****

**《软件工程与开发实践》**

**软件设计报告**

|  |  |
| --- | --- |
| **题 目** | 基于Android开发平台的班级管理系统 |
| **学 院** | 计算机学院 |
| **专 业** | 计算机科学与技术专业 |
| **班 级** | 软件工程实践班 |
| **姓 名** | 李顺(16071209) 崔东彬(16051310)  徐艳玲(16061404) 周平(16051334)  赵婧娟(16051207) |
| **组 长** | 李顺 |
| **指导教师** | 张建海 |
| **完成日期** | 2019 年7月7日 |

**目录**

[1软件设计概述 3](#_Toc13385818)

[1.1项目背景 3](#_Toc13385819)

[1.2项目功能 3](#_Toc13385820)

[1.3项目意义 3](#_Toc13385821)

[1.4项目任务 4](#_Toc13385822)

[1.5开发环境 4](#_Toc13385823)

[1.6参考资料 5](#_Toc13385824)

[2可行性研究与需求分析 5](#_Toc13385825)

[2.1可行性研究 5](#_Toc13385826)

[2.1.1技术可行性 5](#_Toc13385827)

[2.1.2操作可行性 5](#_Toc13385828)

[2.1.3社会可行性 5](#_Toc13385829)

[2.2需求分析 6](#_Toc13385830)

[2.2.1总体功能需求 6](#_Toc13385831)

[2.2.2软件开发平台需求 6](#_Toc13385832)

[2.2.3软件需求分析 6](#_Toc13385833)

[2.2.4外部接口需求 8](#_Toc13385834)

[2.2.5系统功能需求 9](#_Toc13385835)

[2.2.6其他非功能需求 14](#_Toc13385836)

[3软件设计的基本原理和采用的主要方法与技术 15](#_Toc13385837)

[3.1基本原理 15](#_Toc13385838)

[3.2方法技术 16](#_Toc13385839)

[4软件架构设计和模型设计 17](#_Toc13385840)

[4.1软件架构 17](#_Toc13385841)

[4.2数据流程图 18](#_Toc13385842)

[4.3模型设计 19](#_Toc13385843)

[4.4算法设计 19](#_Toc13385844)

[5软件模块划分与人员任务分工 19](#_Toc13385845)

[5.1模块划分 19](#_Toc13385846)

[5.2小组分工 20](#_Toc13385847)

[6软件实现 20](#_Toc13385848)

[6.1UI设计 20](#_Toc13385849)

[6.2数据库实现 21](#_Toc13385850)

[6.3用户模块 21](#_Toc13385851)

[6.4显示资料列表模块 22](#_Toc13385852)

[6.5统计模块 22](#_Toc13385853)

[6.6添加修改资料模块 23](#_Toc13385854)

[6.7搜索模块 24](#_Toc13385855)

[6.8排序模块 25](#_Toc13385856)

[7软件集成与测试 25](#_Toc13385857)

[7.1基本情况 25](#_Toc13385858)

[7.1.1单元测试概况 25](#_Toc13385859)

[7.1.1集成测试概况 27](#_Toc13385860)

[7.2单元测试计划 27](#_Toc13385861)

[7.2.1登录单元 27](#_Toc13385862)

[7.2.2密码修改单元 28](#_Toc13385863)

[7.2.3查找学生信息单元 29](#_Toc13385864)

[7.2.4排序单元测试 31](#_Toc13385865)

[7.2.5删除信息单元测试计划： 32](#_Toc13385866)

[7.2.6通讯单元测试计划： 33](#_Toc13385867)

[7.2.7模块票务预定单元测试计划： 34](#_Toc13385868)

[7.3集成测试 35](#_Toc13385869)

[7.3.1增添学生信息测试计划 35](#_Toc13385870)

[7.3.2修改学生信息测试计划 37](#_Toc13385871)

[8个人工作总结 38](#_Toc13385872)

[9附录 48](#_Toc13385873)

[9.1会议记录（签名） 48](#_Toc13385874)

[9.2个人完成的程序模块核心代码（含注释） 51](#_Toc13385875)

[10声明承诺 63](#_Toc13385876)

### 1软件设计概述

#### 1.1项目背景

随着智能手机等的日益普及，移动应用发展快速。使用移动端为广大师生提供更好的校园信息服务已经成为数字校园建设的重要组成部分。互联网+与班级管理相结合，能够更加快捷的完成班级管理任务。在高校班级管理中，智能手机的及和网络化校园的实现为网络媒体的运用提供了客观的条件，拓展网络平台的班级管理新模式是做好大学生思想教育和才教育的迫切要求。另外，随着学分制管理的全面推行，学生在更加自主化的完成课堂学习的同时，行政班集的概念淡化，班级凝聚力降低，这为班级管理带来了新的挑战。

开发一款班级管理 APP，可以实现开放、平等的交流模式。它能为师生提供便捷的交流平台，使其摆脱时间和空间的限制，使班风更加开放和民主，更能进一步激发学生参加班级管理的热情，有助于培养同学们的主人翁精神。同时有助于及时了解学生状况，为辅导员提高工作效率，促进师生关系进一步稳定。班会等传统班级活动所提供的师生、生生交流的时间与话题非常有限。而班级管理 APP 提供的“社区”功能，方便了师生的信息发布。这样，当你有困难或困惑时，老师和同学们通过 APP 可以非常方便为你排忧解难，这样可以进一步增进师的感情，也便于问题的解决。

#### 1.2项目功能

本产品作为班级信息管理系统，可以使管理员对学生资料进行录入、删除、更改、查询、统计等功能，并具备通讯功能方便教师与学生之间的信息交流，且有些功能涉及到排序操作；而学生账号则可以查询、更改自己的个人资料以及对班级资料的统计。

#### 1.3项目意义

通过该应用系统的研究与开发，将大大方便老师的工作需求，顺应了时代发展的潮流；充分利用班级信息资源，减轻教师和老师工作的负担，提高老师的工作效率；将更进一步推进学校管理现代化和信息化，为用户提供充足的信息和快捷的查询方式。

班级信息管理是班级管理不可缺少的部分，它的内容对于班级的老师和学生都非常重要，它能高效快速的提供数字信息化管理，能够有效的提高班级管理的质量和效率，并且在这个基础上实现了信息共享，为老师更好地进一步幵展学生工作打下良好基础 。

#### 1.4项目任务

* 工作环境的搭建；
* 可行性研究；
* 需求分析；
* 功能模块划分；
* 建立系统逻辑模型；
* 进行架构、接口、界面、数据库、模块等设计；
* 对数据库SQLite、服务器以及客服端编码；
* 集成测试与bug修正；
* 文档撰写。

#### 1.5开发环境

* JDK（Java Development Kit）；
* SDK（Software Development Kit）;
* Android Studio3.4.1。

#### 1.6参考资料

[1]马洪连.基于Android平台的班级管理系统的设计与实现[D].大连:大连理工大学,2012.

[2]李刚.疯狂Android讲义(第3版).北京:电子工业出版社,2015.

[3]张海藩,牟永敏.软件工程导论(第6版).北京:清华大学版社,2013.

### 2可行性研究与需求分析

#### 2.1可行性研究

##### 2.1.1技术可行性

安卓(Android)系统是一种由Google公司和开放手机联盟领导及开发的基于 Linux内核的开源操作系统。目前，安卓系统占据了智能手机市场的主导地位，而基于 Android 平台开发的APP也极其丰富。Android系统以其开源性与丰富的扩展性受到用户的支持。因此，掌上校园 APP 选择了基于安卓平台进行设计和开发。

在进行了系统分析和设计之后，安装配置JDK，在Eclipse中安装Android SDK 工具包和ADT插件就可以完成班级管理APP的开发。

##### 2.1.2操作可行性

该系统的界面友好，窗口操作简单、直观、易于掌握，帮助信息详尽，一般人员都可以使用；并且该系统开发前的准备活动、开发以及系统的运行等活动不违反现行的法律法规，没有对他人的专利、版本构成侵犯，与现行的管理制度没有冲突。

##### 2.1.3社会可行性

随着校园内智能手机的迅速普及，服务于学生校园生活的APP也日益受到学生的青睐。针对班级管理实际开发的班级管理APP可以服务于学校的学生和教师，广大师生随时随地掌握校园和班级的信息，方便了师生的校园生活，提高了班级管理的质量，提高了校园信息化水平，优化了校园服务质量。

#### 2.2需求分析

##### 2.2.1总体功能需求

该产品主要是一个班级信息管理系统，具有对班级学生资料录入的基本功能（学生的学号，姓名，性别，生日，籍贯，专业，成绩，电话等），并将学生基本信息保存在SQLite数据库中；针对录入的学生信息，要有信息管理：修改、查询、统计、删除。针对统计，比如对班级考试成绩的查询可以获得相应的最高分、平均分、最低分，以及及格人数、不及格人数等基本信息，并且学生信息涉及到了排序操作。而拓展功能有：针对信息管理的权限，实现登录账号权限识别及授权功能；在学生信息编辑页面中，实现直接拨打学生电话功能，满足老师与学生的信息交流需求等。

##### 2.2.2软件开发平台需求

开发者开发的软件必须能够在android平台上正常运行。

目前开发软件平台为：

* 数据库管理系统：SQLite；
* 中间件(应用服务器)系统：IBM WebSphere；
* Android系统：基于Linux的自由及开放源代码的操作系统；
* 网络架构：完全支持TCP/IP协议；

开发工具或技术体系：为保证软件的上下兼容性，开发者应选择较通用的开发工具的较新版本进行开发，android studio3.4.1。

##### 2.2.3软件需求分析

**a软件范围**

该软件在功能上的范围为：

* 班级学生信息管理；
* 班级学生成绩统计；
* 账号权限识别及授权；
* 拨打电话；

**b软件风险**

* 软件管理的过程中：

软件开发在不同的工程阶段需要的人员不同，需要配合的方面也不同，所有这些都需要行之有效的软件管理的保证，进而使得App软件要能够按工期的要求完成。并且由于此次试验的时间原因，app的软件的需求调研并不是很透彻，不能保证开发的软件为用户所接受。

* 建立运营的过程中：

对学生信息的保密安全需求较高，这方面如果较为薄弱则会产生学生个人信息泄露的风险，对学生的前程有一定威胁，进而被破坏软件的信用等级。甚至可能会有部分不法分子盗取这些信息以用作某种盈利行为。

**c.软件的功能**

这款软件主要的功能就在于为用户建立一个班级信息管理平台，为老师提供信息管理服务，为学生提供信息查询的服务以及根据学生资料直接拨打电话方便老师与学生的交流，当然也为学生的基本信息（学生的学号，姓名，性别，生日，籍贯，专业，成绩，电话等）进行一定程度的保密服务。

**d.用户类和特性**

该软件针对的用户群体指向各个高校大学，主要针对教师的管理学生信息需求与和学生信息交流的需求，也针对于学生查询或更改关于自身的某项重要信息的需求。

**e.运行环境需求**

本软件的运行环境，一般包括：

* 操作系统和版本：Windows 7及以上、android 4.0及以上；
* 支撑环境和版本：数据库软件Oracle 9i以上版本；
* 其它与该软件有关的软件组件：应用服务器系统采用IBM WebSphere；

**f.设计和实现上的限制**

在开发过程中可能的限制包括下列内容：

* 必须使用JDK，本产品使用AS+SDK，也可以使用Eclipse+SDK；
* 要求遵循安卓软件的开发规范和标准；

例如，如果由客户的公司或者转交给教育局负责软件维护，就必须定义转包者所使用的设计符号表示和编码标准；

* 企业策略的限制；
* 政府法规的限制；
* 工业标准的限制；
* 硬件的限制

例如，定时需求或存储器限制；

* 数据转换格式标淮的限制。

##### 2.2.4外部接口需求

**a.用户界面**

* 以下是用户界面可能包括的一些特征内容：
* 将要采用的图形用户界面(GUl)标准或者产品系列的风格；
* 有关屏幕布局或者解决方案的限制；
* 将要使用在每一个屏幕(图形用户界面)上的软件组件，可能包括：
* 选单；
* 标准按钮
* 导航链接；
* 各种功能组件；
* 快捷键；
* 各种排序方式的规定，按照成绩大小；
* 不同情况下数字的表现格式与对齐方式：学号，班级号，年龄左对齐；
* 日期的表现方法与格式：示例：2019年3月26日
* 错误信息显示标准：弹出错误提示界面，显示错误原因。

**b.硬件接口**

* 支持的硬件类型；
* 软、硬件之间交流的数据；
* 控制信息的性质；
* 使用的通讯协议：TCP/IP协议

**c.软件接口**

* 操作系统：基于Linux的自由及开放源代码的操作系统，android系统；
* 数据库：SQLite数据库管理系统；
* 工具：Android studio ；
* 函数库；
* 集成的商业组件

**d.通讯接口**

描述与软件产品所使用的通讯功能相关的需求，遵循各种网络协议，满足各种浏览器的需求，包括：

* 电子邮件；
* WEB浏览器；
* 网络通讯标准或者协议；
* 数据交互用电子表格；

##### 2.2.5系统功能需求

* 用户登录功能需求：用户通过自己账号和密码登录，对app进行操作；
* 用户修改密码功能需求：已登录用户可通过输入旧密码以及新密码，进而修改密码成功；
* 账号权限识别授权功能需求：对教师和学生这些不同身份的账号进行权限识别以及授权功能，使得教师可以实现学生信息管理和统计的各种操作，而学生只能更改自己的个人信息以及信息统计；
* 通讯功能需求：已登录的教师用户可以在编辑学生信息页面直接拨打学生电话，进而实现信息交互；
* 搜索功能需求：已登录的教师用户可直接在搜索框内按照学生学号或姓名搜索学生，进而对学生进行信息管理等。
* 学生信息管理以及后台数据库存储功能需求：已登录的用户根据权限可以实现学生信息的添加、删除、更改、查询等功能，其中可能会涉及到排序的操作，而系统则根据用户要求实现后台数据库的更改存储。
* 学生信息统计功能需求：已登录的用户可以根据权限实现信息统计功能，对班级考试成绩的查询可以获得相应的最高分、平均分、最低分以及及格人数、不及格人数等。

**a.说明和优先级**

* 本系统根据最终用户所具有的不同功能将用户分为以下三类：
* 教师用户：从产品获取、上传、管理信息等；
* 学生用户：从产品获取、管理自己的个人信息等；
* 系统管理员：维护系统的正常；
* 优先级：管理员 高于 教师用户 高于 学生用户
* 注：产品在做完基础的架构之后，应该优先做出用户的模块，然后才是管理用户的功能。

**b.激励响应序列**

角色：用户

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 角色 | 触发条件 | 前置条件 | 基本事件流描述 | 后置条件 |
| 用 户 | 登录 | 无 | 1. 输入账号、密码，单击log in按钮； 2. 系统验证用户合法性； 3. 获取用户权限和数据库信息； 4. 转向系统主页面。 | 成功：显示“登录成功！”，并返回系统主页面；  失败：显示“登录失败，密码错误。” |
| 修改 | 在系统主页 面 | 1. 点击某个学生； 2. 获取用户权限和数据库信息； 3. 进入编辑学生信息页面； | 成功：框内显示修改后的内容。 |
| 修改成功按钮 | 在信息编辑页面 | 1. 点击“√”； 2. 将信息保存入数据库； 3. 进入系统主页面。 | 成功：显示“数据修改成功”，并返回系统主页面。 |
| 查询或搜索 | 在菜单栏中或在主页面中 | 1. 点击“查询信息”或搜索按钮； 2. 弹出搜索类型询问框； | 成功：弹出询问框。 |
| 确定搜索类型 | 在搜索类型询问框中 | 1. 点击“学号”或姓名； 2. 弹出学号或姓名输入框； 3. 键入学号或姓名； 4. 获取数据库信息； 5. 进入该学生信息编辑界面。 | 成功：进入学生信息编辑界面并显示学生信息。 |
| 增添 | 在系统主页面或菜单栏中 | 1. 点击“+”或“增添信息”； 2. 进入空白学生信息编辑页面； | 成功：框内显示输入的内容。 |
| 增添成功按钮 | 在信息编辑页面 | 1. 点击“√”； 2. 将信息保存入数据库； 3. 进入系统主页面。 | 成功：并返回系统主页面。 |
| 排序 | 在系统主页 面，且无顺序或降 序 | 1. 点击排序按钮； 2. 获取用户权限和数据库信息； 3. 将升序后信息显示。 | 成功：显示“按升序排序成功！”，并进入系统主页面。 |
| 排序 | 在系统主页 面，且升 序 | 1. 点击排序按钮； 2. 获取用户权限和数据库信息； 3. 将降序后信息显示。 | 成功：显示“按升序排序成功！”，并进入系统主页面。 |
| 删除 | 在信息编辑页面 | 1. 点击“删除该学生信息”按钮； 2. 弹出“取消”“确定”询问窗口。 |  |
| 删除确定 | 在“取消”“确定”询问窗口 | 1. 点击“确定”按钮； 2. 数据库删除该学生信息； 3. 返回主页面。 | 成功：进入系统主页面。 |
| 删除取消 | 在“取消”“确定”询问窗口 | 1. 点击“取消”按钮； 2. 弹窗取消。 | 成功：还在编辑页面。 |
| 拨打 | 在信息编辑页面 | 1. 点击拨打按钮； 2. 获取用户权限和数据库电话信息； 3. 进入手机系统拨打界面； 4. 将数据库电话信息复制入拨打框中。 | 成功：进入手机系统拨打界面，并显示电话。 |
| 统计 | 在系统主页面 | 1. 点击统计按钮； 2. 获取用户权限和数据库信息； 3. 计算统计信息； 4. 进入学生成绩统计页面。 | 成功：进入学生成绩统计页面，并显示统计信息。 |
| 修改密码 | 在菜单栏中 | 1. 点击修改密码按钮； 2. 弹出修改登录密码框； 3. 键入原密码与新密码； 4. 获取用户权限； 5. 点击“确定”。 | 成功：显示“修改密码成功”，并返回系统主界面。  失败：显示“账号获取失败”或“原密码错误”，并返回系统主界面。 |
|  | 退出登录 | 在菜单栏中 | 1. 点击退出登录按钮； 2. 弹出退出与否询问框。 | 成功：弹出退出与否询问框。 |
|  | 退出确定 | 在退出与否询问窗口 | 1. 点击“确定”按钮； 2. 返回用户登录界面。 | 成功：进入用户登录界面。 |
|  | 退出取消 | 在退出与否询问窗口 | 1. 点击“取消”按钮； 2. 返回系统主界面。 | 成功：在系统主界面。 |

**c.输入输出数据**

* **用户：**

输入数据：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据名称 | 数据类型 | 数据格式 | 数据约束 | 备注 |
| 用户账号 | String | 11{数字}11 | 非空 | 不可修改 |
| 用户密码 | String | 1{[字母|数字]}10 | 非空 | 可修改 |
| 学生学号 | Integer | 11{数字}11 | 非空 | 可修改 |
| 学生姓名 | String | 2{字符}4 | 非空 | 可修改 |
| 学生性别 | String | 12{字符}14 | 非空 | 可修改 |
| 学生生日 | String | 1{[数字|字符]} | 非空 | 可修改 |
| 学生籍贯 | String | 4{字符}6 | 非空 | 可修改 |
| 学生专业 | String | 8{字符}8 | 非空 | 可修改 |
| 学生联系方式 | String | 11{数字}11 | 非空 | 可修改 |
| 学生成绩 | String | 1{数字}3 | 非空 | 可修改 |

输出数据：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据名称 | 数据类型 | 数据格式 | 数据约束 | 备注 |
| 学生学号 | Integer | 11{数字}11 | 非空 | 可修改 |
| 学生姓名 | String | 2{字符}4 | 非空 | 可修改 |
| 学生性别 | String | 12{字符}14 | 非空 | 可修改 |
| 学生生日 | String | 1{[数字|字符]} | 非空 | 可修改 |
| 学生籍贯 | String | 4{字符}6 | 非空 | 可修改 |
| 学生专业 | String | 8{字符}8 | 非空 | 可修改 |
| 学生联系方式 | String | 11{数字}11 | 非空 | 可修改 |
| 学生成绩 | String | 1{数字}3 | 非空 | 可修改 |
| 最高分 | String | 1{数字}3 | 非空 | 不可修改 |
| 最低分 | String | 1{数字}3 | 非空 | 可修改 |
| 平均分 | String | 1{数字}3 | 非空 | 可修改 |
| 及格人数 | String | 1{数字}100 | 非空 | 不可修改 |
| 不及格人数 | String | 1{数字}100 | 非空 | 不可修改 |

* **外部接口：**

无

##### 2.2.6其他非功能需求

非功能需求主要包括可靠性、安全性、可维护性、可扩展性、可测试性等。

**a.性能需求**

* 相互合作的用户数量；
* 系统支持的并发操作数量；
* 响应时间：页面刷新时间不能超过3秒，页面的直接跳转不超过4秒，满足用户需要快速得到信息的要求，页面的优化要做好，使得每个用户能感觉到能够流畅迅速的查找到自己所需要的信息；
* 故障处理：页面长时间无响应，提出警告，要求有统一的错误提示界面并有返回操作，要求用户刷新，要求有在线帮助。页面载入时间过长，要有提示，防止用户认为系统有问题；
* 与实时系统的时间关系：要求该系统具有实时性，信息实时更新性；
* 容量需求
* 存储器；
* 磁盘空间；
* 数据库中表的最大行数。

**b.安全措施需求**

* 应用日志必须在现有基础上进行精简，对于重复日志记录、无用日志记录进行删减；
* 应用日志必须按日保存，每个文件保存应用一天的完整日志；
* 历史应用日志必须压缩后按日保存，以减小历史日志文件对硬盘空间的占用；
* 历史应用日志必须保留一个月（30天），以保证问题出现后，日志可以得到较好的保存。

**c.安全性需求**

* 系统要有足够的防御能力，防御普通黑客的攻击，尽量减少各种可能出现的漏洞，防止学生信息被不良分子窃取。
* 保证用户权限的正确设置，防止出现各种越权行为，保证系统的正常良好运行。用户的信息保存采用国际流行的MD5加密，保证安全；每个用户在第一次登录后，必须更改系统预置登录密码，系统预置的登录密码不能重用。

**d.操作需求**

* 系统能被现有的工作人员快速掌握并使用。
* 第三方控件的使用：原则是不使用，但必须使用时必须通知提出，讨论通过后方可用。
* 点对点通讯：私聊、公聊、从吧台获取信息都属于点对点通讯范畴，采用TCP协议

**e.软件质量属性**

易用性优于易学性，或者可移植性优于有效性。

### 3软件设计的基本原理和采用的主要方法与技术

#### 3.1基本原理

1）抽象：软件设计中考虑模块化解决方案时，可以定出多个抽象级别。抽象的层次从概要设计到详细设计逐步降低。

2）模块化：模块是指把一个待开发的软件分解成若干小的简单的部分。模块化是指解决一个复杂问题时自顶向下逐层把软件系统划分成若干模块的过程。

3）信息隐蔽：信息隐蔽是指在一个模块内包含的信息（过程或数据），对于不需要这些信息的其他模块来说是不能访问的。

4）模块独立性：模块独立性是指每个模块只完成系统要求的独立的子功能，并且与其他模块的联系最少且接口简单。模块的独立程度是评价设计好坏的重要度量标准。衡量软件的模块独立性使用耦合性和内聚性两个定性的度量标准。内聚性是信息隐蔽和局部化概念的自然扩展。一个模块的内聚性越强则该模块独立性越强。一个模块与其他模块的耦合性越强则该模块的模块独立性越弱。

内聚性是度量一个模块功能强度的一个相对指标。内聚是从功能角度来衡量模块的联系，它描述的是模块内的功能联系。内聚有如下种类，它们之间的内聚度由弱到强排列：偶然内聚、逻辑内聚、时间内聚、过程内聚、通信内聚、顺序内聚、功能内聚。

耦合性是模块之间互相连接的紧密程度的度量。耦合性取决于各个模块之间接口的复杂度、调用方式以及哪些信息通过接口。耦合可以分为下列几种，它们之间的耦合度由高到低排列：内容耦合、公共耦合、外部耦合、控制耦合、标记耦合、数据耦合、非直接耦合。

在程序结构中，各模块的内聚性越强，则耦合性越弱。一般较优秀的软件设计，应尽量做到高内聚，低耦合，即减弱模块之间的耦合性和提高模块内的内聚性，有利于提高模块的独立性。

#### 3.2方法技术

设计过程中用以促成模块化设计的四个区域：模块、数据、体系和程序设计。

1）模块设计降低了复杂性、便于修改、且使得支持系统不同部分的并行开发实现起来更容易。模块类型提供的操作特性通过结合时间历史、激活机制、和控制模式来表现。

2）数据设计首先并且有些人也坚信，是最重要的设计行为。数据结构的影响和程序上的复杂性导致数据设计对软件质量有着深远的影响。这种质量由以下的原理来实施：

a.适用于功能和行为分析的系统分析原理同样应该适用于数据。

b.所有的数据结构，以及各自所完成的操作都应该被确定。

c.创建数据词典并用来详细说明数据和程序的设计。

d.底层的数据设计决定应该延迟至设计过程的后期。

e.数据结构的陈述（具体说明）应该只被那些直接使用包含在此结构内的数据的模块所知道。

f.有用的数据结构和操作库可以在适当的时候使用。

g.软件设计和编程语言应该支持抽象数据类型的规范和实现。

3）体系设计的主要目标是开发模块化的程序结构并表达出模块间的控制相关性。另外，体系设计融合了程序结构与数据结构，以及使得数据得以在程序中流动的界面定义。这种方法鼓励设计者关注系统的整体设计而不是系统中单独的组件。选用不同的方法会采用不同的途径来接近体系的原点，但所有这些方法都应该认识到具有软件全局观念的重要性。

4）程序设计在数据、程序结构、和陈述详细算法的说明都已使用类似英语的自然语言来呈现后，再确定程序设计。

### 4软件架构设计和模型设计

#### 4.1软件架构



#### 4.2数据流程图





#### 4.3模型设计



#### 4.4算法设计

无

### 5软件模块划分与人员任务分工

#### 5.1模块划分



#### 5.2小组分工

* 周平：显示资料列表模块，UI设计，通讯录；
* 李顺：用户权限控制模块，用户密码修改模块，添加修改资料模块；
* 崔东彬：数据库，排序模块；
* 徐艳玲：查询模块，编写文档；
* 赵婧娟：统计模块，编写文档。

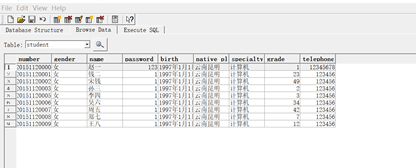
### 6软件实现

#### 6.1UI设计



#### 6.2数据库实现

用SQLiteDatabase来管理Sqlite database，SQLiteDatabase提供了基本的增删改查方法，可以执行SQL命令，执行其他常见数据库管理任务的方法。创建一个数据库表名student，其主键为学生学号，并有着性别，姓名，密码，生日，籍贯，专业，成绩，电话等列。



#### 6.3用户模块

**public void** onClick(View v) {  
 **if**(**"0000"**.equals(**etAccount**.getText().toString())){  
 *//从本地获取到密码，如果没有设置过密码，就默认为1* String oldPassword = **mSharedPreferences**.getString(**"password"**, **"1"**);  
 **if** (oldPassword.equals(**etPassword**.getText().toString())) {  
 Intent intent = **new** Intent(LogInActivity.**this**, MainActivity.**class**);  
 startActivity(intent);  
 finish();  
 Toast.*makeText*(LogInActivity.**this**, **"登录成功！"**, Toast.***LENGTH\_SHORT***).show();  
 } **else** {  
 Toast.*makeText*(LogInActivity.**this**, **"登录失败，密码错误。"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
 **else**{  
 String[] my\_account ={**etAccount**.getText().toString()};  
 Cursor mCursor = **mSQLiteDatabase**.query(**"student"**, **null**, **"number = ?"**, my\_account, **null**, **null**, **null**);  
 String mypassword=**"9999"**;  
 **while**(mCursor.moveToNext()){  
 mypassword = mCursor.getString(mCursor.getColumnIndex(**"password"**));  
 }  
 **if**(mypassword.equals(**etPassword**.getText().toString())){  
 Intent intent = **new** Intent(LogInActivity.**this**, MainActivity.**class**);  
 intent.putExtra(**"flag"**, 1);*//quanxian* startActivity(intent);  
 finish();  
 Toast.*makeText*(LogInActivity.**this**, **"登录成功！"**, Toast.***LENGTH\_SHORT***).show();  
 SharedPreferences.Editor mEditor = **mSharedPreferences**.edit();  
 mEditor.putString(**"account"**, my\_account[0]);  
 mEditor.commit();  
 }  
 **else** {  
 Toast.*makeText*(LogInActivity.**this**, **"登录失败，账号或密码错误"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
 }

#### 6.4显示资料列表模块

*//使用Adapter来显示MainActivity要显示的列表***public class** ShowAdapter **extends** RecyclerView.Adapter<ShowAdapter.ViewHolder> {  
 **private** Context **mContext**;  
 *//定义要显示的学号，名字，成绩* **private** ArrayList<String> **mNumber**;  
 **private** ArrayList<String> **mName**;  
 **private** ArrayList<String> **mGrade**;  
 **private int mflag**;  
 **public final static int *maxSize*** = 60;*//最多显示60条*

*…*

**public** ViewHolder(View itemView) {  
 **super**(itemView);  
 **cvShow** = (CardView) itemView.findViewById(R.id.***cv\_show***);  
 **tvNumber** = (TextView) itemView.findViewById(R.id.***tv\_show\_number***);  
 **tvName** = (TextView) itemView.findViewById(R.id.***tv\_show\_name***);  
 **tvGrade** = (TextView) itemView.findViewById(R.id.***tv\_show\_grade***);  
 }  
  
}

#### 6.5统计模块

**for**(**int** i = 0;i <size;i++){mCursor.moveToNext();  
 grades[i] = Integer.*parseInt*(mCursor.getString(mCursor.getColumnIndex(**"grade"**)));  
 sum += grades[i];  
 }  
 mCursor.close();  
 *//判断及格人数* **for**(**int** i = 0;i < size;i++) {  
 **if** (grades[i] >= 60 && grades[i + 1] < 60) {  
 pass = i + 1;  
 } **else if** (grades[size - 1] >= 60) {  
 pass = size;  
 } **else if** (grades[0] < 60){  
 pass = 0;  
 }  
 }  
 *//计算平均分、最高分、最低分、及格人数、不及格人数* String averageNum = String.*valueOf*(sum/size);  
 String maxNum = String.*valueOf*(grades[0]);  
 String minNum = String.*valueOf*(grades[size-1]);  
 String passNum = String.*valueOf*(pass);  
 String failNum = String.*valueOf*(size-pass);

#### 6.6添加修改资料模块

**if** (notNull(number, name, gender, nativePlace, specialty, grade, birth)) {  
 **if** (notSameNumber(number)) {  
 ContentValues values = **new** ContentValues();  
 values.put(**"number"**, number);  
 values.put(**"name"**, name);  
 values.put(**"gender"**, gender);  
 values.put(**"native\_place"**, nativePlace);  
 values.put(**"specialty"**, specialty);  
 values.put(**"grade"**, grade);  
 values.put(**"birth"**, birth);  
  
 **switch** (**currentType**) {  
 **case *TYPE\_ADD***:  
 **mSQLiteDatabase**.insert(**"student"**, **null**, values);  
 Toast.*makeText*(EditActivity.**this**, **"添加数据成功"**, Toast.***LENGTH\_SHORT***).show();  
 **break**;  
 **case *TYPE\_EDIT***:  
 **mSQLiteDatabase**.update(**"student"**, values, **"number=?"**, **new** String[]{**initNumber**});  
 Toast.*makeText*(EditActivity.**this**, **"数据修改成功"**, Toast.***LENGTH\_SHORT***).show();  
 **break**;  
 }  
  
 finish();  
 } **else** {  
 Toast.*makeText*(EditActivity.**this**, **"该学号的学生已经存在！"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 } **else** {  
 Toast.*makeText*(EditActivity.**this**, **"数据不可以为空！"**, Toast.***LENGTH\_SHORT***).show();  
 }  
}

#### 6.7搜索模块

**switch** (**currentSearchType**) {  
 **case *TYPE\_SEARCH\_NUMBER***:  
 mCursor = **mSQLiteDatabase**.query(**"student"**, **null**, **"number like ?"**, **new** String[]{**"%"** + s + **"%"**}, **null**, **null**, **null**);  
 **break**;  
 **case *TYPE\_SEARCH\_NAME***:  
 mCursor = **mSQLiteDatabase**.query(**"student"**, **null**, **"name like ?"**, **new** String[]{**"%"** + s + **"%"**}, **null**, **null**, **null**);  
 **break**;  
 **default**:  
 mCursor = **mSQLiteDatabase**.query(**"student"**, **null**, **"number like ?"**, **new** String[]{**"%"** + s + **"%"**}, **null**, **null**, **null**);  
 }  
  
 **int** size = mCursor.getCount() < ShowAdapter.***maxSize*** ? mCursor.getCount() : ShowAdapter.***maxSize***;  
  
 **while** (**true**) {  
 **if** (size-- == 0)  
 **break**;  
 mCursor.moveToNext();*//逐条读取数据* number.add(mCursor.getString(mCursor.getColumnIndex(**"number"**)));*//获取序列号* name.add(mCursor.getString(mCursor.getColumnIndex(**"name"**)));  
 }  
 mCursor.close();  
  
 **rvSearch**.setAdapter(**new** SearchAdapter(SearchActivity.**this**, number, name));}

#### 6.8排序模块

**private void** sort() {  
 …  
 **if**(!*changeOrder*){  
 mCursor = **mSQLiteDatabase**.query(**"student"**,**null**,**null**,**null**,**null**,**null**,**"grade ASC"**);  
 }**else** {  
 mCursor = **mSQLiteDatabase**.query(**"student"**, **null**, **null**, **null**, **null**, **null**, **"grade DESC"**);  
 }  
 **while** (**true**) {  
 **if** (size-- == 0)  
 **break**;  
 mCursor.moveToNext();  
 number.add(mCursor.getString(mCursor.getColumnIndex(**"number"**)));  
 name.add(mCursor.getString(mCursor.getColumnIndex(**"name"**)));  
 grade.add(mCursor.getString(mCursor.getColumnIndex(**"grade"**)));  
 }  
 mCursor.close();  
 **rvMain**.setAdapter(**new** ShowAdapter(MainActivity.**this**, number, name, grade,*flag*));  
}

### 7软件集成与测试

#### 7.1基本情况

##### 7.1.1单元测试概况

单元测试集中监测软件设计的最小单元——模块。通常，单元测试和编码属于软件过程的同一个阶段。在编写出源程序代码并通过了编译程序的语法检查之后，就可以用详细设计描述作指南，对重要的执行通路进行测试，以便发现模块内部的错误。可以应用人工测试和计算机测试这样两种不同类型的测试方法，完成单元测试工作。这两种测试方法各有所长，互相补充。通常，单元测试主要使用白盒测试技术，而且对多个模块的测试可以并行地进行。

在单元测试期间着重从下述5个方面对模块进行测试。

**a.模块接口**

首先应该对通过模块接口的数据流进行测试，如果数据不能正确地进出，所有其他测试都是不切实际的。

在对模块接口进行测试时主要检查下述几个方面：参数的数目、次序、属性或单位系统与变元是否一致；是否修改了只作输入用的变元；全局变量的定义和用法在各个模块中是否一致。

**b.局部数据结构**

对于模块来说，局部数据结构是常见的错误来源。应该仔细设计测试方案，以便发现局部数据说明、初始化、默认值等方面的错误。

**c.重要的执行通路**

由于通常不可能进行穷尽测试，因此，在单元测试期间选择最有代表性、最可能发现错误的执行通路进行测试就是十分关键的。应该设计测试方案用来发现由于错误的计算、不正确的比较或不适当的控制流而造成的错误。

**d.出错处理通路**

好的设计应该能预见出现错误的条件，并且设置适当的处理错误的通路，以便在真的出现错误时执行相应的出错处理通路或干净地结束处理。不仅应该在程序中包含出错处理通路，而且应该认真测试这种通路。当评价出错处理通路时，应该着重测试下述一些可能发生的错误。

* 对错误的描述是难以理解的;
* 记下的错误与实际遇到的错误不同;
* 在对错误进行处理之前，错误条件已经引起系统干预;
* 对错误的处理不正确。
* 描述错误的信息不足以帮助确定造成错误的位置。

需要测试的模块有：登录模块、密码修改模块、个人信息模块、学生信息修改（增添）模块、查找学生信息模块、班级学生成绩统计模块、学生成绩排名模块。

**e.边界条件**

边界测试时单元测试中最后的也可能是最重要的任务。软件常常在它的边界上失效，例如，处理n元数组的第n个元素时，或做到i次循环中的第i次重复时，往往会发生错误。使用刚好小于、刚好等于和刚好大于最大值或最小值的数据结构、控制量和数据值的测试方案，非常可能发现软件中的错误。

##### 7.1.1集成测试概况

集成测试是测试和组装软件的系统化技术，例如，子系统测试即是在把模块按照设计要求组装起来的同时进行测试，主要目标是发现与接口有关的问题（系统测试与此类似），例如，数据穿过接口时可能丢失；一个模块对另一个模块可能由于疏忽而造成有害影响，把子功能组合起来可能不产生预期的主功能；个别看来是可以接受的误差可能累积到不能接受的程度；全程数据结构可能有问题等。不幸的是，可能发生的接口问题多得不胜枚举。

在本次集成计划当中，使用了自顶向下集成的测试方式。

#### 7.2单元测试计划

##### 7.2.1登录单元

输入：登录密码

输出：登录是否成功

测试数据：1（初始密码）或更改之后的密码 预期结果：登录成功 覆盖路径：1 2 5 6 7

其他错误密码（字母数字的结合），预期结果：登录失败，密码输入错误，覆盖路径：1 2 3 8



##### 7.2.2密码修改单元

输入：A：原密码 B：新密码

输出：密码修改成功

测试数据：A=正确原密码，B=不与原密码重复的新密码 预期结果：更改密码成功，覆盖路径：1 2 3 5 7

A=错误原密码，B=符合要求新密码 预期结果：更改失败，原密码输入错误，覆盖路径：1 2 4 9

A=正确原密码，B=与原密码重复的新密码或其他不符合要求新密码 预期结果：更改密码失败，新密码不符合要求，覆盖路径：1 2 3 6 8

A=错误原密码，B=与原密码重复的新密码或其他不符合要求新密码 预期结果：更改失败，原密码输入错误，覆盖路径：1 2 4 9



##### 7.2.3查找学生信息单元

输入：A：学生学号

输出：学生相关信息

测试数据：A：20151120003， 预期结果：学号：20151120003，姓名：孙三

覆盖路径：1 2 3 5

A：008， 预期结果：（学号：20151120008，姓名：吴六）（学号：20151120089，姓名：赵十三） 覆盖路径：1 2 3 5

A：1130， 预期结果：显示没有符合学号的学生信息

覆盖路径：1 2 4 6

A：201511300000， 预期结果：显示没有符合学号的学生信息

覆盖路径：1 2 4 6



输入：B：学生姓名

输出：学生相关信息

测试数据：B：孙三，预期结果：学号：20151120003，姓名：孙三

覆盖路径：1 2 3 5

B：六， 预期结果：学号：20151120006，姓名：吴六

覆盖路径：1 2 3 5

A：十一， 预期结果：显示没有符合学号的学生信息

覆盖路径：1 2 4 6

A：赵十一， 预期结果：显示没有符合学号的学生信息

覆盖路径：1 2 4 6



##### 7.2.4排序单元测试

动作：点击排序按钮

输出：排序情况

预期结果：学生信息按照成绩降序排序，覆盖路径：1 2 3 5

学生信息没有按照成绩降序排序，覆盖路径：1 2 4 6



##### 7.2.5删除信息单元测试计划：

动作：点击删除按钮

输出：删除情况

预期结果：删除成功，覆盖路径：1 2 3 5

删除失败，覆盖路径：1 2 4 6



##### 7.2.6通讯单元测试计划：

动作：点击“电话”按钮

输出：跳转情况

预期结果：跳转成功，可以开始拨打电话，覆盖路径：1 2 3 5

跳转失败，无法进行通话，覆盖路径：1 2 4 6



##### 7.2.7模块票务预定单元测试计划：

动作：点击统计按钮

输出：班级学生成绩统计情况

预期结果：班级学生成绩包括平均分、最高分、最低分、及格人数、不及格人数都显示无误，覆盖路径：1 2 3 5

班级学生成绩包括平均分、最高分、最低分、及格人数、不及格人数其中某一项或某几项出现错误，覆盖路径：1 2 4 6



#### 7.3集成测试

##### 7.3.1增添学生信息测试计划

输入：A学号，B姓名，C性别，D生日，E籍贯，F专业，G成绩，H电话

输出：X1添加成功，X2添加失败

**等价类划分**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 输入条件 | | 有效等价类 | | 无效等价类 | |
| 要求学号为11位数字、籍贯为中国34个省市、专业为给出的四个专业，电话为11位数字 | | A=8{数字}8 （1）  B=1{字符}n （2）  C=女或男 （3）  D=xxxx年xx月xx日（4）  E=河北省等34个省市（5）  F=软件工程等所有专业（6）  G=0-100 （7）  H=11{数字}11 （8） | | A≠11{数字}11 （9）  B=空 （10）  E≠河北省等34个省市（11）  F≠软件工程等所有专（12）  G<0 (13)  G>100 (14)  H=0{数字}10 （15）  H=12{数字}n （16） | |
| 序号 | “A ，B ，C，D，E，F，G，H” | | 覆盖等价类 | | 输出 |
| 1 | “16051098，宋钱，男，  2000年5月11日，河北省，软件工程，89,18768121732” | | （1）（2）（3）（4）（5）（6）（7）（8） | | X1 |
| 2 | “1605109823455657，宋钱，男，  2000年5月11日，河北省，软件工程，89,18768121732” | | （9）（2）（3）（4）（5）（6）（7）（8） | | X2 |
| 3 | “16051098，NULL，男，  2000年5月11日，河北省，软件工程，89,18768121732” | | （1）（10）（3）（4）（5）（6）（7）（8） | |
| 4 | “16051098，宋钱，男，  2000年5月11日，火星，软件工程，89,18768121732” | | （1）（2）（3）（4）（11）（6）（7）（8） | |
| 5 | “16051098，宋钱，男，  2000年5月11日，河北省，资源管理，89,18768121732” | | （1）（2）（3）（4）（5）（12）（7）（8） | |
| 6 | “16051098，宋钱，男，  2000年5月11日，河北省，软件工程，-1,18768121732” | | （1）（2）（3）（4）（5）（6）（13）（8） | |
| 7 | “16051098，宋钱，男，  2000年5月11日，河北省，软件工程，101,18768121732” | | （1）（2）（3）（4）（5）（6）（14）（8） | |
| 8 | “16051098，宋钱，男，  2000年5月11日，河北省，软件工程，89,1876812” | | （1）（2）（3）（4）（5）（6）（7）（15） | |
| 9 | “16051098，宋钱，男，  2000年5月11日，河北省，软件工程，89,187681217326789” | | （1）（2）（3）（4）（5）（6）（7）（16） | |

**边界值分析**

根据等价类划分的结果，测试数据设计如下：

1）学号=123456789

2）学号=1234567

3）姓名=NULL

4）籍贯=火星

5）专业=环境资源

6）成绩=-1

7）成绩=101

8）电话=00000000000

9) 电话=100000000000

##### 7.3.2修改学生信息测试计划

输入：A学号，B姓名，C性别，D生日，E籍贯，F专业，G成绩，H电话

输出：X1修改成功，X2修改失败

**等价类划分**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 输入条件 | | 有效等价类 | | 无效等价类 | |
| 要求学号为11位数字、籍贯为中国34个省市、专业为给出的四个专业，电话为11位数字 | | A=8{数字}8 （1）  B=1{字符}n （2）  C=女或男 （3）  D=xxxx年xx月xx日（4）  E=河北省等34个省市（5）  F=软件工程等所有专业（6）  G=0-100 （7）  H=11{数字}11 （8） | | A≠11{数字}11 （9）  B=空 （10）  E≠河北省等34个省市（11）  F≠软件工程等所有专（12）  G<0 (13)  G>100 (14)  H=0{数字}10 （15）  H=12{数字}n （16） | |
| 序号 | | “A ，B ，C，D，E，F，G，H” | | 覆盖等价类 | |
| 1 | “16051098，宋钱，男，  2000年5月11日，河北省，软件工程，89,18768121732” | | （1）（2）（3）（4）（5）（6）（7）（8） | | X1 |
| 2 | “1605109823455657，宋钱，男，  2000年5月11日，河北省，软件工程，89,18768121732” | | （9）（2）（3）（4）（5）（6）（7）（8） | | X2 |
| 3 | “16051098，NULL，男，  2000年5月11日，河北省，软件工程，89,18768121732” | | （1）（10）（3）（4）（5）（6）（7）（8） | |
| 4 | “16051098，宋钱，男，  2000年5月11日，火星，软件工程，89,18768121732” | | （1）（2）（3）（4）（11）（6）（7）（8） | |
| 5 | “16051098，宋钱，男，  2000年5月11日，河北省，资源管理，89,18768121732” | | （1）（2）（3）（4）（5）（12）（7）（8） | |
| 6 | “16051098，宋钱，男，  2000年5月11日，河北省，软件工程，-1,18768121732” | | （1）（2）（3）（4）（5）（6）（13）（8） | |
| 7 | “16051098，宋钱，男，  2000年5月11日，河北省，软件工程，101,18768121732” | | （1）（2）（3）（4）（5）（6）（14）（8） | |
| 8 | “16051098，宋钱，男，  2000年5月11日，河北省，软件工程，89,1876812” | | （1）（2）（3）（4）（5）（6）（7）（15） | |

**边界值分析**

根据等价类划分的结果，测试数据设计如下：

1）学号=123456789

2）学号=1234567

3）姓名=NULL

4）籍贯=火星

5）专业=环境资源

6）成绩=-1

7）成绩=101

8）电话=00000000000

9) 电话=100000000000

### 8个人工作总结

成员1：崔东彬

（1）本人实现的过程与步骤

数据库模块的完成：软件制作初期优先搭建android studio的学生信息数据库，添加表名列名主键声明等，因为这个软件系统很多功能都要用到数据库，所以数据库模块要首先完成。

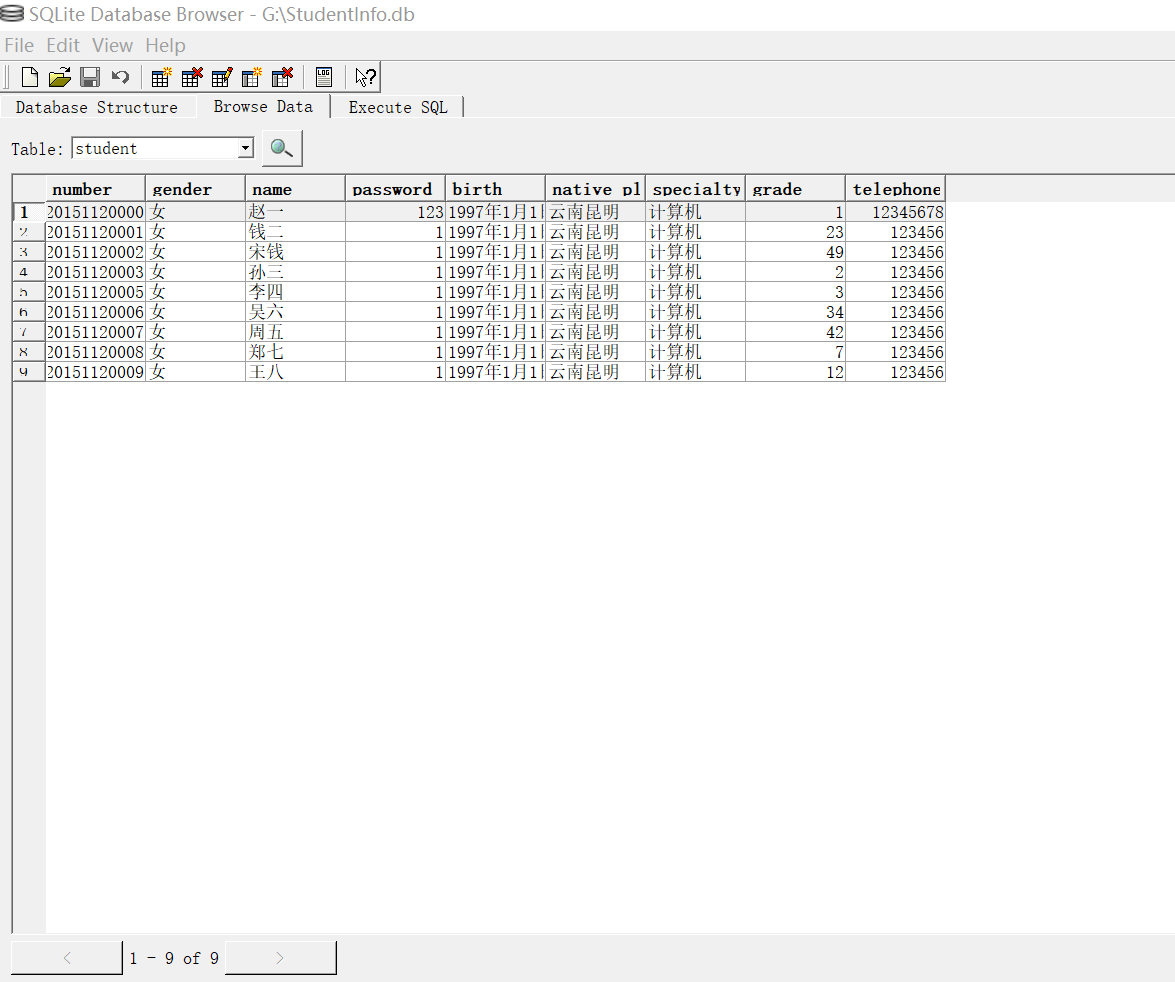
成绩排序模块的完成：在主界面展示条目列表完成后对其进行排序功能完成。

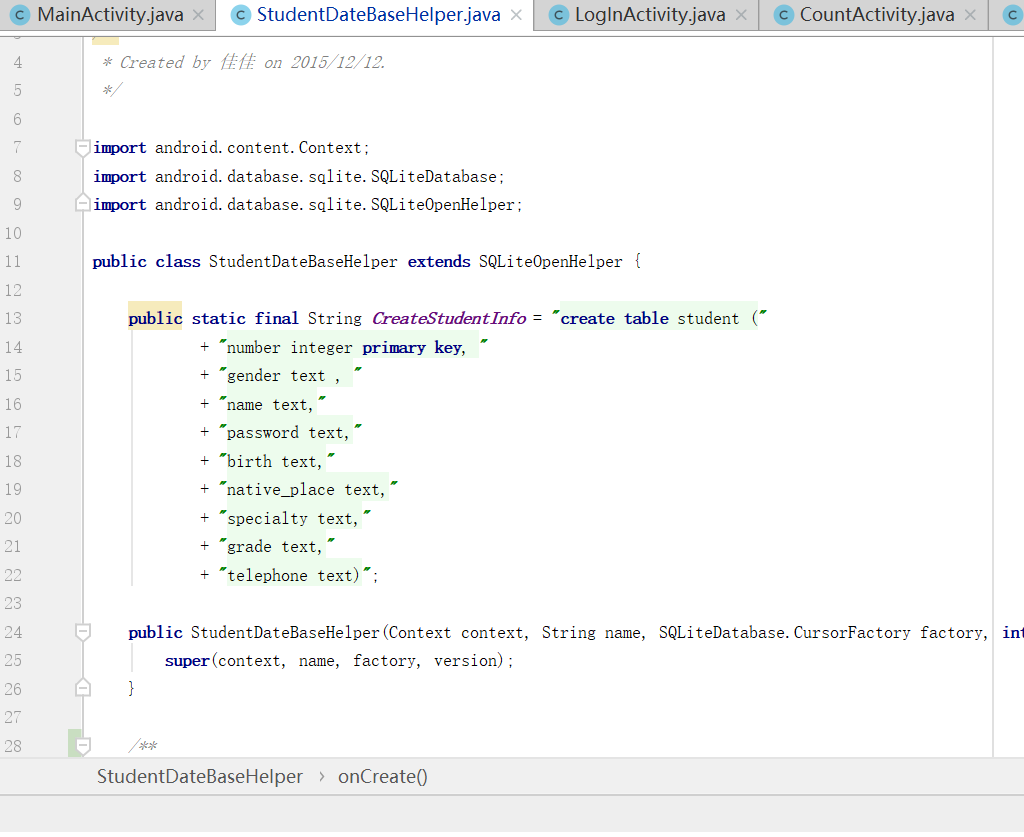
与其它模块的对接：协助其他人完成设计数据库的操作，诸如遍历数据库信息来设置主界面的条目列表，统计搜索等功能。

（2）个人遇到的困难与获得的主要成果

因为数据库在别的功能上应用广，所以经常要参与进其他人负责的功能模块中，并提供数据库操作方面的援助，比如使用游标cursor进行query查询操作来完成统计，排序之类的，这要求对其他人的代码有一定了解与一些团队协作能力。

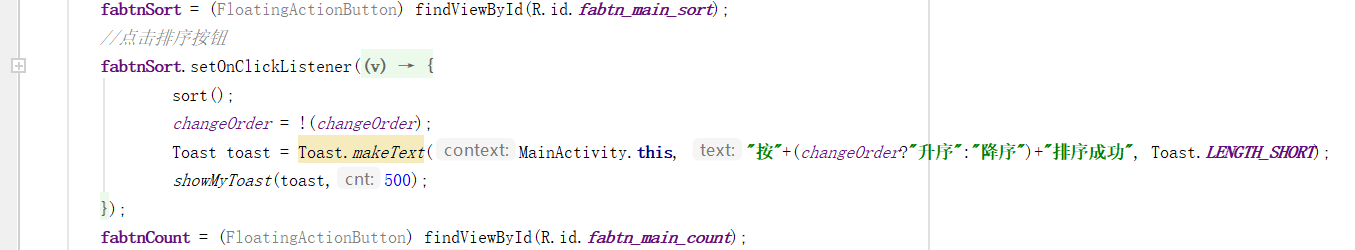
1. 个人测试与运行记录





数据库的搭建展示，采用device file explorer讲数据库文件另存并展示。

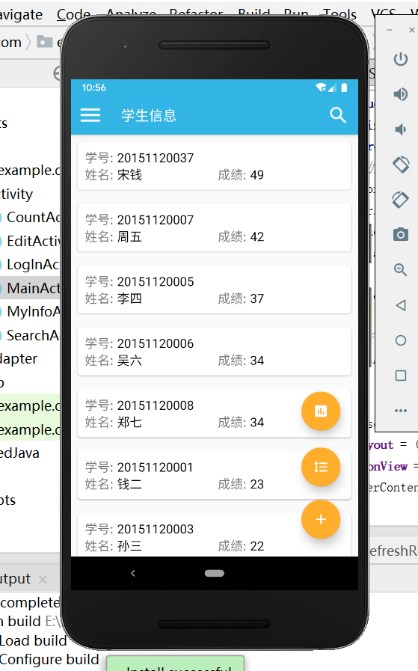
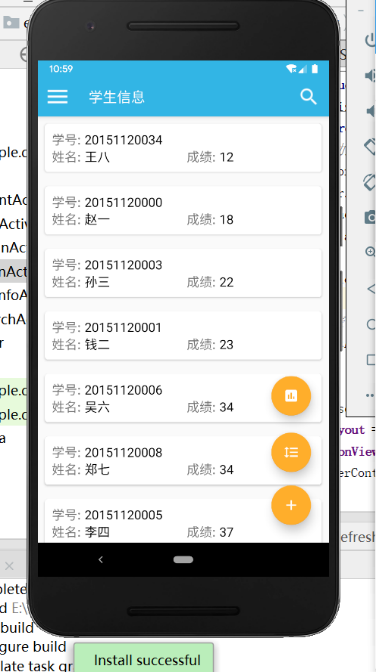
主界面信息条目陈列后编写排序sort函数和排序按钮事件





完成排序模块后协助其他人提供数据库操作援助

（4）结果分析与个人小结

完成排序模块，点击一次右边橘色中间按钮换一次升降序。

个人小结：在小组完成该项目的工程中，组内进行了明确的分工，包括项目初期的分析、文档撰写及项目后期的开发测试过程中的模块划分。在小组中，我负责的部分为：项目初期的数据库分析、数据库设计功能协助。本次的实践让我学到了一些我之前不了解的东西。明确的分工要求了彼此的模块在要有独立性的同时又要能够彼此联系，加上我负责的部分为数据库模块这个要广泛应用到的。在能够让数据库在activity顺利运行遇到过不小的困难，通过与同学的交流，加上项目不需要太过复杂的操作得以完成任务。在这次的软件工程实践里，也跟着整组人完整地经历了一遍软件开发的流程，学到了其他一些书本上没有的东西。

成员2：李顺

（1）本人实现的过程与步骤

用户权限控制模块:通过对数据库的运用，将学生信息存储在sqlite数据库中，管理员账户存在本地，通过通过一个flag标识表明当前账号是学生还是管理员，达到权限控制的目的。

用户密码修改模块:通过对数据库的调用，验证原密码与输入密码，完成用户密码修改。

与其它模块的对接：本模块属于初始化模块，用过intent方法跳转到其他页面。

添加修改资料模块:在mainactivity中确定添加或者修改，跳转至修改页面后完成对数据库的操作

（2）个人遇到的困难与获得的主要成果

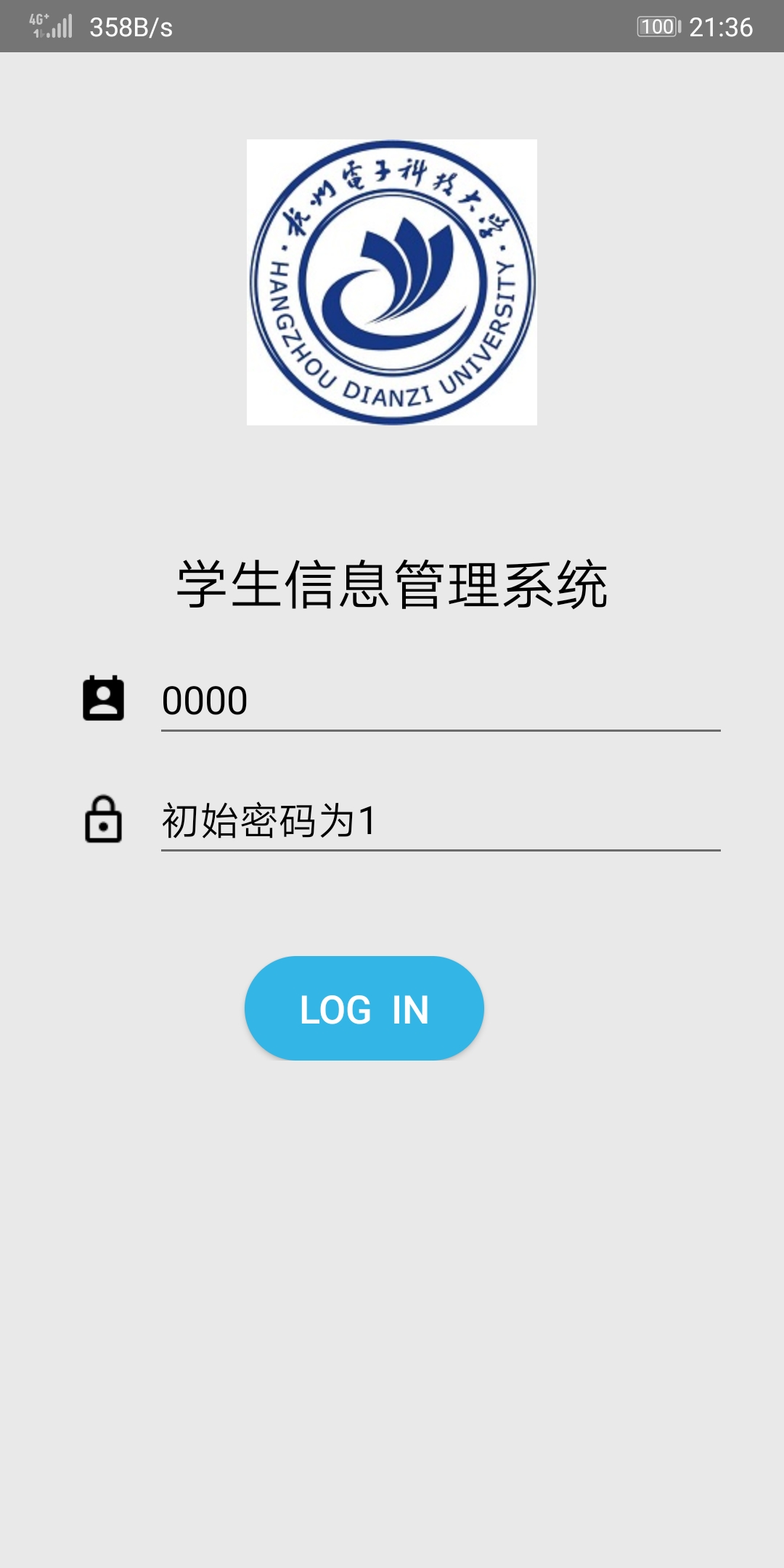
主要还是基本没有android基础，在同学的协助，查阅了一些资料后，对基本功能的添加有了了解后就比较得心应手了。

（3）个人测试与运行记录

修改资料界面：点击后能正常修改



登陆界面：能根据账号密码自动识别账户权限。



修改密码界面：能正确识别是否与原密码相同，并完成修改



（4）结果分析与个人小结

我是我们小组的组长，项目初期我们一丝不苟地完成了任务划分和需求分析，这对我们后来的开发有很大帮助，后来在开发过程中，与同学的交流也让我收益颇多，尽管一开始并不android基础，但是在学习过程中，越来越得心应手了，自己也掌握了一门技能。

成员3：周平

（1）本人实现的过程与步骤

* 1. 根据需求分析文档来设计出满足客户需求以及交互体验的UI
  2. 设计编写UI所需的style,drawable资源
  3. 完成UI界面的编辑并输出每个UI组件的id表格
  4. 测试UI组件的显示效果
  5. 创建ShowAdapter.java来格式化显示每个学生的学号、姓名、成绩
  6. 测试显示效果
  7. 创建MainActivity.java来实例化数据库，通过ShowAdapter来显示出列表
  8. 测试显示效果
  9. 整体测试效果

（2）个人遇到的困难与获得的主要成果

* 1. 在设计UI的风格时很难做到简洁美观
  2. 所用到的drawable资源需要满足不同分辨率屏幕的要求
  3. 编写ShowAdapter.java的代码需要用到很多陌生的知识
  4. 主要成果是完成了满足整体需求的UI的设计编写
  5. 整体的风格和资料列表的显示简洁大方

（3）个人测试与运行记录

7/4 完成了UI的显示效果测试

7/5 完成了正确显示列表并能正确响应列表项点击的测试

7/6 完成了项目整合后的黑盒测试。完成了项目打包签名后，安装到真机运行的测试

（4）结果分析与个人小结

软件安装后运行稳定，经测试没有发先导致崩溃的输入。测试结果显示：对于用户的合理输入，软件有正常合理的输出。对于用户较大范围的不合理输入，软件会给出正确积极并且不影响客户使用体验的响应。

个人小结;

我个人主要完成了UI的设计以及显示列表的模块，涉及到界面的设计和代码的逻辑，在两个方面都学到很多东西。

安卓项目的开发门槛和难度并不大，但是很花时间，在编程过程中要有耐心。汲取大佬们开发的经验技巧是非常重要的，学会利用网上的资源来设计界面，，学习其他项目的代码结构都能提高开发效率。团队协作对于开发项目至关重要，整合测试要注意方法。

成员4：徐艳玲

（1）本人实现的过程与步骤

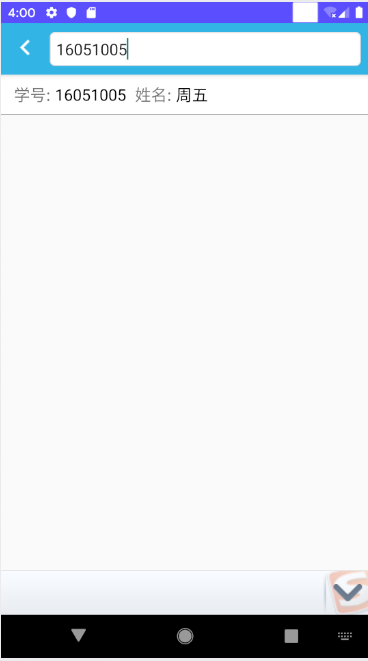
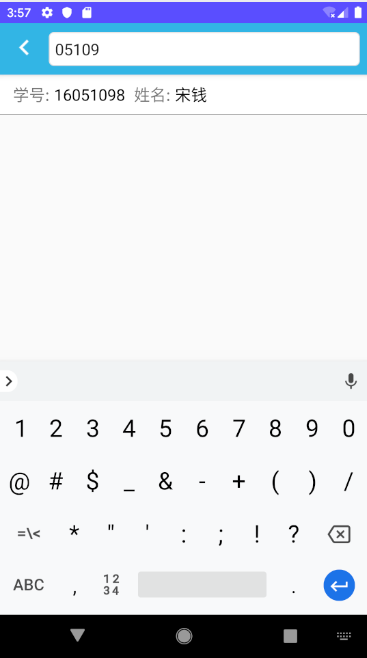
查询模块:先获知查询的字段是学生学号，还是学生姓名，再通过addTextChangedListener实时监听用户输入情况。输入完成后，用mSQLiteDatabase.query识别需要查询的字段并进行匹配，然后将匹配到的学生逐条读取，获取他们的序列号，放入setAdapter适配器中，最后在显示在SearchActivity界面上。当匹配到的学生信息数目比适配器容量大时，则选取适配器最大容量数目的信息显示。

（2）个人遇到的困难与获得的主要成果

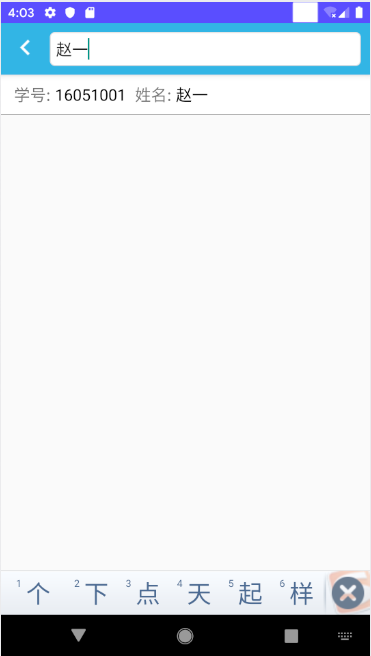
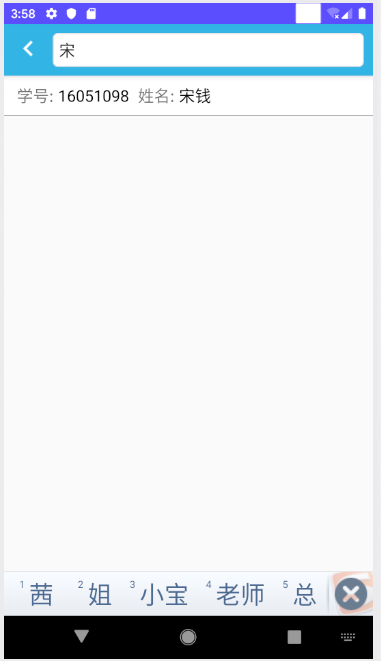
不管是对Java还是Android，我都是属于初学者，多亏组员的帮助，看了一些课件和书籍之后，我了解了Android的结构和一些Java的基础知识。在磕磕绊绊的边学习、边查阅知识点的过程中，我完成了查询模块的代码部分。学到了一些以前没有接触到的知识，比如RecyclerView界面，addTextChangedListener用户监听组件，Cursor类，自动刷新等等

（3）个人测试与运行记录

查询学号，可以输入完整学号，也可以输入学号的连续部分



查询姓名，可以输入完整姓名，也可以输入姓名一部分



（4）结果分析与个人小结

我一开始定的主要任务是为小组编写文档，因此需求分析也是我来编辑，但是越写越发现我们的app描述的有点模糊，于是我不知不觉成了小组的产品经理，对这个app提了很多建议（可能会被吐槽话多），但是很多都被驳回了，毕竟开发时间相对来说不是很足，于是本来想做的社区功能和联系人功能都被卡掉了。

但是总体来说，这次短学期还是很有意义的，实践是促使进步的最快方法，在这短短的时间，我了解了Android的结构和一些Java的基础知识，能够理解一些大概的轮廓，剩下的更细节的知识则需要专门的去学习和铭记。我也发现我的一个学习上的缺点，就是对大部分东西都泛泛而学，并没有精益求精，在遇到问题的第一反应还是会有些迷茫。

成员5：赵婧娟

（1）本人实现的过程与步骤

创建CountActivity.java来完成统计模块；

通过activit\_count.xml显示统计结果；

单元测试，

小组实验报告的编写。

（2）个人遇到的困难与获得的主要成果

在安装Android Studio时，遇到cpu不支持HAX的困难；数据库相关的知识有些遗忘；需要学习有关安卓开发操作SQLite相关内容；文档十分难写。在将成绩降序时，由于string类型，100会排到最后面，9分会排在80分前面；排序时添加abs（）函数将其转为数值再排序。

解决了安装问题，使用Genymotion+VirtualBox作为安卓虚拟器代替Android Studio本身安卓模拟器；完成了统计模块的编码；完成了文档的编写。

（3）个人测试与运行记录





（4）结果分析与个人小结

对于学生成绩资料，能够实现最高分、最低分、平均分的统计，以及及格人数和不及格人数的统计。

复习了数据库相关知识；对于安卓开发以及操作SQLite有了初步的了解；对以后写文档的需求分析、测试、数据流图设计等有了更加清晰的把握。

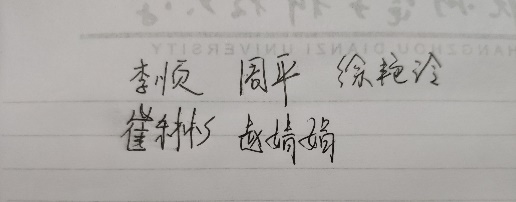
### 9附录

#### 9.1会议记录（签名）

课题研究的活动及会议记录（一）

|  |  |  |  |
| --- | --- | --- | --- |
| 时间 | 2019年7月1日 | 地点 | 讨论小组 |
| 主持 | 周平 | 记录 | 赵婧娟 |
| 参与人员 | 崔东彬、李顺、徐艳玲、周平、赵婧娟 | | |
| 会议主题 | 分组和课题选择 | | |
| 会议内容 | 1. 推选李顺作为组长； 2. 根据题目难易程度以及结合组员情况，确定题目一； 3. 确定开发环境：Android Studio。 | | |

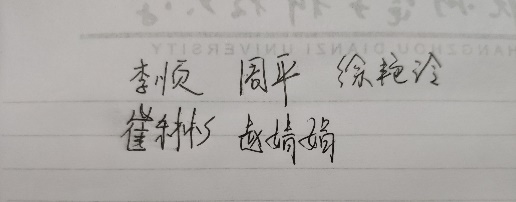
签字：



课题研究的活动及会议记录（二）

|  |  |  |  |
| --- | --- | --- | --- |
| 时间 | 2019年7月2日 | 地点 | 1教115 |
| 主持 | 李顺 | 记录 | 赵婧娟 |
| 参与人员 | 崔东彬、李顺、徐艳玲、周平、赵婧娟 | | |
| 会议主题 | 需求分析和模块划分 | | |
| 会议内容 | 一、查找关于课题的资料等；  二、确定需求分析：信息的增、删、改、查、统计、权限设置、直接通讯等；  三、确定模块：登陆、个人信息、通讯录、成绩、查找、数据库；  四、讨论数据存储形式：数据库；  五、确定集成测试方案：黑盒测试+白盒测试；  六、确定小组分工：崔东彬、李顺、周平主要负责代码实现；徐艳玲、赵婧娟负责文档编写，后期实际操作小组可能互有补充，分工并不固定；  七、确定该日任务：整理需求分析，可行性报告，完成工作环境的搭建。 | | |

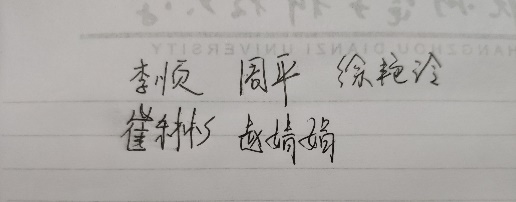
签字：



课题研究的活动及会议记录（三）

|  |  |  |  |
| --- | --- | --- | --- |
| 时间 | 2019年7 月3日 | 地点 | 1教115 |
| 主持 | 李顺 | 记录 | 赵婧娟 |
| 参与人员 | 崔东彬、李顺、徐艳玲、周平、赵婧娟 | | |
| 会议主题 | 系统开发 | | |
| 会议内容 | 1. 学习讨论安卓开发技术以及安卓SQLite数据库基本用法； 2. 完善需求分析； 3. 确定系统逻辑模型图； 4. 进行模块划分，以及逻辑模型、架构、接口、界面、数据库、模块设计； 5. 规划完成初步的系统开发计划。 | | |

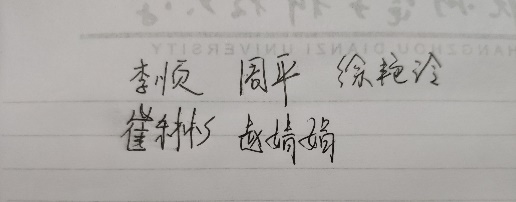
签字：



课题研究的活动及会议记录（四）

|  |  |  |  |
| --- | --- | --- | --- |
| 时间 | 2019年7 月4日 | 地点 | 1教115 |
| 主持 | 李顺 | 记录 | 赵婧娟 |
| 参与人员 | 崔东彬、李顺、徐艳玲、周平、赵婧娟 | | |
| 会议主题 | 系统开发 | | |
| 会议内容 | 一、继续学习讨论安卓开发技术以及安卓SQLite数据库基本用法；  二、确定编码任务：数据库，UI代码，增删改查、统计模块，登录，显示资料列表模块、排序模块；  三、确定编写文档任务：软件设计概述，软件设计的基本原理、方法技术，软件架构设计和模型设计，软件模块划分与人员分工；  四、制定测试计划和方案制定。 | | |

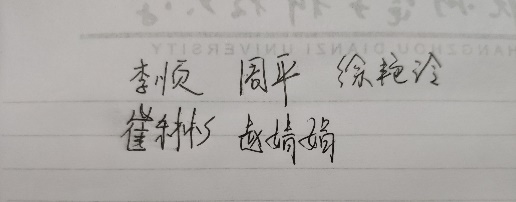
签字：



课题研究的活动及会议记录（五）

|  |  |  |  |
| --- | --- | --- | --- |
| 时间 | 2019年7 月5日 | 地点 | 1教115 |
| 主持 | 周平 | 记录 | 赵婧娟 |
| 参与人员 | 崔东彬、李顺、徐艳玲、周平、赵婧娟 | | |
| 会议主题 | 系统开发 | | |
| 会议内容 | 一、李：实现不同账户登录功能，解决权限及修改密码的问题；  二、崔：添加补充数据库表项，实现电话号码存储和修改功能；  三、周：添加软件信息和退出登录功能，添加涉及图标并优化UI显示效果；  四、赵，徐：继续文档编写任务：软件设计的基本原理、方法技术，软件算法设计，软件模块实现、测试计划等； | | |

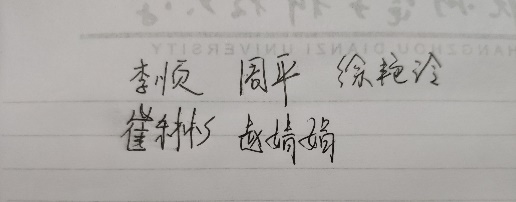
签字：



课题研究的活动及会议记录（六）

|  |  |  |  |
| --- | --- | --- | --- |
| 时间 | 2019年7 月6日 | 地点 | 1教115 |
| 主持 | 周平 | 记录 | 赵婧娟 |
| 参与人员 | 崔东彬、李顺、徐艳玲、周平、赵婧娟 | | |
| 会议主题 | 系统开发 | | |
| 会议内容 | 一、优化UI显示效果；  二、实现拨打电话功能；  三、系统测试并修改bug完善代码；  四、撰写个人实验报告文档；  五、继续编写小组报告，将个人文档整理、会议记录、代码等整理进小组实验报告。 | | |

签字：



#### 9.2个人完成的程序模块核心代码（含注释）

1、李顺

**private void** initView() {  
  
 **mSharedPreferences** = **this**.getSharedPreferences(**"student"**, ***MODE\_PRIVATE***);  
 **etAccount** = (EditText) findViewById(R.id.***et\_login\_account***);  
 **etPassword** = (EditText) findViewById(R.id.***et\_login\_password***);  
 **btnLogIn** = (Button) findViewById(R.id.***btn\_login\_log\_in***);  
 **btnLogIn**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **if**(**"0000"**.equals(**etAccount**.getText().toString())){  
 *//从本地获取到密码，如果没有设置过密码，就默认为1* String oldPassword = **mSharedPreferences**.getString(**"password"**, **"1"**);  
  
 **if** (oldPassword.equals(**etPassword**.getText().toString())) {  
 Intent intent = **new** Intent(LogInActivity.**this**, MainActivity.**class**);  
 startActivity(intent);  
 finish();  
 Toast.*makeText*(LogInActivity.**this**, **"登录成功！"**, Toast.***LENGTH\_SHORT***).show();  
 } **else** {  
 Toast.*makeText*(LogInActivity.**this**, **"登录失败，密码错误。"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
 **else**{  
 String[] my\_account ={**etAccount**.getText().toString()};  
 Cursor mCursor = **mSQLiteDatabase**.query(**"student"**, **null**, **"number = ?"**, my\_account, **null**, **null**, **null**);  
 String mypassword=**"9999"**;  
 **while**(mCursor.moveToNext()){  
 mypassword = mCursor.getString(mCursor.getColumnIndex(**"password"**));  
 }  
 **if**(mypassword.equals(**etPassword**.getText().toString())){  
 Intent intent = **new** Intent(LogInActivity.**this**, MainActivity.**class**);  
 intent.putExtra(**"flag"**, 1);*//quanxian* startActivity(intent);  
 finish();  
 Toast.*makeText*(LogInActivity.**this**, **"登录成功！"**, Toast.***LENGTH\_SHORT***).show();  
 SharedPreferences.Editor mEditor = **mSharedPreferences**.edit();  
 mEditor.putString(**"account"**, my\_account[0]);  
 mEditor.commit();  
 }  
 **else** {  
 Toast.*makeText*(LogInActivity.**this**, **"登录失败，账号或密码错误"**, Toast.***LENGTH\_SHORT***).show();  
 }  
  
 }  
  
 }  
 });

**private void** changePasswordDialog() {  
 **final** TableLayout tlPassword = (TableLayout) getLayoutInflater().inflate(R.layout.***dialog\_main\_password***, **null**);  
 **final** EditText etOldPassword = (EditText) tlPassword.findViewById(R.id.***et\_main\_old\_password***);  
 **final** EditText etNewPassword = (EditText) tlPassword.findViewById(R.id.***et\_main\_new\_password***);  
 **new** AlertDialog.Builder(MainActivity.**this**)  
 .setTitle(**"修改登录密码"**)  
 .setView(tlPassword)  
 .setPositiveButton(**"确定"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 String myaccount = **mSharedPreferences**.getString(**"account"**, **"9998"**);  
 **if**(myaccount.equals(**"9998"**)){  
 Toast.*makeText*(MainActivity.**this**, **"账号获取失败"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 **else**{  
 String[] my\_account ={myaccount};  
 Cursor mCursor = **mSQLiteDatabase**.query(**"student"**, **null**, **"number = ?"**, my\_account, **null**, **null**, **null**);  
 String mypassword=**"9999"**;  
 **while**(mCursor.moveToNext()){  
 mypassword = mCursor.getString(mCursor.getColumnIndex(**"password"**));  
 }  
 **if**(mypassword.equals(etOldPassword.getText().toString())){  
 String newpassword = etNewPassword.getText().toString();  
 ContentValues values = **new** ContentValues();  
 values.put(**"password"**,newpassword);  
 **mSQLiteDatabase**.update(**"student"**,values,**"number = ?"**,my\_account);  
 Toast.*makeText*(MainActivity.**this**, **"修改密码成功！"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 **else**{  
 Toast.*makeText*(MainActivity.**this**, **"原密码错误"**, Toast.***LENGTH\_SHORT***).show();  
 }

**private void** btnSureAction() {  
 String number = **etNumber**.getText().toString();  
 String name = **etName**.getText().toString();  
 String gender = **tvGender**.getText().toString();  
 String nativePlace = **etNativePlace**.getText().toString();  
 String specialty = **etSpecialty**.getText().toString();  
 String grade = **etgrade**.getText().toString();  
 String birth = **tvBirth**.getText().toString();  
  
 **if** (notNull(number, name, gender, nativePlace, specialty, grade, birth)) {  
 **if** (notSameNumber(number)) {  
 ContentValues values = **new** ContentValues();  
 values.put(**"number"**, number);  
 values.put(**"name"**, name);  
 values.put(**"gender"**, gender);  
 values.put(**"native\_place"**, nativePlace);  
 values.put(**"specialty"**, specialty);  
 values.put(**"grade"**, grade);  
 values.put(**"birth"**, birth);  
  
 **switch** (**currentType**) {  
 **case *TYPE\_ADD***:  
 **mSQLiteDatabase**.insert(**"student"**, **null**, values);  
 Toast.*makeText*(EditActivity.**this**, **"添加数据成功"**, Toast.***LENGTH\_SHORT***).show();  
 **break**;  
 **case *TYPE\_EDIT***:  
 **mSQLiteDatabase**.update(**"student"**, values, **"number=?"**, **new** String[]{**initNumber**});  
 Toast.*makeText*(EditActivity.**this**, **"数据修改成功"**, Toast.***LENGTH\_SHORT***).show();  
 **break**;  
 }  
  
 finish();  
 } **else** {  
 Toast.*makeText*(EditActivity.**this**, **"该学号的学生已经存在！"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 } **else** {  
 Toast.*makeText*(EditActivity.**this**, **"数据不可以为空！"**, Toast.***LENGTH\_SHORT***).show();  
 }  
}

2、周平

ShowAdapter.java

*//使用Adapter来显示MainActivity要显示的列表***public class** ShowAdapter **extends** RecyclerView.Adapter<ShowAdapter.ViewHolder> {  
 **private** Context **mContext**;  
 *//定义要显示的学号，名字，成绩* **private** ArrayList<String> **mNumber**;  
 **private** ArrayList<String> **mName**;  
 **private** ArrayList<String> **mGrade**;  
 **private int mflag**;  
 **public final static int *maxSize*** = 60;*//最多显示60条*

*…*

**public** ViewHolder(View itemView) {  
 **super**(itemView);  
 **cvShow** = (CardView) itemView.findViewById(R.id.***cv\_show***);  
 **tvNumber** = (TextView) itemView.findViewById(R.id.***tv\_show\_number***);  
 **tvName** = (TextView) itemView.findViewById(R.id.***tv\_show\_name***);  
 **tvGrade** = (TextView) itemView.findViewById(R.id.***tv\_show\_grade***);  
 }  
  
}

…

MainActivity.java

//初始化MainActivity界面

**private void** initView() {  
 **if** (Build.VERSION.***SDK\_INT*** >= Build.VERSION\_CODES.***LOLLIPOP***) {  
 *//将状态栏颜色设置为与toolbar一致* getWindow().setStatusBarColor(getResources().getColor(R.color.***holo\_blue\_light***));  
 }  
 setToolBar();  
 setNavigationView();  
 **rvMain** = (RecyclerView) findViewById(R.id.***rv\_main***);  
 **rvMain**.setLayoutManager(**new** LinearLayoutManager(MainActivity.**this**));  
//添加add新资料按钮  
 **fabtnAdd** = (FloatingActionButton) findViewById(R.id.***fabtn\_main\_add***);  
 **fabtnAdd**.setOnClickListener(**new** View.OnClickListener() {  
 @Override

//设置add监听器，flag决定账号权限  
 **public void** onClick(View v) {  
 **if**(*flag* == 0){  
 Intent intent = **new** Intent(MainActivity.**this**, EditActivity.**class**);  
 intent.putExtra(**"type"**, EditActivity.***TYPE\_ADD***);  
 startActivity(intent);  
 }  
 **else**{  
 Toast.*makeText*(MainActivity.**this**, **"权限不足"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
 });  
//添加排序按钮  
 **fabtnSort** = (FloatingActionButton) findViewById(R.id.***fabtn\_main\_sort***);  
 *//排序按钮监听器，用changeOrder来改变升序降序排序方式* **fabtnSort**.setOnClickListener(**new** View.OnClickListener(){  
 @Override  
 **public void** onClick(View v){  
 sort();  
 *changeOrder* = !(*changeOrder*);  
 Toast toast = Toast.*makeText*(MainActivity.**this**, **"按"**+(*changeOrder*?**"升序"**:**"降序"**)+**"排序成功"**, Toast.***LENGTH\_SHORT***);  
 *showMyToast*(toast,500);  
 }  
 });

//添加统计成绩按钮  
 **fabtnCount** = (FloatingActionButton) findViewById(R.id.***fabtn\_main\_count***);  
 **fabtnCount**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View view) {  
 Intent intent = **new** Intent(MainActivity.**this**,CountActivity.**class**);  
 startActivity(intent);  
 }  
 });  
  
}

*//排序函数部分节选***private void** sort() {  
 …  
 **if**(!*changeOrder*){  
 mCursor = **mSQLiteDatabase**.query(**"student"**,**null**,**null**,**null**,**null**,**null**,**"grade ASC"**);  
 }**else** {  
 mCursor = **mSQLiteDatabase**.query(**"student"**, **null**, **null**, **null**, **null**, **null**, **"grade DESC"**);  
 }  
 **int** size = mCursor.getCount() < ShowAdapter.***maxSize*** ? mCursor.getCount() : ShowAdapter.***maxSize***;  
  
  
 **while** (**true**) {  
 **if** (size-- == 0)  
 **break**;  
 mCursor.moveToNext();  
 number.add(mCursor.getString(mCursor.getColumnIndex(**"number"**)));  
 name.add(mCursor.getString(mCursor.getColumnIndex(**"name"**)));  
 grade.add(mCursor.getString(mCursor.getColumnIndex(**"grade"**)));  
 }  
 mCursor.close();  
 **rvMain**.setAdapter(**new** ShowAdapter(MainActivity.**this**, number, name, grade,*flag*));  
  
}

…

*//显示菜单视图***private void** setupDrawerContent(NavigationView navigationView) {  
 navigationView.setNavigationItemSelectedListener(  
 **new** NavigationView.OnNavigationItemSelectedListener() {  
 @Override  
 **public boolean** onNavigationItemSelected(MenuItem menuItem) {  
 Intent intent,intent2;  
 **switch** (menuItem.getItemId()) {  
 **case** R.id.***nav\_my\_info***://软件信息  
 intent=**new** Intent(MainActivity.**this**,MyInfoActivity.**class**);  
 startActivity(intent);  
 **break**;  
 **case** R.id.***nav\_password***://修改密码  
 changePasswordDialog();  
 **break**;  
 **case** R.id.***nav\_logout***://退出登录  
 **new** AlertDialog.Builder(MainActivity.**this**)  
 .setTitle(**"确认退出吗？"**)  
 .setPositiveButton(**"确定"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 Intent intent1 = **new** Intent(MainActivity.**this**,LogInActivity.**class**);  
 startActivity(intent1);  
 }  
 }).setNegativeButton(**"取消"**, **null**)  
 .create()  
 .show();  
 **break**;  
 **case** R.id.***nav\_search***://搜索  
 searchAction();  
 **break**;  
 **case** R.id.***nav\_add***://添加  
 **if**(*flag* == 0){  
 intent2 = **new** Intent(MainActivity.**this**, EditActivity.**class**);  
 intent2.putExtra(**"type"**, EditActivity.***TYPE\_ADD***);  
 startActivity(intent2);  
 }  
 **else**{  
 Toast.*makeText*(MainActivity.**this**, **"权限不足"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 **break**;  
 }  
 **mDrawerLayout**.closeDrawers();  
 **return true**;  
 }  
 });  
}

3、崔东彬

***Studentdatabasehelper.java***

**package** com.example.double2.studentinfomanager.db;  
*//SQLiteDatabase是Android提供的用来管理Sqlite database的一种数据持久化解决方案。  
// SQLiteDatabase提供了基本的增删改查方法，可以执行SQL命令，执行其他常见数据库管理任务的方法。*

**import** android.content.Context;  
**import** android.database.sqlite.SQLiteDatabase;  
**import** android.database.sqlite.SQLiteOpenHelper;  
  
**public class** StudentDateBaseHelper **extends** SQLiteOpenHelper {  
  
 **public static final** String ***CreateStudentInfo*** = **"create table student ("** + **"number integer primary key, "** + **"gender text , "** + **"name text,"** + **"password text,"** + **"birth text,"** + **"native\_place text,"** + **"specialty text,"** + **"grade text,"** + **"telephone text)"**;  
*//创建一个数据库表名student，其主键为学生学号，并有着性别，姓名，密码，生日，籍贯，专业，成绩，电话等列。*

**public** StudentDateBaseHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, **int** version) {  
 **super**(context, name, factory, version);  
 }  
 *//需要定义一个继承自SQLiteOpenHelper的子类(在实例中为DatabaseOpenHelper类)，并在此类中重写onCreate()和onUpgrade()方法  
 /\*\*  
 \* context 上下文  
 \* name 数据库的名字  
 \* factory 工厂游标  
 \* version 数据库的版本 >=1  
 \*/* @Override  
 **public void** onCreate(SQLiteDatabase db) {  
 db.execSQL(***CreateStudentInfo***);  
 }  
  
 @Override  
 **public void** onUpgrade(SQLiteDatabase arg0, **int** arg1, **int** arg2) {  
  
 }  
}

Mainactivity.java中：

*//数据存储***private** StudentDateBaseHelper **mStudentDateBaseHelper**;  
**private** SQLiteDatabase **mSQLiteDatabase**;

*//控件*

**private** FloatingActionButton **fabtnSort**;

**protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***act\_main***);

*//在Activity中实例化自定义的DBHelper就可以通过getWritableDatabase方法获得对应的SQLiteDatabase对象，  
// 通过这个对象就可以对数据库进行增删改查操作了。（insert：插入）（update：更新）（delete：删除）（query：查询）  
//当数据库不可写入时（如磁盘空间已满）getReadableDatabase( )方法返回的对象将以只读的方式去打开数据库*

**mStudentDateBaseHelper** = **new** StudentDateBaseHelper(**this**, **"StudentInfo.db"**, **null**, 1);  
 **mSQLiteDatabase** = **mStudentDateBaseHelper**.getReadableDatabase();  
 **mSharedPreferences** = **this**.getSharedPreferences(**"student"**, ***MODE\_PRIVATE***);  
 setTestData();  
 initView();  
 receiveFlag();  
}

**fabtnSort** = (FloatingActionButton) findViewById(R.id.***fabtn\_main\_sort***);  
*//点击排序按钮***fabtnSort**.setOnClickListener(**new** View.OnClickListener(){  
 @Override  
 **public void** onClick(View v){  
 sort();  
 *changeOrder* = !(*changeOrder*);  
 Toast toast = Toast.*makeText*(MainActivity.**this**, **"按"**+(*changeOrder*?**"升序"**:**"降序"**)+**"排序成功"**, Toast.***LENGTH\_SHORT***);  
 *showMyToast*(toast,500);  
 }  
});

*//排序函数***private void** sort() {  
 ArrayList<String> number = **new** ArrayList<>();  
 ArrayList<String> name = **new** ArrayList<>();  
 ArrayList<String> grade = **new** ArrayList<>();

*//字符串数组*  
 Cursor mCursor;  
 **if**(!*changeOrder*){

*// query后面参数的定义：*

*//table: 执行查询数据的表名  
//colummns: 要查询出来的列名。如select语句select关键字后面的部分  
//whereClause: 查询条件子句，where关键字后面的部分，在条件子句中允许使用占位符”?”  
//selectionArgs: 用于为whereClause子句中的占位符传入参数，值在数组中的位置与占位符在语句中的位置必须一致。否则就会有异常。  
//groupBy： 相当于select语句groupby关键字后面的部分  
//having： 相当于select语句having关键字后面的部分  
//orderBy： 相当于select语句orderby关键字后面的部分  
//limit： limit参数控制最多查询几条记录(用于控制分页的参数).*

*//此处在order by处添加grade asc与desc实现游标查询的按成绩（升降交替）排序*  
 mCursor = **mSQLiteDatabase**.query(**"student"**,**null**,**null**,**null**,**null**,**null**,**"grade ASC"**);  
 }**else** {  
 mCursor = **mSQLiteDatabase**.query(**"student"**, **null**, **null**, **null**, **null**, **null**, **"grade DESC"**);  
 }  
 **int** size = mCursor.getCount() < ShowAdapter.***maxSize*** ? mCursor.getCount() : ShowAdapter.***maxSize***;

*//获得数据库行数作为班级人数与showadapter定义的最大可显示列表数比较*  
  
  *// 你必须知道每一列的数据类型。Cursor是一个随机的数据源。  
 //Cursor翻译成中文是"游标,光标"的意思，在Android中它代表了一行或多行的数据集合，  
 // 比方说从数据库中查询的数据放在一个表中，我们可以通过使用Cursor游标去指定每一行的数据，并通过属性名称去查询指定所在的行的属性值。  
 // 所有的数据都是通过下标取得。  
 //·close()——关闭游标，释放资源  
 //·copyStringToBuffer(int columnIndex, CharArrayBufferbuffer)——在缓冲区中检索请求的列的文本，将将其存储  
 //·getColumnCount()——返回所有列的总数  
 //·getColumnIndex(String columnName)——返回指定列的名称，如果不存在返回-1  
 //·getColumnIndexOrThrow(StringcolumnName)——从零开始返回指定列名称，如果不存在将抛出IllegalArgumentException 异常。  
 //·getColumnName(int columnIndex)——从给定的索引返回列名  
 //·getColumnNames()——返回一个字符串数组的列名  
 //·getCount()——返回Cursor 中的行数* **while** (**true**) {  
 **if** (size-- == 0)  
 **break**;  
 mCursor.moveToNext();  
 number.add(mCursor.getString(mCursor.getColumnIndex(**"number"**)));  
 name.add(mCursor.getString(mCursor.getColumnIndex(**"name"**)));  
 grade.add(mCursor.getString(mCursor.getColumnIndex(**"grade"**)));  
 }  
 mCursor.close();  
 **rvMain**.setAdapter(**new** ShowAdapter(MainActivity.**this**, number, name, grade,*flag*));  
}

4、徐艳玲

**etSearch** = (EditText) findViewById(R.id.***et\_search***);*//选择学号或姓名进行搜索***switch** (**currentSearchType**) {  
 **case *TYPE\_SEARCH\_NUMBER***:  
 **etSearch**.setHint(**"请输入你要搜索的学生学号"**);  
 **break**;  
 **case *TYPE\_SEARCH\_NAME***:  
 **etSearch**.setHint(**"请输入你要搜索的学生姓名"**);  
 **break**;  
}

**rvSearch** = (RecyclerView) findViewById(R.id.***rv\_search***);*//获取对象，循环视图* **rvSearch**.setLayoutManager(**new** LinearLayoutManager(SearchActivity.**this**));*//垂直布局* **etSearch**.addTextChangedListener(**new** TextWatcher() {*//监听用户输入状态* @Override  
 **public void** beforeTextChanged(CharSequence s, **int** start, **int** count, **int** after) {  
 }  
 @Override  
 **public void** onTextChanged(CharSequence s, **int** start, **int** before, **int** count) {  
 }  
 @Override  
 **public void** afterTextChanged(Editable s) {  
 refreshRecyclerView(s + **""**);  
 }  
 });  
 }  
  
 **private void** refreshRecyclerView(String s) {  
 ArrayList<String> number = **new** ArrayList<>();  
 ArrayList<String> name = **new** ArrayList<>();  
 **mStudentDateBaseHelper** = **new** StudentDateBaseHelper(**this**, **"StudentInfo.db"**, **null**, 1);*//创建数据库* **mSQLiteDatabase** = **mStudentDateBaseHelper**.getReadableDatabase();*//打开数据库* Cursor mCursor;*//当数据发生改变的时候，可以实现列表数据自动刷新* **switch** (**currentSearchType**) {  
 **case *TYPE\_SEARCH\_NUMBER***:  
 mCursor = **mSQLiteDatabase**.query(**"student"**, **null**, **"number like ?"**, **new** String[]{**"%"** + s + **"%"**}, **null**, **null**, **null**);  
 **break**;  
 **case *TYPE\_SEARCH\_NAME***:  
 mCursor = **mSQLiteDatabase**.query(**"student"**, **null**, **"name like ?"**, **new** String[]{**"%"** + s + **"%"**}, **null**, **null**, **null**);  
 **break**;  
 **default**:  
 mCursor = **mSQLiteDatabase**.query(**"student"**, **null**, **"number like ?"**, **new** String[]{**"%"** + s + **"%"**}, **null**, **null**, **null**);  
 }  
 **int** size = mCursor.getCount() < ShowAdapter.***maxSize*** ? mCursor.getCount() : ShowAdapter.***maxSize***;  
 **while** (**true**) {  
 **if** (size-- == 0)  
 **break**;  
 mCursor.moveToNext();*//逐条读取数据* number.add(mCursor.getString(mCursor.getColumnIndex(**"number"**)));*//获取序列号* name.add(mCursor.getString(mCursor.getColumnIndex(**"name"**)));  
 }  
 mCursor.close();  
 **rvSearch**.setAdapter(**new** SearchAdapter(SearchActivity.**this**, number, name));   
}

5、赵婧娟

**private void** countInfo(){  
 *//将数据库中的表项按成绩降序排列* Cursor mCursor = **mSQLiteDatabase**.query(**"student"**, **null**, **null**, **null**, **null**, **null**, **"grade desc"**);**int** size = mCursor.getCount();**int** sum = 0;**int** pass = 0;**int**[] grades = **new int**[size];  
 *//初始化成绩数组grades[i]* **for**(**int** i = 0;i <size;i++){  
 *//移动光标到下一行* mCursor.moveToNext();  
 grades[i] = Integer.*parseInt*(mCursor.getString(mCursor.getColumnIndex(**"grade"**)));  
 sum += grades[i];  
 }  
 mCursor.close();  
 *//判断及格人数* **for**(**int** i = 0;i < size;i++) {  
 **if** (grades[i] >= 60 && grades[i + 1] < 60) {  
 pass = i + 1;  
 } **else if** (grades[size - 1] >= 60) {  
 pass = size;  
 } **else if** (grades[0] < 60){  
 pass = 0;  
 }  
 }  
 *//计算平均分、最高分、最低分、及格人数、不及格人数* String averageNum = String.*valueOf*(sum/size);  
 String maxNum = String.*valueOf*(grades[0]);  
 String minNum = String.*valueOf*(grades[size-1]);  
 String passNum = String.*valueOf*(pass);  
 String failNum = String.*valueOf*(size-pass);  
  
 *//显示平均分、最高分、最低分、及格人数、不及格人数* **average**.setText(averageNum);  
 **highest**.setText(maxNum);  
 **lowest**.setText(minNum);  
 **pass\_num**.setText(passNum);  
 **fail\_num**.setText(failNum);  
 Toast.*makeText*(CountActivity.**this**, **"成绩统计成功！"**, Toast.***LENGTH\_SHORT***).show();

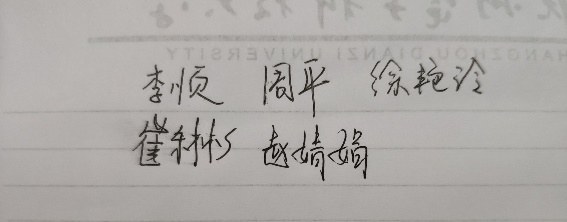
}  
  
 *//界面* **private void** initView() {  
 **if** (Build.VERSION.***SDK\_INT*** >= Build.VERSION\_CODES.***KITKAT***) {  
 *//透明状态栏* getWindow().addFlags(WindowManager.LayoutParams.***FLAG\_TRANSLUCENT\_STATUS***);  
 }  
 **btnBack** = (Button) findViewById(R.id.***btn\_edit\_back***);  
 **average** = (TextView) findViewById(R.id.***average***);  
 **highest** = (TextView) findViewById(R.id.***highest***);  
 **lowest** = (TextView) findViewById(R.id.***lowest***);  
 **pass\_num** = (TextView) findViewById(R.id.***pass\_num***);  
 **fail\_num** = (TextView) findViewById(R.id.***fail\_num***);  
 }

### 10声明承诺

**诚 信 承 诺**

谨在此承诺：本课题《基于Android开发平台的班级管理系统》均系本团队成员独立合作开发完成，没有抄袭行为，凡涉及其他作者的观点和材料，均作了注释，若有不实，后果由团队成员集体承担。

**承诺人（签名）：**



**2019 年 07 月 07 日**