中北大学软件学院

实训说明书

实训名称: 面向对象程序设计实训

题目名称: JAVA 考试系统

专　　业: 软件工程

班　　级: 18130041

学号: 1813004113 姓名: 卢泽华 成绩:

指导教师: 李华玲

2020 年 1 月

实 训 任 务 书

1．实训的目的和要求：

通过实训，使学生进一步理解和巩固面向对象程序设计的基本概念、思想和核心技术；掌握面向对象程序设计的知识和技能；熟悉使用面向对象技术进行软件开发的过程；加强学生对 Java 知识的全面掌握；提高分析问题、解决问题的能力；锻炼动手能力、创新能力和综合应用的能力。

基本要求：

（1）题目需要用到数据结构与算法中的某种结构，线性结构、树结构或者图结构。

（2）使用 Java 完成开发，数据使用文件持久存储，界面必须是图形用户界面；

（3）按照软件工程的思想完成系统开发过程（包括需求分析、系统设计、编码实现和测试）；

（4）最终的软件系统要求数据充实、界面友好、使用方便。

2．实训的具体工作内容：

设计内容：

设计一个考试系统,实现人工与电脑组合进行判卷。

主要功能及要求：

（1）能够实现学生老师和管理员三种类型人员登录和学生的注册功能。

（2）学生可以通过登录进行题目解答,选择题进行系统自动判分。

（3）老师可以通过登录对学生的大题进行判分,最后给学生一个总分。

（4）管理员可以发布题目,以便更新题库。

（5）学生的答题过程中会有一个倒计时,时间到了之后不可以在进行答题,右边会显示当前题号、已答题数和未答题数来提醒学生。

实 训 任 务 书

3．对实训成果的要求：

提交实训说明书和光盘

每组的光盘目录内含：

（1）源程序文件；

（2）数据备份文件；

（3）实训说明书文档（另需交打印文档一份）。

4．实训工作进度计划：

起 迄 日 期	工 作 内 容
2019 年 12 月 19 日 ~ 12 月 22 日 12 月 23 日 ~ 12 月 31 日 **2020 年** 1 月 2 日 1 月 3 日 1 月 4 日 ~ 1 月 5 日	选择实训题目，明确实训任务，完成需求分析及系统分析与设计； 功能模块代码实现、调试，完善实例数据，系统测试； 撰写实训说明书文档； 验收程序、答辩； 修改、上交说明书

目录

1 绪论

1.1 概述

信息技术高度发达的今天,利用信息技术对大量复杂的信息进行有效的管理成为一种普遍而实用的手段。一方面，这极大的减少了簿记和人力的开销,另一方面,现代计算机强大的计算能力和网络的普遍部署，大大简化了大量信息的处理和流动。学生考试系统是评测学生能力的一个重要组成部分，他对老师的工作效率有很大的提高,它可以降低对纸质试卷的要求,同时也体现了节约型社会的要求。

该系统涉及了学生在线考试学习成绩查询以及很多相关信息的综合处理。为了方便配合教师对学生成绩的进一步了解,开发学生考试系统是当务之急。学生考试系统把学科、试题、电脑改卷、成绩查询的部分管理工作集成到一个统一的平台，各管理人员分工协作、相互配合，及时了解学生学习生活情况。同时,也可以方便老师针对学生个体不同情况进行分层次指导。

考试系统是针对学生和老师的大量考试处理工作而开发的管理系统。根据用户的要求，实现对学校学生的考试界面管理,。

用户在学生管理模块里面，通过输入学生的基本信息,登录考试界面,在进行题的做完之后会进行评分。在教师管理模块里，用户可以对学生所做的题进行评判。另外，管理员模块将进行题目的发布。

通过这个系统将更好的为学生和老师提供便利的服务。

1.2 项目背景

在大一C语言考试之后，发现电脑判卷的好处，随时出成绩，当你答完所有的提交之后，就知道自己考得好坏，不用再考试完在煎熬的等着成绩，但是我发现这个判卷系统是统一进行电脑判卷，尤其是在写大题的时候，好像是比较优化的匹配判卷，不管最后写的是否正确，只要和答案不一样，就会扣一些自己不知道的分，当学了数据结构和java之后，就萌发了这样一个想法，是否可以电脑与教师人工判卷，选择题进行人工判卷，而大题则进行老师判卷，这样就不会出现大题不知自己错在哪了的感觉，而且在一定情况下也提高了判卷的效率，为了更好的实现相应的功能，我将使用java的图形化界面来实现一些基础的功能，再根据文件操作的相互配合，模拟数据库，做成一个比较友好的界面，实现初步的功能，在之后的一段时间里，将会把所有的功能完善。

2 需求分析

软件需求分析就是通过调查研究和有效的沟通以及建立系统的数据模型、功能模型和行为模型，使用户和开发人员就“系统必须做什么”这个问题达成一致意见，最后生成软件需求规格说明的过程。软件需求分析的过程就是对用户或企业意图不断揭示的过程，需要对系统进行可行性研究之后，对系统目标做出分为详细的描述。

2.1 考试系统需求描述

考试系统主要是学生模块的设计与实现，学生模块的答题过程中，将分为两个部分，一为选择题部分，二位大题部分，这两部分将统一计时，当做完选择题是，将跳转到大题部分，选择题有 A、B、C、D 四个选项，点击下一题进入到下一个题，右侧会设置一个倒计时和未答的题来提醒学生的做题情况。当然学生也可以重新登录查看自己的成绩个个人信息。

老师可以通过登录教师系统进行判卷和查看所有学生的成绩。

管理员可以进行题目和人员管理。

2.2 系统需求分析

软件需求分为四个层次，分别是：业务需求、用户需求、功能需求和非功能需求。

2.2.1 业务需求

业务需求针对的是项目组或者公司，描述是公司想如何解决用户的问题，如何满足用户的欲望，并将利益最大化。

本项目课题是考试系统的设计与实现，在业务需求方面，第一，学生可以进行考试和查询成绩，选择题成绩在学生做完题时就可查看，而大题得需要老师的批判之后才能看见，两个的分加起来为学生的总分。第二，老师可以批判学生未被打分的试卷。第三，管理员进行题目和人员的管理。

2.2.2 用户需求

在该考试系统中，将实现用户注册和登录，学生修改信息、查看成绩、进行考试，退出登录等功能，用例图如图 2.1。

2.2.3 功能需求

本系统将划分为以下几个模块，如图 2.2 系统功能模块图。将考试系统按照功能划分为：

个人信息模块：对个人信息进行完善和修改。

随机试卷模块：在学生进入考试模块之前，将会随机生成一份试卷。

答案记录模块：完成考试之后会有该生专门的记录信息模块。

计时模块：在规定时间内做完试卷，否则将会强制退出。

分数模块：提交试卷之后将会记录一个分数。

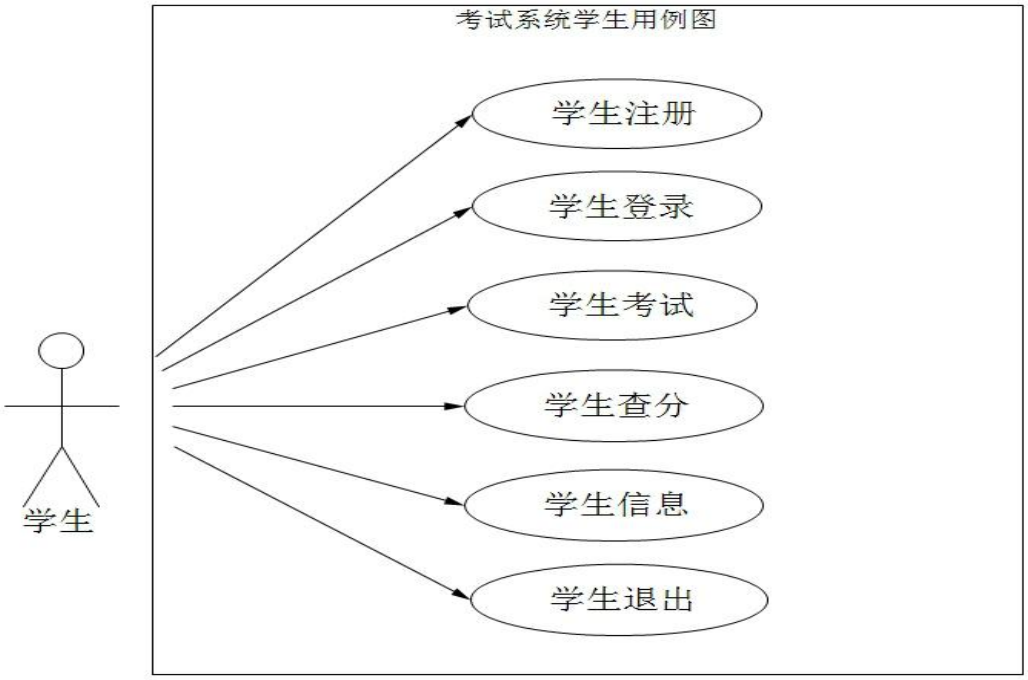

图 2.1 学生用例图

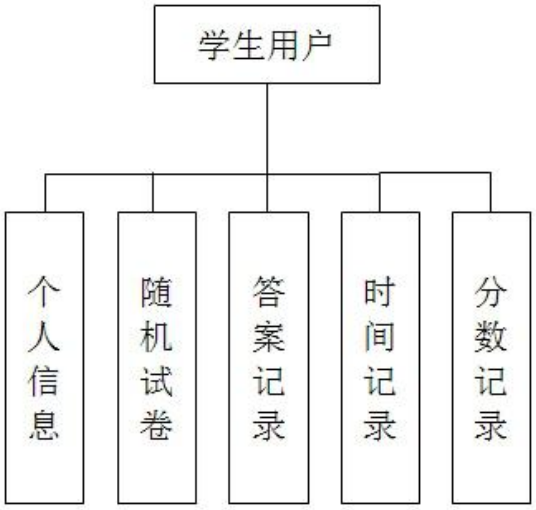

图 2.2 学生功能模块图

2.2.4 非功能需求

非功能需求包括产品必须遵从的标准、规范和约束，操作界面的具体细节和构造上的限制。

首先，软件制作为了设计、实现和后期维护的方便，以及软件用户使用的便利，须采取一定的设计原则。

其次，考试系统的界面也必须合理并且友好，否则也会出现一些不必要的麻烦，不会出现卡顿等现象。方便满足学生和老师之间相互答卷判卷之间的需求。

3 总体设计

通过本章将需求分析中的系统进行模块化，完成对一个复杂系统按功能进行模块划分、建立模块的层次结构及调用关系、确定模块间的接口及调用界面等。

3.1 模块设计

此次设计的考试系统的主体构成有学生老师和管理员模块。界面布局比较复杂，学生的答题界面分为三个面板，题目区域，选项区域和提示区域，老师的判卷区域分为两大区域，学生的原题及答案，还有打分区域，管理员的区域较为简单，只有管理信息的区域。本考试系统的用例图模块如图3.1。

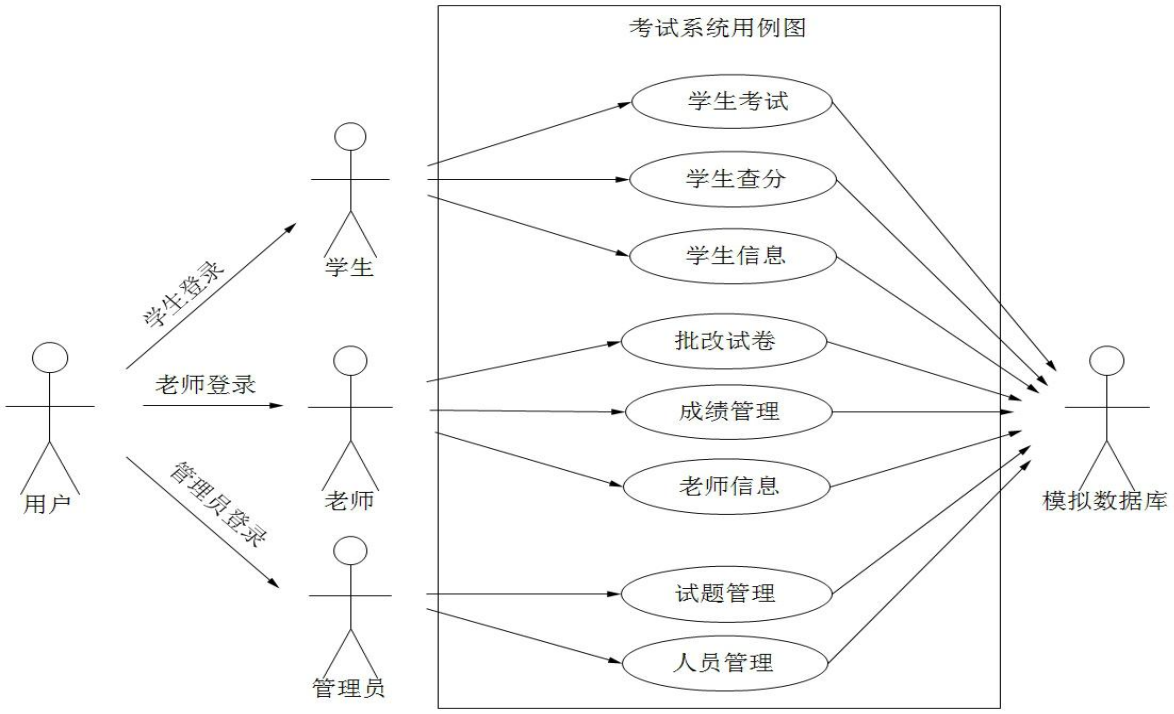

图 3.1 考试系统用例图

通过写的用例图实现了该考试系统的功能模块图，功能模块图将大体的实现功能列出来，根据不同层块实现不同的界面，功能模块图如图3.2所示。

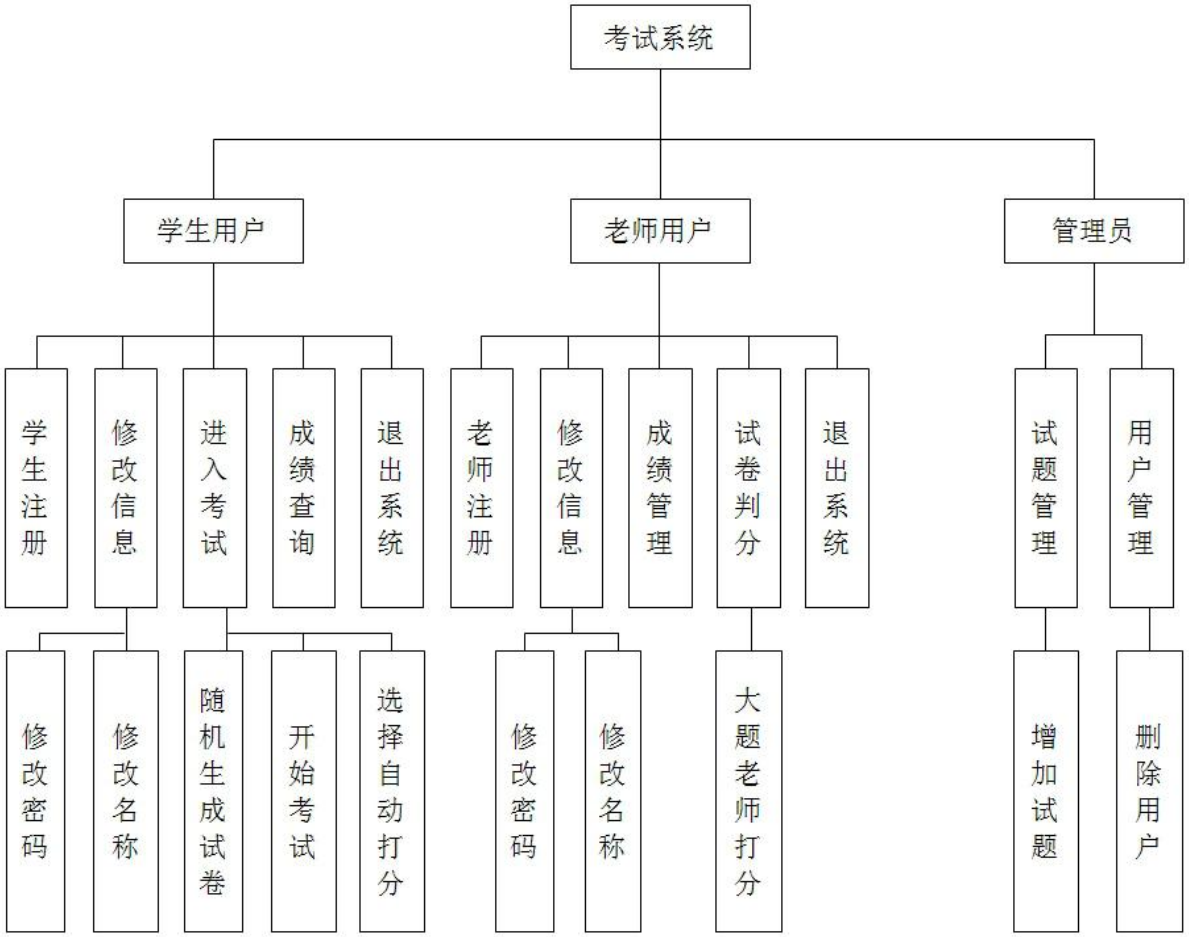

图 3.2 考试系统模块图

为了实现各模块的相应功能，必须写出相应的算法，最主要的是学生进行答题模块的功能，如何进行随机出题，学生答题和老师判分的顺序，以及管理员增加的题目等等，首先是登录模块，该部分功能将根据不同的用户登录不同的模块进入不同的页面，考生可以进行答题查分和修改自己的信息，老师可以修改信息和对学生的试卷进行判卷，最后管理员登录可以增加题型，选择题可以上传图片，下面我将以学生为主体，将该系统的所有信息联系起来，老师什么时候开始判卷，试卷的信息将必须在管理员添加相应的题目之后学生才能进行答题，答完题提交之后学生重新登录则会看到自己的选择题成绩，等待老师判卷之后学生将能看见自己的答题成绩，为了防止学生对同一套题不断的刷新，老师将只会看到学生最新的一套题，以及给该学生判分，判完分之后老师那将显示信息已经，具体如图3.3所示。

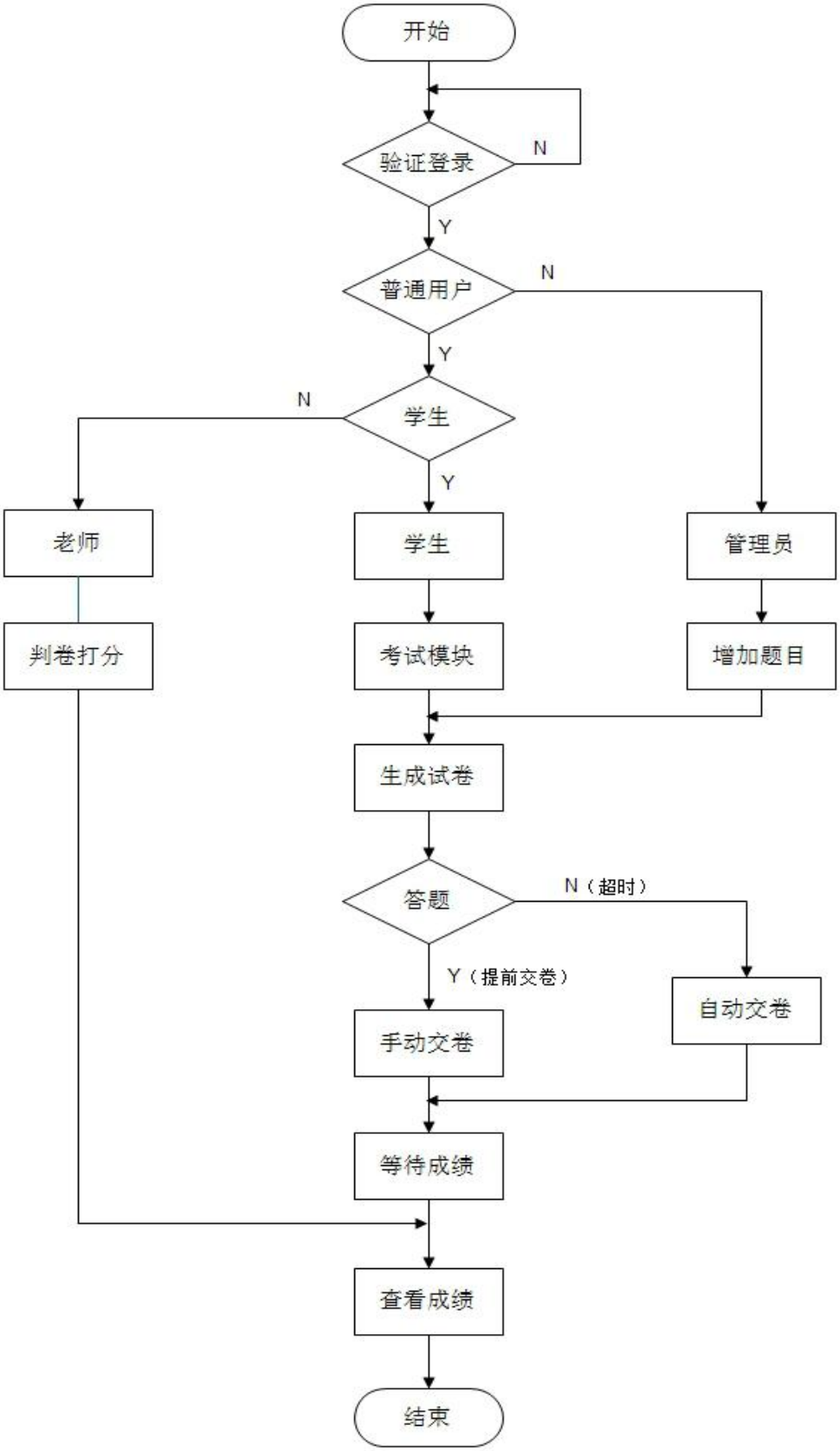
开始
验证登录
N
Y
普通用户
N
Y
学生
N
Y
老师
学生
管理员
判卷打分
考试模块
增加题目
生成试卷
答题
N（超时）
Y（提前交卷）
自动交卷
手动交卷
等待成绩
查看成绩
结束

图3.3 考试系统流程图

3.2 模块介绍

3.2.1 学生考试模块

学生考试模块分为3大部分，题目区域，按钮区域，计时提醒区域。

题目区域：从文件中读取题目，随机出现。

按钮区域：上下题按钮，ABCD选项按钮，提交按钮，大题回答按钮。

计时区域：有倒计时功能，当前题号，总共题数，已经回答数目，未回答数目。

3.2.2 老师判卷模块

老师判卷模块分为2大部分，题目区域，打分区域。

题目区域：题目和学生的答案两部分。

打分区域：输入分数区域和按钮区域。

3.2.3 管理员模块

管理员管理分为两大部分，人员管理和题目管理。

人员管理：包括学生和老师的删除等功能。

题目管理：包括题目的增加等功能。

3.3 确定类

在实现类图是将要用到用户类，选择题类、大题类和分数类，如图 3.1，图 3.1，图 3.3，图 3.4。

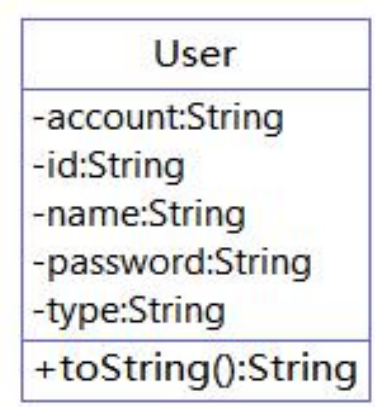

图 3.1 用户类图

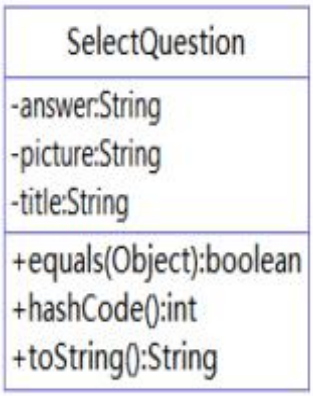

图 3.2 选择题类

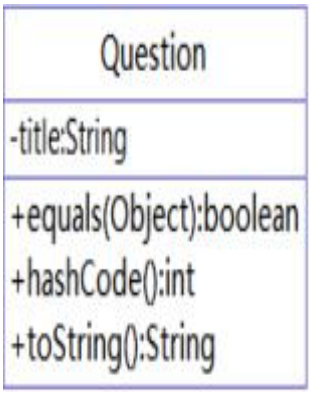

图 3.3 大题类

除此之外，还有一些 dao 层的文件操作类，service 层的服务类等等，在实现框架类的时候，为了方便，写了一个基础类共之后的类进行继承，这样就不容易写错，如图 3.5 中的部分展示。

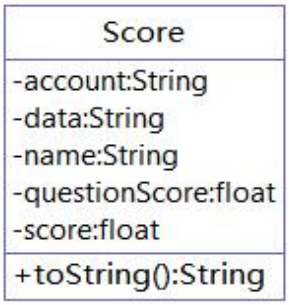

图 3.4 分数类

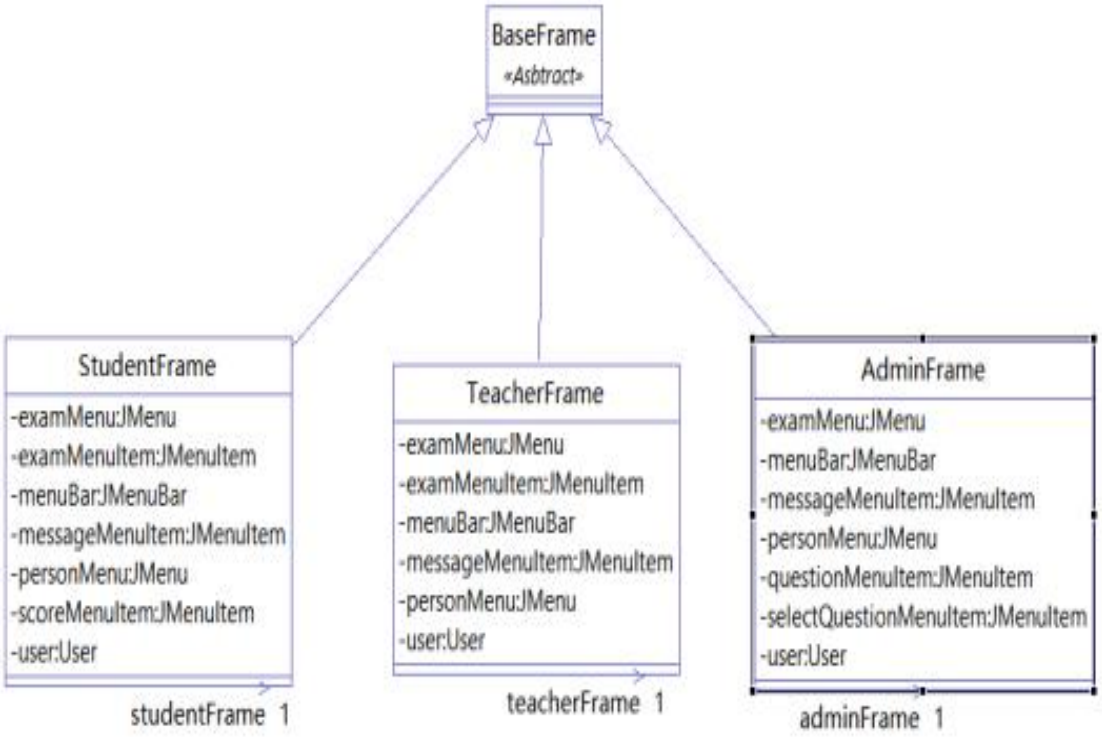

图 3.5 部分 GUI 图形界面图的类

3.4 本章小结

本章划分了考试系统的总体结构。通过对考试系统中各个部分的具体分析，设计并描述了各个小模块的概要内容，进而划分出此次项目中的类，绘制了类图。

4 详细设计

4.1 注册和登录模块的设计与实现

4.1.1 注册和登录界面

（1）注册

用户在进行注册时，需要通过该界面输入帐号、密码和用户身份，然后点击“注册”按钮进行注册，注册界面设计如图 4.1 所示。

(2)登录

用户在进行系统登录时，需要通过该界面输入帐号、密码和用户身份，然后点击“登录”按钮进行登录，登录界面设计如图 4.2 所示。

4.1.2 持久层

用文件 user.txt 持久存储用户的信息，文件中以卢泽华-140122-lzh-166666- 学生的形式存储，其中卢泽华为名字，140122 为身份证号，lzh 为用户账号，166666 为用户密码，学 生为用户类型。所有用户的信息均以这样的格式存储，且每个用户的信息在文件中占一行。

图 4.1 注册页面

图 4.2 登录页面

为了方便，在类 DatabaseConfig 中使用静态常量描述了文件 user.txt 的详细路径。

```
public class DataBaseConfig {
    public static final String
   USER_FILE_PATH=“src\\nuc\\ss\\examination\\db\\User.txt”
}
```

4.1.3 实体类 User

该类主要用于封装用户的信息：帐号、密码和用户类型等信息。类图如图3.1。

4.1.4 文件操作层（Dao层）

该层涉及到接口IUserDao和实现该接口的类UserDaoImpl，主要用来完成对文件

user.txt的读和写操作。

为了分离文件的读和写的操作，我有加了一个util层，把读和写分开

IUserFileReader和UserFileReaderImpl负责读文件的操作IUserFileWriter和UserFileWriterImpl负责写文件的操作

（1）读文件的方法：public User getUser(String accoun);

public HashMap<String, User> getAllUser()

（2）读文件的方法：public void addUser(User user)和

public void revise(User user);

public void remove (User user) ;

（3）核心代码：// static修饰，是为了保证这个集合是唯一的

```
private static HashMap<String, User> userBox = new HashMap<>();

{
    BufferedReader reader = null;
    try {
        reader = new BufferedReader(new FileReader(DatabaseConfig.USER_FI
        LE_PATH));
        String message = reader.readLine();
        while (message != null) {
            String[] values = message.split("-");
            String name = values[0];
            String id = values[1];
            String account = values[2];
            String password = values[3];
            String type = values[4];
            userBox.put(account, new User(name, id, account, password,type));
            message = reader.readLine();
        }
    } catch (Exception e) {
```

```
            e.printStackTrace();
        } finally {
            ...
        }
    }
    public User getUser(String accoun) {
        return userBox.get(accoun);
    }
    public HashMap<String, User> getAllUser() {
        return userBox;
    }
  public void addUser(User user) {
        BufferedWriter writer = null;
        try {
        writer = new BufferedWriter(new
  FileWriter(DatabaseConfig.USER_FILE_PATH, true));
            writer.write(user.toString());
            writer.newLine();
        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            ...
        }
    }
public void revise(User user) {
        IUserFileReader userFileReader = new UserFileReaderImpl();
        HashMap<String, User> users = userFileReader.getAllUser();
        StringBuffer stringBuffer = new StringBuffer();
        BufferedWriter writer = null;
```

```
        Iterator<Entry<String, User>> iter = users.entrySet().iterator();

        try {
         writer = new BufferedWriter(new FileWriter(DatabaseConfig.USE
         R_FILE_PATH));
            while(iter.hasNext()) {
                Entry<String, User> entry = iter.next();
                if(user.getAccount().equals(entry.getValue().getAccount())) {
                    stringBuffer.append(user.getName() + "-" + user.getId() + "-"
+ user.getAccount() + "-" + user.getPassword() + "-" + user.getType() + "\n");
                }else {
                    stringBuffer.append(entry.getValue().getName() + "-" +
entry.getValue().getId() + "-" + entry.getValue().getAccount() + "-" +
entry.getValue().getPassword() + "-" + entry.getValue().getType() + "\n");
                }
            }
            writer.write(stringBuffer.toString());
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            ...
        }
    }
    @Override
    public void remove(User user) {
        File f = new File(DatabaseConfig.USER_FILE_PATH);
        FileWriter fw = null;
        BufferedReader br = null;
```

```
        BufferedWriter bw = null;
        StringBuffer buff = new StringBuffer();

        try {
            br = new BufferedReader(new FileReader(f));
            for (String str = br.readLine(); str != null; str = br.readLine()) {
                String [] strr = str.split("-");
                if (strr[2].equals(user.getAccount()));
                else {
                    buff.append(str+ "\r\n");
                }
            }
            fw = new FileWriter(f);
            bw = new BufferedWriter(fw);
            bw.write(buff.toString());
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            ...
        }
    }
```

4.1.5 业务逻辑层（Service层）

该层涉及到接口 IUserService 和实现该接口的类 UserServiceImpl，主要用来完成注册逻辑和登录逻辑。

（1）注册逻辑

从业务角度，在进行注册时，首先需要判断用户名和密码是否符合约定的格式（用户名：字母组成；密码：6~12 位字母数字构成。需要使用正则表达式。），如果格式正确，需要判断该用户是否已经注册过，没有注册过，写入文件，返回注册成功；如果格式不正确或者已经注册过，均需要返回错误提示信息。

public String revise(User user);该方法实现用户 user 的注册逻辑。

（2）登录方法

public String login(User user)

从业务角度，在进行登录时，需要判断该用户是否是一个合法用户，合法，允许登录系统，否则，不允许登录。

(3)删除用户

public String remove(User user)

管理员登录可以删除用户

4.1.6 界面层（窗体层）

本层包括三个窗体类：LoginFrame、AdminFrame、TeacherFrame 和 StudentFrame，运行结果如图 4.1，4.3，4.4，4.5。

4.2 学生答题模块的设计与实现

4.2.1 选择题和大题界面

（1）选择题

学生登录成功后点击进入考试，将会进入选择题考试系统，选择题考试系统如图 4.6 所示。

（2）大题界面

学生点击提交试卷之后将进入大题页面，大题考试系统如图 4.7 所示。

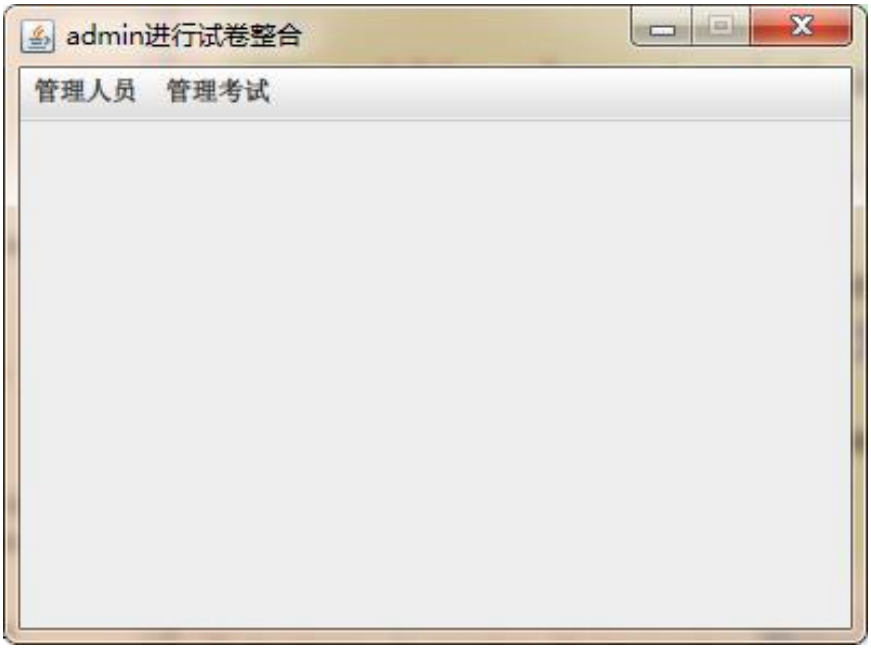

图 4.3 管理员登录系统

图 4.4 学生登录系统

图 4.5 老师登录系统

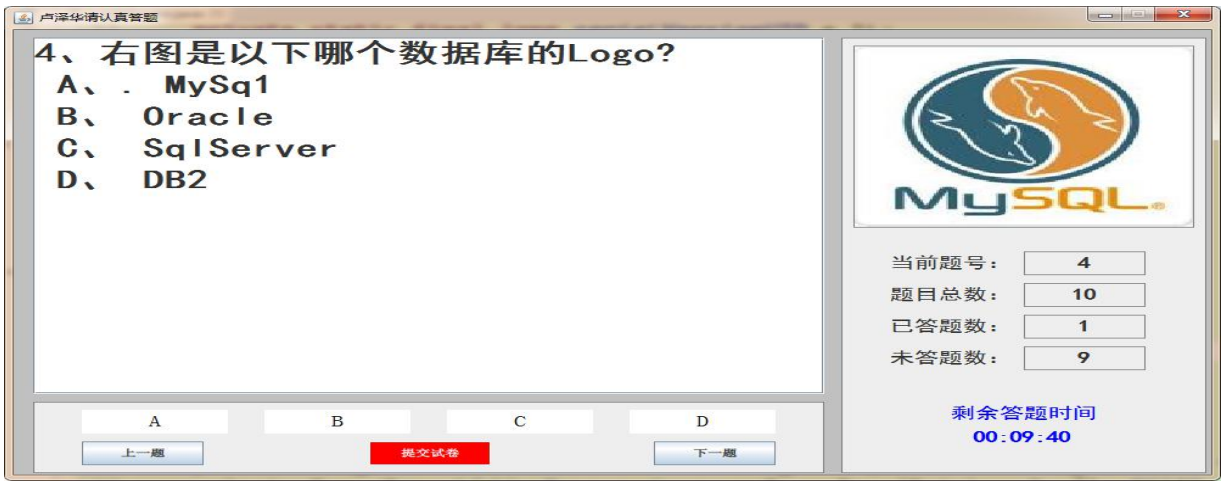

图 4.6 选择题界面

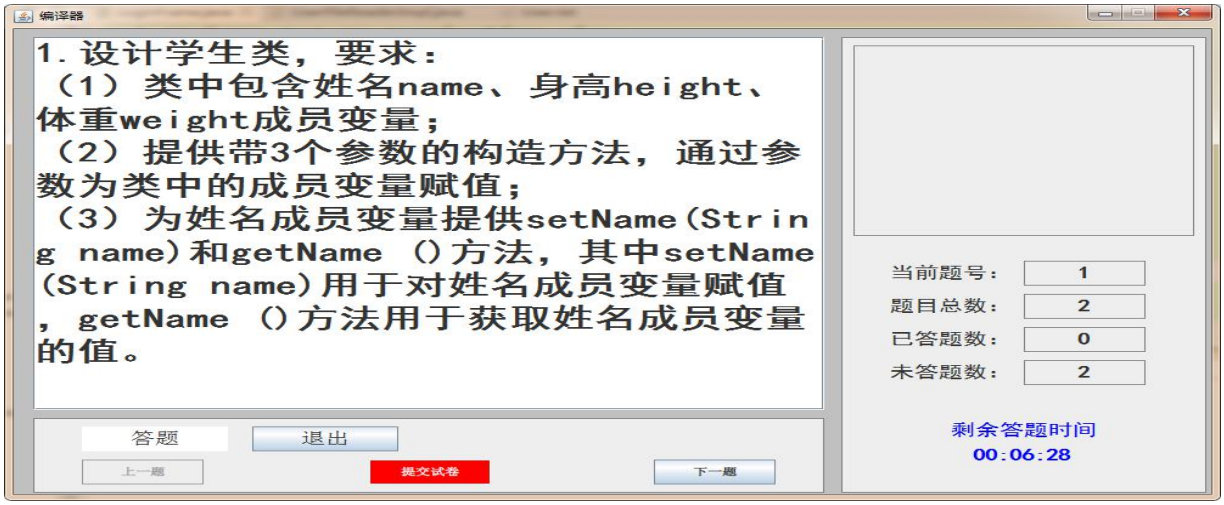

图 4.7 大题界面

4.2.2 持久层

用文件 selectQuestion.txt 和 question.txt 持久存储选择题和大题题型，文件中选择题以“右图是以下哪个语言的 Logo?
 A、 C
 B、 Java
 C、 PHP
 D、VB#B#Java.jpg”的形式存储，
表示换行，B 表示答案，Java.jpg 表示图片信息。文件中大题以“设计学生类，要求：

（1）类中包含姓名 name、身高 height、体重 weight 成员变量；

（2）提供带 3 个参数的构造方法，通过参数为类中的成员变量赋值；

（3）为姓名成员变量提供 setName(String name)和 getName ()方法，其中 setName(String name)用于对姓名成员变量赋值，getName ()方法用于获取姓名成员变量的值。*”的形式存储，
表示换行，*表示这个题目结束。

为了方便，在类 DatabaseConfig 中使用静态常量描述了文件 user.txt 的详细路径。

```
public class DataBaseConfig {
public static final String SELECT_QUESTION_FILE_PATH =
"src\\nuc\\ss\\examination\\db\\SelectQuestion.txt";
public static final String QUESTION_FILE_PATH =
"src\\nuc\\ss\\examination\\db\\Question.txt";
}
```

4.2.3 实体类 Question 和 SelectQuestion

该类主要用于封装问题的题目，答案和图片。类图如图 3.2 和 3.3。

4.2.4 文件操作层（Dao 层）

该层涉及到接口ISelectQuestionDao和实现该接口的类SelectQuestionDaoImpl，主要用来完成对文件selectQuestion.txt的读和写操作,该层涉及到接口IQuestionDao和实现该接口的类QuestionDaoImpl，主要用来完成对文件question.txt的读和写操作。

为了分离文件的读和写的操作，我有加了一个util层，把读和写分开。

接口ISelectQuestionFileReader和实现该接口的SelectQuestionFileReaderImpl完成选择题的读入操作，接口ISelectQuestionFileWriter和实现该接口的类的QuestionWriterImpl完成选择题的写入操作,接口IQuestionFileReader和实现该接口的类QuestionFileReaderImpl完成大题的读入操作,接口IQuestionFileWriter和实现该接口的类QuestionWriterImpl完成大题的写入操作。

（1）文件写入方法：public void addSelectQuestion(SelectQuestion

selectQuestion)管理员增加题目；

public void addQuestion(Question question)管理员增加大题；

（2）文件读入方法：public ArrayList<SelectQuestion> getSelectPaper(int count)随机产生选择题，顺序不一，题目不一；

public ArrayList<Question> getPaper(int count);随机产生大题，题目不一，大题不一。

（3）核心代码：

①public HashSet<SelectQuestion> getSelectQuestions(){

```
BufferedReader reader = null;
 try {
     reader = new BufferedReader(new
FileReader(DatabaseConfig.SELECT_QUESTION_FILE_PATH));
     String question = reader.readLine();
     while(question != null){
         String[] values = question.split("#");
         String title = values[0];
         String answer = values[1];
         if(values.length == 2){//没有图片信息，只有题干和答案
             questionBox.add(new SelectQuestion(title,answer));
         }else if(values.length == 3){//含有图片信息
             questionBox.add(new
SelectQuestion(title,answer,values[2]));
         }
         question = reader.readLine();
     }

 } catch (Exception e) {
     e.printStackTrace();
 }finally {
```

```
            ...
        }
        return questionBox;
    }
    ②public void addSelectQuestion(SelectQuestion selectQuestion) {
          BufferedWriter bWriter = null;
          try {
              bWriter = new BufferedWriter(new
    FileWriter(DatabaseConfig.SELECT_QUESTION_FILE_PATH,true));
    bWriter.write(selectQuestion.getTitle()+"#"+selectQuestion.getAnswer());
              bWriter.newLine();
          } catch (IOException e) {
              e.printStackTrace();
          } finally {
              if(bWriter != null) {
                  try {
                      bWriter.close();
                  } catch (IOException e) {
                      e.printStackTrace();
                  }
              }
          }
      }
     //获取缓存对象
    ③ private ISelectQuestionFileReader selectQuestionFileReader = new
SelectQuestionFileReaderImpl();
    //将缓存中的集合临时改为list集合(有序可重复)
    private ArrayList<SelectQuestion> selectQuestions = new
    ArrayList<SelectQuestion>(selectQuestionFileReader.getSelectQuestions())
```

```
    public ArrayList<SelectQuestion> getSelectPaper(int count){
        HashSet<SelectQuestion> paperSelectQuestions = new HashSet<>();//用来存储最终的试卷题目（无序无重复）
        while(paperSelectQuestions.size() != count){
            Random r = new Random();
            int index = r.nextInt(selectQuestions.size());//随机数范围[0,集合长度)
            paperSelectQuestions.add(selectQuestions.get(index));
        }
        return new ArrayList<SelectQuestion>(paperSelectQuestions);
    }
```

④ private IQuestionFileReader questionFileReader = new QuestionFileReaderImpl();

```
    //将缓存中的集合临时改为list集合(有序可重复)
    private ArrayList<Question> questions = new ArrayList<Question>(questionFileReader.getQuestions());
    public ArrayList<Question> getPaper(int count) {
        HashSet<Question> paperQuestions = new HashSet<>();//用来存储最终的试卷题目（无序无重复）
        while(paperQuestions.size() != count){
            Random r = new Random();
            int index = r.nextInt(questions.size());//随机数范围[0,集合长度)
            paperQuestions.add(questions.get(index));
        }
        return new ArrayList<Question>(paperQuestions);
    }
```

⑤private HashSet<Question> questionBox = new HashSet<>();

```
    //获取全部集合
    public HashSet<Question> getQuestions(){
```

```
        BufferedReader reader = null;
         try {
             reader = new BufferedReader(new FileReader(DatabaseConfig.QUESTI
ON_FILE_PATH));
             StringBuffer buffer = new StringBuffer("");
             String line = reader.readLine();
             while(line != null){
               if(line.equals("*")) {
                   questionBox.add(new Question(buffer.toString()));
                   line = reader.readLine();
                   buffer.setLength(0);
                   continue;
               }
                 buffer.append(line);
                 line = reader.readLine();
             }
         } catch (Exception e) {
             e.printStackTrace();
         }finally {
             ...
         }
         return questionBox;
    }
    ⑥public void addQuestion(Question question) {
           BufferedWriter bWriter = null;
           try {
              bWriter = new BufferedWriter(new
FileWriter(DatabaseConfig.QUESTION_FILE_PATH, true));
              bWriter.write(question.getTitle() + "\n*");
```

```
            bWriter.newLine();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            ...
        }
    }
```

4.2.5 业务逻辑层（Service层）

该层涉及到接口IQuestionService和实现该接口的类QuestionServiceImpl，接口ISelectQuestionService和实现该接口的类SelectQuestionServiceImpl,主要用来完成大题和选择题的生成试卷逻辑和增加题目逻辑。

4.2.6 界面层（窗体层）

本层包括三个窗体类：ExamSelectQuestionFrame、ExamQuestionFrame、和CompilerFrame，运行结果如图4.6，4.7，4.8。

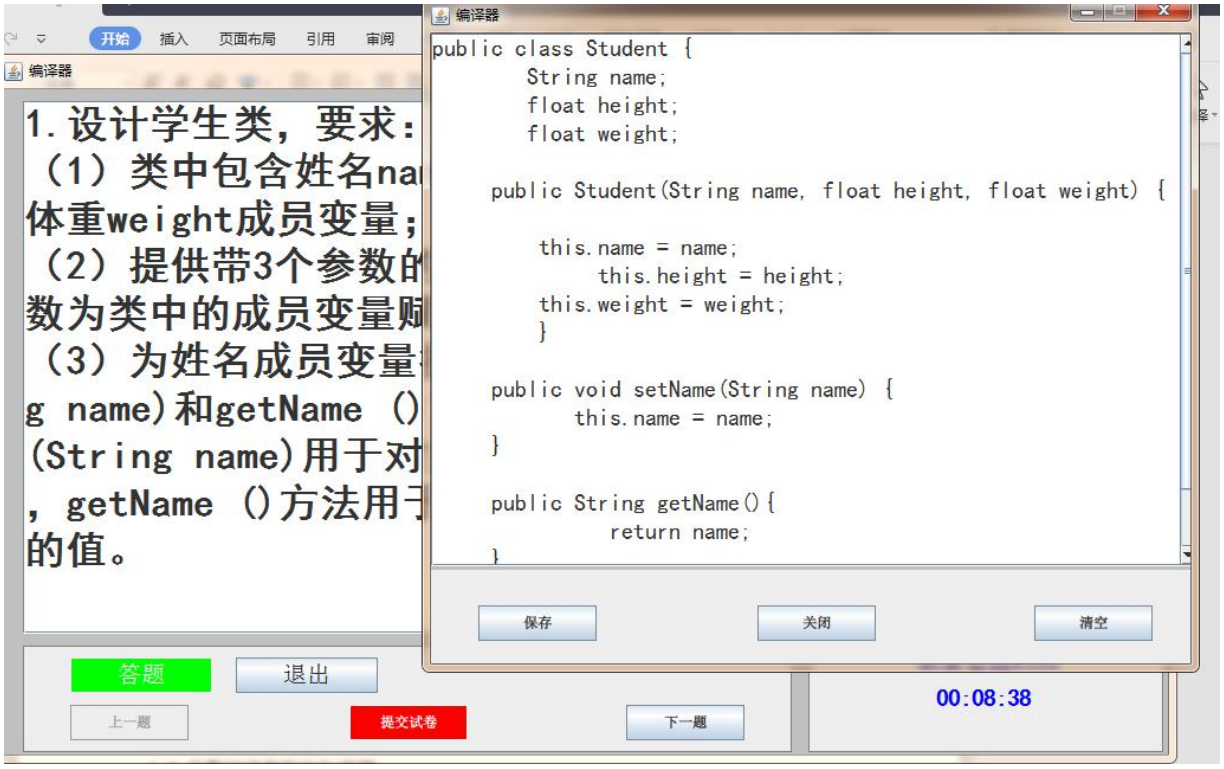

图4.8 编译器界面

当点击答题的时候会弹出一个编译器，当你再次进入的时候写的东西依然存在。

4.3 分数记录的设计与实现

4.3.1 学生查分和老师查分判分界面

学生登录成功后通过查看成绩可以看到自己所有已经答题的记录，如图4.9所示

图4.9学生查自己的分数

老师进行判卷之前，也会先进入一个分数记录的列表，这个列表是最新一次学生的答题情况。老师可点击打分进行判卷，如图4.10所示。

图4.10老师判卷前学生的分数

4.3.2 持久层

用文件分数.txt持久存储用户的信息，文件中以lzh-卢泽华-2020年01月01日19时27分-32.0-40.0来存储学生的成绩，“lzh”表示该生的账号，“卢泽华”表示该生的姓名，

“2020年01月01日19时27分”表示该生交卷的时间，“32.0”表示该生的选择题成绩，“40.0”表示该生的答题成绩。

所有用户的信息均以这样的格式存储，且每个用户的信息在文件中占一行。

4.3.3 实体类Score

该类主要用于封装分数。类图如图 3.4。

4.3.4 文件操作层（Dao 层）

该层涉及到接口 IStudentScoreDao 和实现该接口的类 StudentScoreDaoImpl 来对学生的分数进行读和写的操作，显示学生的成绩。

（1）读得操作：

public Score selectone(String account)读取学生的一个分数；

public HashMap<String, Score> selectAll()读取学生的一组不重复分数；

public List<Score> scores()读取学生的所有分数。

（2）写的操作：

public void saveScore(float score,String data,User user)学生提交时存储选择题分数，电脑判卷；

public void saveScore(float score, User user)老师判卷后存储大题分数。

（3）核心代码：

```
public void score(float score,String data, User user) {
    BufferedWriter writer = null;
    try {
        writer = new BufferedWriter(new FileWriter(DatabaseConfig.STUDENT + "分数 .txt",true));
        writer.write(user.getAccount() + "-" + user.getName() + "-" + data + "-" + score);
        writer.newLine();
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
        ...
    }
```

```
}
    public void selectAnswer(String data, User user, String[] str) {
            BufferedWriter writer = null;
            try {
                writer = new BufferedWriter(new
FileWriter(DatabaseConfig.STUDENT + "选择 .txt",true));
                writer.write(data + "-" + user.getAccount() + "-" + data + "-");
                int i = 0;
                for (i = 0; i < str.length-1; i++) {
                    writer.write(str[i] + "-");
                }
                writer.write(str[i]);
                writer.newLine();
            } catch (IOException e) {
                e.printStackTrace();
            } finally {
                ...
            }
        }
```

4.3.5 业务逻辑层（Service层）

该层涉及到接口IStudentScoreService和实现该接口的类StudentScoreServiceImpl，主要用来完成学生分数的存储。

4.3.6 界面层（窗体层）

本层包括三个窗体类：StudentScoreFrame、TeacherExamScoreFrame 和 TeacherExamFrame，运行结果如图 4.9，4.10 所示，当老师点击打分的时候，会进入相应的打分界面以及打完分之后的学生信息如图 4.11 和图 4.12 所示。

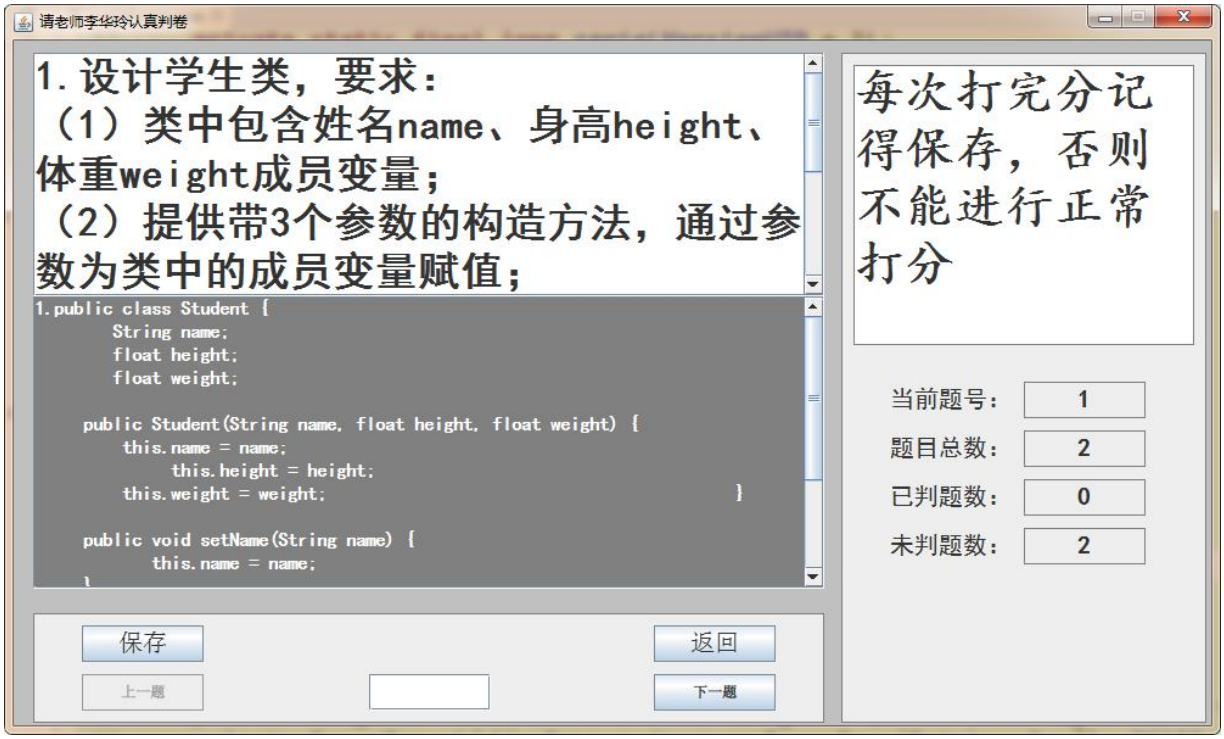

图4.11老师打分界面

图4.12老师打完分后的界面

心得体会

通过这次java课程设计训练，让我受益匪浅，由于这次我是自拟题目，很多事情都必须自己去构思并且实现，在写代码的过程中再一次熟悉了图形化界面，线程，集合框架等知识，重要的是还与之前的数据结构知识相联系起来，为了实现倒计时的功能，用到了内部类和线程的结合，而且还遇到了不少问题，比如如何在一个界面中有多个题目的同一组选项如何保留记忆，比如中英文字符的不同，比如如何将HashSet<>()和ArrayList<>()结合起来让试卷出现的顺序是随机的等等，不过最后在上网查阅，查找书籍和老师的帮助下都已经解决。

参考文献

[1] 孙连英 刘畅 彭涛．Java 面向对象程序设计．北京：清华大学出版社，2017

[2] 严蔚敏 吴伟民．数据结构．北京：清华大学出版社，2018

[3] 张海藩 牟永敏．软件工程导论（第六版）．北京：清华大学出版社，2013

[4] Cay S.Horstmann Gary Cornell．Java 核心技术 I: ．北京：机械工业出版社，2008

[5] Cay S.Horstmann Gary Cornell．Java 核心技术 II．北京：机械工业出版社，2008