**详细设计**

**用户界面草图**

屏幕上有字

描述已自动生成手机屏幕截图

描述已自动生成手机屏幕截图

描述已自动生成手机屏幕截图

描述已自动生成手机屏幕的截图

描述已自动生成手机屏幕的截图

描述已自动生成手机屏幕截图

描述已自动生成

**直接实现全部功能的方法**

此部分提供给用户界面直接调用，实现功能可调用相应的底层查询方法

*//处理用户的登陆*

public boolean Login(String acc, String pas)

*// 用学号查询学生*

public Student Get\_stu(int num)

*// 添加学生*

public void Add\_stu(Student stu)

*// 修改学生信息*

public void Modify\_stu(Student bef, Student aft)

*// 学生日程*

public Course[][][] Get\_course(int num)

*// 添加日程*

public void Add\_course(Course cou, int num, Boolean week, int day, int time)

*// 修改日程*

public void Modify\_course(Course aft, int num, Boolean week, int day, int time)

*// 导入学生资料*

public void Import\_stu(String file)

*// 导入课程表*

public void Import\_course(String file)

*// 批量查找*

public Student[] Get\_Many\_stu(String num)

*// 批量导出学生*

public void Export\_stu(int[] num)

*// 查找社团列表*

public String[] Get\_union\_name()

*// 批量加入社团*

public boolean Join\_union(String num)

*// 从社团移除学生*

public void Remove\_from\_union(Student[] stu)

*// 获取社团的学生列表*

public Student[] Get\_union\_stu(String name)

*// 获取社团信息*

public Union Get\_Union(String name)

*// 用一组学号列表查空*

public int[][][] Get\_free(int[] num)

*// 查一个社团里面的空档*

public int[][][] Get\_free(Union uin)

**查询方法**

//检查当前账号和密码是否有权限进入系统

public boolean check\_user(String acc, String pas)

因为可以登陆系统的用户不太多，直接用一个链表存储分别放有账号和密码的大小为2的String数组即可

*//通过学号获取一个学生对象*

public Student get\_stu(int num)

*//通过姓名获取一个学生对象*

public Student get\_stu(String name)

这个操作需要从整个学生数据库里面查找学生。通过学号查找的我打算用Red-Black Tree以来处理查找操作，预计即使数据量很大，也能快速处理。通过姓名查找的重载函数我打算用基于Linear Probing的Symbol Table来实现。

*//删除一个给定学号的学生*

public void del\_stu(int num)

*//添加一个学生给定学号的学生*

public void add\_stu(int num, Student stu)

两个操作在更改时都要同时修改到Red Black Tree和Symbol Table中的信息。

*// 返回一个可以遍历的所有社团的名字*

public Iterable<String> get\_uni\_name()

*// 通过社团名获取指定的一个社团对象*

public Union get\_nui(String name)

*// 添加新的社团*

public void add\_uni(Union uni)

*// 删除社团*

public void del\_uni(String name)

存储社团用的是跟上面一样的Symbol Table。第一个方法可以把所有社团的名字塞进一个链表里再返回。

private RedBlackBST<Integer, Course[][][]> stu2course;

private RedBlackBST<Integer, Integer>[][][] course2stu;

*// 获取指定学生的课程*

public Course[][][] get\_cour(int num)

*// 添加课程*

public void add\_cour(Course cour, int num, Boolean week, int day, int time)

*// 删除课程*

public void del\_cour(Course cour, int num, Boolean week, int day, int time)

*//查空*

public int[][][] get\_free(int[] stu\_num)

在实现这些功能的时候我用了两个特殊的数据结构来存储，为的就是让程序无论怎么查找，都能快速得到结果。所有涉及修改的方法在进行的时候都需要对两个结构中的值同时修改。