

# 1. 需求分析：

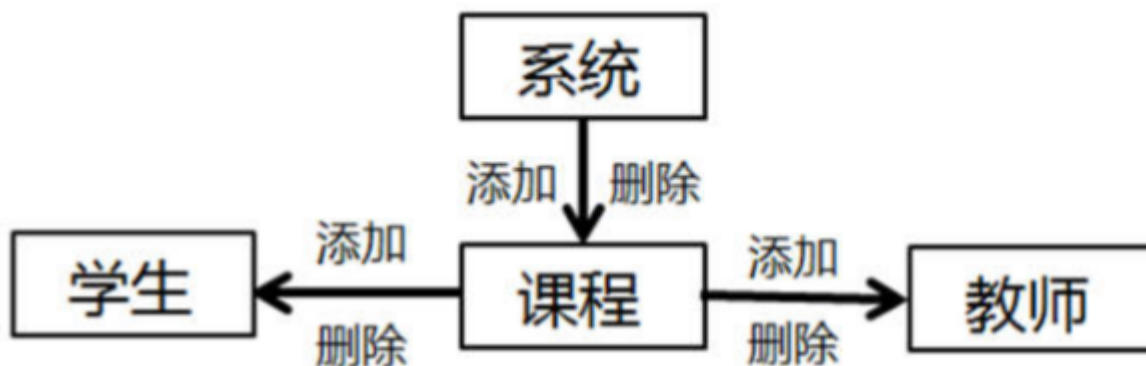
---

选课系统需求分析如下：

- 系统添加课程。将一门课程加入到系统数据中。课程提交重复时给出提示信息。
- 系统删除课程。以课程编号为索引删除课程。系统无此课程时给出提示。
- 课程添加学生。把学生的姓名、学号等信息加入到课程中。学号重复时给出提示信息。
- 课程删除学生。以学号为索引从课程中删除学生。课程无此学生时给出提示。
- 课程添加教师。把教师的姓名、教工号等信息加入到课程中。教工号重复时给出提示信息。
- 课程删除教师。以教工号为索引从课程中删除教师。课程无此教师时给出提示。
- 课程输出数据。提供某课程的学生、教师信息。

## 2. 功能结构图

---



- $A \rightarrow B$ 表示A包含B
- 箭头旁文字表示A对B的操作

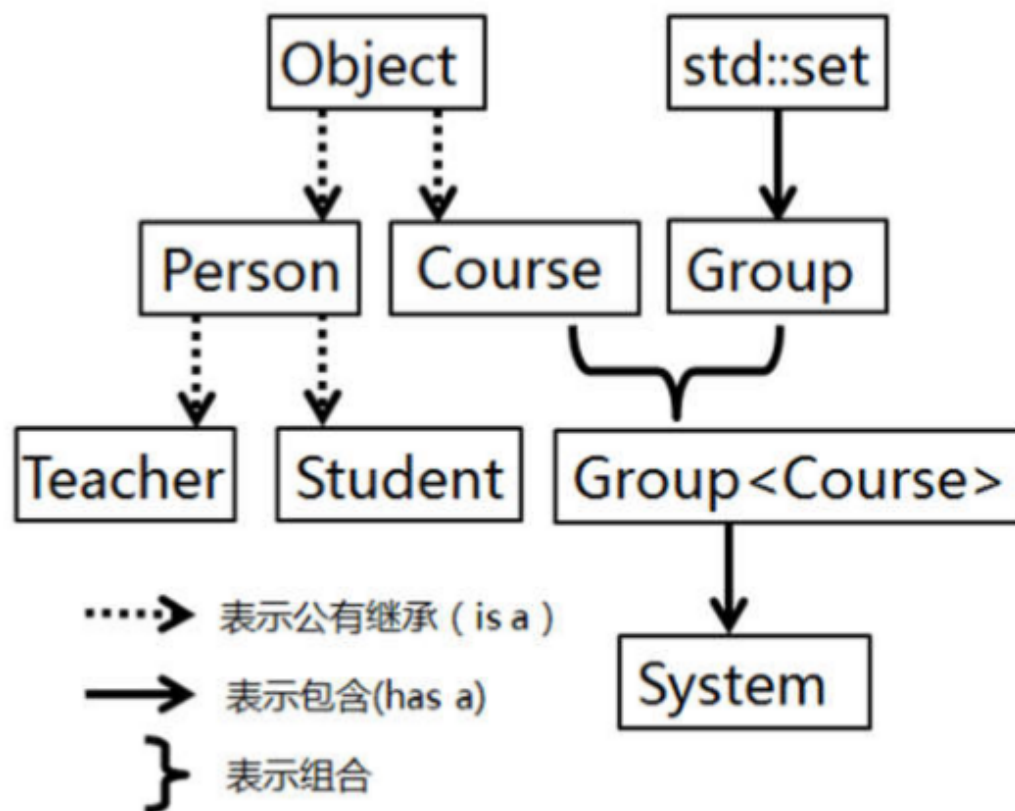
## 3. 实现思路

---

- 结构间关系：系统中包含了一系列课程，课程由名称和编号构成；课程 包含了任课教师和选课学生，学生和教师都含有名称和学号/教工号等信息。系统负责管理课程，课程负责管理教师和学生。
- 具体逻辑：“添加”操作需要“名称”和“编号”两项信息。“删除”操作只需要“编号”作为索引。编号保证唯一，是确定身份的凭证。

## 4. 数据设计

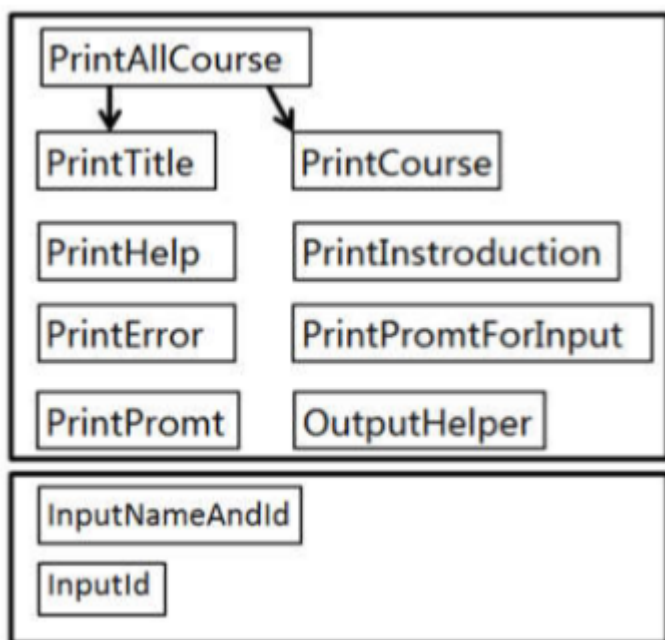
---



- Object(abstract base class):抽象基类。定义了“姓名”和“编号”两种属性。它是 Student (class) , Teacher(class), Course(class) 最终的父类
- Person (class) :表示人。没有额外定义属性。Student(class)和 Teacher(class)直接继承这个类。
- Teacher(class):表示教师。没有额外定义属性。
- Student(class):表示学生。没有额外定义属性。
- std::set (templateclass)
- Group(template class):对 std::set 作轻度封装
- Course(class):表示课程。包含成员 Group和 Group, 定义一系列添加、删除、访问的操作。
- System (class) :表示整个系统的类。是整个程序的核心。包含成员 Group。定义了一系列添加、删除、访问操作。

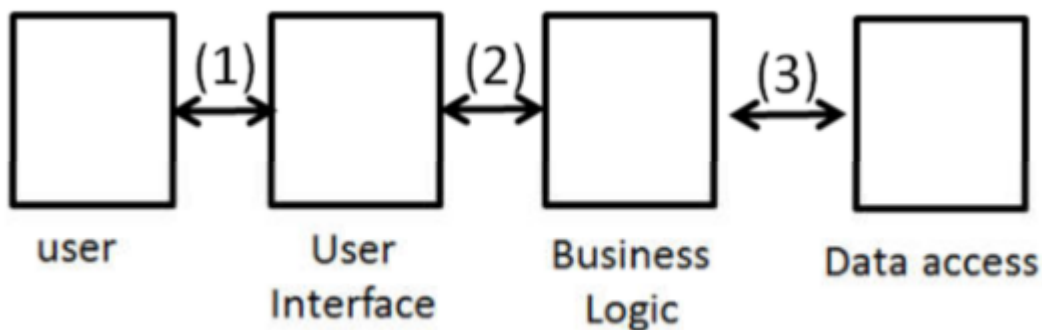
## 5. 函数设计（文件 IOHelper.hpp/.cpp 中定义了许多与 IO 有关的函数）

---



- `void OutputHelper(const string& s1, const string& s2)` //格式化输出的辅助函数。以上几乎所有函数都调用 `OutputHelper`。
- `void PrintAllCourse(const System& sys)` //调用 `PrintTitle` 和 `PrintCourse` 以表格的形式打印所有的课程
- `void PrintCourse(const Course& crs)` //输入一个课程的全部信息。
- `inline void PrintPromt();` //打印“>>>”用以实现交互界面。
- `inline void PrintPromtForInput();` //打印“...”用以等待用户输入更多信息
- `void PrintIntroduction();` //打印指导信息。每次运行该系统都会输出该内容。
- `void PrintHelp();` //打印帮助信息。在系统中输入 `help` 会调用该函数输出帮助。
- `inline void PrintError(const string& s);` //打印错误信息。该函数会将参数 `s` 直接打印出来（加上换行符）。
- `inline void PrintTitle()` //打印表格的题头。执行实际的打印行为。
- `T InputNameAndId()` //模板函数，接受 `Name` 和 `Id`，返回 `T`。`T` 是应当是 `Student`, `Teacher`, `Course` 中的一个
- `T InputId()` //模板函数，接受 `Id`，返回 `T`。同上

## 6. 数据交流

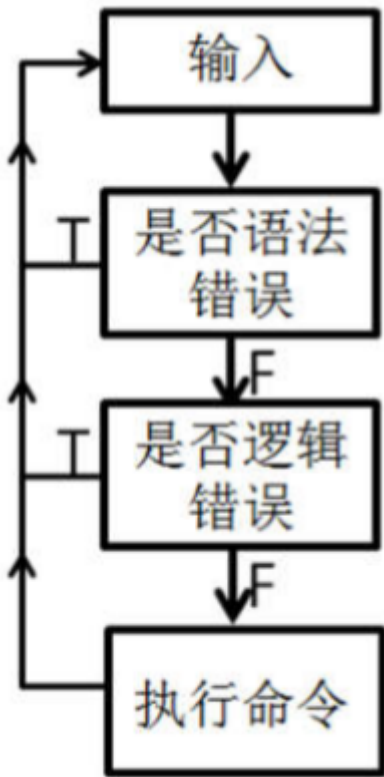


- `User` 指用户，即系统管理员 `User Interface` 指程序的交互层，包括文件 `main.cpp`, `IOHelper.*`。
- `Business Logic` 指业务逻辑层。包括除上述文件以外的所有文件。
- `Data access` 数据访问层。系统采用文件 `IO` 的方式储存数据。

说明：

- User Interface: 定义了一系列函数，构成交互界面，检测流过的信息合法，并负责 User 和 Business Logic 的数据交换。
- Business Logic: 定义了 System (class) 及与其相关的一系列 class。处理内部数据。
- Data Access: 由于系统功能的特殊性，System (class) 的构造和析构函数分别负责从文件读写数据。
- User: User 为 admin，有修改数据的全部权限。

## 7. main 函数的算法流程：



- 输入：输入使用 `getline` 读取整行。
- 是否语法错误：使用正则表达式检查语法错误。若格式不匹配，输出错误提示信息，重新等待输入。
- 是否逻辑错误：调用 `System` 类并查看返回值。若返回 `false` 表示出现逻辑错误。输出提示信息，重新等待输入。
- 执行命令：命令被执行。

## 8. 命令

说明：

- `:` 尖括号内的单词为一个命令。即 `or1` 是一个命令。
- `|` 或运算符链接的 `n` 个命令表示需要且只能出现其中一个。
- `stu`, `tea`, `crs` 分别为 `Student`, `Teacher`, `Course` 的缩写。
- `prta` 为 `print all` 的缩写。

合法命令如下：

- 
-

- 
- 
- 
- 

## 9. 使用的正则如下:

---

匹配整个输入的格式:

```
^help|quit|prta|(add|rm|prt)(stu|tea|crs)$
```