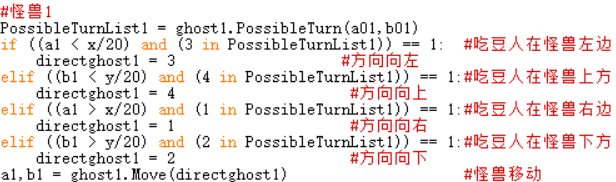
**1**．**四只怪兽对吃豆人的追踪功能**

这是通过比较怪兽与吃豆人的横纵坐标，以此来确定怪兽的移动方向实现的。

以怪兽1为例：

（a1、b1分别为怪兽1的横、纵坐标，以20个像素为单位；

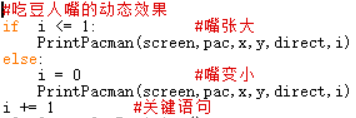
x、y分别为吃豆人的横纵坐标，以一个像素为单位。）



一个小细节是，当两只怪兽的坐标重合时，编号靠后的怪兽会移动到左上角的（1,1）位置，这样做的原因是怪兽的移动算法是相同的，若不作处理两只怪兽就会一直重合下去。

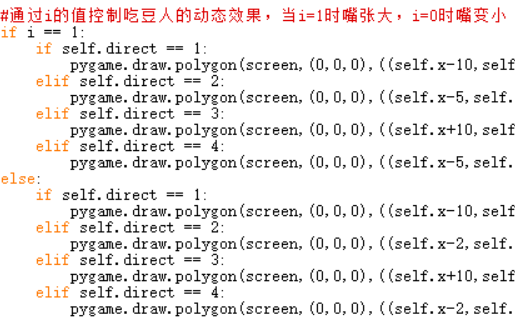
**2**．**吃豆人嘴的动态效果**

这里定义了一个变量i，每执行一个此if else 语句体，i+=1；以此实现i在0与1中转换。



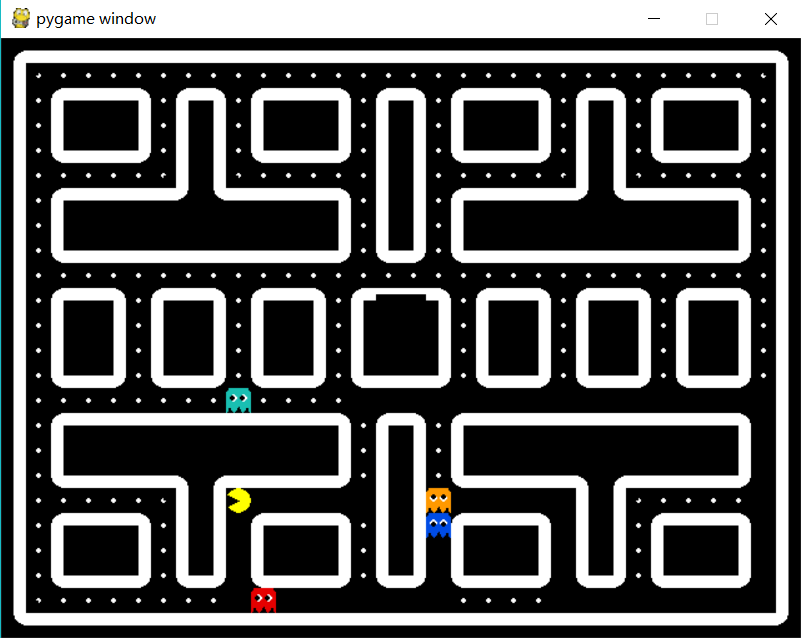
然后在绘制吃豆人的函数中以i的值作为标志，画嘴张开或嘴变小的情况。

通过调用polygon时传入的参数不同来画嘴的大小。



1. 结果和效果

游戏进行界面



张着嘴的黄色的小圆球是吃豆人，其他颜色的是怪兽。

白色的线条是迷宫障碍物。

1. 总结和不足

1．收获

前面几天一直在学Python的语法，后来又学习了用pygame写游戏的流程等，学到了很多知识，对Python这门语言有了了解。收获很大。

这次游戏设计没有贴图，全是用像素画的，这让我的代码行数大大增加，但同时，我自己已经能熟练运用画图函数rect()、circle()、polygon()了，对它们的参数有了深入的了解。

2．不足

由于自己的时间安排不很合理，花了四五天在学习Python及pygame上，因此游戏中还有一些简单的功能如开始界面、结束界面、暂停及恢复功能、计分功能有待实现。此外，没有创建多线程去实现音效的播放。

我的程序还有一个问题是没有很好地实现数据的封装，导致很多地方都重复使用一大段语句，代码行数有许多可避免的增加，后续会进行改进。

画地图的文件有很多地方过于繁琐，特别简单粗暴地计算、画图。后续还会尝试简化。

3．总结

对比吃豆豆游戏的两种实现方法，C语言和Python，最大的感受在于，Python比C语言好用的地方是，省去了很多不必要的定义、删除、释放等。例如，要画一个物体，C语言还需定义句柄、笔、刷还有初始化，用完之后还要删除句柄，比较麻烦；而Python只需调用pygame的draw函数就可以了，简单方便。

发现编程的乐趣是最大的学习动力。