**安徽财经大学字**

**软件工程设计**

**题 目**  基于python实现的连连看游戏开发

**学 院**  管理科学与工程学院

**专 业**  计算机科学与技术专业

**班 级**  20计科4班

**姓 名**  窦建邺

目录

[1.**系统流程图** 4](#_Toc132637987)

[2.**可行性研究** 4](#_Toc132637988)

[2.1.硬件、技术可行性分析 4](#_Toc132637989)

[2.2.运行可行性分析 5](#_Toc132637990)

[2.3.经济可行性分析 5](#_Toc132637991)

[2.4.法律可行性分析 5](#_Toc132637992)

[3.**需求分析** 5](#_Toc132637993)

[3.2数据流图 5](#_Toc132637994)

[3.2.1顶层数据流图 5](#_Toc132637995)

[3.2.2 0层数据流图 6](#_Toc132637996)

[3.2.3数据流图分解 7](#_Toc132637997)

[3.3数据字典 8](#_Toc132637998)

[3.3.1数据流 8](#_Toc132637999)

[3.3.2数据元素 10](#_Toc132638000)

[3.3.3数据存储 11](#_Toc132638001)

[3.3.4数据处理 12](#_Toc132638002)

[3.4实体联系分析 13](#_Toc132638003)

[4**系统设计** 14](#_Toc132638004)

[4.1总体设计 14](#_Toc132638005)

[4.1.1 系统功能结构 14](#_Toc132638006)

[4.1.2 系统层次图 17](#_Toc132638007)

[4.2详细设计 20](#_Toc132638008)

[4.2.1 代表性模块设计 20](#_Toc132638009)

[4.2.2 系统数据库设计 23](#_Toc132638010)

[5 **编码与测试** 24](#_Toc132638011)

[5.1 编码 24](#_Toc132638012)

[5.1.1 编码规则简介 24](#_Toc132638013)

[5.1.2代表性模块示例 26](#_Toc132638014)

[5.2测试 26](#_Toc132638015)

[5.2.1 白盒测试 27](#_Toc132638016)

[5.2.2黑盒测试 32](#_Toc132638017)

[6 **系统使用说明** 35](#_Toc132638018)

[6.1 系统运行环境和配置 35](#_Toc132638019)

[6.2 系统操作说明（按照结构图或层次图的框架依次介绍） 35](#_Toc132638020)

[6.2.1 XX1模块说明 35](#_Toc132638021)

[6.2.2 XX2模块说明 35](#_Toc132638022)

[6.2.3 XX3模块说明 35](#_Toc132638023)

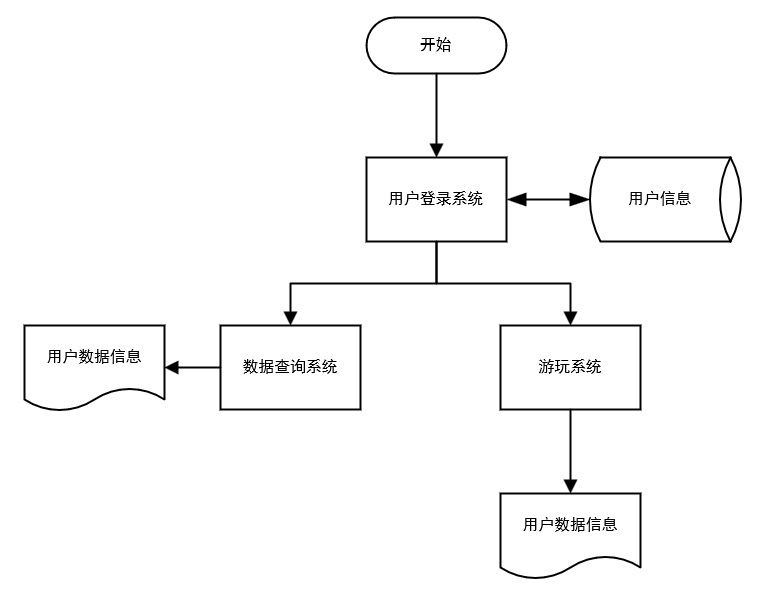
[6.2.4 XX4模块说明 35](#_Toc132638024)

[7 **总结** 35](#_Toc132638025)

[参考文献 35](#_Toc132638026)

# 1.系统流程图

本软件本质为基于python实现的连连看游戏开发，其系统结构简单、直观、易懂。如下图1-1为系统流程图，本软件主要分为用户信息、游戏主体两大部分。

图1-1系统流程图

# 2.可行性研究

## 2.1.硬件、技术可行性分析

(1)需要装有可运行python程序的编译器，如pycharm软件的电脑；

(2)需要实现直观的游戏图形界面，技术上不是很复杂，因此技术上是没问题，是可行的；

(3)需要掌握连连看图片相连并消除的算法原理，这需要一定的python基础，但其本质上是一个"锻炼"型算法，属于入门级，因此仅需一个月的python水平便可掌握；

(4)综上，在硬件与技术上，是没有任何问题的。

## 2.2.运行可行性分析

(1)需要一个可运行python程序的电子设备，如windows、mac、Linux、Ubtunus；

(2)在性能上，达到了目前所有的电脑设备都可完美运行的性能需求；

(3)对于游戏主体来说，玩家操作简单，点击"Start Game"便可开始游戏，因此在游戏体验上没有运行问题；

(4)综上，在运行上，不存在任何困难的问题。

## 2.3.经济可行性分析

作为一款脍炙人口的小游戏，不存在任何经济上的支出与收益，因此在经济上是可行的。

## 2.4.法律可行性分析

基于python开发的连连看游戏，不存在侵权问题——相反，它常常作为程序员锻炼自己的第一块入门砖而存在，因此不存在法律上或存在的问题与纠纷。

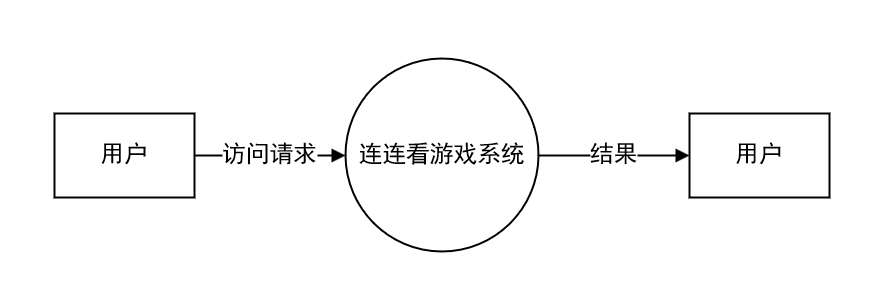
# 3.需求分析

## 3.2数据流图

### 3.2.1顶层数据流图

本软件的顶层数据流图如图所示：

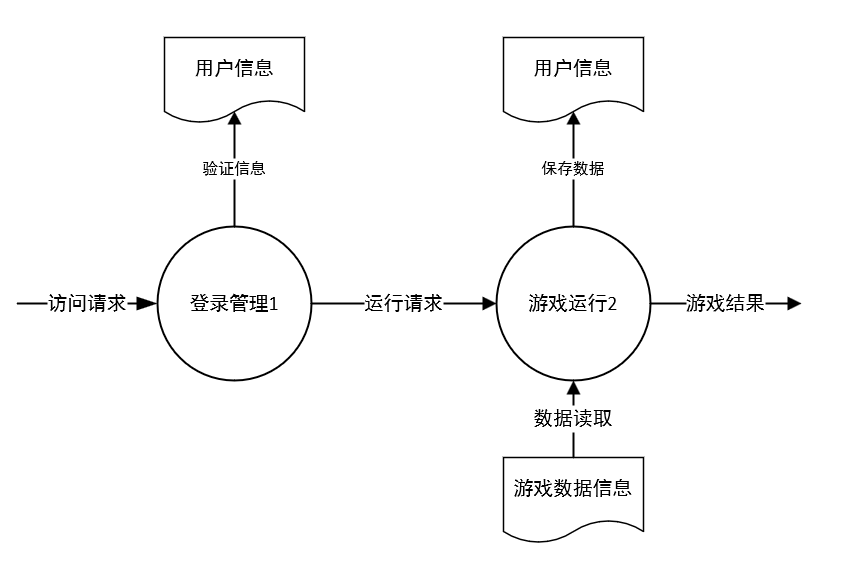
图3-1 顶层数据流图



系统的源点和终点均为用户，用户提交访问请求，等待系统处理请求，若流程正常则返回游戏结果。

### 3.2.2 0层数据流图

图3-2 0层数据流图



基于python的连连看游戏开发的核心功能是连连看游戏的正常运行和带给用户良好的游戏体验，当用户提交访问请求后，系统先对用户信息进行验证，验证成功后，再向系统发送运行请求，系统根据用户信息的正确性决定是否启动游戏主体程序，最后返回游戏的结果并将结果上传至用户信息数据库。

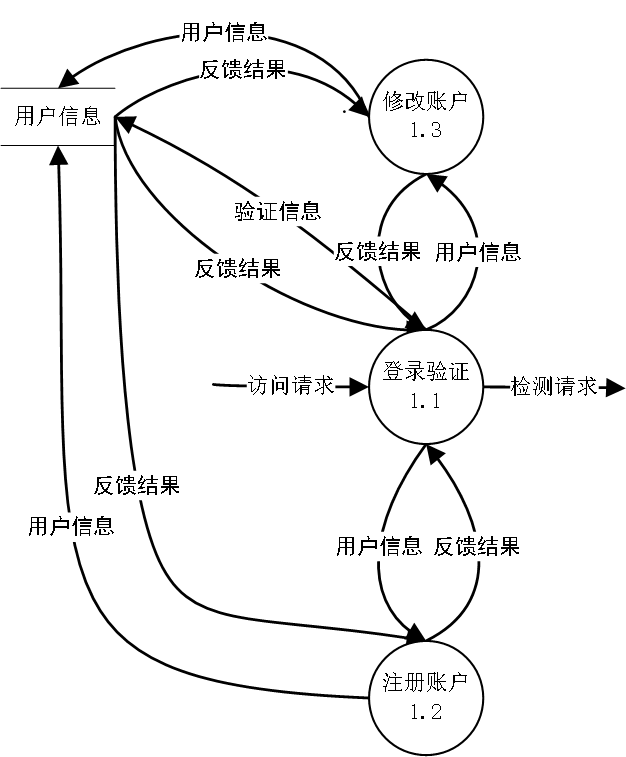
### 3.2.3数据流图分解

根据顶层数据流图和0层数据流图对数据处理进行进一步的分解，让数据流图更加清晰。

基于python的连连看游戏开发系统有登录管理、游戏运行两个子功能。

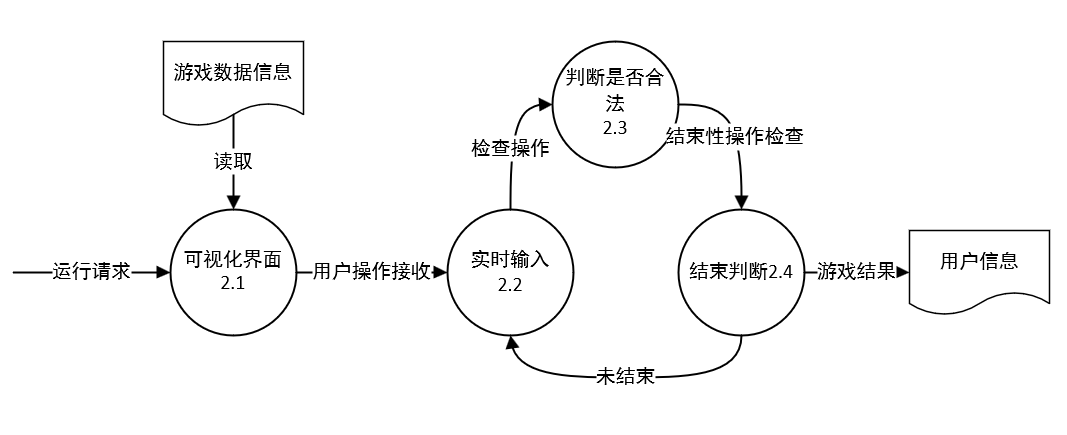
1.登录管理分为登录验证、注册账户、修改账户

图3-3 登录管理分解数据流图



2.游戏运行分为可视化界面、实时输入、判断是否合法和结束判断。

图3-4 游戏运行分解数据流图



## 3.3数据字典

数据字典是描述数据信息定义的集合,也就是对数据流图中包含的所有元素定义的集合。数据字典包含对数据流、数据元素、数据存储、数据处理的描述，数据流、数据元素、数据存储、数据处理的结果如下表3-1所示：

表3-1 数据字典提取结果

|  |  |  |  |
| --- | --- | --- | --- |
| 数据流 | 数据元素 | 数据存储 | 数据处理 |
| 访问请求 | 用户ID | 用户信息 | 登录管理 |
| 运行请求 | 用户已解锁等级 | 游戏数据信息 | 游戏运行 |
| 修改请求 | 关卡ID |  |  |
|  | 填充量 |  |  |

### 3.3.1数据流

根据3.2节数据流图分析情况可知，系统主要涉及的数据流包括：访问请求，运行请求，修改请求分别定义如图3-2~3-6所示。

表3-2 数据流表1

名称：访问请求

别名：验证信息

描述：用户向连连看软件系统发送的用户信息

数据流组成：用户ID

数据流来源：用户

数据流去向：登录管理

表3-3 数据流表2

名称：运行请求

别名：无

描述：用户游玩时选择游戏难度的信息

数据流组成：用户ID+关卡ID

数据流来源：登录验证

数据流去向：游戏运行

表3-4 数据流表3

名称：修改请求

别名：无

描述：用户修改当前已解锁关卡难度的信息

数据流组成：用户ID

数据流来源：用户信息

数据流去向：用户信息

### 3.3.2数据元素

根据3.2节数据流图分析情况可知，系统主要涉及的数据元素包括：用户ID，用户名，密码，产品ID，研磨率，等级，原因分析，百分比，样本ID，相关参数和检测ID分别定义如图3-7~3-17所示。

表3-5 数据元素表4

名称：用户ID

别名：无

含义：唯一的确定用户的编号

定义：用户ID = 8{字符}8

位置：用户信息

表3-6 数据元素表5

名称：用户已解锁关卡等级

别名：无

含义：用户持有的关卡解锁凭证

定义：用户已解锁关卡等级 = 0{数字}5

位置：用户信息

表3-7 数据元素表6

名称：关卡ID

别名：无

含义：关卡所唯一对应的编号

定义：关卡ID = 0{数字}5

位置：游戏数据信息

图3-8 数据元素表7

名称：填充量

别名：无

含义：关卡所唯一对应的图片填充数目

定义：填充量 = 0{数字}100

位置：游戏数据信息

### 3.3.3数据存储

根据3.2节数据流图分析情况可知，系统主要涉及的数据存储包括：用户信息，游戏数据信息分别定义如图3-18~3-20所示。

表3-9 数据存储表8

存储文件名：用户信息

说明：无

流入的数据流：用户信息

流出的数据流：无

组成：用户ID+用户已解锁等级

表3-10 数据存储表9

存储文件名：游戏数据信息

说明：无

流入的数据流：游戏数据信息

流出的数据流：无

组成：关卡ID+填充量

### 3.3.4数据处理

根据3.2节数据流图分析情况可知，系统主要涉及的数据处理包括：登录管理，游戏运行分别定义如图3-21~3-24所示。

表3-11 数据处理表10

处理名：登录管理

说明：对用户提交的访问请求进行处理

流入的数据流：访问请求

流出的数据流：运行请求

处理：对用户提交的访问请求进行处理

表3-12 数据处理表11

处理名：游戏运行

说明：启动游戏

流入的数据流：运行请求

流出的数据流：游戏结果、数据信息

处理：启动游戏程序

## 3.4实体联系分析

由需求分析阶段所得E-R图导出过程如下图所示：

图3-5 用户属性图

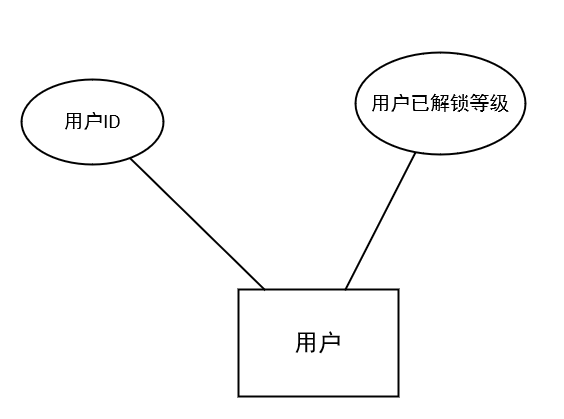
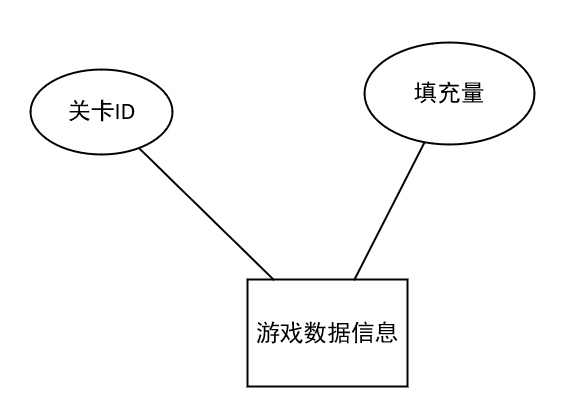


图3-6 游戏数据信息属性图



# 4系统设计

经过系统分析，系统架构必须从各个方面满足实际需求，这是总体设计的目标。同时在设计与开发的过程中严格遵守架构的特殊约束，从而保证实现目标。系统设计是把用户需求转为软件系统重要环节部分，是系统的物理设计阶段，应遵循系统的合理性、统一性、扩展性、兼容性原则。

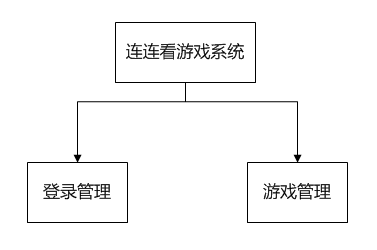
## 4.1总体设计

系统通过计算机对人们日常生活的各项收入和支出进行信息化管理，方便进行信息的录入和查询以及数据的导入和统计，从而减少人们繁琐的计算和记录。系统的整体设计实现，划分为若干个模块，每个模块完成子功能，最后把这些模块集成起来构成一个整体，完成指定的功能满足用户的需求。

### 4.1.1 系统功能结构

面向数据流的设计方法把信息流映射成软件结构，信息流的类型决定了映射的方法。

图4-1 第一级分解结构图



（1）参考图3-2，信息沿输入通路进入系统，同时由外部形式变换为内部形式，进入系统的信息通过变换中心，经加工处理以后再沿输出通路变换成外部形式离开软件系统，可见这些信息流具有交换流的总特征。

（2）确定输入流和输出流的边界，从而孤立出变换中心。

（3）完成各级分解，得到未经精化的软件结构图。

图4-2未经精化的输入结构 登录管理

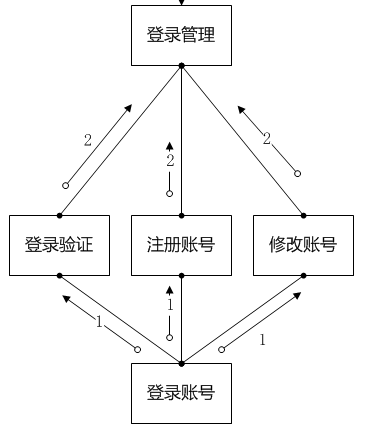


图4-3未经精化的输出结构 游戏管理

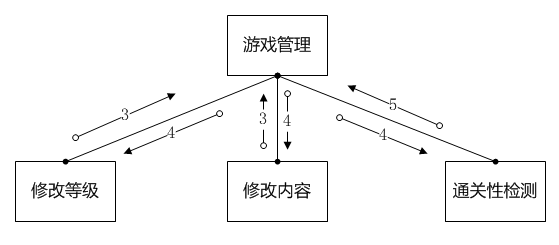
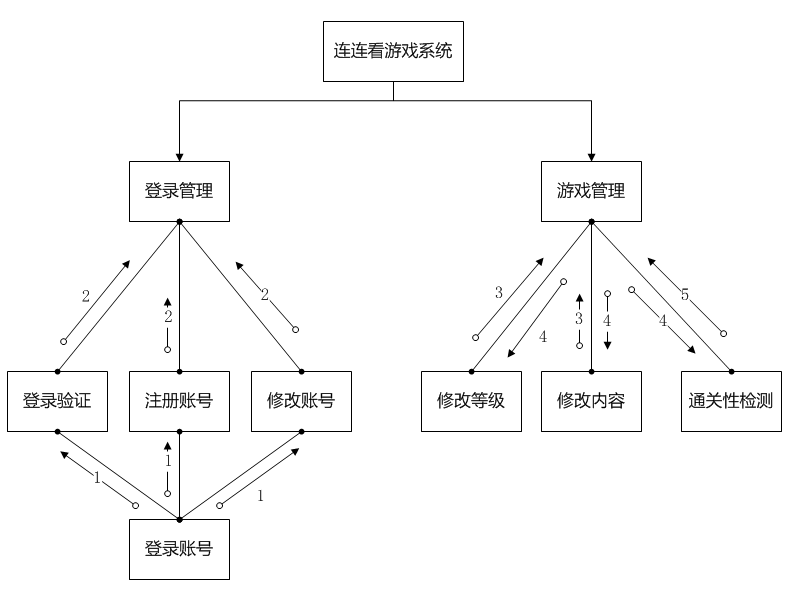


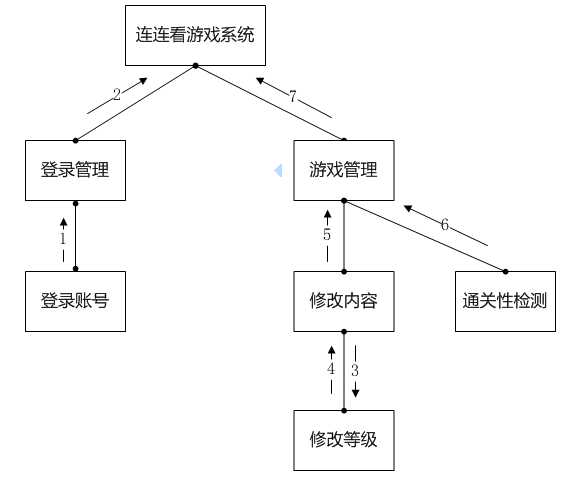
图4-4基于python实现的连连看游戏开发系统未经精化的软件结构图



软件结构进一步精化，对初步分割得到的模块进行再分解或合并：

1. 输入结构中的“登录验证”、“修改账户”和“注册账户”可以合并为“登录账号”；
2. 输入结构中的“修改等级”可并入“修改内容”下；

图4-5基于python实现的连连看游戏开发系统精化后的软件结构图



其中：

1：用户信息 2：反馈结果 3：用户ID

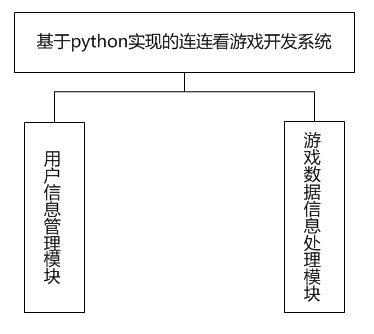
4：用户信息 5：游戏数据信息 6：游戏数据信息

7：反馈结果

### 4.1.2 系统层次图

根据前期需求分析可知基于python实现的连连看游戏开发系统包括用户信息管理模块、游戏数据信息处理模块2大子模块。每个子模块又包含并调用相应下层模块完成程序子功能，底层模块完成基本功能。

图4-6 基于python实现的连连看游戏开发系统的一级功能结构图



用户信息管理模块包括用户注册、用户登录、找回密码、修改密码四个部分。

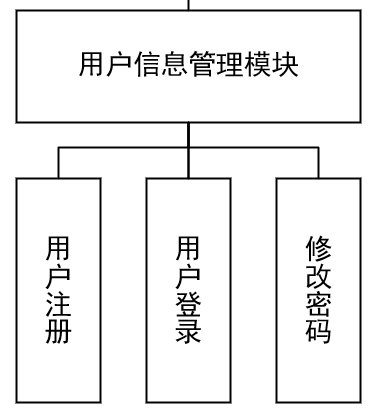
（1）用户注册时，系统将引导用户进行注册，注册成功后返回用户登录界面进行登录。

（2）用户输入账号密码后，系统会与用户信息库中的信息进行比对，比对成功则登录成功。

（3）若对比失败，用户可以通过找回密码功能，进行再次登录。

（4）登陆成功后，用户可以进行修改密码操作。

图4-7 用户信息管理模块结构图

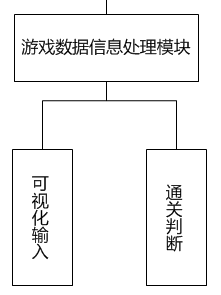


游戏数据信息处理模块包括可视化输入模块和通关判断模块两大功能模块。

（1）用户可通过可视化输入模块进而调用游戏代码，从而合法进行游戏。

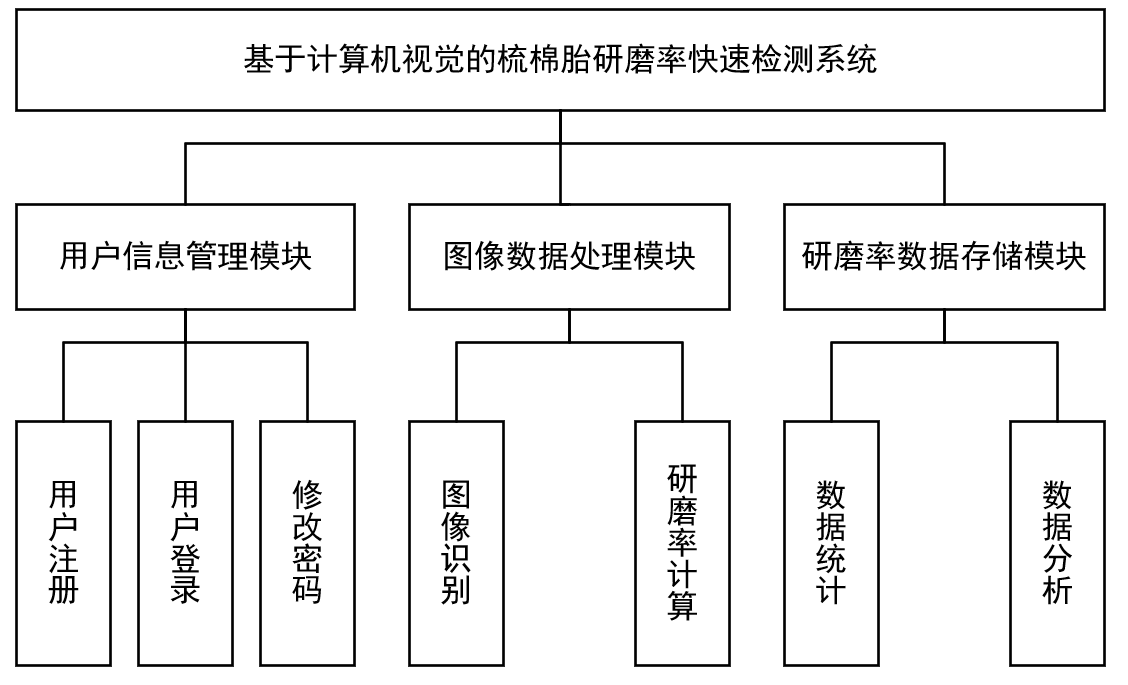
（2）经历一段时间的游玩后，系统会对是否通关进行判断，从而调用通关判断模块。

图4-8 游戏数据信息处理模块结构图



综上所述，将基于python实现的连连看游戏开发系统进行模块化结构表示，并对每一个子模块在进行模块表示，构建出整体系统结构图，并实现完整的子功能，用户使用系统进行游戏时结构更加清晰明了，并以弹出窗口的形式将结果展现给用户，以达到最直观的效果。

图4-9 基于python实现的连连看游戏开发系统整体系统结构图



## 4.2详细设计

### 4.2.1 代表性模块设计

#### 4.2.1.1 用户信息管理模块设计

（1）模块简介

**用户注册**：用户可在登录界面点击“注册账户”按键开启注册流程。在注册界面，用户依次输入用户名、密码、手机号等信息。系统在确认手机号合法后，发送验证码，用户在正确输入验证码后，即完成注册。

**用户登录**：用户在登录界面输入手机号和密码开始登陆，系统会对用户所输入的信息进行验证。系统将通过访问数据库，判断用户信息并进行下一步操作，当系统判断用户信息输入正确，即可允许登录。

**修改密码**：当用户想修改密码时，可通过点击“修改密码”按键开启找回密码流程。用户可输入手机号，随后系统发送验证码，确认用户身份信息，确认成功后进行重置密码。

（2）用户信息管理模块流程图

图4-10用户注册程序流程图

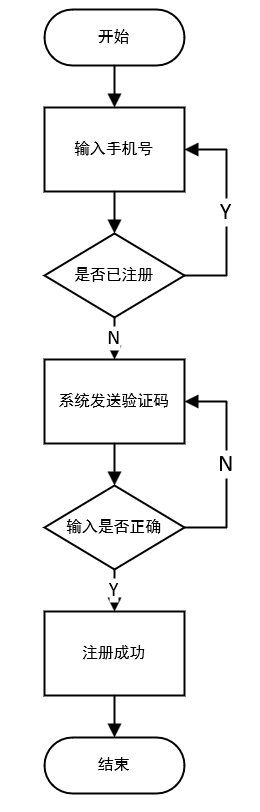


图4-11 用户登录程序流程图

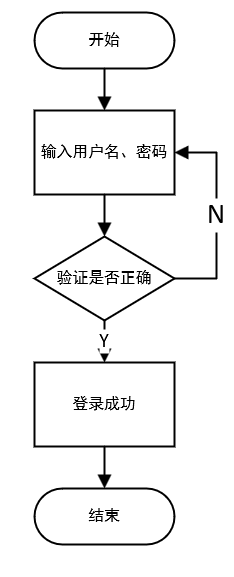
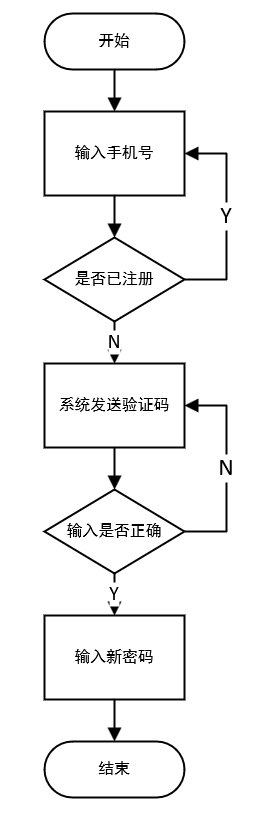


图4-12 修改密码程序流程图



### 4.2.2 系统数据库设计

#### 4.2.2.1 数据库的逻辑设计

用户（用户ID，用户已解锁等级）

游戏数据信息（关卡ID，填充量**）**

#### 4.2.2.2 数据库的表设计及表之间的关联

表4-1 用户表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **类型** | **长度** | **键** | **其他约束** |
| 成员ID | varchar | 8 | 主键 | Unique Key |
| 用户已解锁等级 | int | 3 | 外键 | >0 and <=3 |

表4-2游戏数据信息表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名** | **类型** | **长度** | **键** | **其他约束** |
| 关卡ID | int | 3 | 主键 | Unique Key &  >0 and <=3 |
| 填充量 | int | 10 | 无 | NOT NULL |

# 5 编码与测试

## 5.1 编码

### 5.1.1 编码规则简介

源程序代码的逻辑简明清晰、易读易懂是好程序的一个重要标准，为了做到这一点，应该遵循下述规则。

（1）程序内部的文档

选取含义鲜明的名字，使它能正确地提示程序对象所代表的实体，这对于帮助阅读者理解程序是很重要的。如果使用缩写，那么缩写规则应该一致，并且应该给每个名字加注解。

命名：驼峰法命名。

注解：通常在每个模块开始处有一段序言性的注解，简要描述模块的功能、主要算法、接口特点、重要数据以及开发简史。插在程序中间与一段程序代码有关的注解，主要解释包含这段代码的必要性。对于用高级语言书写的源程序，不需要用注解的形式把每个语句翻译成自然语言，应该利用注解提供一些额外的信息。应该用空格或空行清楚地区分注解和程序。注解的内容一定要正确，错误的注解不仅对理解程序毫无帮助，反而会妨碍对程序的理解。

布局：程序清单的布局对于程序的可读性也有很大影响，应该利用适当的阶梯形式使程序的层次结构清晰明显。

（2）数据说明

数据说明的次序应该标准化，有次序就容易查阅，因此能够加速测试、调试和维护的过程。当多个变量名在一个语句中说明时，应该按字母顺序排列这些变量。如果设计时使用了一个复杂的数据结构，则应该用注解说明用程序设计语言实现这个数据结构的方法和特点。

（3）语句构造

设计期间确定了软件的逻辑结构，然而个别语句的构造却是编写程序的一个主要任务。构造语句时应该遵循的原则是，每个语句都应该简单而直接，不能为了提高效率而使程序变得过分复杂。下述规则有助于使语句简单明了。

·不要为了节省空间而把多个语句写在同一行。

·尽量避免复杂的条件测试。

·尽量减少对“非”条件的测试。

·避免大量使用循环嵌套和条件嵌套。

·利用括号使逻辑表达式或算术表达式的运算次序清晰直观。

（4）输入输出

在设计和编写程序时应该考虑下述有关输入输出风格的规则。

·对所有输入数据都进行检验。

·检查输入项重要组合的合法性。

·保持输入格式简单。

·使用数据结束标记，不要要求用户指定数据的数目。

·明确提示交互式输入的请求，详细说明可用的选择或边界数值。

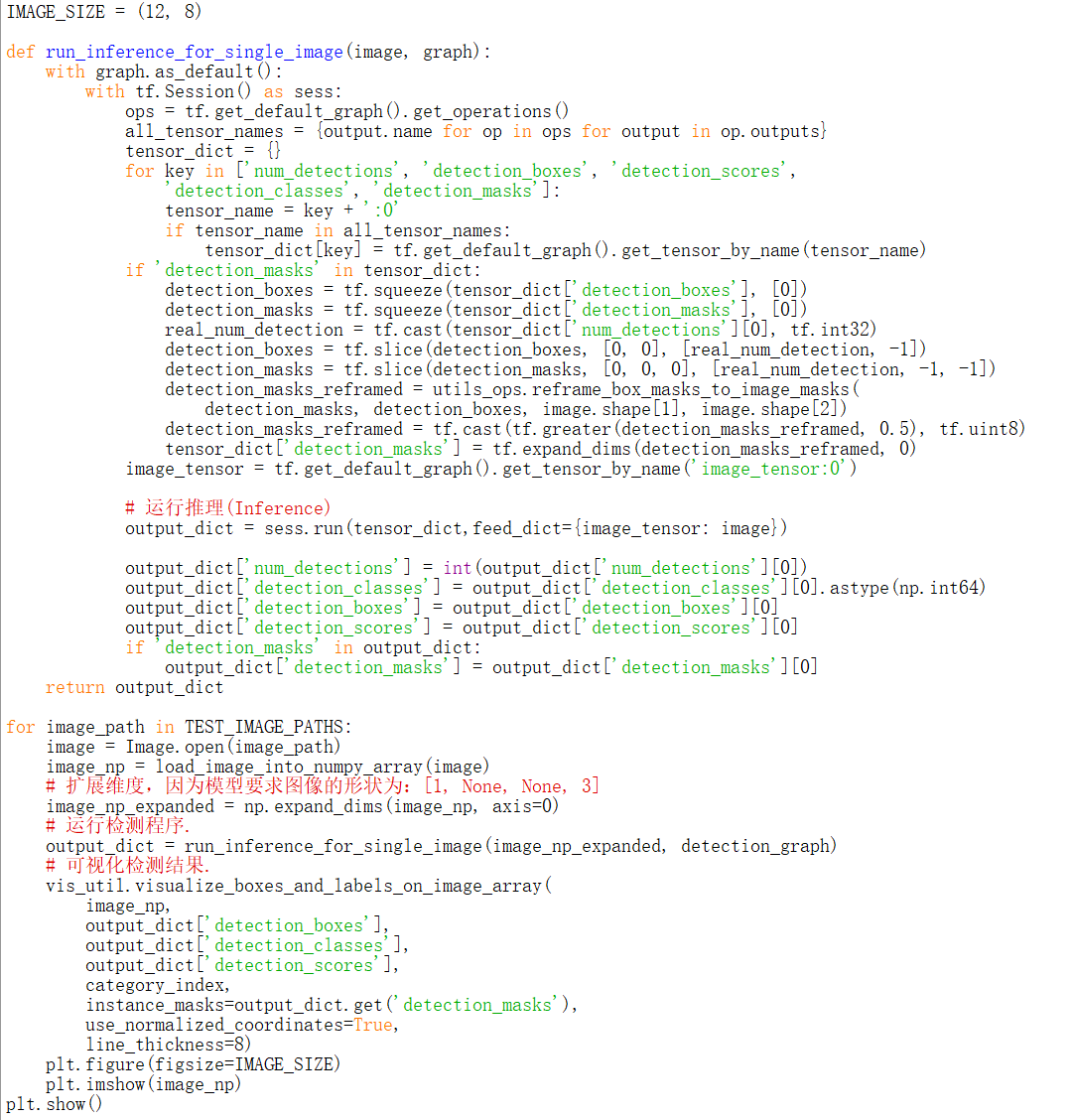
·当程序设计语言对格式有严格要求时，应保持输入格式一致。

·设计良好的输出报表。

·给所有输出数据加标志。

### 5.1.2代表性模块示例

图5-1 图像识别模块核心代码



## 5.2测试

在软件初步成型时应该进行软件测试。软件测试是为了发现错误而执行程序的过程。或者说，软件测试是根据软件开发各阶段的规格说明和程序的内部结构而精心设计一批测试用例（即输入数据及其预期的输出结果），并利用这些测试用例去运行程序，以发现程序错误的过程。

总的来说以最少的时间和人力，系统地找出软件中潜在的各种错误和缺陷，以考虑用户是否能接受该产品。

对于本系统的测试采用两种方法：一种方法称为黑盒测试，已知产品应该具有的功能，通过测试来检验是否每个功能都能正常使用；另一种方法称为白盒测试，已知产品的内部工作过程，通过测试来检验产品内部动作是否按照规格说明书的规定正常进行。

对于软件测试而言，黑盒测试法把程序看作一个黑盒子，完全不考虑程序的内部结构和处理过程。也就是说，黑盒测试是在程序接口进行的测试，它只检查程序功能是否能按照规格说明书的规定正常使用，程序是否能适当地接收输入数据并产生正确的输出信息，程序运行过程中能否保持外部信息(例如数据库或文件)的完整性。黑盒测试又称为功能测试。

白盒测试法与黑盒测试法相反，它的前提是可以把程序看成装在一个透明的白盒子里，测试者完全知道程序的结构和处理算法。这种方法按照程序内部的逻辑测试程序，检测程序中的主要执行通路是否都能按预定要求正确工作。白盒测试又称为结构测试。

### 5.2.1 白盒测试

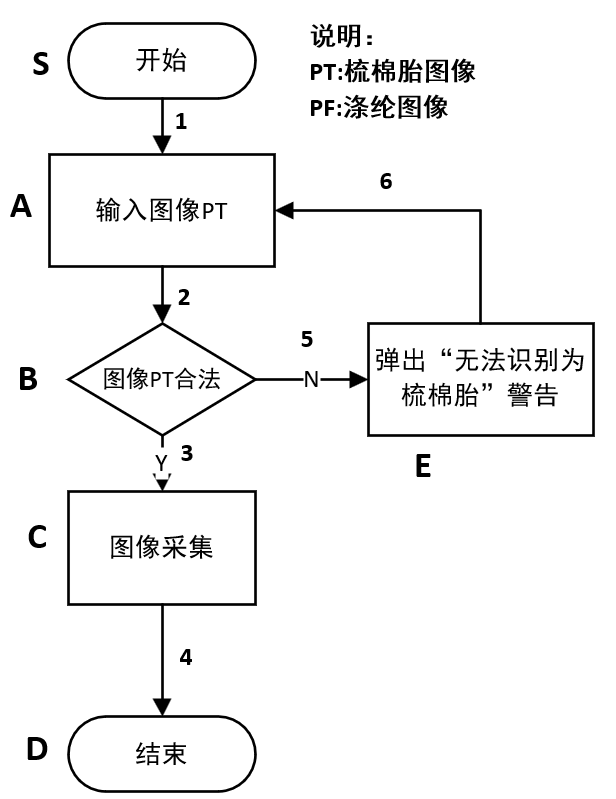
测试方案包括具体的测试目的，应该输入的测试数据和预期的结果。不同的测试数据发现程序错误的能力差距甚远，应当选用最简便、最高效的测试数据，做到尽可能完备、全面的测试。白盒测试把程序看成透明的白盒子，原理是按照程序内部的逻辑测试程序，检测程序中主要执行通路是否能按预定要求正确工作。下面主要介绍**图像数据处理模块**的白盒测试流程，其他模块功能均以此功能测试过程为模板进行测试。

（1）测试目标：从软件设计者角度检测图像数据处理功能是否存在错误;

（2）测试数据：图像PT=梳棉胎图像；图像PF=涤纶图像;

（3）测试对象：被测试模块程序流程图如图所示：

图5-2 基于计算机视觉的梳棉胎研磨率快速检测系统-图像数据处理模块的流程图

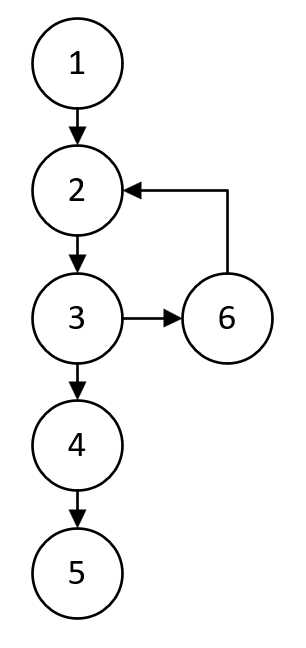


#### 5.2.1.1基本路径测试

基本路径测试是Tom McCabe提出的一种白盒测试技术。使用这种技术设计测试用例，首先计算程序的环形复杂度，并用该复杂度为指南定义执行路径的基本集合，从该基本集合到处的测试用例可以保证程序中的每条语句至少执行一次，而且每个条件在执行时都将分别取真假两种值。

1. 根据过程设计结果画出相应的流图

图5-3 基于计算机视觉的梳棉胎研磨率快速检测系统-图像数据处理模块的流图



（2）计算流图的环形复杂度

根据：V（G）= E-N+2（E:流图的边数 N：节点数）

可得该流图的环形复杂度为V=6-6+2=2，即有2条独立路径条数。

（3）确定线性独立路径的基本集合

由于环形复杂度为2，因此共有2条独立路径，如下列出具体路径：

①SABCD

②SABE

（4）设计可强制执行基本集合中每条路径的测试用例：

表5-1 图像数据处理模块白盒测试-测试用例

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 图像类型 | 是否重新输入 | 独立路径 |
| 1 | 梳棉胎图像(PT) | 否 | SABCD |
| 2 | 涤纶图像(PF) | 是 | SABE |

#### 5.2.1.2测试用例设计与情况分析

测试步骤依次为语句覆盖测试、判定覆盖测试、条件覆盖测试、判定/条件覆盖测试、条件组合覆盖测试、路径覆盖。测试用例以路径覆盖为原则。在每个步骤中需要设计相应的测试用例，使其达到不同的覆盖标准。

（1）语句覆盖测试

语句覆盖需要将程序中每一可执行语句至少执行一次。登录模块流程图中共有两条相互独立路径，因此本轮测试共需要设计2组测试用例。

表5-2图像数据处理模块白盒测试-语句覆盖测试

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 图像类型 | 是否重新输入 | 独立路径 | 期望结果 |
| 1 | 梳棉胎图像(PT) | 否 | SABCD | 图像采集 |
| 2 | 涤纶图像(PF) | 是 | SABE | 弹出警告 |

（2）判定覆盖测试

判定覆盖需要将程序中每个判断的取真分支和取假分支至少经历一次。

应执行路径：

①1-2-3-4-5

②1-2-3-6

图中共有2条路径可经过程序中的每个判断的取真/取假分支。因此本轮测试设计2组测试用例。

表5-3图像数据处理模块白盒测试-判定覆盖测试

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 图像类型 | 是否重新输入 | 覆盖路径 | 期望结果 |
| 1 | 梳棉胎图像(PT) | 否 | ① | 图像采集 |
| 2 | 涤纶图像(PF) | 是 | ② | 弹出警告 |

（3）条件覆盖测试

条件覆盖需要将程序中每个判断的每个条件的可能取值至少执行一次。

B点可能结果：图像合法/图像不合法

图中有1个判断，共计1个判断条件，因此本轮测试设计2组测试用例。

表5-4 图像数据处理模块白盒测试-条件覆盖测试

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 图像类型 | 是否重新输入 | 条件结果 | 期望结果 |
| 1 | 梳棉胎图像(PT) | 否 | True | 图像采集 |
| 2 | 涤纶图像(PF) | 是 | False | 弹出警告 |

（4）判定/条件覆盖测试

判定/条件覆盖需要使得判断中的每个条件的所有可能取值至少执行一次，每个判断中的每个分支至少执行一次。

应满足以下覆盖情况：

1）条件：

图像合法/图像不合法

2）应执行路径：

①1-2-3-4-5

②1-2-3-6

图中共有2条路径可经过程序中的每个判断的取真/取假分支，有1个判断，共计1个判断条件，因此本轮测试设计2组测试用例。

表5-5 图像数据处理模块白盒测试-判定/条件覆盖测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 图像类型 | 是否重新输入 | 覆盖路径 | 条件结果 | 期望结果 |
| 1 | 梳棉胎图像 | 否 | ① | True | 图像采集 |
| 2 | 涤纶图像 | 是 | ② | False | 弹出警告 |

（5）条件组合覆盖测试

条件组合覆盖需要使得每个判断的所有可能的条件取值组合至少执行一次。

1）条件：

图像合法/图像不合法

2）所有条件组合情况：

①图像合法

②图像不合法

图5-2中共有1个判断，其中每个判断只有一个条件。因此本轮测试设计2组测试用例。

表5-6 图像数据处理模块白盒测试-条件组合覆盖测试

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序号 | 图像类型 | 是否重新输入 | 覆盖路径 | 条件 | 期望结果 |
| 1 | 梳棉胎图像 | 否 | SABCD | ① | 图像采集 |
| 2 | 涤纶图像 | 是 | SABE | ② | 弹出警告 |

（6）路径覆盖测试

路径覆盖需要使得每一条可能的路径至少执行一次。

所有可能路径：

①1-2-3-4-5

②1-2-3-6

图中共有2条可能路径，故测试用例如下。

表5-7 图像数据处理模块白盒测试-路径覆盖测试

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 图像类型 | 是否重新输入 | 覆盖路径 | 期望结果 |
| 1 | 梳棉胎图像 | 否 | ① | 图像采集 |
| 2 | 涤纶图像 | 是 | ② | 弹出警告 |

### 5.2.2黑盒测试

黑盒测试着重测试软件功能，黑盒测试并不能取代白盒测试，黑盒测试与白盒测试相互补充，黑盒测试可能发现白盒测试不容易发现的其他类型的错误。黑盒测试是从用户的角度出发，设计相关的测试条件，发现具体的错误。如功能不正确、界面错误、数据结构错误或外部数据库访问错误及性能错误。

白盒测试在测试过程的早期阶段进行，而黑盒测试主要用于测试过程的后期。

黑盒测试的测试用例集相较与白盒测试应满足两个条件：

条件一：测试用例应能减少为达到合理测试所需要设计的测试用例的总数。

条件二：所测试的用例条件应能够告诉人们是否存在某些类型的错误。

#### 5.2.2.1等价划分

等价划分法力图设计出能发现若干类程序错误的测试用例，从而减少必须设计的测试用例数目。

根据等价划分类的原则，划分出下列对应的有效等价类和无效等价类：

（1）账号信息

1）有效输入等价类有

①8位数字

②账号存在

③仅含数字不含其他字符

2）无效输入等价类有

①含7位数字

②含9位数字

③账号不存在

④除数字外含其他字符

（2）密码信息

1）有效输入等价类

①20位数字

②由数字开头

③仅由数字组成

2）无效输入等价类

①含19位数字

②含20位数字

③以其余字母开头

④含数字外的其余字符

根据上述划分的等价类，可以设计出下述的测试方案

表5-8 用户登录模块黑盒测试-等价类的划分

| **输入条件** | **有效等价类** | | **编号** | **无效等价类** | **编号** |
| --- | --- | --- | --- | --- | --- |
| 账号 | 8个数字 | （1） | | 7个数字  9个数字 | （4）  （5） |
| 纯数字 | （2） | | 除数字外的  其他符号 | （6） |
| 存在 | （3） | | 不存在 | （7） |
| 密码 | 20个数字 | （8） | | 19位数字  21位数字 | （11）  （12） |
| 以数字开头 | （9） | | 由字母开头  特殊字符开头 | （13）  （14） |
| 仅由数字组成 | （10） | | 还有除数字外  的其他字符 | （15） |
| 账号、密码  匹配 | 匹配 | （16） | | 不匹配 | （17） |

表5-9 用户登录模块黑盒测试的输出结果分析

|  |  |  |  |
| --- | --- | --- | --- |
| **编号** | **输入数据** | **覆盖等价类** | **预计输出** |
| （1） | 账号：12345678ab | (3)(4)(6) | 拒绝登录 |
| 密码：a1234567 | (8)(13)(15) |
| （2） | 账号：123456789102 | (1)(2)(5) | 拒绝登录 |
| 密码：1234567 | (9)(10)(11) |
| （3） | 账号： | (7) | 拒绝登录 |
| 密码：\*12345678 | (12)(14) |
| （4） | 账号：123456789102 | (2)(3)(5) | 拒绝登录 |
| 密码：12345678 | (8)(9)(10) |
| （5） | 账号：12345678 | (1)(2)(3) | 成功登录 |
| 密码：1234567910111213111 | (8)(9)(10) |

# 6 系统使用说明

概述段落

## 6.1 系统运行环境和配置

## 6.2 系统操作说明（按照结构图或层次图的框架依次介绍）

### 6.2.1 XX1模块说明

### 6.2.2 XX2模块说明

### 6.2.3 XX3模块说明

### 6.2.4 XX4模块说明

# 7 总结

# 参考文献