**Super Mario in Python World作品实习报告**

作者：吴悦欣 王旌 张启哲 包逸博

【摘要】本小组以马里奥为题材，Python为编程语言，制作了内含5个关卡的过关小游戏，同时，作品内嵌了商城与卡牌收集系统，玩家可以使用在各关卡中收集到的金币在商城中抽取卡牌，作品也具有存档与重置整个游戏的功能，方便玩家反复游玩，提升游戏体验。

1. 选题及创意介绍：

本小组的选题是以马里奥作为主要风格的系列型闯关游戏，主要构想是利用Pygame Zero库及其他内置库实现游戏框架、画面以及与玩家间的交互。项目的创意之处体现在作品内含了5个小关卡，利用外部框架整合后形成了系列闯关型游戏，全作中有致敬经典的部分，也有自己的创新；同时，作品内嵌了商城系统，玩家可利用在小关卡内获得的金币进行抽卡，丰富了游戏体验；作品还在隐藏了部分彩蛋等待玩家探索，并自己创作了许多背景、按钮与动画效果，完善了细节。

1. 设计方案：
2. 外部框架：

游戏通过开始按钮进入主界面，在主界面展示了各关卡的通关情况，以及背包、商城的入口、重置游戏、查看帮助以及退出游戏的按钮。作为闯关式游戏，玩家需按照关卡的顺序依次通过，因此在最开始的游戏界面玩家只能进入第一关，其余关卡随着之前关卡的通关逐步开放。在5个小关卡中，包括了3个主关卡与2个奖励关卡，以主界面地图的主线和支线方式呈现。游戏中玩家在各关卡中获得的金币可累计，并用于在商城中抽取奖励（马里奥卡牌），抽到的奖励可在背包中查看。游戏在各关卡及抽卡后有自动存档功能，确保玩家在本地重新打开游戏时仍保留之前的游戏进度，玩家也可以选择点击主界面的Reset按钮重置整个游戏的进度与金币数。

1. 第一关——Mario\_VS\_Lakitu（简称MVL）:

玩家通过键盘的向左、向右、Z键控制马里奥的移动，来应对Lakitu（刺猬云）的两种不同攻击方式：扔板栗仔和扔刺球。Lakitu的平均攻击速度会随剩余时间的减少而小幅度上升。界面会由上至下随机掉落金币供玩家收集。游戏中每过一定时间并当马里奥受伤时会掉落蘑菇，拾取蘑菇可使马里奥恢复。玩家坚持2min即可过关。

1. 第二关——迷宫：

程序随机生成迷宫，保证一定有从起点到终点的通路。玩家通过键盘的上下左右键控制人物移动。游戏界面每次仅显示迷宫的四分之一，且在此基础上进一步增加局部可见的效果来提升游戏的难度和趣味。迷宫中随机放置一定数量的金币供玩家收集。玩家在3min内到达终点即可过关。

1. 第三关——Boss战：

玩家可通过各方向键操作马里奥躲避Boss的攻击并通过X键攻击Boss，Boss初始血量为100，以随机的速度在岩浆上运动，且有两种攻击方式：喷射斜向运动的火焰或会被岩浆反弹一次的火球。马里奥共有5条生命来对Boss造成伤害，此间Boss的平均移动速度会随剩余血量的降低而上升，击败Boss即可过关。

1. 奖励关卡一——音游模拟：

程序随机生成带有不同方向键的方块或随机道具，玩家通过上下左右与空格键拾取对应方块或道具，不同的道具有不同的加成效果。游戏中物品的刷新间隔、密集程度和下落速度均会随分数增加而增加，开始时玩家共有3条生命，在漏击方块使生命减为0前尽可能获取高分，最终得分将会以500:1的方式折算为金币作为奖励。

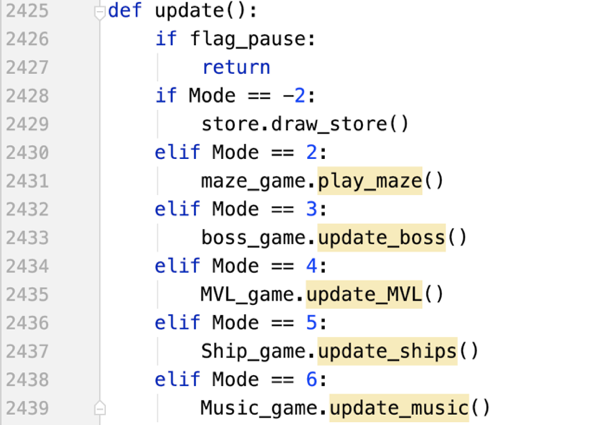
1. 奖励关卡二——炸船：

程序在10\*10的海面上随机生成5艘敌舰，玩家需要在40步之内通过鼠标点击摧毁隐藏的敌舰，每次点击投弹后均会有命中反馈（击中不消耗步数），玩家需要根据反馈和海面情况综合评估局势，尽快确定敌舰位置。击沉全部敌舰即获胜，玩家可在获胜后利用剩余步数探索随机出现的隐藏金币，最终所得金币作为奖励赠与玩家。

1. 实现方案及代码分析：

1. 外部框架：

程序运行后首先会调用read函数进行读档操作，读取之前的通关状态、收集到的卡片和累积的积分，并将五个游戏的类对象都先初始化为None，接下来的代码由Mode变量进行模式的选择。运行pgzrun.go后，程序不断调用draw和update函数；由on\_mouse\_down响应鼠标操作，生成需要的游戏类对象并进行游戏画面和模式之间的切换。Mode及对应的模式如下：1-主菜单，2-迷宫，3-Boss战，4-MVL，5-炸船，6-音游模拟，-1-游戏帮助，-2-商店，-3-背包。draw/update函数会调用对应模式下各个对象内部的draw/update函数，或是直接展示对应模式界面所需元素。





除了这些函数外，外部还有：save函数来实现在每次游戏通关后或是抽卡后保存记录，pause\_menu对应游戏内部的暂停操作，loading\_menu对应主界面的显示，on\_key\_down响应部分键盘操作（包括P键暂停和继续，以及游戏中的部分键盘操作）。

附：存档与读档功能：



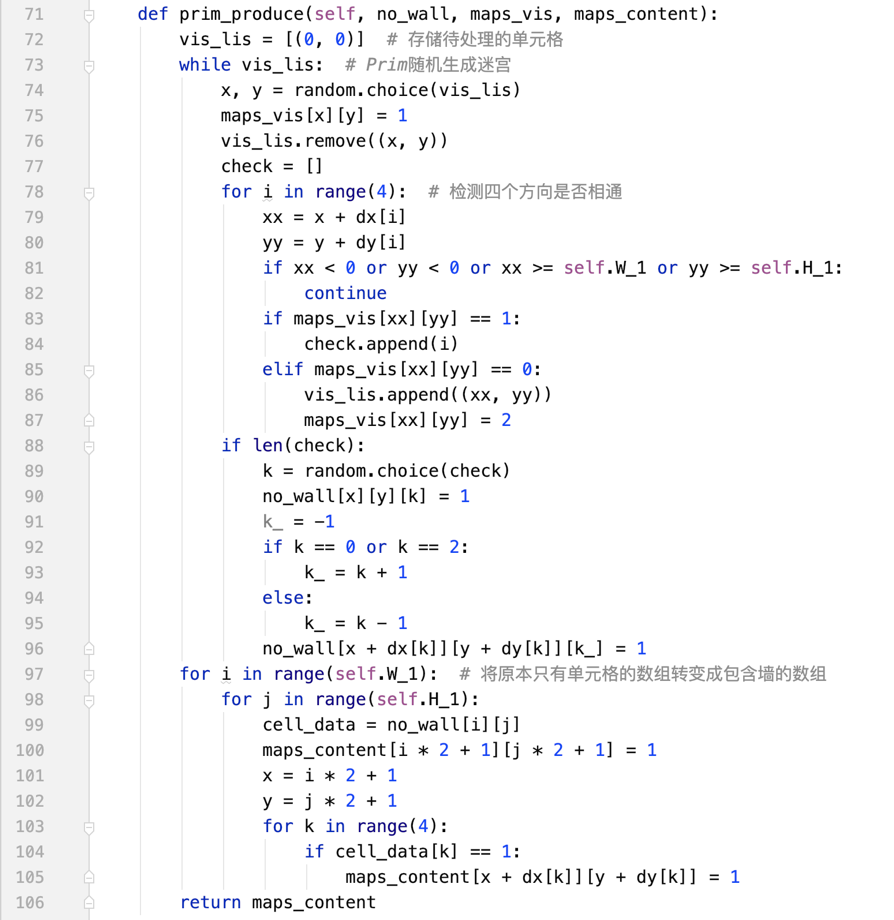
利用基本的文件读写操作，将游戏运行核心的参数（各关卡的通关情况、各卡牌的获取情况、金币数）通过文本文件进行存储，实现游戏的存档与读档功能。

2. 迷宫MAZE类：

* 1. 随机Prim生成算法：

算法核心：初始化时使所有地方都是墙，即不可走通，之后选择一个单元格作为迷宫的通路，将与其相邻的四面墙放入列表中。当列表不为空的时候，不断从中选取一个：若该墙面两侧的单元格只有一个被访问过，则从列表中移除该墙，并使得两边打通；若都已经访问过，则可以直接移除该墙面。

在实际的算法实现过程中，采用了优化策略，将对墙的维护改成对于单元格四周的维护，即在已经能走通的单元格随机选择一面打通。随机Prim算法生成的墙面足够曲折和复杂，能够较好的满足游戏设计中随机生成迷宫的需求。



* 1. 代码结构：

init函数初始化加载规则、人物和遮挡画布的Actor类，并将游戏内部的mode置为1，表示新建游戏后先进入规则界面。外部创建MAZE类的实例后，先调用其成员函数main\_maze对人物的初始状态、游戏的开始时间、游戏积分进行初始化，并在main\_maze中进一步调用prim\_produce成员函数用随机Prim算法生成迷宫，存储在numpy数组中，之后再随机生成金币放入数组。

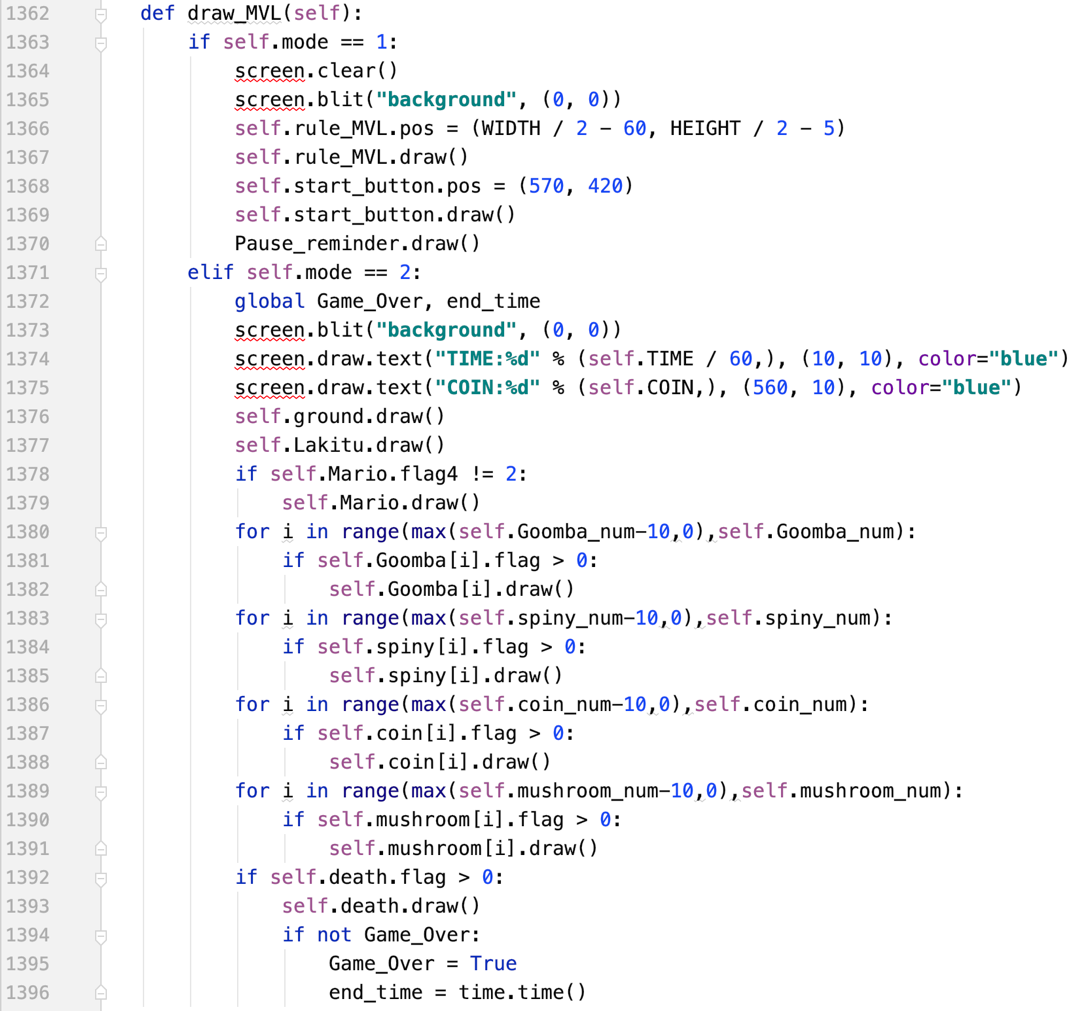
在外部函数draw不断调用draw\_maze成员函数时，根据mode的值确定生成规则界面还是游戏界面，其中游戏界面除人物之外的场景在没有切换1/4迷宫区域的情况下不会重新对地图数组（Actor类的数组）赋值，这一判断在show\_ground函数中实现，节省了时间开销；而外部的update函数持续调用play\_maze的成员函数来感应键盘的操作并判断操作的合法性，进而对人物的行动产生预判。人物之后主要由move\_player函数进行控制并通过animation函数判断微小位移的次数，来实现画面的人物的动画效果。

游戏最终的胜负在draw\_maze中判断，并调用win函数和lose函数来播放对应的音效，游戏的通关记录也将在win函数中进行更新。

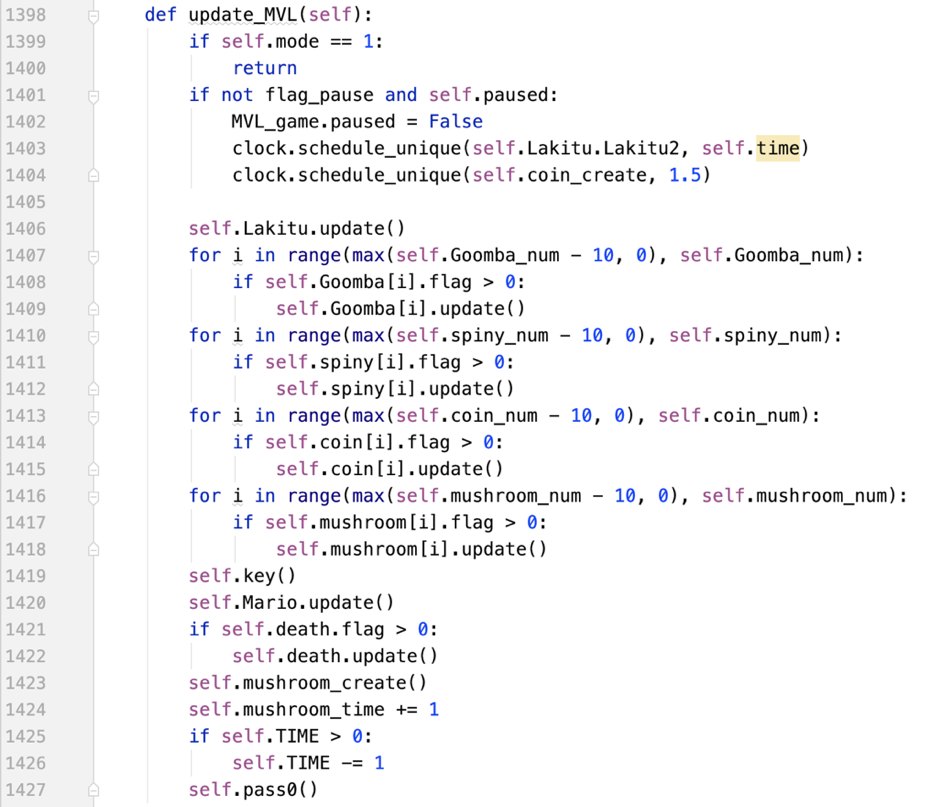
3. MARIO\_VS\_LAKITU类(简称MVL)：

在生成MVL类对象后调用init函数初始化游戏规则界面，在点击START按钮后进入游戏，并调用main\_MVL函数对成员变量进行赋值，其中包括但不限于金币、刺猬云、板栗仔、人物的生成及初始属性的设定。

后续运行过程中，外部的draw函数不断调用draw\_MVL成员函数来显示或消除部分画面元素，并进行人物存货状态的判定。



外部的update函数则在调用update\_MVL成员函数的过程中定时生成金币、板栗仔等对象、响应键盘操作和调用pass0进行游戏过关的判定。



其中，金币等对象由对应的内部类生成，分别为：\_Goomba, \_coin, \_spiny, \_Lakitu, \_Mario, \_mushroom类，以及在人物死亡后出现的\_death类。这些类在各自的初始化中实现类对象属性的初始赋值，并通过函数互相调用的方式实现动画（下图为coin金币动画的实现方式，其他类对象是类似的）



除上述内容外，MVL类中还有的成员函数有：coin\_create控制金币的随机生成和初始属性；mushroom\_create控制蘑菇的定时生成；key函数为update\_MVL所调用来实现键盘的响应；pass0函数进行成功过关的判定。

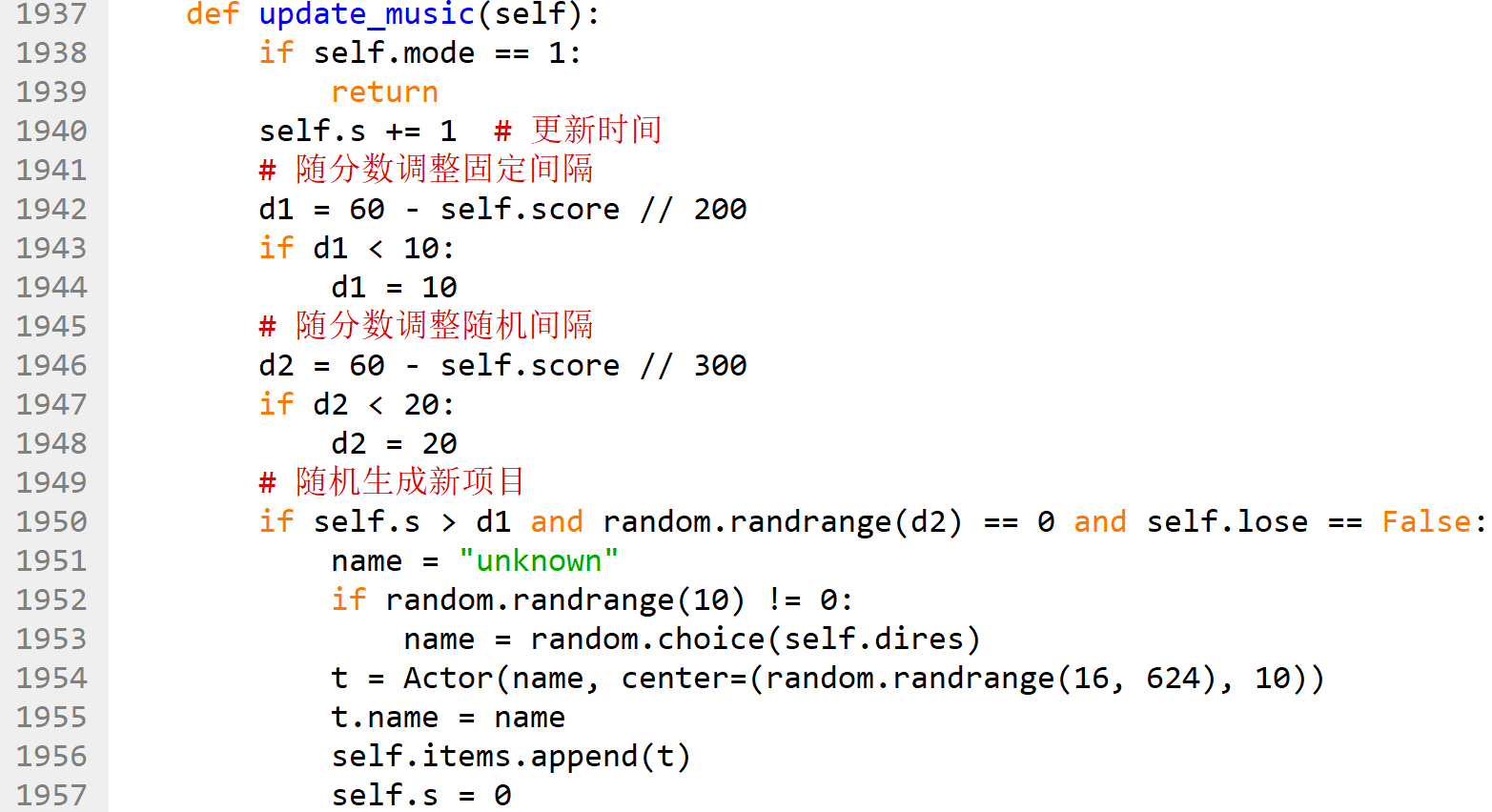
4. BOSS类：

BOSS类的代码结构与MVL类基本一致，也是在初始化后生成各内部类的对象，通过内部类的初始化和函数调用实现包括移动和动画在内的对画面元素的控制。

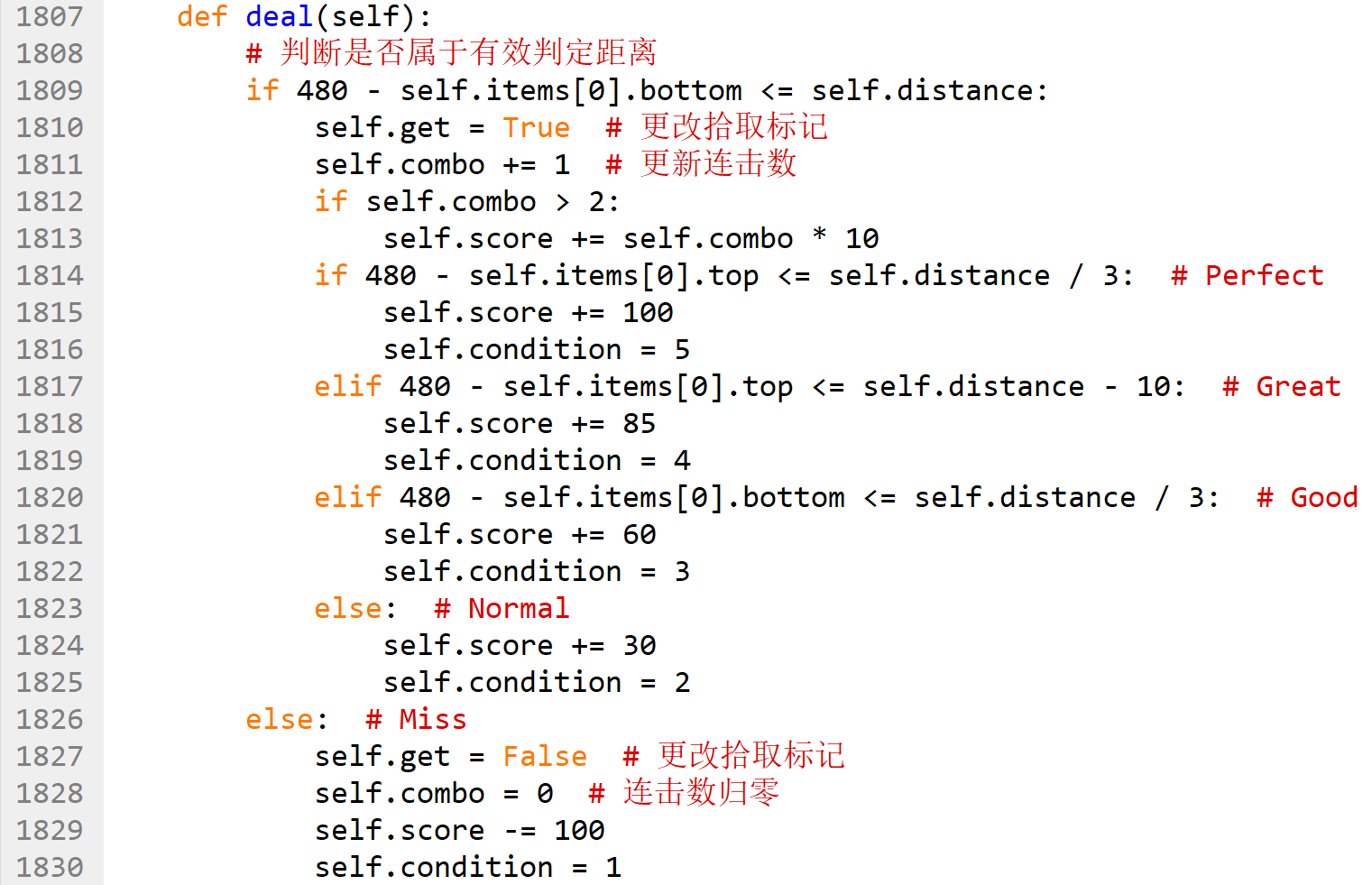
外部的draw / update函数通过调用draw\_boss / update\_boss成员函数实现画面的不断更新。

5. ImitateMusic类:

游戏运行时，根据玩家得分实时更新物品下落间隔、密度与速度，在update\_music函数中随机生成箭头方块或道具。



随后通过on\_key\_down\_music函数捕捉键盘响应，根据最低物品的状态判定响应正误，并更新拾取标记get，连击数combo，玩家得分score与拾取等级condition，更新后交由得分处理函数deal统一处理。



最后由draw\_music函数根据各参数状态显示相应图像与文本。同时，拾取道具触发的效果由计时器定时调用set\_speed\_normal与set\_distance\_normal实现下落速度与判定距离的动态变化。

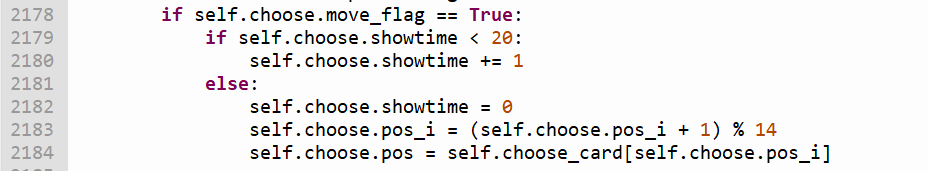
6. BattleShip类:

游戏运行时，首先调用generate\_ships函数，利用随机枚举加循环的方式实现敌舰的随机生成。

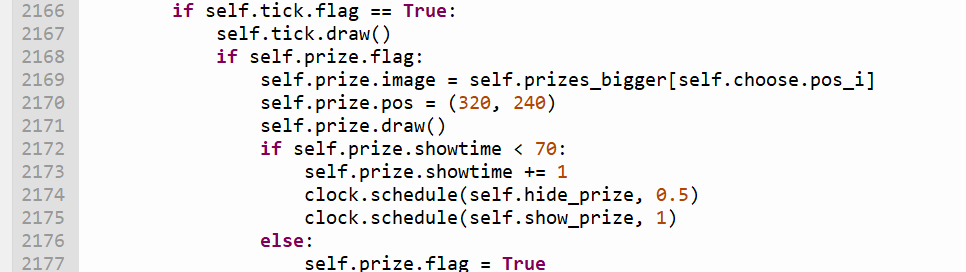
算法主要是利用循环随机选择方向优先填充海面，直至达到敌舰长度，再生成下一艘敌舰，否则更换方向重新填充，实现海面上随机生成预定长度敌舰的效果。

1. STORE类：

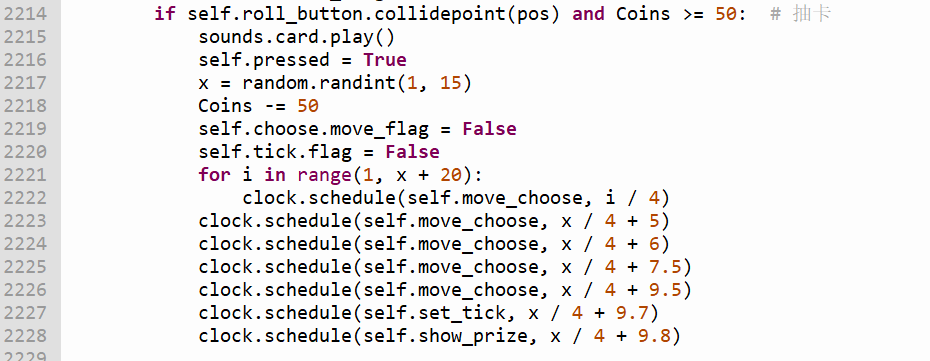
STORE类实现的主要功能是: 1、标记不断转动，显示抽卡的位置2、点击抽卡键后，抽卡并且将抽到的图像闪烁并放在界面正中3、对全局变量金币（coins）的处理



Choose进行标记，当没有抽卡时（move\_flag==True），标记正常转动，一共有14个位置，利用数组choose.pos记录位置，choose.pos\_i记录序号，利用数组的序号不断+1%14。为了使标记移动后有短暂停留，采用了showtime来记录停留时间，进行判断：如果停留时间超过20/60s那么就移动到下一个位置。



先是根据choose.pos\_i来获取抽奖结果，并且用与前面类似的方式控制图像的显示时间，clock函数实现了闪烁的效果



抽卡函数，当点击抽卡按钮且金币>=50的时候可以抽卡，将抽卡状态记录下来（move\_flag=False），使得显示标记进入抽卡状态的移动用多个clock函数实现标记逐渐减速，最终停下的动画效果，同时在函数中记录下抽奖的结果

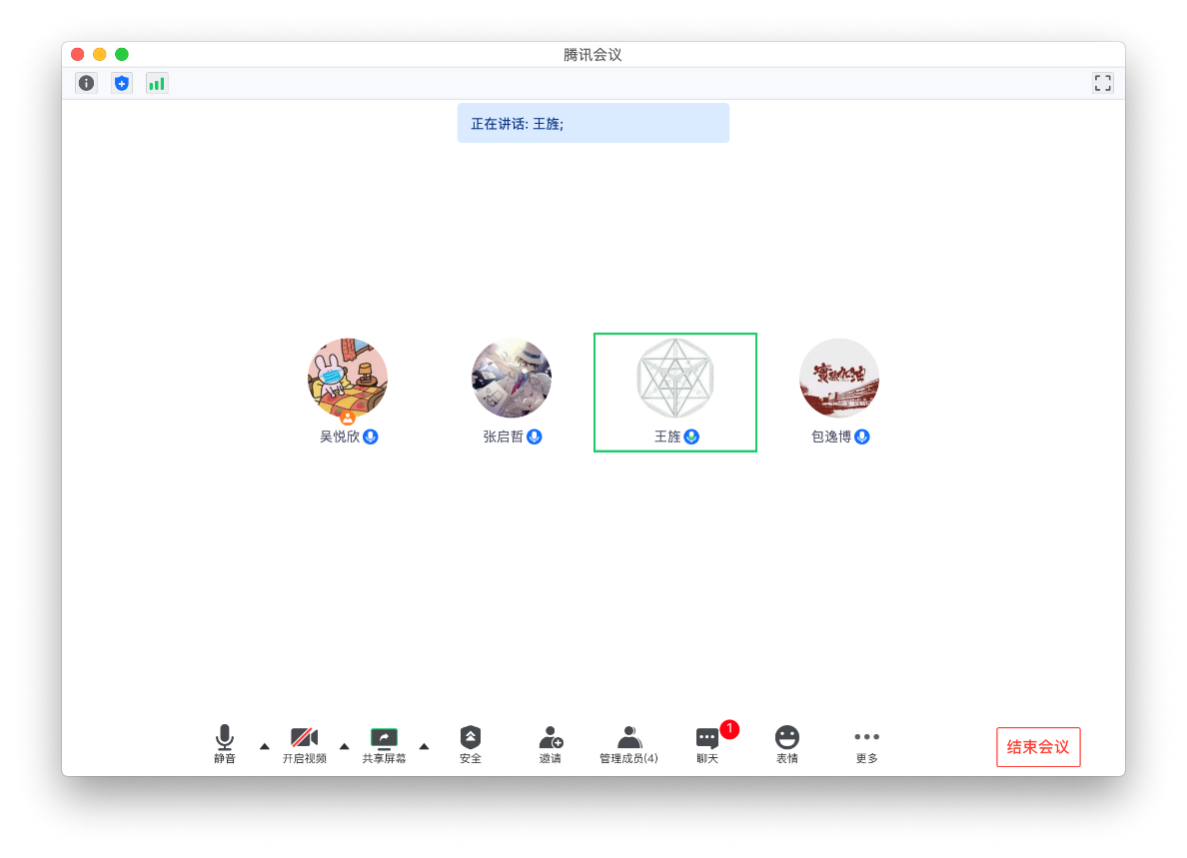
1. 后续工作展望：

该项目后续仍有许多值得扩展改进的方面：内容上，可以增加更多类型的游戏关卡来丰富游戏体验；功能上，可以扩展商店的内容和更多由玩家自定义的部分，如音效与音乐的控制开关、游戏的难度选择等，提升玩家对于整个游戏的掌控性；画面上，可以增添更多的动画效果，增添转场动画、通关动画等，提高游戏的观赏性。

1. 小组分工合作(含讨论照片)：

组长吴悦欣主要负责外部框架与主界面功能设计、迷宫关卡编写、代码改进、部分素材制作、视频剪辑与项目规划；王旌同学主要负责项目图片与音频素材提供、MVL与boss关卡编写，项目海报制作与设计方案的提出；张启哲同学主要负责外部框架设计、两个奖励关卡编写、部分素材制作与代码改进；包逸博同学主要负责商店功能编写、部分素材制作、游戏介绍和帮助撰写与视频剪辑；实习报告的撰写所有成员均有参与。





1. 致谢：

感谢Python语言基础与应用课程中陈斌老师与各位助教的讲解与帮助！