

**基于 DFS 算法的迷宫游戏设计**  
**——期末大作战**

Spico

2017 年 7 月 8 日

# 目录

一、游戏内容及特点描述.....	1
1、游戏背景介绍.....	1
2、游戏玩法及规则介绍.....	1
2.1、大小及材质说明.....	1
2.2、游戏键位定义.....	2
2.3、游戏总体玩法及具体参数.....	2
3、游戏特色.....	8
3.1、产生随机迷宫.....	8
3.2、答题模式.....	8
4、游戏原理说明.....	9
4.1、游戏编程环境.....	9
4.2、pygame.sprite.Sprite 精灵.....	9
4.3、DFS 迷宫产生算法.....	10
4.4、答题过程.....	10
5、游戏特色分析总结.....	10
6、游戏测试及改进方向.....	10
6.1、游戏测试.....	10
6.2、游戏漏洞及待改进部分.....	11
二、游戏流程（流程图）.....	13
1、游戏运行流程.....	13
2、DFS 迷宫产生流程.....	14
四、参考文献.....	15
五、附录.....	16
1、游戏实现界面图.....	16
2、查重情况.....	21
3、游戏源码.....	23

一、游戏内容及特点描述

1、游戏背景介绍

故事发生在一个叫做大数据的王国，这里的人们以知识和技术为尊，人们享受着新兴技术给他们带来的生活便利,安居乐业。而这时，王宫中却传来了国王的叹息声，原来国王唯一的女儿小小在闯关期末考核时发生了意外，被困在知识迷岛里面，国王不但担忧着女儿的安危，更担忧女儿无法继承他的王位，思虑多时，国王最终决定，贴榜昭告天下，凡是能救出公主的，就可以成为王国的驸马。万千勇士收到消息便不及待地踏上了征程，而这当中就有一位名唤木木的少年……

勇士木木来到了知识迷岛，却发现这座岛比他想象的要复杂得多，整座岛是一个非常大的迷宫，道路错综复杂，更麻烦的是重要的关卡还有擅长不同学科的老师把守，要想让老师放路，就只能乖乖的通过老师的考试，通过考试就可以得到迷宫道路中的提示，找到公主，完成游戏。反之，则游戏失败。

2、游戏玩法及规则介绍

2.1、大小及材质说明

游戏幕布大小为 800\*600，道路宽度为 50\*50。游戏男主角名称为木木，为了使其灵活移动，将其大小设为 30\*30，女主角名称为小小，大小为 50\*50。道路中路设考官，大小为 30\*30。具体信息图示见表 1 材质大小说明表、表 2 颜色定义表：

表 1 材质大小说明表

名称	大小	图示
木木	30*30	
小小	50*50	
考官	30*30	
道路	50*50	
墙体	50*6	


游戏主窗口	800*600	
-------	---------	--

表 2 颜色定义表

名称	RGB 定义
道路	(239, 240, 220)
墙体	(0, 0, 0)
红色提示文字	(235, 63, 47)

## 2.2、游戏键位定义

游戏中在不同的界面有着不同的键位说明，详见表 3 游戏键位定义表。

表 3 游戏键位定义表

场合	名称	键位
游戏主界面	上移	↑
	下移	↓
	左移	←
	右移	→
	退出游戏	ESC
答题界面	A 选项	a
	B 选项	b
	C 选项	c
	D 选项	d
	跳过此问题	ESC
结束界面	退出游戏	ESC

## 2.3、游戏总体玩法及具体参数

打开游戏，首先进行游戏背景介绍，因为篇幅原因，将背景分为两个幕布分

别说明，幕布 1 持续时长为 3 秒，幕布 2 持续时长为 4 秒。此阶段响起背景音乐并暂不可进行窗口拖动。幕布 1 如图 1 幕布 1 所示，幕布 2 如图 2 幕布 2 所示。

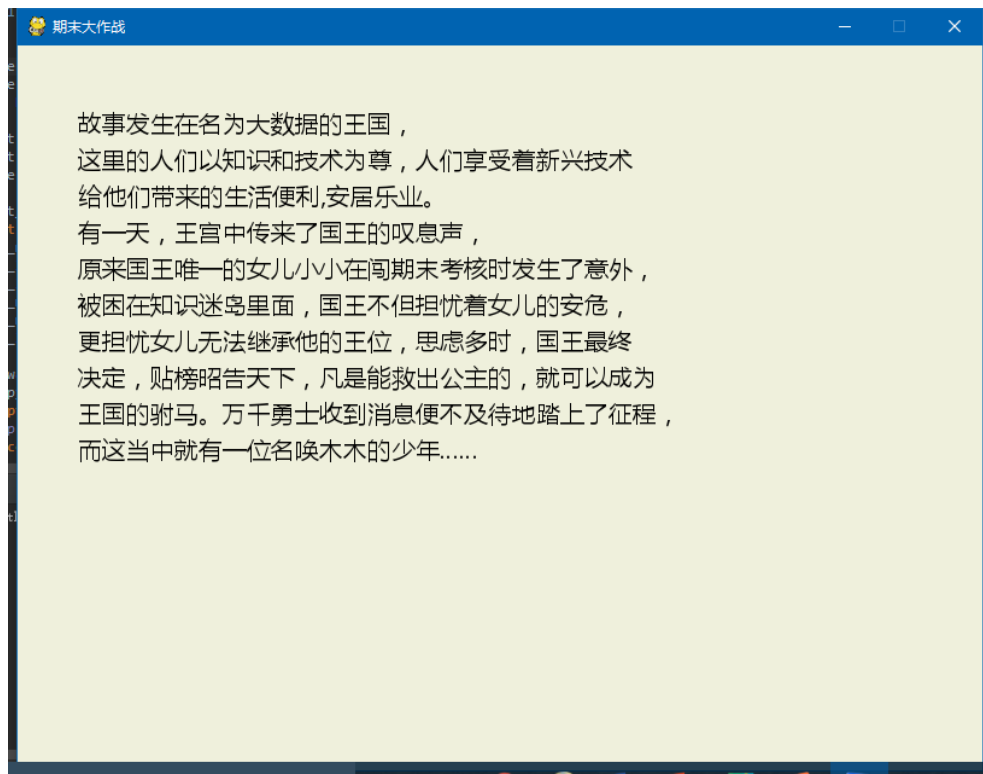


图 1 幕布 1

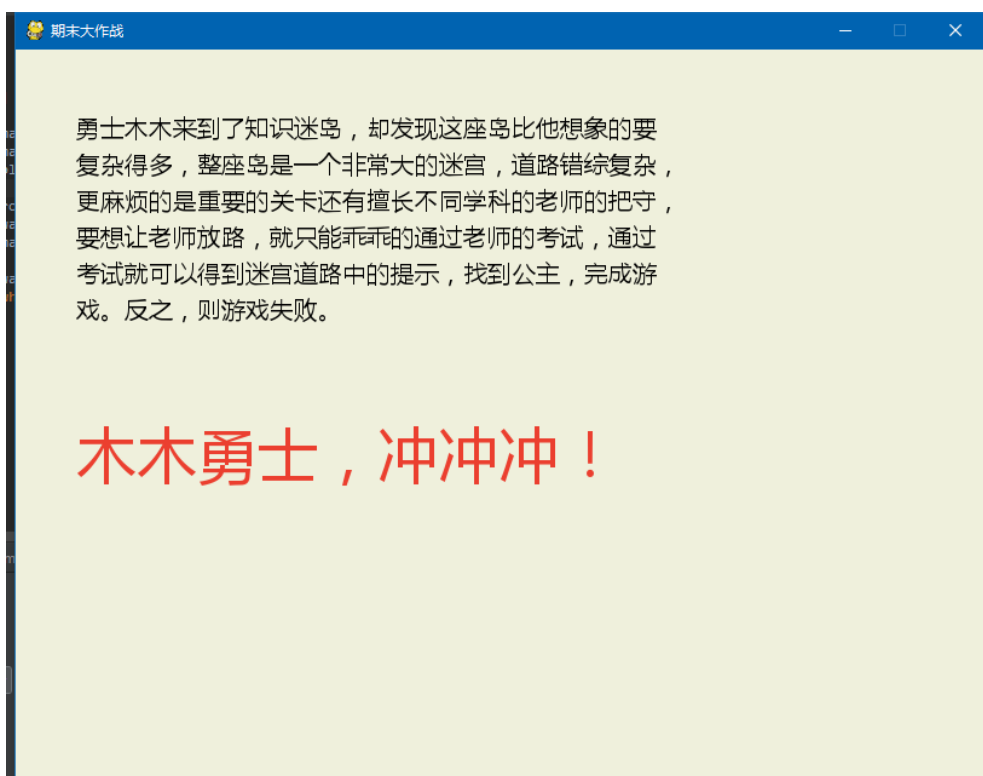


图 2 幕布 2

背景介绍之后自动跳入主界面。

主界面左上角显示当前血量。进入主界面后根据方向键移动木木战士，每次移动的像素点为 3，刷新频率为 60 帧/s。游戏墙体不可穿越，游戏迷宫随机生成，不重复不固定，矩阵大小为 13\*17。主界面如图 3 主界面所示。

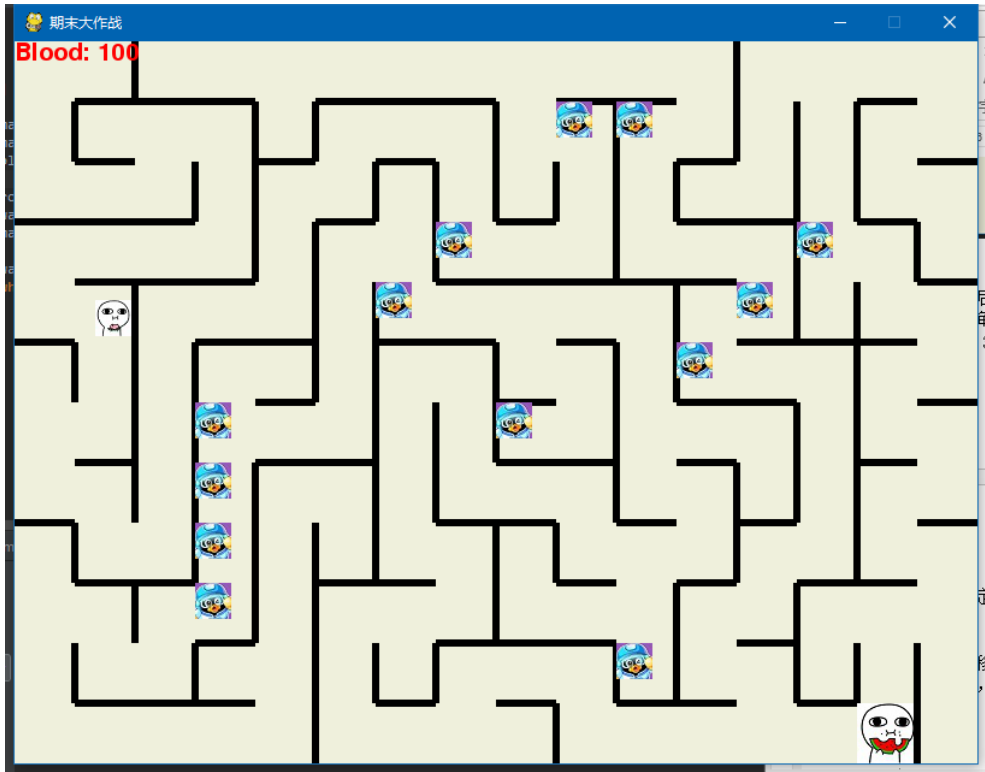


图 3 主界面

在主界面中移动木木，在每格道路的左上角设有考官，与考官发生碰撞接触即触发答题界面，答题环节不设时长限制，键位说明参照表 3 游戏键位定义表，若输入其余键位，则默认无效。答题界面如图 4 答题界面所示。

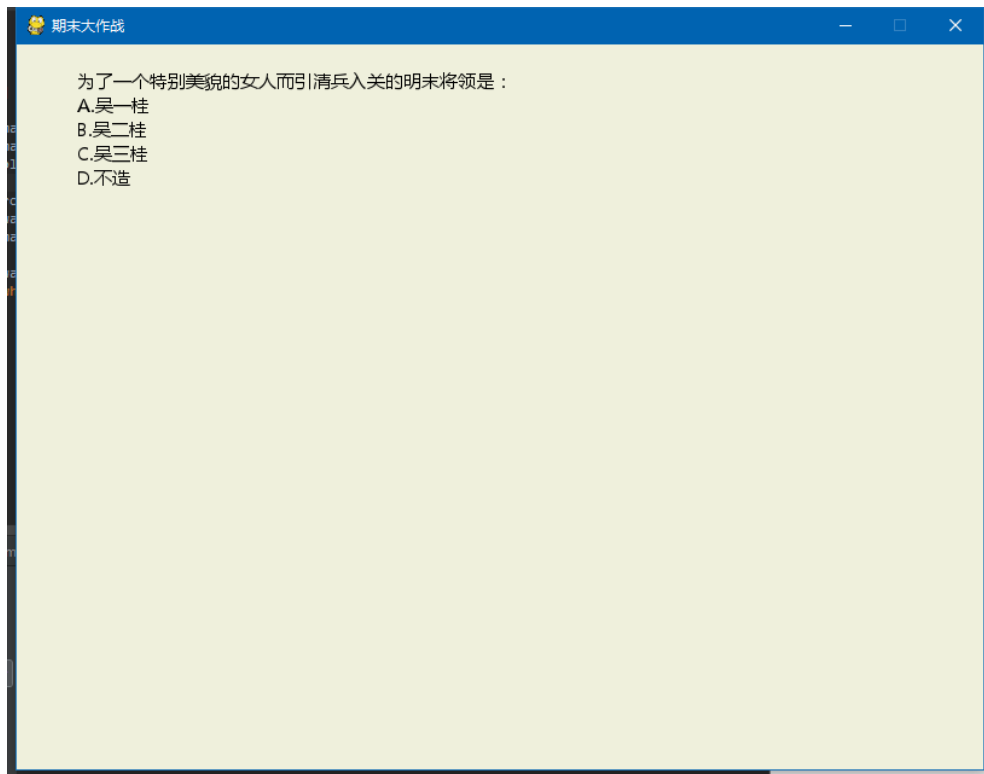


图 4 答题界面

答题共有两种结果，分别为答错与答对。

答错后会如图 5 答题错误提示显示“你个渣渣”“BLOOD -20~”等字样。并在此引入减血机制。退出之后会在主界面左上角看到自己的血量减少 20 如图 6 答题错误后血量变化。答对后会如图 7 答题成功图显示“You’re right~”字样，无奖励，不减血或加血。

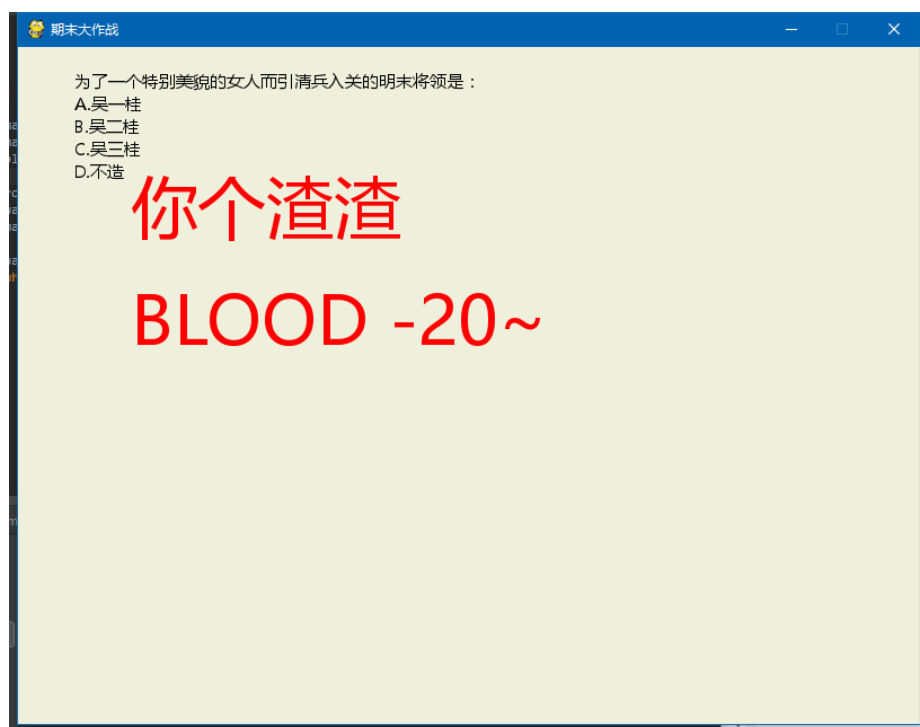


图 5 答题错误提示

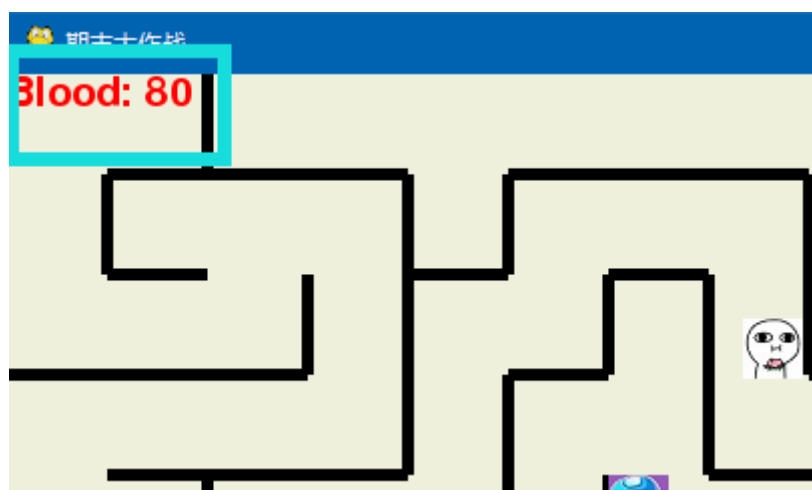


图 6 答题错误后血量变化



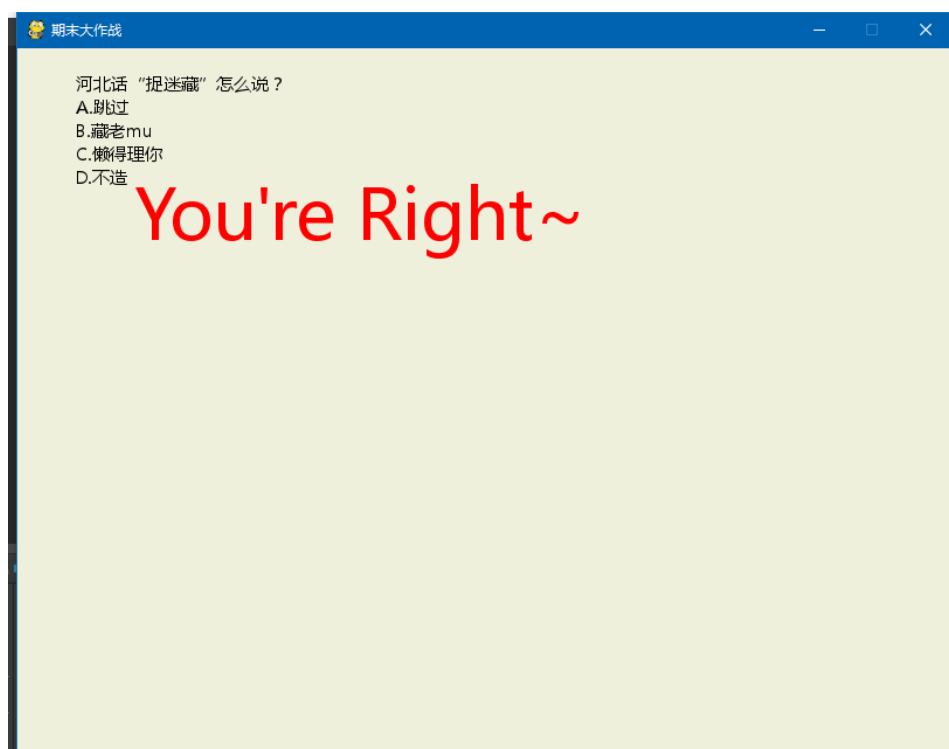


图 7 答题成功图

当答错 5 题，即血量变为 0 时，进入失败提示界面如图图 8 失败显示界面，播放失败音乐。音乐列表如表 4 游戏音乐信息表所示。

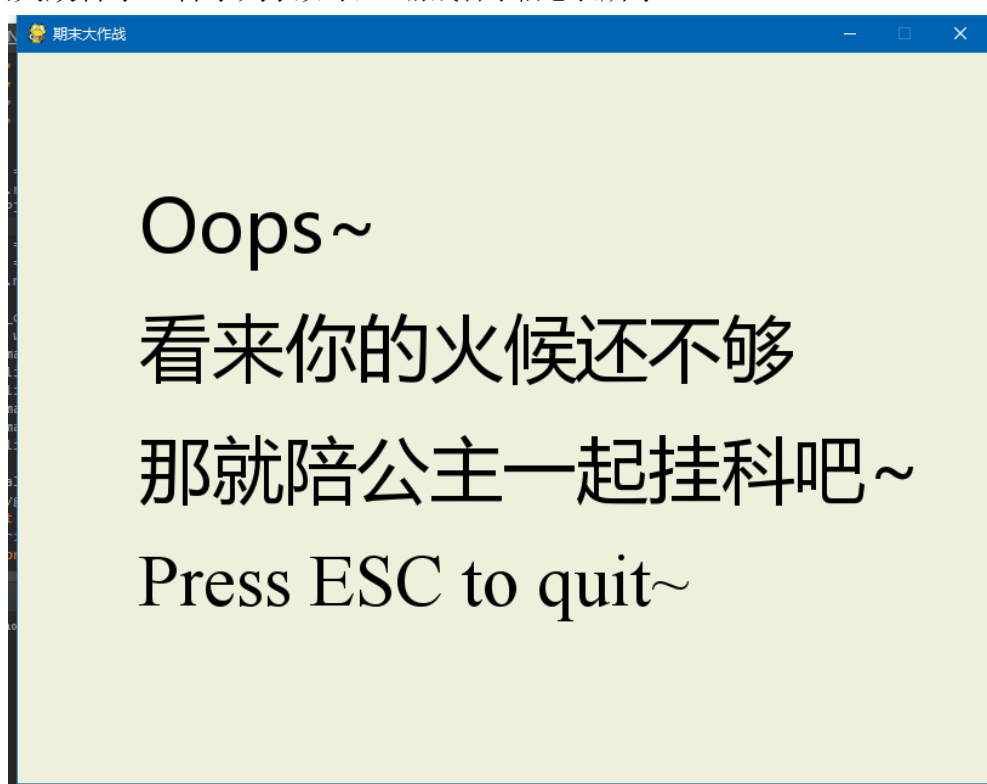


图 8 失败显示界面

表 4 游戏音乐信息表

文件名	原名称	场合
bgm_zalababa.mp3	짜라빠빠	游戏主背景音乐
victory.ogg	O sole mio	游戏胜利
lose.mp3	超级玛丽游戏失败音乐	游戏失败

若答题成功，则提示 “You’re right~”

若血量大于 0 且木木到达小小处，则游戏成功，播放游戏胜利音乐，并显示游戏胜利界面如图 9 游戏胜利界面所示。



图 9 游戏胜利界面

### 3、游戏特色

#### 3.1、产生随机迷宫

从 13\*17 的矩阵开始由 DFS 算法（深度优先搜索算法）进行遍历寻找可通向出口的路径，并由 random 模块<sup>[1]</sup>内的 random.randrange()方法产生随机数，选择 DFS 之中的深入路径。具体流程见算法流程图。此算法可保证每次产生不同的迷宫，并保证有通向小小的通路。

#### 3.2、答题模式

由 openpyxl 模块<sup>[2]</sup>内的 load\_workbook()方法导入 excel (.xlsx) 文件，并从中提取题目、选项以及正确答案等信息，为游戏的二次定制开发开辟了 API。

在与考官发生碰撞时，利用上述 random.randrange()方法随机抽选题目，

但目前并未创建禁忌表(Taboo List)，故会出现不同考官出同一道题目的现象。

## 4、游戏原理说明

### 4.1、游戏编程环境

本游戏利用 Python 语言编写，编译环境 Python 3.5.2，编译器 cPython.构建游戏编程环境时还运用了 PyGame、math、random、openpyxl.load\_workbook 等第三方库及模块，本游戏窗口由 pygame 库直接生成，不经过 PyQt、tkinter 等图形窗口产生模块，故不具备窗口提示及消息窗提示功能。若有提示需要，如答题模块，则直接利用幕布进行覆盖。

### 4.2、pygame.sprite.Sprite 精灵

PyGame 是基于 Python 的第三方库，它具有良好的社区支持，多核 CPU 运算支持，跨平台应用等多种优良特性，且不需要 OpenGL 支持<sup>[3][4]</sup>。加之 Python 胶水语言的特点，可在游戏中方便地加入许多特性。

利用 OOP 程序设计思想，为了方便对迷宫图之中的各个部分进行操作，利用了 pygame 的 sprite 精灵模块及精灵组 (sprite.Group())<sup>[5]</sup>，可对各个部分进行批量、单独删改。下面表 5 类定义之中具体介绍游戏之中的类。

表 5 类定义

类	继承关系	定义说明	说明		
Maze	Object	迷宫	属性	neighbor	相邻元素
				visited	DFS 中已标记的元素
				unvisited	DFS 中未被标记的元素
				wall_table	墙体 Sprite Group
				road_mat	迷宫矩阵
			方法	__init__(self)	构造函数
				isexit(self, pos_now)	出口位置比对判断
				dfs_maze_generate(self, start_pos, wall_list)	DFS 算法对墙体进行删除处理
				neighbor_select(self, pos_now)	DFS 算法中挑选相邻元素
Road	pygame.sprite.Sprite	道路	方法	wall_break(self, pos1, pos2, wall_list)	DFS 算法中删除相邻元素之间的墙体
				__init__(self, x, y)	构造函数，产生 (x, y) 位置上的道路块
Wall	pygame.sprite.Sprite	墙体	方法	__init__(self, x, y, mode)	在(x, y)位置产生墙体，mode 限定墙体形状

Teacher	pygame.sprite.Sprite	考官	方法	__init__(self)	载入图像
Player	pygame.sprite.Sprite	玩家	属性	change_x	X 方向上的速度
				change_y	Y 方向上的速度
			方法	__init__(self, x, y)	构造函数, 产生 (x, y) 位置上的玩家
				changespeed(self, x, y)	改变速度
				move(self, walls)	改变位置

另外, 还对游戏之中的各个阶段的显示进行了函数封装, 具体代码见附录。

### 4.3、DFS 迷宫产生算法

利用深度优先搜索算法对迷宫矩阵进行处理, 具体算法过程描述如下:

①创建 visited 和 unvisited 列表进行标记, 当 unvisited 列表为空时, 算法结束。另外, 还需创建 current\_stack 栈, 保存当前位置方便后期回溯;

②令(0, 1)位置为迷宫起点, 入栈并标记;

③当 unvisited 不为空时, 若存在未被标记的邻格, 则随机选择其中的一个, 入栈、拆墙、标记并设邻格为当前格; 当没有空闲邻格时, 若栈不为空, 则使栈顶元素出栈并设其为当前格, 循环重复。若栈空了, 则从 unvisited 之中随机选择一个作为当前格, 重复<sup>[6]</sup>。

具体流程图与算法代码见游戏流程及附录

### 4.4、答题过程

由于数据库开发的门槛较高, 对普通开发者来说难度较大, 故选取 Excel 表格作为数据存储位置。利用 openpyxl 模块导入数据表, 并使用 random 模块进行随机抽题, 自带检测功能, 达到测试效果。

## 5、游戏特色分析总结

- ①游戏玩法简单, 入手容易;
- ②通过闯关, 在娱乐的同时巩固大家的专业基本常识;
- ③错综复杂的迷宫, 生动有趣的剧情;
- ④打破传统的老鼠走迷宫设定, 结合校园学习生活, 增添趣味性。

## 6、游戏测试及改进方向

### 6.1、游戏测试

为了探求游戏运行性能, 对游戏进行了 10 次的运行测试, 测试环境: 具体运行参数见表 6 游戏测试信息表、图 10 游戏测试情况信息图。

表 6 游戏测试信息表

序号	游戏开始时		游戏稳定后	
	CPU 占用率(%)	内存占用情况(MB)	CPU 占用率(%)	内存占用情况(MB)
1	4.2	20	6.9	41.5
2	8.4	41.4	7	41
3	4.5	41.5	6.6	41.1
4	5.9	41.9	6.3	41.5
5	3.5	40.9	6.8	40.5
6	5.2	41.3	5.5	40.9
7	6.8	41.6	6.8	41.1
8	5.5	41.5	6.8	41
9	8.9	41.9	6.8	41.4
10	4.5	41.7	8	41.4
平均值	5.74	39.37	6.75	41.14

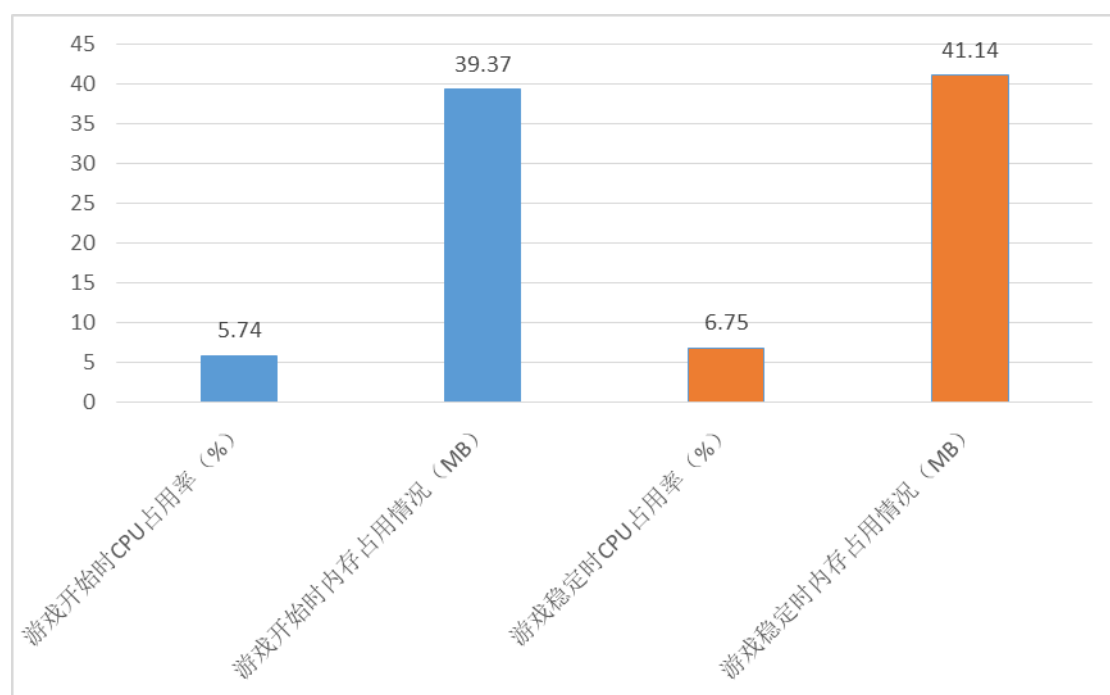


图 10 游戏测试情况信息图

## 6.2、游戏漏洞及待改进部分

经过数次测试，得到游戏 BUG 缺陷及相关可优化问题如下：

①由于游戏优化不当，加之迷宫计算及玩家状态的循环判断，导致 CPU 占用率过高。可通过降低帧率、优化算法加以改进。

②由于没有通过提示窗口产生提示信息，致使在进入答题界面时存在着玩家移动的按键释放问题，答题结束后可能会看到木木战士贴着一边走，无法进行行动控制的问题。

③由于对 pygame 和 random 第三方包的掌握度不够，在加入答题禁忌表时

会出现游戏运行卡死的情况。故在此游戏中暂不加入禁忌表设置，即答题时有可能会抽到相同题目。

④由于对 OOP 程序设计及软件开发的思想领悟并不透彻，致使游戏代码不能做到真正的高内聚、低耦合。即模块与模块之间的耦合度略高。

## 二、游戏流程（流程图）

### 1、游戏运行流程

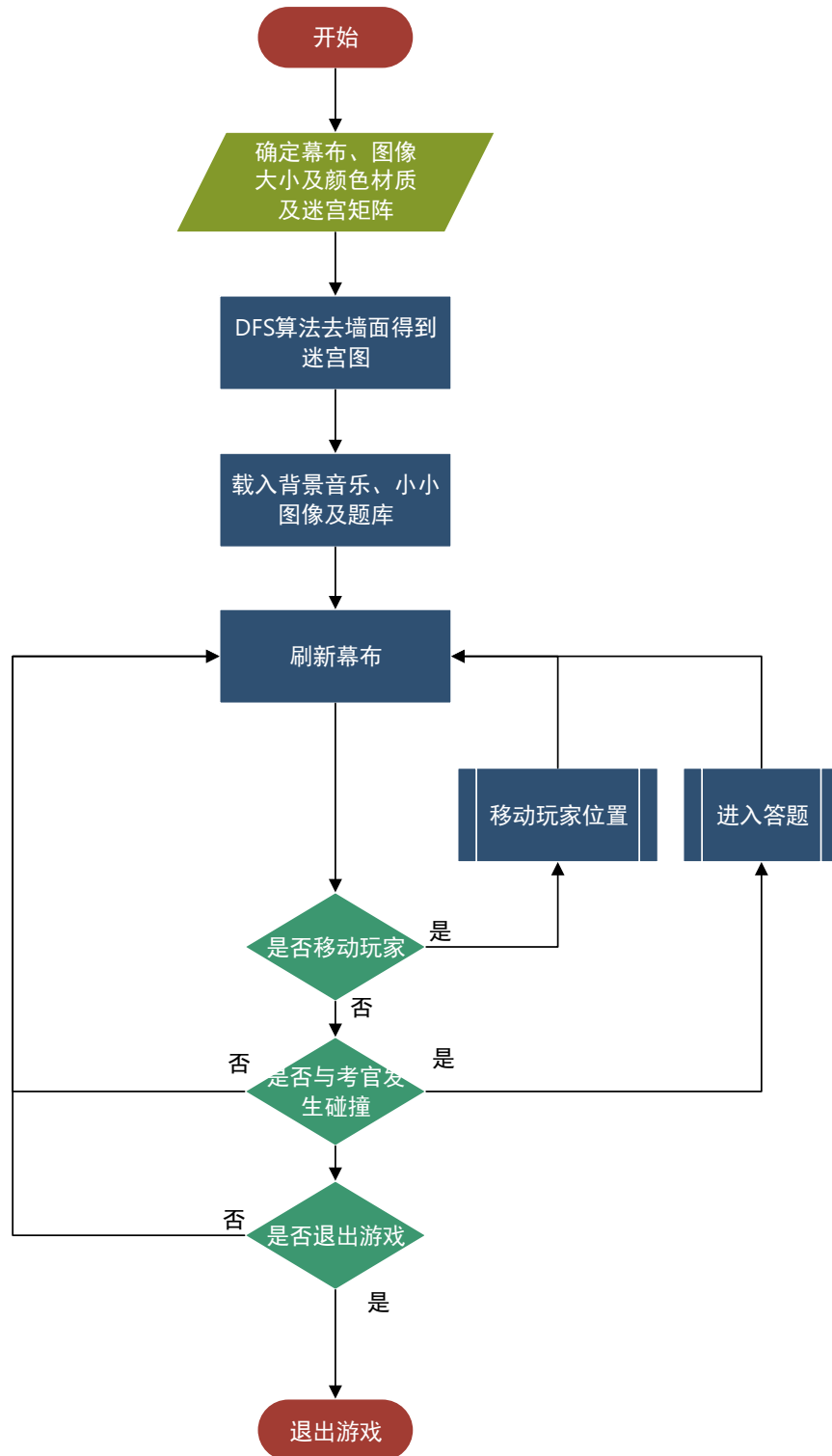


图 11 游戏运行流程图

## 2、DFS 迷宫产生流程

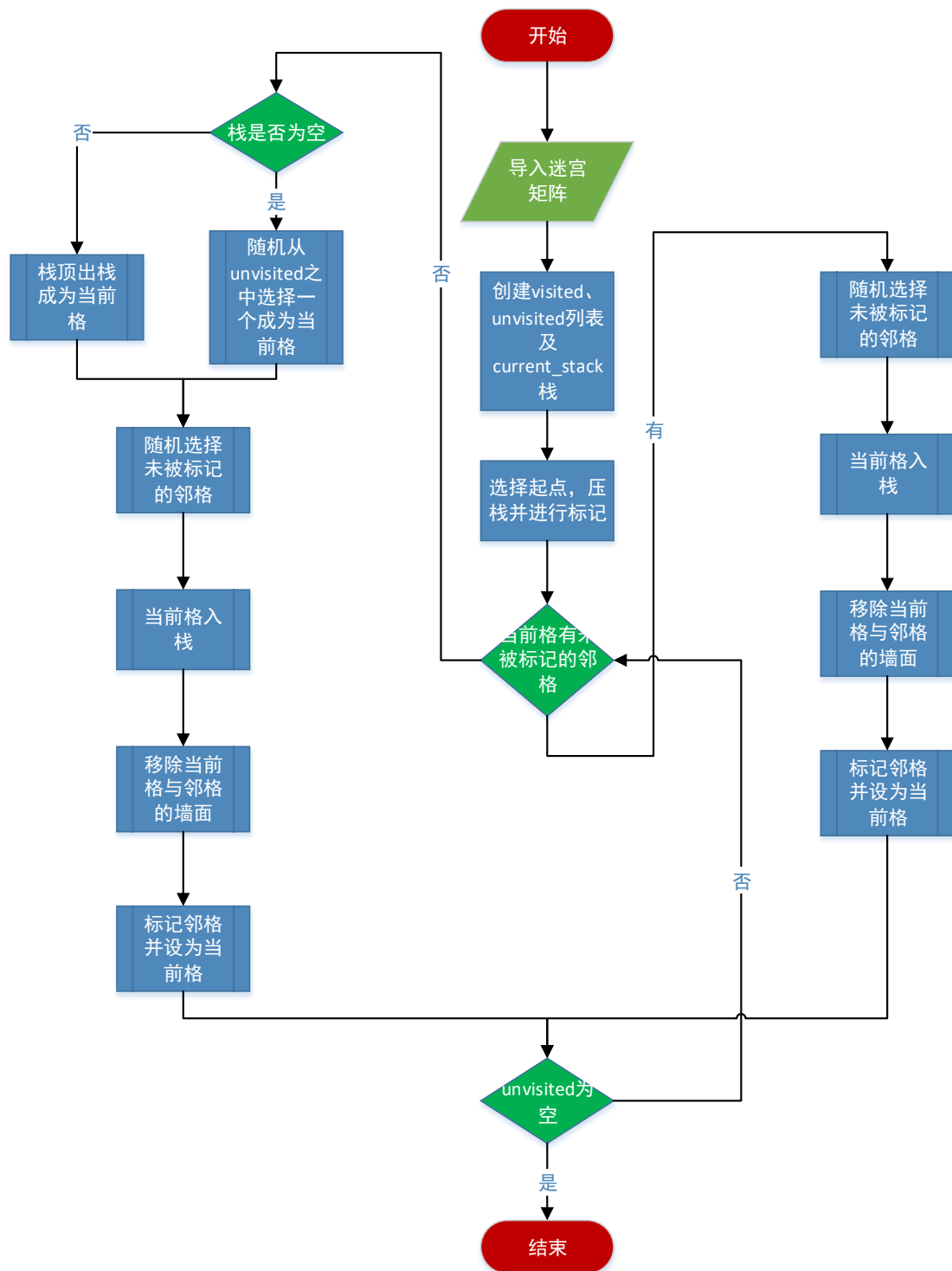


图 12 DFS 算法流程图



### 三、参考文献

- [1] Capricorn.Python 中的 random 模块[EB/OL].  
<http://cnblogs.com/yd1227/archive/2011/03/18/1988015.html>, 2011
- [2] Openpyxl.openpyxl docs[EB/OL]. <https://openpyxl.readthedocs.io/en/default>,  
2017
- [3] PyGame.pygame[EB/OL]. <http://www.pygame.org>, 2017
- [4] PyGame.PyGame About[EB/OL]. <http://www.pygame.org/wiki/about>, 2017
- [5] PyGame.pygame.sprite[EB/OL]. <http://www.pygame.org/docs/ref/sprite.html>,  
2017
- [6] Cosven.Play maze with python and pygame[EB/OL].  
<https://my.oschina.net/zjuysw/blog/496942>, 2015

## 四、附录

### 1、游戏实现界面图

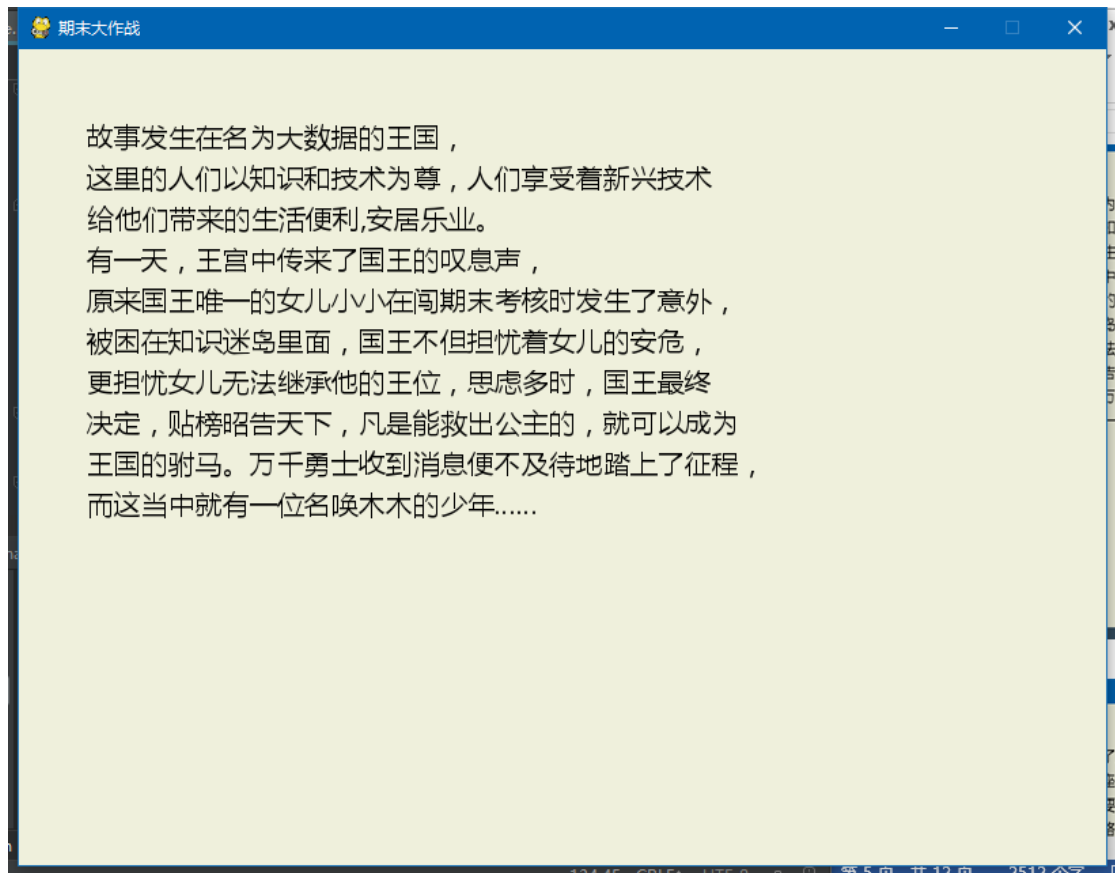


图 13 开始幕布 1

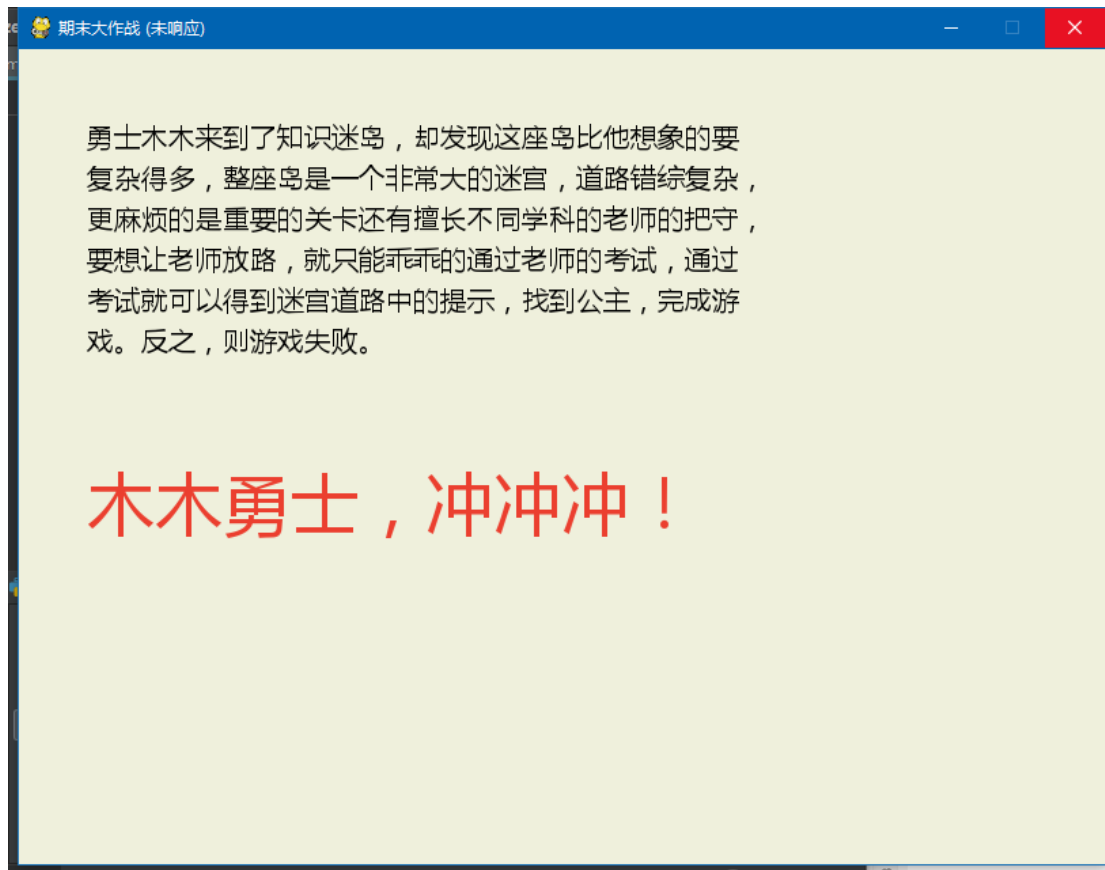


图 14 开始幕布 2

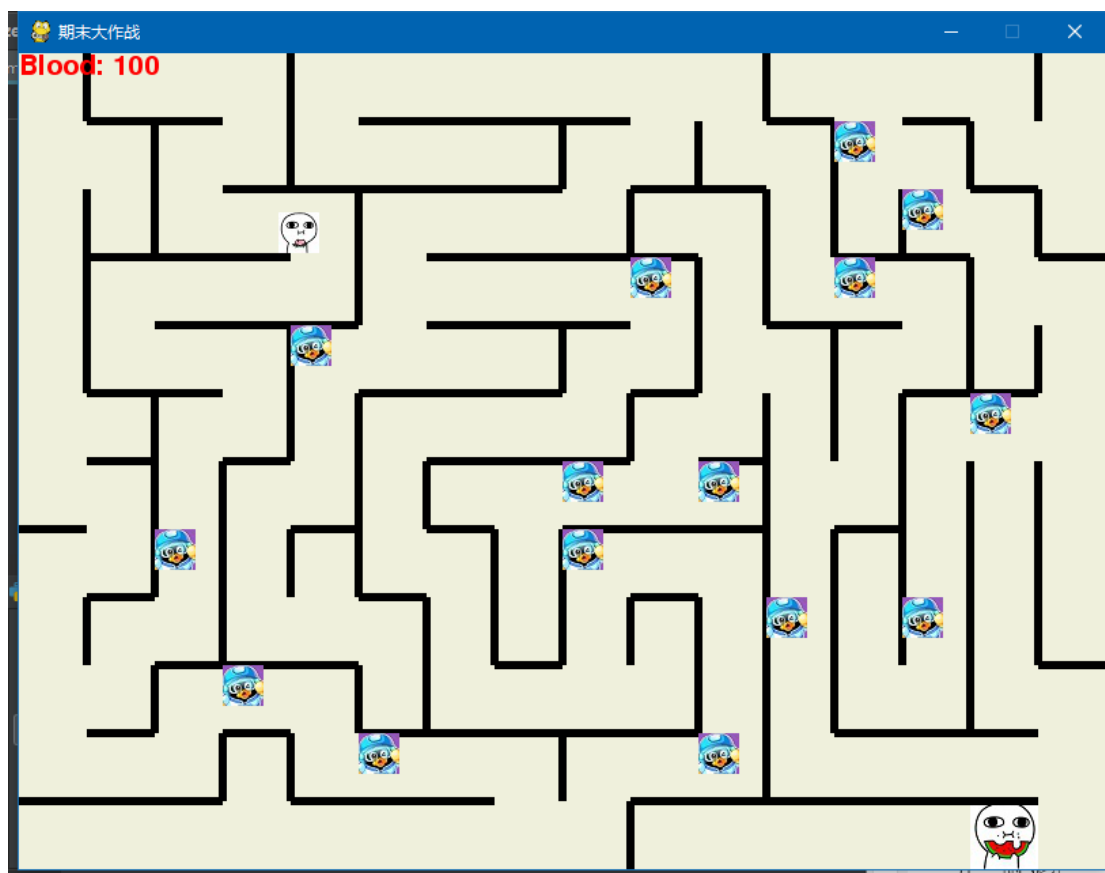


图 15 游戏主界面

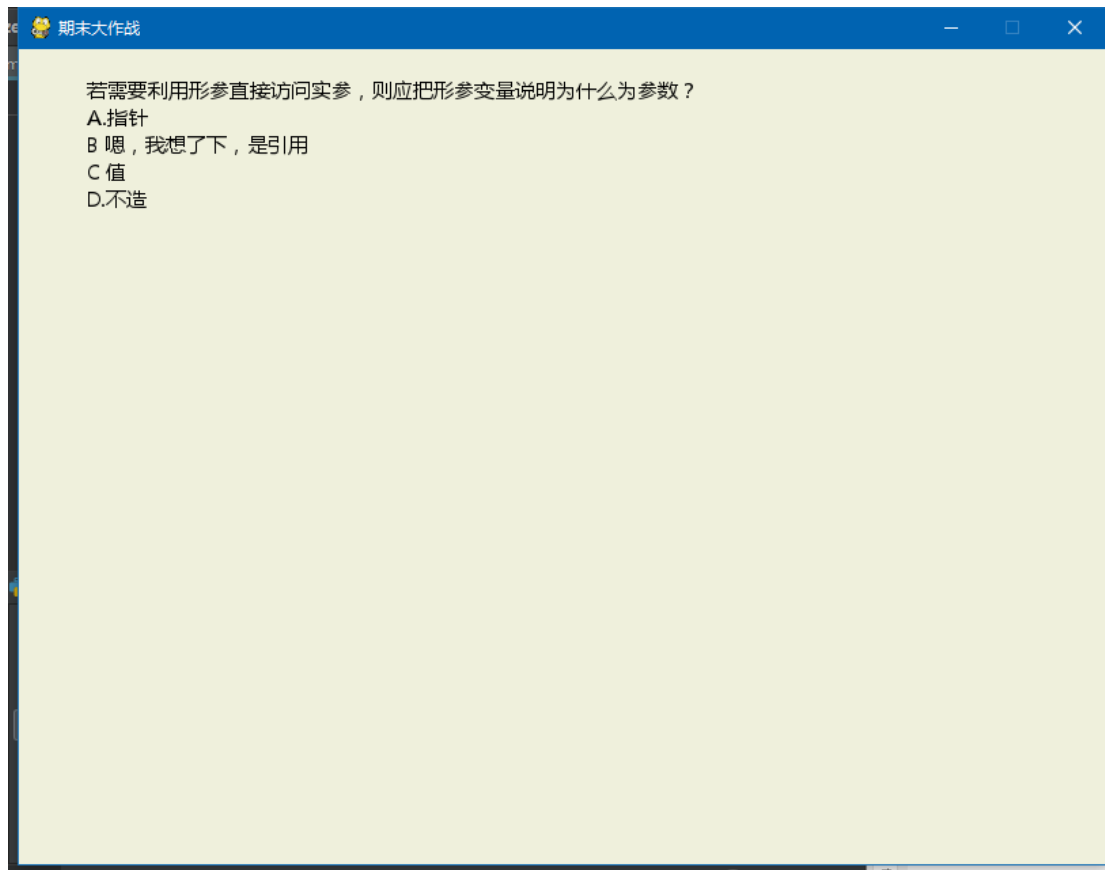


图 16 答题界面

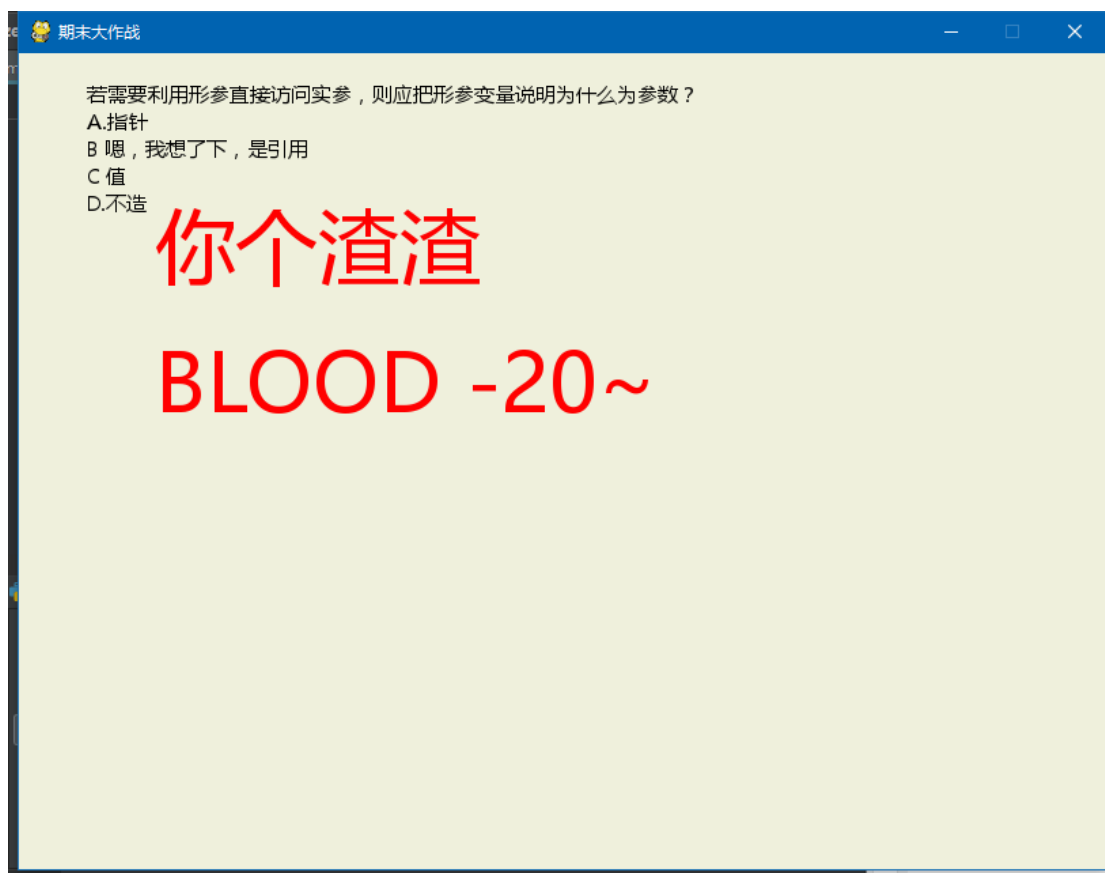


图 17 答题答案错误提示

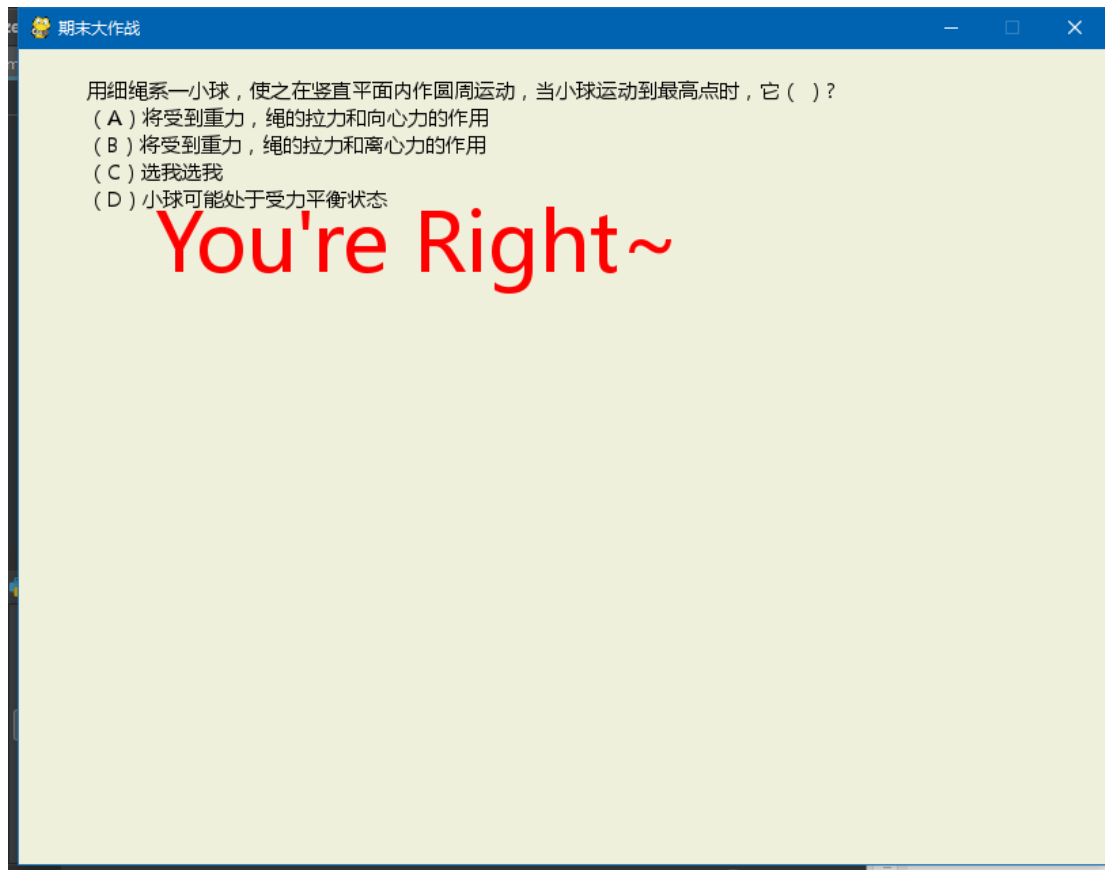


图 18 答题答案正确提示

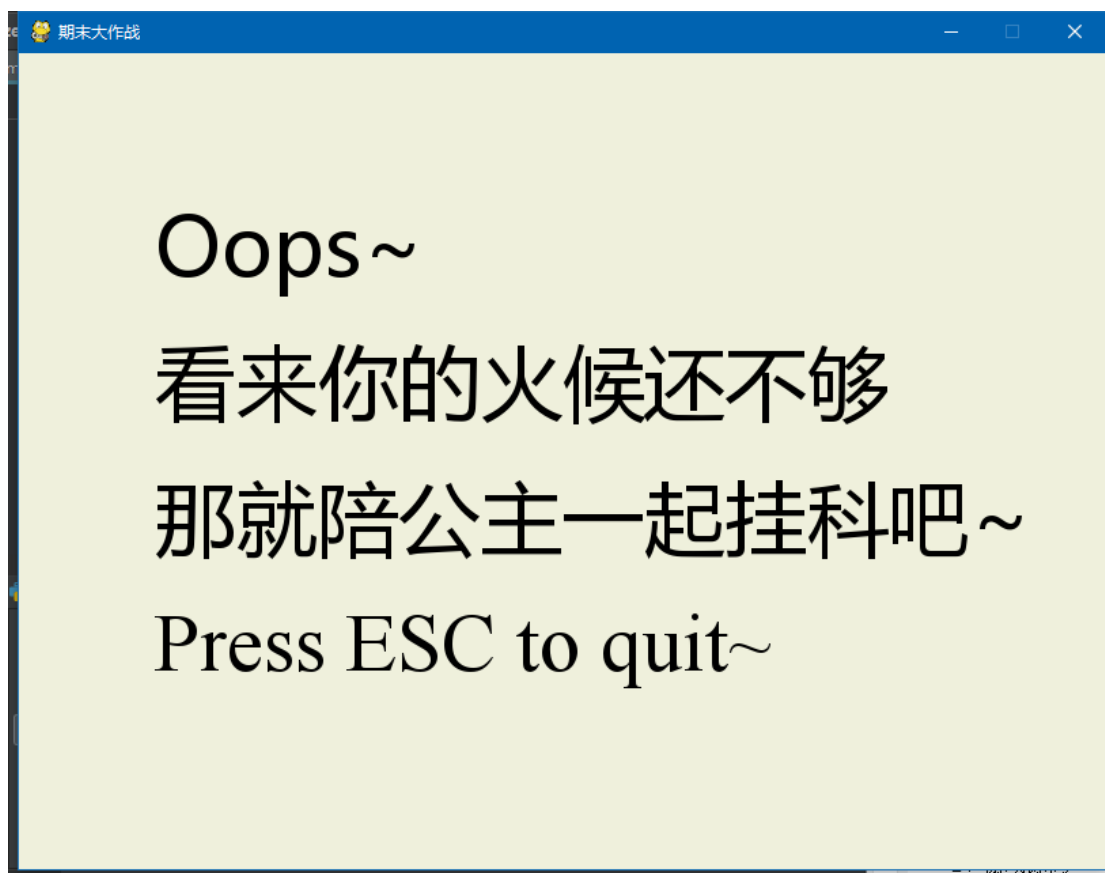


图 19 游戏失败界面



图 20 游戏胜利界面

## 2、查重情况



ID: 5961ADBE13DDDVWHY www.paperpass.com 1 / 9

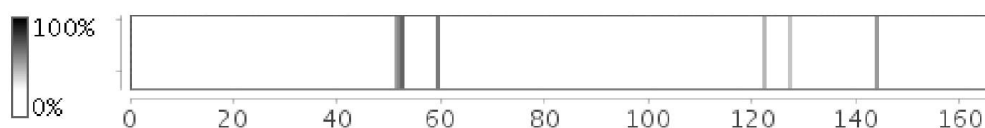
# PaperPass专业版检测报告 简明打印版

### 比对结果（相似度）：

总体：2 %（总体相似度是指本地库、互联网的综合比对结果）  
本地库：2 %（本地库相似度是指论文与学术期刊、学位论文、会议论文数据库的比对结果）  
期刊库：1 %（期刊库相似度是指论文与学术期刊库的比对结果）  
学位库：2 %（学位库相似度是指论文与学位论文库的比对结果）  
会议库：1 %（会议库相似度是指论文与会议论文库的比对结果）  
互联网：0 %（互联网相似度是指论文与互联网资源的比对结果）

编号：5961ADBE13DDDVWHY  
版本：专业版  
标题：基于DFS算法的迷宫游戏设计  
作者：朱桐  
长度：4067 字符(不计空格)  
句子数：167句  
时间：2017-7-9 12:14:54  
比对库：学术期刊、学位论文（硕博库）、会议论文、互联网资源  
查真伪：<http://www.paperpass.com/check>

### 句子相似度分布图：



### 本地库相似资源列表（学术期刊、学位论文、会议论文）：

- 相似度：1 % 篇名：《基于XEN的入侵检测服务研究》  
来源：学位论文 太原理工大学 2009 作者：张志新
- 相似度：1 % 篇名：《基于轻量级消息中间件技术的O2O结算系统研究与实现》  
来源：学位论文 华中科技大学 2015 作者：张焜
- 相似度：1 % 篇名：《关于WLAN组网和数据转发方式的研究》  
来源：学位论文 北京邮电大学 2010 作者：霍群松
- 相似度：1 % 篇名：《针对数字电视码流的硬加速与同步控制策略研究》  
来源：学位论文 湖南大学 2015 作者：周霞
- 相似度：1 % 篇名：《局域网智能监控系统的应用》  
来源：学位论文 山东师范大学 2010 作者：张晶

6. 相似度：1% 篇名：《虚拟机健壮日志安全研究》  
来源：学位论文 太原理工大学 2008 作者：高丹
7. 相似度：1% 篇名：《Intranet网络安全智能监控系统的研究》  
来源：学位论文 东北电力大学 2008 作者：郭良
8. 相似度：1% 篇名：《基于内容的发布/订阅系统中事件快速匹配算法研究》  
来源：学位论文 河北师范大学 2008 作者：张彩云
9. 相似度：1% 篇名：《迷宫机器人系统设计与搜索算法研究》  
来源：学位论文 北京工业大学 2007 作者：祝尚臻
10. 相似度：1% 篇名：《迷宫搜索算法的比较研究》  
来源：学术期刊 《计算机应用研究》 2011年12期 作者：龚道雄 刘翔
11. 相似度：1% 篇名：《深度优先搜索算法和A\*算法在迷宫搜索中的仿真研究》  
来源：学术期刊 《制造业自动化》 2011年11期 作者：刘翔 龚道雄
12. 相似度：1% 篇名：《新型电脑鼠的设计开发》  
来源：学位论文 中国海洋大学 2013 作者：宫兆俭
13. 相似度：1% 篇名：《基于人工势场法的迷宫路径搜索算法设计》  
来源：学术期刊 《北京交通大学学报》 2014年5期 作者：李晓光 姚自强 杨旭
14. 相似度：1% 篇名：《机器人走迷宫最优路径算法研究》  
来源：学位论文 南京工业大学 2010 作者：杨新
15. 相似度：1% 篇名：《并发启发式搜索蛙跳算法》  
来源：会议论文 2008年中国信息技术与应用学术论坛 2008-04-01 作者：蔡阳波 邓一贵 王康
16. 相似度：1% 篇名：《迷宫机器人路径规划算法研究》  
来源：学位论文 东华大学 2007 作者：朱德良

#### 互联网相似资源列表：

没有找到与互联网相似度高的资源！

#### 全文简明报告：

##### 一、游戏内容及特点描述

##### 1、游戏背景介绍

故事发生在一个叫做大数据的王国，这里的人们以知识和技术为尊，人们享受着新兴技术给他们带来的生活便利，安居乐业。而这时，王宫中却传来了国王的叹息声，原来国王唯一的女儿小小在闯关期末考核时发生了意外，被困在知识迷岛里面，国王不但担忧着女儿的安危，更担忧女儿无法继承他的王位，思虑多时，国王最终决定，贴榜昭告天下，凡是能救出公主的，就可以成为王国的驸马。万千勇士收到消息便不及待地踏上了征程，而这当中就有一位名唤木木的少年...

勇士木木来到了知识迷岛，却发现这座岛比他想象的要复杂得多，整座岛是一个非常大的迷宫，道路错综复杂，更麻烦的是重要的关卡还有擅长不同学科的老师把守，要想让老师放路，就只能乖乖的通过老师的考试，通过考试就可以得到迷宫道路中的提示，找到公主，完成游戏。反之，则游戏失败。

##### 2、游戏玩法及规则介绍



### 3、游戏源码

代码行数共 791 行，以下行号仅供参考。由于 Python 语言对缩进要求极为严格，所以若有源代码查看及使用需求请勿直接复制粘贴如下代码。下文代码及相应资料文件已上传至 Github 仓库，地址如下：

[https://github.com/Spico197/maze\\_runner](https://github.com/Spico197/maze_runner)

```
1  # !python3
2  # coding=utf-8
3
4  """
5      This is a python3 and pygame based maze runner game lib
6      for the Programming Course Design.
7      @author: Spico
8      @date: 2017/07/06
9  """
10
11 # ----- Lib Import ----- #
12 import pygame
13 import math
14 import random
15 from openpyxl import load_workbook
16
17 # ----- Color Definition ----- #
18 BLACK = (0, 0, 0)
19 WHITE = (255, 255, 255)
20 GREEN = (0, 255, 0)
21 RED = (255, 0, 0)
22 WARM_GREY = (239, 240, 220)
23 WARM_ORANGE = (242, 159, 63)
24 WARM_YELLOW = (255, 233, 87)
25
26 # ----- Global Variables Definition
27 ----- #
28 # Screen dimensions
29 ROAD_THICKNESS = 50
30 WALL_THICKNESS = 6
31
32 SCREEN_WIDTH = 800
33 SCREEN_HEIGHT = 600
34
35 blood = 100
```

```

36
37 done = False # main-loop flag
38
39
40 # ----- Class Definition ----- #
41
42 class Maze(object):
43     """
44     maze class
45     """
46     def __init__(self):
47         """
48         constructor init
49         """
50         self.neighbor = []
51         self.visited = []
52         self.unvisited = []
53         self.wall_table = []
54         self.road_mat = []
55
56     def maze_generate(self, road_list, wall_list):
57         """
58         generate the road, wall from road list
59         @param road_list: road sprite group
60         @param wall_list: wall sprite group
61         @return: road_list, wall_list
62         """
63         rows = len(self.road_mat)
64         cols = len(self.road_mat[0])
65
66         for row in range(rows): # row&col start from 0 to
67             cols-1
68                 for col in range(cols):
69                     if self.road_mat[row][col] == 1:
70                         road = Road(ROAD_THICKNESS*col,
71 ROAD_THICKNESS*row)
72                         road_list.add(road)
73                         self.unvisited.append([row, col]) # get in
74 all the matrix points
75
76         # wall maze drawing
77         for row_inv in range(1, rows):
78             for col_inv in range(0, cols):
79                 wall = Wall(col_inv*ROAD_THICKNESS,

```

```

80 row_inv*ROAD_THICKNESS-WALL_THICKNESS/2, 1)
81     wall_list.add(wall)
82     self.wall_table.append(wall)
83     for col_inv in range(1, cols):
84         for row_inv in range(0, rows):
85             wall
86 Wall(col_inv*ROAD_THICKNESS-WALL_THICKNESS/2,
87 row_inv*ROAD_THICKNESS, 2)
88     wall_list.add(wall)
89     self.wall_table.append(wall)
90
91     return road_list, wall_list
92
93     def isexit(self, pos_now):      # row&col start from 0
94         """
95         test if the pos_now is the exit
96         @param pos_now: the position remaining to be tested
97         @return: flag-the bool variable
98         """
99
100         flag = False
101         row = pos_now[0]
102         col = pos_now[1]
103         # find the exit
104         exit_row = 0
105         exit_col = 0
106
107         for row_mat in range(len(self.road_mat)):
108             for col_mat in range(len(self.road_mat[0])):
109                 if self.road_mat[row_mat][col_mat] == -1:
110                     exit_row = row_mat
111                     exit_col = col_mat
112
113             # if (exit_row == row and math.fabs(exit_col - col) ==
114 1) or \
115             #         (exit_col == col and math.fabs(exit_row - row)
116 == 1):
117                 #         flag = True
118
119             if exit_row == row and exit_col == col:
120                 flag = True
121
122         return flag
123

```

```

124     def dfs_maze_generate(self, start_pos, wall_list):
125         """
126         dfs algorithm to generate the wall group
127         @param start_pos: the position that mumu is ready to
128     go
129         @param wall_list: wall group remaining to change
130         @return: if the maze is good to run, return wall_list,
131     else return False
132         """
133         rows = len(self.road_mat)
134         cols = len(self.road_mat[0])
135
136         self.unvisited.remove(start_pos)
137         self.visited.append(start_pos) # maze.visited add
138     the start_pos and remove the start_pos in unvisited
139         current_stack = [start_pos]
140
141         find = False
142         while self.unvisited:
143             next_pos = self.neighbor_select(start_pos)
144
145             if not next_pos:
146                 if current_stack:
147                     start_pos = current_stack.pop()
148                 else:
149                     start_pos =
150     self.unvisited[random.randrange(0, len(self.unvisited))]
151                 continue
152
153             current_stack.append(next_pos)
154
155             self.road_mat[next_pos[0]][next_pos[1]] = 0
156             self.visited.append(next_pos)
157             self.unvisited.remove(next_pos)
158             wall_list = self.wall_break(start_pos, next_pos,
159     wall_list)
160
161             if not wall_list:
162                 return False
163
164             if self.isexit(next_pos):
165                 find = True
166
167             start_pos = next_pos

```

```

168         # dfs_maze_generate(next_pos, maze, maze_matrix)
169     if find:
170         return wall_list
171     else:
172         return False
173
174     def neighbor_select(self, pos_now):
175         """
176         select a neighbor near pos_now
177         @param pos_now: the current position
178         @return: the neighbor position
179         """
180
181         row = len(self.road_mat)
182         col = len(self.road_mat[0])
183         row_now = pos_now[0]
184         col_now = pos_now[1]
185         tag_up_down = row_now * col + col_now # judge the
186 up&down tag comparision
187         tag_left_right = col_now * row + row_now # judge the
188 left&right tag comparision
189         neighbor = []
190
191         if tag_up_down + col <= row * col + col and [row_now
192 + 1, col_now] in self.unvisited:
193             neighbor.append([row_now + 1, col_now])
194         if tag_up_down - col >= 0 and [row_now - 1, col_now]
195 in self.unvisited:
196             neighbor.append([row_now - 1, col_now])
197         if tag_left_right + row <= row * col + col and [row_now,
198 col_now + 1] in self.unvisited:
199             neighbor.append([row_now, col_now + 1])
200         if tag_left_right - row >= 0 and [row_now, col_now -
201 1] in self.unvisited:
202             neighbor.append([row_now, col_now - 1])
203
204         if not neighbor:
205             return False
206
207         return neighbor[random.randrange(0, len(neighbor))]
208
209     def wall_break(self, pos1, pos2, wall_list):
210         """
211         break the wall between to points

```

```

212         @param pos1: first position
213         @param pos2: second position
214         @param wall_list: the wall group
215         @return: if handy, return wall_list, else return False
216         """
217         rows = len(self.road_mat)
218         cols = len(self.road_mat[0])
219         location = 0
220
221         if pos1[0] == pos2[0] and math.fabs(pos1[1] - pos2[1])
222 == 1: # left&right border
223             if pos1[1] - pos2[1] == -1: # pos1 at the left
224                 location = (rows - 1) * cols + pos1[1] * rows
225 + pos1[0]
226             elif pos1[1] - pos2[1] == 1:
227                 location = (rows - 1) * cols + pos2[1] * rows
228 + pos2[0]
229             elif pos1[1] == pos2[1] and math.fabs(pos1[0] - pos2[0])
230 == 1:
231                 if pos1[0] - pos2[0] == 1: # pos1 at the bottom
232                     location = pos2[0] * cols + pos2[1]
233                 elif pos1[0] - pos2[0] == -1:
234                     location = pos1[0] * cols + pos1[1]
235             try:
236                 wall_list.remove(self.wall_table[location])
237                 return wall_list
238             except Exception as e:
239                 print("wall_remove error!" + str(Exception) +
240 str(e))
241                 return False
242
243
244 class Road(pygame.sprite.Sprite):
245     """
246     @brief: road class inherited from sprite and image
247 loaded
248     @param: x: x-direction position
249     @param: y: y-direction position
250     """
251
252     def __init__(self, x, y): # constructor function
253         """
254         constructor init function
255         @param x: the road x position

```

```

256         @param y: the road y position
257         """
258         super().__init__() # parents' constructor call
259
260         self.image = pygame.Surface((ROAD_THICKNESS,
261 ROAD_THICKNESS))
262         self.image.fill(WARM_ORANGE)
263         self.rect = self.image.get_rect()
264         self.rect.x = x
265         self.rect.y = y
266
267
268 class Wall(pygame.sprite.Sprite):
269     """
270     @brief: wall class inherited from sprite and image
271 loaded
272     @param x: x-direction position
273     @param y: y-direction position
274     """
275
276     def __init__(self, x, y, mode): # constructor function
277         """
278         constructor function
279         @param x: x position
280         @param y: y position
281         @param mode: mode 1: the standing one; mode 2: the lying
282 one
283         """
284         super().__init__() # parents' constructor call
285
286         if mode == 1: # mode = 1, lying down; mode = 2, stand
287 up.
288             self.image = pygame.Surface((ROAD_THICKNESS,
289 WALL_THICKNESS))
290             if mode == 2:
291                 self.image = pygame.Surface((WALL_THICKNESS,
292 ROAD_THICKNESS))
293             self.image.fill(BLACK)
294             self.rect = self.image.get_rect()
295             self.rect.x = x
296             self.rect.y = y
297
298
299 class Player(pygame.sprite.Sprite):

```

```

300         """
301         @brief: the Player class inherited from sprite and
302 image loaded
303         """
304
305         change_x = 0    # speed
306         change_y = 0
307
308         def __init__(self, x, y):
309             """
310             constructor function
311             @param x: x-position
312             @param y: y-position
313             """
314
315             super().__init__() # inherited
316
317             self.image =
318 pygame.image.load('../data/pic/mumu_30.jpg')
319             # image/rect are the properties of the sprite class
320             self.rect = self.image.get_rect() # top left corner
321 location
322             self.rect.y = y
323             self.rect.x = x
324
325         def changespeed(self, x, y):
326             """
327             change the speed of the player
328             @param x: x-speed
329             @param y: y-speed
330             """
331
332             self.change_x += x
333             self.change_y += y
334
335         def move(self, walls):
336             """
337             move the player
338             @param walls: the wall group that player cannot crawl
339 out
340             @return: none
341             """
342
343             # first left/right then up/down

```



```

344
345     # moving left/right
346     self.rect.x += self.change_x
347
348     if self.rect.x < 0:
349         self.rect.left = 0
350     if self.rect.right > SCREEN_WIDTH:
351         self.rect.right = SCREEN_WIDTH
352
353     # create the interact list
354     block_hit_list = pygame.sprite.spritecollide(self,
355 walls, False)
356
357     # set the limitation of the player
358     for block in block_hit_list:
359         # moving right, the hit obj's left side is the
360 player's right side
361         if self.change_x > 0:
362             self.rect.right = block.rect.left
363         else:
364             # moving left, the road's right is the player's
365 left
366             self.rect.left = block.rect.right
367
368     # moving up/down
369     self.rect.y += self.change_y
370
371     if self.rect.y < 0:
372         self.rect.top = 0
373     if self.rect.bottom > SCREEN_HEIGHT:
374         self.rect.bottom = SCREEN_HEIGHT
375
376     # create hit list
377     block_hit_list = pygame.sprite.spritecollide(self,
378 walls, False)
379
380     # limitation in the up&down direction
381     for block in block_hit_list:
382         if self.change_y > 0:
383             # NOTICE: it's moving down
384             self.rect.bottom = block.rect.top
385         else:
386             self.rect.top = block.rect.bottom
387

```

```

388
389 class Teacher(pygame.sprite.Sprite):
390     def __init__(self):
391         """
392         constructor function
393         """
394         super().__init__()
395         self.image =
396         pygame.image.load('../data/pic/ds_30.jpg').convert()
397         self.rect = self.image.get_rect() # top-left corner
398         location
399
400
401 # ----- Function Definition -----
402 #
403
404
405 def welcome(screen_wel):
406     """
407     print the welcome information
408     @param screen_wel: the screen which is ready to make an
409     output
410     @return: none
411     """
412     screen_wel.fill(WARM_GREY)
413
414     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
415     text_wel = font_wel.render("故事发生在名为大数据的王国，",
416 True, BLACK)
417     screen_wel.blit(text_wel, [50, 50])
418     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
419     text_wel = font_wel.render("这里的人们以知识和技术为尊，人们
420 享受着新兴技术", True, BLACK)
421     screen_wel.blit(text_wel, [50, 80])
422     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
423     text_wel = font_wel.render("给他们带来的生活便利，安居乐业。
424 ", True, BLACK)
425     screen_wel.blit(text_wel, [50, 110])
426     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
427     text_wel = font_wel.render("有一天，王宫中传来了国王的叹息声，

```

```

428     ", True, BLACK)
429     screen_wel.blit(text_wel, [50, 140])
430     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
431     text_wel = font_wel.render("原来国王唯一的女儿小小在闯期末考
432 核时发生了意外, ", True, BLACK)
433     screen_wel.blit(text_wel, [50, 170])
434     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
435     text_wel = font_wel.render("被困在知识迷岛里面, 国王不但担忧
436 着女儿的安危, ", True, BLACK)
437     screen_wel.blit(text_wel, [50, 200])
438     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
439     text_wel = font_wel.render("更担忧女儿无法继承他的王位, 思虑
440 多时, 国王最终", True, BLACK)
441     screen_wel.blit(text_wel, [50, 230])
442     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
443     text_wel = font_wel.render("决定, 贴榜昭告天下, 凡是能救出公
444 主的, 就可以成为", True, BLACK)
445     screen_wel.blit(text_wel, [50, 260])
446     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
447     text_wel = font_wel.render("王国的驸马。万千勇士收到消息便不
448 及待地踏上了征程, ", True, BLACK)
449     screen_wel.blit(text_wel, [50, 290])
450     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
451     text_wel = font_wel.render("而这当中就有一位名唤木木的少
452 年.....", True, BLACK)
453     screen_wel.blit(text_wel, [50, 320])
454
455     pygame.display.flip()
456     pygame.time.wait(3000)
457
458     screen_wel.fill(WARM_GREY)
459

```

```

460     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
461     text_wel = font_wel.render("勇士木木来到了知识迷岛，却发现这
462 座岛比他想象的要", True, BLACK)
463     screen_wel.blit(text_wel, [50, 50])
464     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
465     text_wel = font_wel.render("复杂得多，整座岛是一个非常大的迷
466 宫，道路错综复杂，", True, BLACK)
467     screen_wel.blit(text_wel, [50, 80])
468     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
469     text_wel = font_wel.render("更麻烦的是重要的关卡还有擅长不同
470 学科的老师的把守，", True, BLACK)
471     screen_wel.blit(text_wel, [50, 110])
472     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
473     text_wel = font_wel.render("要想让老师放路，就只能乖乖的通过
474 老师的考试，通过", True, BLACK)
475     screen_wel.blit(text_wel, [50, 140])
476     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
477     text_wel = font_wel.render("考试就可以得到迷宫道路中的提示，
478 找到公主，完成游", True, BLACK)
479     screen_wel.blit(text_wel, [50, 170])
480     font_wel = pygame.font.Font('../data/font/msyh.ttc', 20)
481     text_wel = font_wel.render("戏。反之，则游戏失败。", True,
482 BLACK)
483     screen_wel.blit(text_wel, [50, 200])
484     font_wel = pygame.font.Font('../data/font/msyh.ttc', 50)
485     text_wel = font_wel.render("木木勇士，冲冲冲！", True, (235,
486 63, 47))
487     screen_wel.blit(text_wel, [50, 300])
488
489     pygame.display.flip()
490     pygame.time.wait(4000)
491

```

```

492
493 def blood_loss(screen_loss):
494     """
495     if mumu get a wrong answer, mumu get blood loss. This prints
496     the output tip
497     @param screen_loss: the screen you are ready to make an
498     output
499     @return: none
500     """
501     font_loss = pygame.font.Font('../data/font/msyh.ttc',
502 60)
503
504     text_loss = font_loss.render("你个渣渣", True, RED)
505     screen_loss.blit(text_loss, [100, 100])
506     font_loss = pygame.font.Font('../data/font/msyh.ttc',
507 60)
508     text_loss = font_loss.render("BLOOD -20~", True, RED)
509     screen_loss.blit(text_loss, [100, 200])
510
511     pygame.display.flip()
512     pygame.time.wait(700)
513
514 def answer_right(screen_right):
515     """
516     if mumu get a right answer, mumu get a hint
517     @param screen_right: the screen you are ready to make an
518     output
519     @return: none
520     """
521     font_right = pygame.font.Font('../data/font/msyh.ttc',
522 60)
523     text_right = font_right.render("You're Right~", True,
524 RED)
525
526     screen_right.blit(text_right, [100, 100])
527
528     pygame.display.flip()
529     pygame.time.wait(700)
530
531
532 def lose(screen_lose):
533     """
534     if you lose, you will see this

```

```

535     @param screen_lose: the screen you are redy to make an
536 output
537     @return:
538     """
539     global done
540     screen_lose.fill(WARM_GREY)
541
542     pygame.mixer.music.fadeout(500) # bgm music stop
543     pygame.time.wait(500)
544
545     pygame.mixer.music.load('../data/music/lose.mp3')
546     pygame.mixer.music.play()
547
548     font_vic1 = pygame.font.Font('../data/font/msyh.ttc',
549 60)
550     font_vic2 = pygame.font.Font('../data/font/msyh.ttc',
551 60)
552     font_vic3 = pygame.font.Font('../data/font/msyh.ttc',
553 60)
554     font_vic4 = pygame.font.SysFont('TimesNewRoman', 60)
555     text_vic1 = font_vic1.render("Oops~", True, BLACK)
556     text_vic2 = font_vic2.render("看来你的火候还不够", True,
557 BLACK)
558     text_vic3 = font_vic3.render("那就陪公主一起挂科吧~", True,
559 BLACK)
560     text_vic4 = font_vic4.render("Press ESC to quit~ ", True,
561 BLACK)
562
563     screen_lose.blit(text_vic1, [100, 100])
564     screen_lose.blit(text_vic2, [100, 200])
565     screen_lose.blit(text_vic3, [100, 300])
566     screen_lose.blit(text_vic4, [100, 400])
567
568     pygame.display.flip()
569
570     done = False
571     while not done:
572         for event in pygame.event.get():
573             if event.type == pygame.QUIT:
574                 done = True
575             if event.type == pygame.KEYUP:
576                 if event.key == pygame.K_ESCAPE:

```

```

577         done = True
578     pygame.quit()
579
580
581 def victory(screen_vic):
582     """
583     if you have won the game, you will see this hint
584     @param screen_vic: the screen you are willing to make an
585     output
586     @return: none
587     """
588     global done
589     screen_vic.fill(WARM_GREY)
590
591     pygame.mixer.music.fadeout(1000)    # bgm music stop
592     pygame.time.wait(500)
593
594     pygame.mixer.music.load('../data/music/victory.ogg')
595     pygame.mixer.music.play()
596
597     font_vic1 = pygame.font.Font('../data/font/msyh.ttc',
598 60)
599     font_vic2 = pygame.font.Font('../data/font/msyh.ttc',
600 60)
601     font_vic3 = pygame.font.Font('../data/font/msyh.ttc',
602 40)
603     font_vic4 = pygame.font.Font('../data/font/msyh.ttc',
604 40)
605     font_vic5 = pygame.font.Font('../data/font/msyh.ttc',
606 20)
607     font_vic6 = pygame.font.Font('../data/font/msyh.ttc',
608 20)
609     text_vic1 = font_vic1.render("Congratulations!", True,
610 BLACK)
611
612     text_vic2 = font_vic2.render("你拯救了公主!", True, BLACK)
613
614     text_vic3 = font_vic3.render("但由于你胆敢覬覦公主的美色, ",
615 True, BLACK)
616
617     text_vic4 = font_vic4.render("国王决定将你处死...", True,
618 BLACK)
619
620     text_vic5 = font_vic5.render("(原来木木从来都只是国王的工具)")

```

```

617     ", True, BLACK)
618     text_vic6 = font_vic6.render("Press ESC to quit~ ", True,
619     BLACK)
620
621     screen_vic.blit(text_vic1, [100, 100])
622     screen_vic.blit(text_vic2, [100, 200])
623     screen_vic.blit(text_vic3, [100, 300])
624     screen_vic.blit(text_vic4, [100, 350])
625     screen_vic.blit(text_vic5, [100, 400])
626     screen_vic.blit(text_vic6, [100, 470])
627
628     pygame.display.flip()
629
630     done = False
631     while not done:
632         for event in pygame.event.get():
633             if event.type == pygame.QUIT:
634                 done = True
635             if event.type == pygame.KEYUP:
636                 if event.key == pygame.K_ESCAPE:
637                     done = True
638     pygame.quit()
639
640
641 # ----- Main Loop ----- #
642
643
644 def main():
645     """
646     the main loop funtion
647     @return: none
648     """
649     # ----- PyGame Initialization ----- #
650     pygame.init()
651
652     size = (SCREEN_WIDTH, SCREEN_HEIGHT) # window size
653     screen = pygame.display.set_mode(size)
654
655     pygame.display.set_caption("期末大作战")
656
657     road_mat_12_16 = [
658         [1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
659         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
660         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],

```



```

660         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
661         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
662         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
663         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
664         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
665         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
666         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
667         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
668         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1]
669     ]
670     road_mat = [
671         [1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
672         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
673         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
674         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
675         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
676         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
677         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
678         [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, -1, 1],
679     ]
680
681     main_maze = Maze()
682     main_maze.road_mat = road_mat_12_16
683     player = Player(ROAD_THICKNESS, 0)
684
685     road_list = pygame.sprite.Group()
686     wall_list = pygame.sprite.Group()
687     main_maze.maze_generate(road_list, wall_list) #
688     generate the initial state
689
690     wall_list_dfs = main_maze.dfs_maze_generate([0, 1],
691 wall_list)
692     while not wall_list_dfs:
693         main_maze = Maze()
694         road_list = pygame.sprite.Group()
695         wall_list = pygame.sprite.Group()
696         main_maze.road_mat = road_mat_12_16
697         main_maze.maze_generate(road_list, wall_list)
698         wall_list_dfs = main_maze.dfs_maze_generate([0, 1],
699 wall_list)
700         try:
701             wall_list_dfs.draw(screen)
702             pygame.display.flip()
703         except Exception as e:

```

```

704         print("wall_list_dfs error!" + str(Exception) +
705 str(e))
706         continue
707
708     movingsprites = pygame.sprite.Group()
709     movingsprites.add(player)
710
711     # Teacher Section
712     teacher_list = pygame.sprite.Group()
713     for i in range(15):
714         tea = Teacher()
715         tea.rect.x = random.randrange(ROAD_THICKNESS,
716 SCREEN_WIDTH-ROAD_THICKNESS, 50)
717         tea.rect.y = random.randrange(ROAD_THICKNESS,
718 SCREEN_HEIGHT-ROAD_THICKNESS, 50)
719         teacher_list.add(tea)
720
721     clock = pygame.time.Clock()
722
723     # ----- BGM Loaded ----- #
724     bgm =
725     pygame.mixer.music.load('../data/music/bgm_zalababa.mp3')
726     pygame.mixer.music.play()
727
728     # ----- Figure Loaded ----- #
729     xiaoxiao_image =
730     pygame.image.load('../data/pic/xiaoxiao_50.jpg').convert(
731 )
732
733     # ----- Question Loaded ----- #
734     wb =
735     load_workbook(filename='../data/question/question_cn.xlsx
736 ')
737     ws = wb.get_sheet_by_name('Sheet1')
738     question_taboo_list = []
739
740
741     # welcome
742     welcome(screen)
743
744     # ----- Main Loop ----- #
745     global done
746     global blood
747     while not done:

```

```

748     # ----- Event ----- #
749     for event in pygame.event.get():
750         if event.type == pygame.QUIT:
751             done = True
752
753         if event.type == pygame.KEYDOWN:
754             if event.key == pygame.K_LEFT:
755                 player.changespeed(-3, 0)
756             if event.key == pygame.K_RIGHT:
757                 player.changespeed(3, 0)
758             if event.key == pygame.K_UP:
759                 player.changespeed(0, -3)
760             if event.key == pygame.K_DOWN:
761                 player.changespeed(0, 3)
762
763         if event.type == pygame.KEYUP:
764             if event.key == pygame.K_LEFT:
765                 player.changespeed(3, 0)
766             if event.key == pygame.K_RIGHT:
767                 player.changespeed(-3, 0)
768             if event.key == pygame.K_UP:
769                 player.changespeed(0, 3)
770             if event.key == pygame.K_DOWN:
771                 player.changespeed(0, -3)
772             if event.key == pygame.K_ESCAPE:
773                 done = True      # ESC key to quit the game
774
775     # ----- Game Logic ----- #
776     player.move(wall_list_dfs)
777
778     # ask questions
779     question_list = pygame.sprite.spritecollide(player,
780 teacher_list, True)
781     row_rand = random.randrange(1, 21, 1)
782     # while row_rand in question_taboo_list:
783     #     row_rand = random.randrange(1, 21, 1)
784     #
785     # question_taboo_list.append(row_rand)
786     for row in question_list:
787         screen.fill(WARM_GREY)
788         position = ('B' + str(row_rand))
789         font_wb                                     =
790 pygame.font.Font('../data/font/msyh.ttc', 15)
791         text_wb = font_wb.render(ws[position].value, True,

```

```

792     BLACK)
793
794         position_a = ('C' + str(row_rand))
795         font_option_a                                     =
796     pygame.font.Font('../data/font/msyh.ttc', 15)
797         text_option_a                                     =
798     font_option_a.render(ws[position_a].value, True, BLACK)
799         position_b = ('D' + str(row_rand))
800         font_option_b                                     =
801     pygame.font.Font('../data/font/msyh.ttc', 15)
802         text_option_b                                     =
803     font_option_b.render(ws[position_b].value, True, BLACK)
804         position_c = ('E' + str(row_rand))
805         font_option_c                                     =
806     pygame.font.Font('../data/font/msyh.ttc', 15)
807         text_option_c                                     =
808     font_option_c.render(ws[position_c].value, True, BLACK)
809         position_d = ('F' + str(row_rand))
810         font_option_d                                     =
811     pygame.font.Font('../data/font/msyh.ttc', 15)
812         text_option_d                                     =
813     font_option_d.render(ws[position_d].value, True, BLACK)
814
815         screen.blit(text_wb, [50, 20])
816         screen.blit(text_option_a, [50, 40])
817         screen.blit(text_option_b, [50, 60])
818         screen.blit(text_option_c, [50, 80])
819         screen.blit(text_option_d, [50, 100])
820
821     pygame.display.flip()
822     # test if the answer is true
823     ok = False
824     right_answer = ws[('G'+str(row_rand))].value
825     answer = ' '
826     while not ok:
827         for event in pygame.event.get():
828             if event.type == pygame.KEYUP:
829                 if event.key == pygame.K_LEFT:
830                     player.changespeed(3, 0)
831                     ok = False
832                 if event.key == pygame.K_RIGHT:
833                     player.changespeed(-3, 0)
834                     ok = False
835                 if event.key == pygame.K_UP:

```

```

836         player.changespeed(0, 3)
837         ok = False
838     if event.key == pygame.K_DOWN:
839         player.changespeed(0, -3)
840         ok = False
841     if event.key == pygame.K_a:
842         answer = 'A'
843         ok = True
844     elif event.key == pygame.K_b:
845         answer = 'B'
846         ok = True
847     elif event.key == pygame.K_c:
848         answer = 'C'
849         ok = True
850     elif event.key == pygame.K_d:
851         answer = 'D'
852         ok = True
853     elif event.key == pygame.K_ESCAPE:
854         ok = True
855     else:
856         ok = False
857     if answer == right_answer:
858         answer_right(screen)
859         #
860     pygame.mixer.music.load('../data/music/right.mp3')
861     # pygame.mixer.music.play()
862     else:
863         blood -= 20
864         blood_loss(screen)
865         #
866     pygame.mixer.music.load('../data/music/wrong.mp3')
867     # pygame.mixer.music.play()
868
869     if blood <= 0:
870         lose(screen)
871
872     # MARK: \ line break
873     if player.rect.top > SCREEN_HEIGHT - ROAD_THICKNESS
874 and player.rect.left >= \
875         SCREEN_WIDTH - 2*ROAD_THICKNESS and
876 player.rect.right <= SCREEN_WIDTH and blood > 0:
877         victory(screen)
878
879     # ----- Game Graphics ----- #

```

```

880         screen.fill(WARM_GREY)
881
882         screen.blit(xiaoxiao_image,
883 (SCREEN_WIDTH-2*ROAD_THICKNESS,
884 SCREEN_HEIGHT-ROAD_THICKNESS))
885
886         wall_list_dfs.draw(screen)
887         teacher_list.draw(screen)
888         movingsprites.draw(screen)
889
890         # text on screen
891         font = pygame.font.SysFont('../data/font/msyh.ttc',
892 30)
893         text = font.render("Blood: " + str(blood), True, RED)
894         screen.blit(text, [0, 0])
895         # ----- Refresh & Clock Set ----- #
896
897         pygame.display.flip()    # fresh the screen
898
899         clock.tick(60)    # the speed that the screen updates
900
901         # ----- END of Game ----- #
902         pygame.quit()
903
904
905         # ----- Debug Statement ----- #
906         if __name__ == "__main__":
907             main()
908

```