《GIS开发与设计》

俄罗斯方块程序设计报告

**课 题： 俄罗斯方块**

**学 院： 地理空间信息学院**

**专 业： JS地理信息系统**

**小 组： 第二组**

**指导教师： 　 张江水**

目 录

第一章 前言

1.1 编写目的

1.2 项目概述

1.3 组织结构与人员职责

1.4 理解面向对象分析与设计

第二章 面向对象分析

2.1 需求描述

2.2 建立功能模型

2.3 建立动态模型

2.4 建立对象模型

2.5 界面设计

第三章 面向对象设计

3.1 系统架构设计

3.2 模型层设计

3.2.1 方块类（Brick）

3.2.2 墙类（Wall）

3.2.3 提示信息类（Prompt）

3.2.4 业务类及其关系设计

3.3 视图层设计

3.4 控制层设计

第四章 面向对象实现

4.1 模型层实现（cn.usth.tetris.model）

4.2 视图层实现（cn.usth.tetris.view）

4.3 控制层实现（cn.usth.tetris.model）

**前言**

**编写目的**

详细说明用户需求，并将其作为系统分析、设计和测试的依据。

**项目概述**

该项目使用面向对象软件工程方法设计并实现一个俄罗斯方块游戏。项目进程包括面向对象软件工程的三个阶段，即：面向对象分析、面向对象设计和面向对象实现。系统框架易于扩展，可以在此基础上增添各种功能、扩展游戏用户数，并方便集成其它项目中。

该项目用于面向对象软件工程案例教学，达到以下实习目的：

1. 如何使用面向对象软件工程方法进行软件开发；
2. 如何使用面向对象的思想降低问题的复杂性；
3. 如何使用Python语言具体地实现一个俄罗斯方块游戏。

**组织结构与人员职责**

**组长：**

|  |  |  |  |
| --- | --- | --- | --- |
| **班级** | **学号** | **姓名** | **分工** |
| 制本57班 | 2072017009 | 李一非 |  |
| 制本57班 | 2072017015 | 周琦 |  |
| 制本57班 | 2072017007 | 姜笑宇 |  |
| 制本57班 | 2072017012 | 王林翼 |  |
| 制本57班 | 2072017002 | 成文博 |  |

**理解面向对象分析与设计**

面向对象分析与设计方法要求在设计中要映射现实世界中指定问题域中的对象和实体，例如：顾客、汽车和销售人员等。这就需要设计要尽可能地接近现实世界，即以最自然的方式表述实体。所以[面向对象技术](http://baike.baidu.com/view/1520586.htm)的优点即为能够构建与现实世界相对应的问题模型，并保持他们的结构、关系和行为模式。

大师说：“没有不变的需求，世上的软件都改动过3次以上，唯一一个只改动过两次的软件的拥有者已经死了，死在去修改需求的路上。”

需求是不稳定的，那么需求之中是不是没有稳定的东西呢？有的，就是对象。世界都是由对象组成的，而对象都是持久的，例如动物、植物已经有相当长的时间。虽然对象也在变化，动物，植物也在不断的进化。但对象在一个相当长的时期内都存在，动植物的存在时间肯定比任何一家企业长久。面向对象的开发方法的精髓就是从企业的不稳定需求中分析出企业的稳定对象，以企业对象为基础来组织需求、构架系统。这样得出的系统就会比传统的系统要稳定得多，因为企业的模式一旦变化，只需要将稳定的企业对象重新组织就行了。

本段摘自：<http://baike.baidu.com/view/1488767.htm>

**面向对象分析**

**需求描述**

1. 随机产生经典俄罗斯方块图形，每种方块颜色不同；
2. 可以设置游戏难度级别，级别越高方块下落速度越快；
3. 可以暂停游戏或继续游戏；
4. 方向键实现下落方块的左移、右移、加带下落、变形等基本操作；
5. 方块下落后，下方如果有满行，则将满行消除
6. 对游戏成绩进行记分并显示，存储玩家游戏数据，显示排行榜。
7. 添加游戏音效。

**建立功能模型**

1. 系统用例图

使用UML进行系统分析，就是使用面向对象方法来分析系统，以此建立面向对象的系统模型。此处主要采用UML的例图、活动图对俄罗斯方块游戏的需求进行分析，并建立功能模型。

识别参与者：构造系统的用例模型首先要确定参与者，参与者是与系统、子系统或类发生交互的外部用户、进程或其他系统理想化概念。通过上面的用户分析，可以确定该系统的参与者为游戏玩家。

识别用例：用例是规定系统或部分系统的行为，它描述系统所执行的动作序列集，并为执行者产生一个可供观察的结果。

根据需求描述，可以得到以下用例图，系统顶级用例如图2.1所示。

**图2.1 系统顶级用例**



开始游戏用例如图2.2所示。

**图2.2 开始游戏用例**



1. 典型用例描述

（1）开始游戏

**表2-1 开始游戏用例描述**

|  |  |
| --- | --- |
| **用例名称** | 开始游戏 |
| **功能简述** | 用户通过方向键控制方块的旋转、左移、右移、加速下移 |
| **前置条件** | 启动游戏程序 |
| **主要成功场景** | 1．用户开始一个新游戏  2．用户将方块旋转到一个适当的方向  3．用户左移方块  4．用户右移方块  6．用户加速下移方块  7．方块落到最下方 |
| **扩展路径** | 1a.用户暂停游戏  1b.用户继续游戏  7a.系统将满行消除  7b.游戏结束 |
| **后置条件** | 无 |
| **备注** |  |

（2）提示信息

**表2-2 提示信息用例描述**

|  |  |
| --- | --- |
| **用例名称** | 提示信息 |
| **功能简述** | 根据用户操作实时显示相关游戏信息 |
| **前置条件** | 启动游戏程序 |
| **主要成功场景** | 1．用户设置游戏初始级别  2．用户开始一个游戏  3．显示用户下一个将要出现的方块  4．显示用户消除的总行数和总得分 |
| **扩展路径** | 2a. 游戏结束  2a.1 显示游戏结束对话框 |
| **后置条件** | 无 |
| **备注** |  |

**建立动态模型**

在需求描述的基础上，我们使用用例图对需求进行了进一步的刻画，并对主要用例进行了详细说明。为了更深入的理解用户需求，我们还可以使用活动图对用户与系统的交互过程作更具体的分析。

活动图主要用于业务建模阶段，是描述系统在执行某一用例时的具体步骤的，它主要表现的是系统的动作。从活动图中可以看出，系统是如何一步一步的完成用例描述的。活动图描述的是整个系统的事情。可以说活动图是对用例图的一种细化，帮助开发者理解业务领域。

下面我们用活动图进一步描述用户与系统之间的交互过程，游戏过程活动图如图2.3所示。

**图2.3 游戏过程活动图**



状态图是描述某一对象的状态转化的，它主要表现的是该对象的状态。从状态图中可以看出，该对象在接受了外界的某种刺激之后，会做出什么样的反应。描述的是一个对象的事情。下面我们将俄罗斯方块游戏作为一个系统对象，用状态图分析它的所有可有的状态以及引发状态转换的事件，如图2.4所示。



**图2.4 俄罗斯方块游戏系统状态图**

**建立对象模型**

“万物皆对象”，面向对象程序设计的核心思想是使用现实世界中人类分析和处理问题的方式进行程序设计。因此，结合以上需求分析和描述，我们可以形象地把俄罗斯方块的游戏过程比作是在一个房子里使用各种不同形状的砖块砌墙。我们每次拿起一个砖块，适当地对它进行旋转、左右移动，找到恰当的位置后把它固定住。砖块固定后就成为了下面的墙的一部分。然后再拿起一个砖块，重复以上过程，直到把墙砌满时游戏结束。

建立对象模型的目的就是要找出完成业务功能的所有对象，这些对象就像公司里一个部门的所有员工，每一个人都有自己特定的职责，完成指定的任务。我们在这一步就要找出业务模型中所有的对象，并分析这些对象类之间的关系。如果系统需要使用数据库的话，这步分析包括了传统软件工程学中的E-R图所要完成的功能。对象模型将会在下一个阶段，即设计阶段进一步细化。

这样，我们将问题域中主要的对象及它们之间的关系描述如图2.5所示。

**图2.5 核心业务对象类之间的关系**



在俄罗斯方块游戏中，方块的状态是最具有代表性的，我们下面用状态图来对方块类对象的状态进一步描述，如图2.6所示。

**图2.6 方块对象Brick的状态图**



**界面设计**

人机界面设计是接口设计的重要组成部分。对于交互式系统来说，人机界面的设计质量，直接影响用户对软件产品的评价，从而影响软件产品的竞争力和寿命，必须对人机界面设计给予足够重视。

在面向对象分析阶段，我们要对人机界面进行初步设计。界面设计的结果一方面有助于提前找出需求分析人员与用户对系统需求理解中的潜在的不一致；另一方面有助于分析人员对系统更深入更全面的理解。

**面向对象设计**

[面向对象](http://baike.baidu.com/view/125370.htm)[设计](http://baike.baidu.com/view/66964.htm)解决的是“类与相互通信的对象之间的组织关系，包括它们的角色、职责、协作方式几个方面。[面向对象](http://baike.baidu.com/view/125370.htm)[设计模式](http://baike.baidu.com/view/66964.htm)是“好的[面向对象](http://baike.baidu.com/view/125370.htm)设计”，所谓“好的[面向对象](http://baike.baidu.com/view/125370.htm)设计”是那些可以满足“应对变化，提高复用”的设计。

[面向对象](http://baike.baidu.com/view/125370.htm)[设计模式](http://baike.baidu.com/view/66964.htm)描述的是[软件设计](http://baike.baidu.com/view/575391.htm)，因此它是独立于[编程语言](http://baike.baidu.com/view/552871.htm)的，但是面向对象设计模式的最终实现仍然要使用面向对象编程语言来表达，本项目最终使用Python语言实现。

**系统架构设计**

通过图形化界面与用户交互的系统的特点是：用户通过操作图形界面与系统交互；系统响应用户引发的事件，向对象传递消息；对象之间仅通过消息相互通信，协作完成一个具体的业务功能。

俄罗斯方块游戏的设计中，我们可以把问题域的业务逻辑与图形界面展示分离开，这样做的好处是整个系统结构清晰，分工明确；并且问题域的对象类相对独立，有利于扩展功能以及软件复用。这里我们借助目前流行的MVC（模型(model)－视图(view)－控制器(controller)）的思想把完成系统不同功能的类和接口划分到三个包中。

1. 模型实体包

模型实体包封装了实现俄罗斯方块业务功能的所有基础对象类。这个包就像公司里的一个部门，所有对象类就相当于部门中的每一个职位，每个职位都有具体的岗位职责。每个职位对应的员工就是一个具体的对象，是对象类的一个实例。

1. 用户接口包

用户接口包封装了实现俄罗斯方块游戏的用户界面类，它们就像公司对外业务的窗口。公司的客户通过这个窗口向公司的员工提出服务请求，若干个公司员工相互协作完成对顾客的服务，并最终通过窗口将产品交给顾客。

1. 业务逻辑包

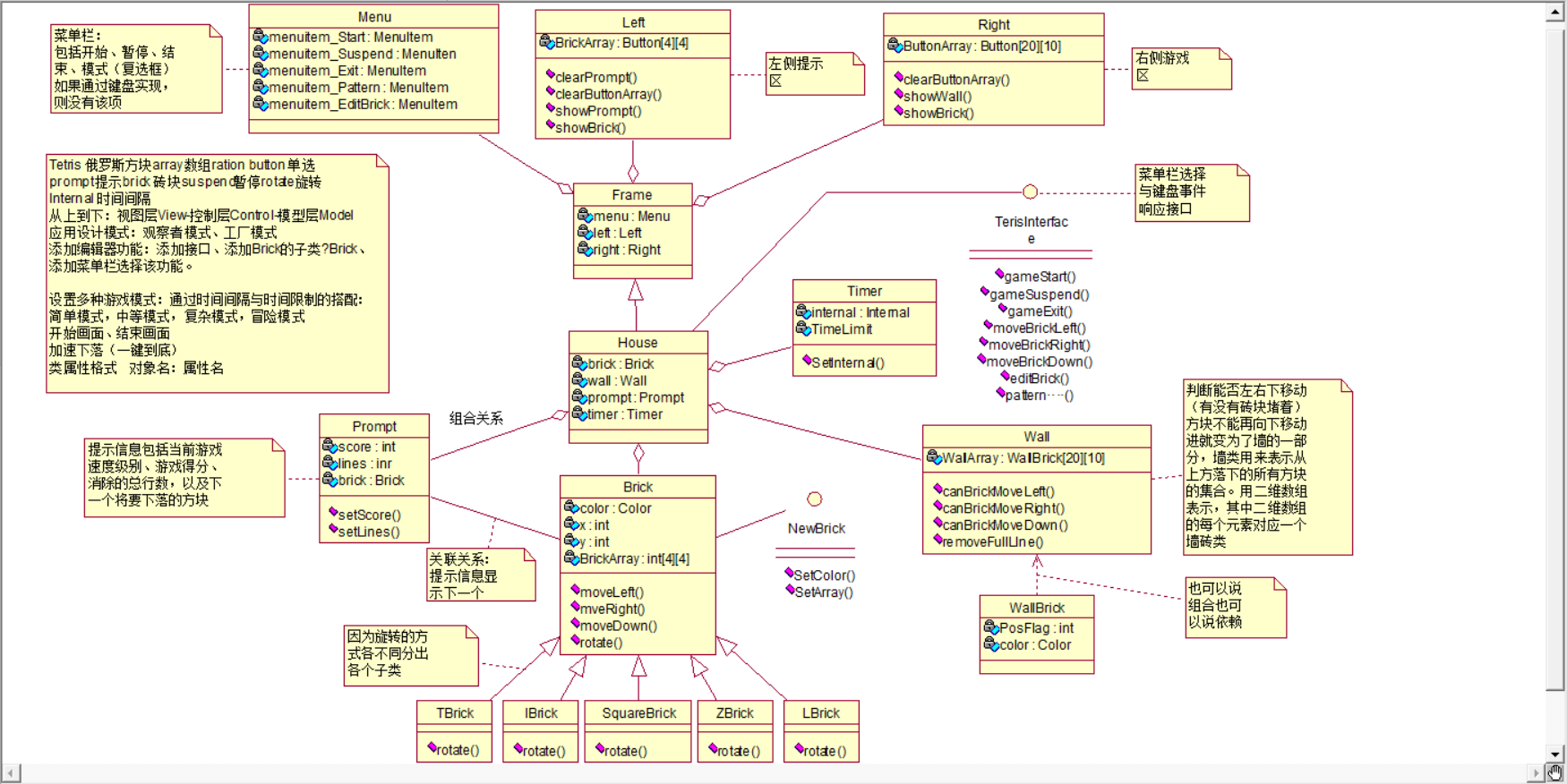
业务逻辑包封装了实现俄罗斯方块具体业务功能的接口（interface），接口中的具体业务就是从用户角度看到的业务逻辑。该包是系统业务的核心实现部分，其他包可以通过实现该包提供的接口，实现具体的业务逻辑内容。

比如，公司的一个部门会有很多员工，每个员工都有自己的本职工作，但仅凭一个员工的岗位职责，还不能胜任为客户服务的任务，必需每个人各司其职，多个员工分工协作来共同完成一项客户请求的服务。

业务逻辑包中的接口就是要合理地组织系统中的各个对象，使它们发挥出集体的力量。具体地说，业务逻辑包中的接口完成两项任务：一是如何组织协调业务实体包中的多个对象共同完成用户请求的服务；二是如何通过用户接口包的图形界面完成与用户的交互。

综上所述，俄罗斯方块游戏系统架构设计如图3.1所示。

此设计图为我们最初版本的设计构造。后续对于House与用户界面皆有功能上的调整，改动较大。



**图3.1 俄罗斯方块游戏系统架构**



**模型（M）**

**视图（V）**

**控制（C）**

**模型层设计**

在面向对象分析阶段，我们确定了问题域中主要的对象类及它们之间的关系。在此基础上，我们进一步添加和完善每个类的属性和行为，并确定类之间的关系。

**方块类（Brick）**

观察一般的俄罗斯方块游戏中的各种方块，可以看出：

1. 不同类别的方块有各种形状和不同的颜色；
2. 方块最大长度或最大宽度都不超过4个小单元；
3. 每种方块旋转后的形状最多有四种状态（如T形方块），最少有1种状态(如田字形方块)。
4. 方块具有左移、右移、旋转、下移、加速下移等行为。

这样，我们对方块类进行以下初步设计：

1. 数据设计
2. 用4×4的整型数组表示一个方块，数组元素取值为0和1，如图3.2所示：

0

0

0

0

0

0

1

0

0

1

1

1

0

0

0

0

0

0

1

0

0

1

1

0

0

0

1

0

0

0

0

0

0

0

0

0

0

1

1

1

0

0

1

0

0

0

0

0

0

0

1

0

0

0

1

1

0

0

1

0

0

0

0

0

**图3.2 T形方块的四种状态表示**

（2）用两个整型变量x和y表示方块矩阵的坐标（x,y），坐标点为矩阵的左上角，即第0行第0列元素的位置。

（3）使用Python.awt.Color类对象color表示方块的颜色

（4）使用整型变量RotateFlag记录方块当前的旋转状态。

1. 行为设计

通过需求分析我们知道：方块具有左移、右移、下移、旋转等行为。在此基础上，我们还要加上系统实现时必需的一些方法，如清空数组、克隆对象等行为。

1. 类的继承关系设计

游戏中的方块尽管相互之间形状、颜色不同，但它们还是有很多相同点，都有(x,y)坐标，都用4×4的整型数组表示、都有旋转标识，都有几种基本行为等。但它们也有不同，比如旋转的方式不同，有的方块根本不用旋转，因为它只有一种旋转状态，而有的方块最多有四种旋转方式。这样，我们把所有方块共性的部分抽象出来定义为父类，每个具体的方块作为子类继承父类的所有属性和行为，并具体定义自己独特的属性和行为。

这样，我们通过方块类的类图将方块类的属性和行为，以及方块类父类与子类之间的关系具体描述如图3.3所示。



**图3.3 方块类的UML类图**

**墙类（Wall）**

方块不能再向下移动进就变为了墙的一部分，墙类用来表示从上方落下的所有方块的集合。用二维数组表示，其中二维数组的每个元素对应一个墙砖类，如图3.4所示。



**图3.4 墙类的UML类图**

**提示信息类（Prompt）**

提示信息包括当前游戏速度级别、游戏得分、消除的总行数，以及下一个将要下落的方块。如图3.5所示。



**图3.5 提示信息类的UML类图**

**实体类及其关系设计**

房子类与方块类、墙类、提示信息类之间是聚集关系中的组合关系。聚集是整体与部分的关系，整体包含部分。组合关系是强聚集关系，如果房子不存在的话，其它三者也将不存在。业务类之间的关系如图3.6所示。

**图3.6 实体类及之间的关系**



**视图层设计**

参照游戏图形界面的初步设计，将图形界面分解为一个窗口，窗口中包括一个菜单条、一个左侧提示区和一个右侧游戏区。窗口类与其它三者之间是聚集关系中的组合关系。设计结果如图3.7所示。



**图3.7 图形接口类及之间的关系**

**控制层设计**

控制器接受用户的输入并调用模型和视图去完成用户的需求。这里将用户的基本控制功能定义在TetrisImpl接口中，然后房子类（House）继承窗口类（TetrisFrame）,并实现TetrisImpl接口。在House类中具体实现控制功能。设计结果如图3.8所示。



**图3.8 控制层设计**

下面将主要控制功能设计结果描述如下。

1. 游戏开始（gameStart）

用户点击界面中的“开始”菜单项开始一个游戏，系统启动定时器；生成游戏区方块并将其显示在游戏区；系统生成提示区方块并将其显示在提示区。这个过程用UML顺序图描述如图3.9所示。



**图3.9 游戏开始顺序图**

1. 方块左移（moveBrickLeft）
2. 方块下移（moveBrickDown）

**面向对象实现**

**模型层实现（cn.usth.tetris.model）**

1. 方块类及其子类的实现
2. 方块工厂类的实现
3. 墙类的实现
4. 提示信息类的实现

**视图层实现（cn.usth.tetris.view）**

1. 窗口类的实现
2. 菜单条的实现
3. 左侧提示区的实现
4. 右侧游戏区的实现

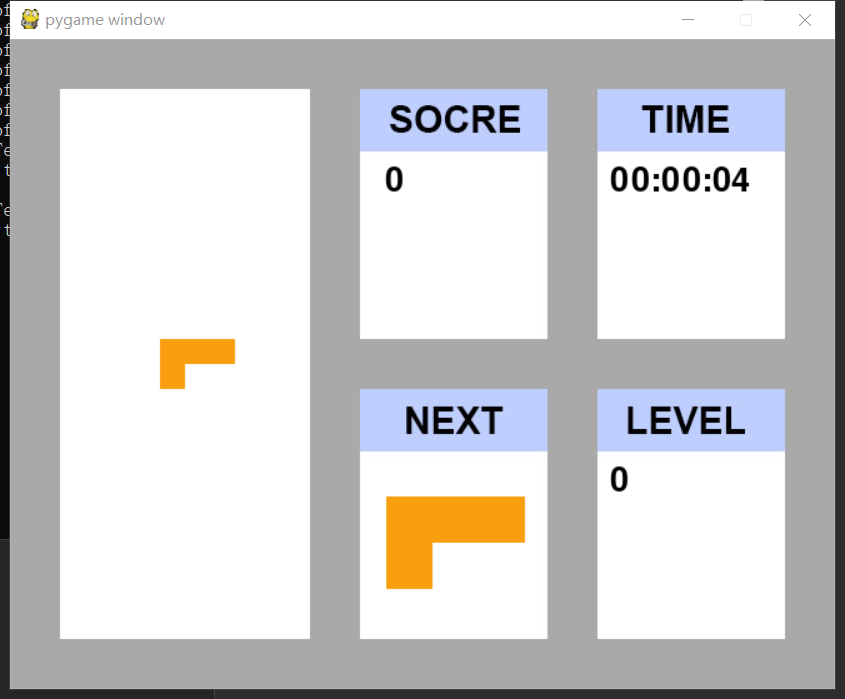
**控制层实现（cn.usth.tetris.model）**

1. 业务接口的定义
2. 房子类的实现

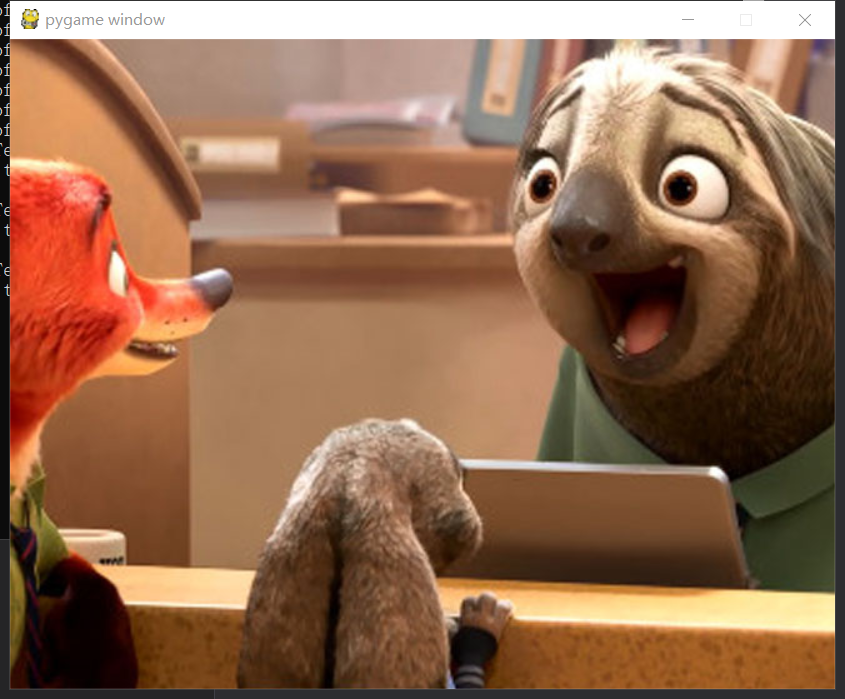
游戏运行的效果图：



游戏初始界面



游戏过程



暂停游戏



排行榜界面