实习报告

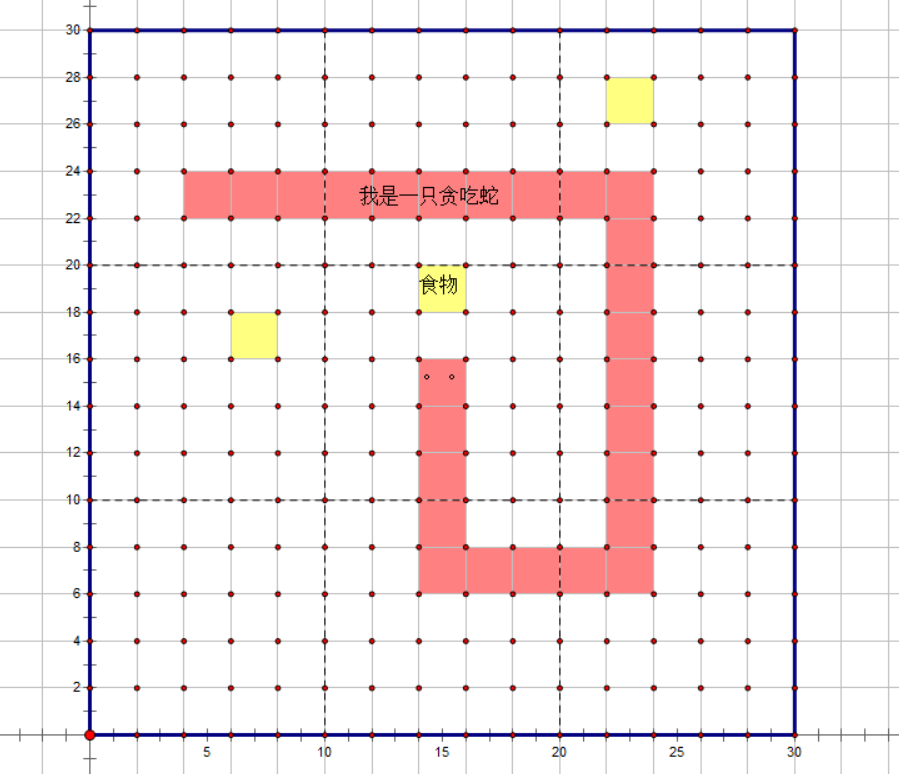
贪吃蛇(microbit 版)

李文睿 物理学院 宋典毅 数学科学学院

摘要：

本作品以经典的贪吃蛇游戏为模板，构建了15X15的游戏区域，设置了三个难度，层层递进。胜利的闯关需要玩家记忆microbit显示的5X5小地图之外的蛇体的状态，还需要玩家具有快速反应的能力。是一个锻炼记忆力与反应力的益智小游戏。

游戏地图示例：



设计方案：

由于贪吃蛇的逻辑较为简单，但是microbit的屏幕只有25个二极管可以使用，为了在有限的显示空间上实现这一款经典游戏，需要对地图和显示空间有更多的安排。本作品采用的是通过局部显示地图的方式实现游戏的显示。通过不同的亮度来分别表示地图的边界，蛇身以及食物。在地图的边界的墙，如果贪吃蛇碰到墙游戏即宣告结束。其余操作均沿用传统贪吃蛇游戏：玩家使用方向键操控一条长长的蛇不断吞下豆子，同时蛇身随着吞下的豆子不断变长，当蛇头撞到蛇身或障壁时游戏结束。

硬件链接：

本作品只需要使用数据线连接电脑即可。在接入电脑后，屏幕显示爱心形状，此时左右两个按键同时按下即可开始游戏。

实现方案：

本作品主要通过矩阵来表示地图，用矩阵元素的值来控制二极管发光的亮度，以表示墙，蛇身和食物。其中墙体最亮，蛇其次，豆子最次。

在游戏开始时玩家即进入难度1，三个难度等级改变的是游戏刷新的帧率，也就是贪吃蛇移动一次所花的时间间隔，难度越高贪吃蛇移动越快，并且不同难度等级获胜条件，也就是需要达到的蛇身长度不同，难度越高贪吃蛇要获胜需要的长度越长。

然后构建turn变量感应左右键来代表改变方向，turn变量是一个模4循环的变量，每次按左右键会+-1，改变蛇头的朝向，如果不按的话蛇就保持原来的方向移动。通过加入一个双端队列来描述蛇身，每次的移动等同于队列头添加，队列的尾部踢出。在挑战成功后屏幕露出笑脸图片并且播放音乐，挑战失败屏幕显示沮丧的表情并且播放音乐。连续挑战成功三个难度将会获得最终的胜利。

以下给出代码

from microbit import \*  
import random  
import music  
  
while True:  
 display.show(Image.HEART)  
 if button\_a.is\_pressed() and button\_b.is\_pressed():  
 break  
  
LEVEL\_list=[1000,700,400]  
win\_list=[20,60,60]  
egg\_list=[50,70,40]  
  
def transformer(matrix):  
 s=''  
 for i in range(5):  
 for j in range(5):  
 s+=str(matrix[i][j])  
 s+=':'  
 return s  
def move(turn):  
 if turn==0:  
 dx=1  
 dy=0  
 elif turn==1:  
 dx=0  
 dy=1  
 elif turn==2:  
 dx=-1  
 dy=0  
 elif turn==3:  
 dx=0  
 dy=-1  
 return dy,dx  
def random\_egg():  
 return [random.randint(1,13),random.randint(1,13)]  
  
class game :  
 def \_\_init\_\_(self,eggs,wins):  
 self.over=False  
 self.body=[[0 for i in range(15)]for j in range(15)]  
 self.body[7][7]=7  
 self.body\_list=[[7,7]]  
 self.egg=[[0 for i in range(15)]for j in range(15)]  
 self.egg\_num=0  
 self.egg\_sum=eggs  
 self.win\_sum=wins  
 while self.egg\_num < self.egg\_sum:  
 l=random\_egg()  
 if self.egg[l[0]][l[1]] == 0:  
 self.egg[l[0]][l[1]]=3  
 self.egg\_num+=1  
 self.egg[7][7]=0  
 self.win=False  
 self.turn=0  
 self.backgrd=[[0 for i in range(15)]for j in range(15)]  
 for i in range(15):  
 self.backgrd[0][i]=9  
 self.backgrd[14][i]=9  
 self.backgrd[i][0]=9  
 self.backgrd[i][14]=9  
 for i in range(15):  
 for j in range(15):  
 self.backgrd[i][j]+=self.body[i][j]+self.egg[i][j]  
 self.showing=[[0 for i in range(5)]for j in range(5)]  
   
 def update(self):  
 if len(self.body\_list)>self.win\_sum:  
 self.win=True  
 return  
 dx,dy=move(self.turn)  
 head=self.body\_list[0]  
 new\_pos=[head[0]+dx,head[1]+dy]  
 if self.backgrd[new\_pos[0]][new\_pos[1]]==9 or self.body[new\_pos[0]][new\_pos[1]]!=0:  
 self.over=True  
 elif self.egg[new\_pos[0]][new\_pos[1]]!=0:  
 self.body\_list.insert(0,new\_pos)  
 self.egg[new\_pos[0]][new\_pos[1]]=0  
 while True:  
 l=random\_egg()  
 if self.egg[l[0]][l[1]] == 0 and self.body[l[0]][l[1]]==0:  
 self.egg[l[0]][l[1]]=3  
 break  
 self.body[new\_pos[0]][new\_pos[1]]=7  
 self.backgrd[new\_pos[0]][new\_pos[1]]=7  
 else:  
 old\_pos=self.body\_list.pop()  
 self.body\_list.insert(0,new\_pos)  
 self.body[new\_pos[0]][new\_pos[1]]=7  
 self.backgrd[new\_pos[0]][new\_pos[1]]=7  
 self.body[old\_pos[0]][old\_pos[1]]=0  
 self.backgrd[old\_pos[0]][old\_pos[1]]=0  
 self.show()  
   
 def show(self):  
 head=self.body\_list[0]  
 n=head[0]//5  
 m=head[1]//5  
 for i in range(5):  
 for j in range(5):  
 self.showing[i][j]=self.backgrd[i+n\*5][j+m\*5]  
 display.show(Image(transformer(self.showing)))  
   
win=False  
for k in range(3):  
 time\_intrv=LEVEL\_list[k]  
 display.scroll('LV')  
 display.scroll(str(k+1))  
 s=game(egg\_list[k],win\_list[k])  
 while s.over==False and s.win==False:  
 t0=running\_time()  
 while running\_time()<t0+time\_intrv:  
 if button\_a.is\_pressed():  
 s.turn-=1  
 if s.turn<0:  
 s.turn=3  
 sleep(80)  
 break  
 elif button\_b.is\_pressed():  
 s.turn+=1  
 if s.turn>3:  
 s.turn=0  
 sleep(80)  
 break  
 else:  
 continue  
 while running\_time()<t0+time\_intrv:  
 sleep(10)  
 s.update()  
 if s.over==True:  
 display.show(Image.SAD)  
 music.play(music.WAWAWAWAA)  
   
 break  
 else:  
 display.show(Image.SMILE)  
 music.play(music.NYAN)  
   
 if k == 2:  
 win=True  
if win :  
 display.scroll('WIN')  
else:  
 display.scroll('LOSE')

后续工作展望：

1976年，Gremlin平台推出了一款经典街机游戏Blockade。游戏中，两名玩家分别控制一个角色在屏幕上移动，所经之处砌起围栏。角色只能向左、右方向90度转弯，游戏目标保证让对方先撞上屏幕或围栏。这一款游戏被认为是贪吃蛇的起源，同时在智能手机上的贪吃蛇大作战也风靡一时。后续可以仿照这些游戏加入两个mirco：bit对战功能，同时借鉴这些规则。可以考虑的方向是当对方的贪吃蛇碰到己方贪吃蛇身体（非头部）即判定为失败。这里需要用到radio方法。

小组分工合作：

李文睿：负责整合游戏功能

宋典毅：负责部分游戏内容的逻辑