

课程设计报告

**2023—2024学 年 第 二 学 期**

**《软件工程课程设计》报告**

专业班级 \*\*\*\*\*\*\*\*\*\*\*\*\*\*

姓 名 \*\*\*

学 号 \*\*\*\*\*\*\*\*\*

开课系室 软件工程

考核日期 \*\*\*\*年\*\*月\*日

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 报告内容 | 需求分析 | 概要设计 | 详细设计 | 测试 | 工具使用 | 总分 |
| 得 分 | 0 | 0 | 0 | 0 | 0 | 0 |
| 阅卷人 |  | | | | | |

|  |  |
| --- | --- |
| 需求分析评语 |  |
| 概要设计评语 |  |
| 详细设计评语 |  |
| 软件测试评语 |  |
| 工具使用评语 |  |

[**1.前言 1**](#_Toc15190)

[1.1 项目描述 2](#_Toc2618)

[1.2 项目背景 2](#_Toc2495)

[1.3项目目的 2](#_Toc14416)

[**2.需求分析 4**](#_Toc4147)

[2.1功能需求分析 4](#_Toc23799)

[2.2其他需求分析 4](#_Toc15313)

[2.3 实体-联系图（E-R图） 5](#_Toc4021)

**3.逻辑模型 6**

[3.1 业务流程分析 6](#_Toc18561)

[3.2 数据流图 6](#_Toc31913)

[3.3 数据字典 10](#_Toc19832)

[**4.概要设计 13**](#_Toc24359)

[**5.详细设计 17**](#_Toc32631)

[5.1 模块详细设计 17](#_Toc10147)

[**6.系统测试 21**](#_Toc2172)

[5.1 白盒测试 21](#_Toc19768)

[5.2 黑盒测试 22](#_Toc30932)

1. 前言

**1.项目描述：**

本系统主要由管理员模块、教师模块、学生模块，对于不同的模块设计功能包括了管理员对教师、课程、学生的管理，教师对所教授课程的学生成绩管理，学生对自己的成绩等信息的查询和基本信息的维护。

2.项目背景：

随着教育领域的不断发展和数字化趋势的兴起，学校和教育机构面临着越来越复杂的教务管理挑战。传统的手动管理方式已经无法满足现代教育的需求，因此有必要引入一种高效、自动化的系统来简化和改进教务管理流程。

3.项目目的：

教务管理系统的目的是提供一种全面、灵活且易于使用的解决方案，以便学校和教育机构能够更好地管理学生信息、课程安排、考试成绩、教职工信息等方面的任务。该系统旨在优化教务流程，提高管理效率，同时提供及时、准确的信息以支持教学和决策过程。

关键问题和需求：

1. 学生信息管理： 系统应能够有效管理学生的基本信息、选课情况、学术记录等，以确保学生档案的完整性和准确性。

2. 课程安排和管理： 实现课程的排班、教室分配、教学计划的制定，以满足学校的教学需求，同时确保合理的资源利用。

3. 考试和成绩管理：\*系统需要支持考试安排、成绩录入、学生成绩查询等功能，以便教育工作者能够更好地跟踪学生的学术表现。

4. 教职工管理：\*管理教师和其他教育工作者的信息，包括聘用信息、授课安排、教学评估等。

二、教务管理系统的需求分析

1：功能需求：

学校教务处可通过软件向各个学院发布信息。

工作量计算子系统可以协调。

学院教学主管人员可以发布一些要求。

任课教师可以向学院主管人员汇报工作。

学院主管人员可向教务处汇报。

2：性能需求：

（1）原则上，每位教师每学期主讲的课程门数不超过2门。

（2）严格限制：每位教师每年教学工作量不得低于300，不得超过600。

（3）本学期出现严重教学事故的教师不能承担下各学期的主讲任务。

（4）本系统的输入项至少包括：教务处布置的教学计划和工作量计算子系统、学院教师自报的授课计划和学院制定的有关授课限制条件。

（5）本系统的输出项至少包括：教务处最终下达全院教师的教学任务书和学院各个班级下各学期的课程表（可以不含上课地点）。

3：环境需求：

操作简便，界面简洁大方，快捷的信息管理，系统运行稳定，安全可靠，并且视觉上易于理解。

4：接口需求：

要求有工作量计算子系统：

a. 单班课 计划学时\*1.0

双班课 计划学时\*1.3

三班课 计划学时\*1.6

四班课 计划学时\*2.0

b.双语教学

计划学时\*合班系数\*2.0

c.精品课程

国家级精品课程 计划学时\*合班系数\*1.5

省级精品课程 计划学时\*合班系数\*1.1

5：用户界面需求：

对用户登录密码要有安全保护

**6：数据需求：**

为了能够准确的获取数据需求，采用E-R图的形式对需求进行分析和描述。具体分析如下图所示：

|  |
| --- |
|  |
| 教务管理系统之子系统——学院课程安排的E-R图 |

## 三、教务管理系统的逻辑模型

## 1.基础应知：

：箭头，表示数据流；

：圆或椭圆，表示变换数据的处理；

：方框，表示数据的三原点或者终点；

：双杠或者单杠，表示数据存储文件。

源点和终点是系统之外的实体，表明数据的来源和去向；

加工是对数据处理的单元，通常为动词短语；

数据流是数据在系统中运动的方向，通常为名词或者名词短语；

数据存储是暂时存储数据的。

### 2.数据流图的生成

（1）首先确定：

实体：教务处，学院，任课教师，工作量计算子系统以及学生；

加工主要分为：下发有效的课程计划，生成有效的教学任务书，产生课表。

（2）数据流图：

#### 第0层

|  |
| --- |
|  |
| 图0 教务管理系统顶层数据流程图 |

#### 第1层

如下图所示：

对于教务管理系统进行进一步划分：

|  |
| --- |
|  |
| 图1 教务管理系统1层数据流程图 |

#### 第2层

如下图所示：

主要是对1，2，3，4进行了进一步的划分：

|  |
| --- |
|  |
| 图2 教务管理系统2层数据流程图 |

### 3.数据字典

（1）数据流条目

教学计划=课程名称+课程代码+课时+班级类别+班号

班级类别=本科+专科+成人教育+研究生

工作量=计划学时合计之和

学院教学主管人员=学院教学主管人员

相关限制=任课老师的职称+上课的班数+最高周学时数+最低周学时数

任课老师的职称=教授+副教授+讲师

授课计划=课程名称+课程代码+课时+班级类别+班号

教学计划=课程名称+课程代码+课时+班级类别+班号

有效教学计划=课程名称+课程代码+课时+班级类别+班号

教学任务书=课程名称+课程代码+课时+班级类别+班号

课表=课程名称+课程代码+课时+班级类别+班号

（2）数据存储条目

文件名：教学任务

组成：任课老师的职称+上课的班数+最高周学时数+最低周学时数

组织方式：索引文件，以任课老师的职称为关键

文件名：相关限制

组成：教工号+课程名称+课程代码+课时+班级类别+班号+限制条件+计划学时+合班系数

组织方式：索引文件，以课程代码为关键

文件名：教师信息

组成：教工号+课程名称+课程代码+课时+班级类别+班号+计划学时+合班系数

组织方式：索引文件，以课程代码为关键

文件名：教学任务书

组成：教工号+课程名称+课程代码+课时+班级类别+班号+计划学时+合班系数

组织方式：索引文件，以课程代码为关键

文件名：工作量计算子系统

组成：工作量

组织方式：索引文件，以工作量为关键

（3）数据项

课程名称：别名：无

类型：字符串

长度：2{汉字}15

课程代码：别名：无

类型：整型

长度：5位

课时：别名：无

类型：整型

长度：2位

（4）加工条目

加工名：教学管理系统

编号：0

输入：教学计划，授课计划，相关限制，工作量

输出：教学任务书，课表

加工名：下发教学计划

编号：1

输入：教学任务，教学计划

输出：课程相关限制，教学任务

加工逻辑：将教学计划放入教务系统当中

加工名：教师上报

编号：2

输入：相关限制，工作量

输出：授课计划

加工逻辑：按照相关限制和自己的工作量，上交自己的授课计划

加工名：生成教学任务书

编号：3

输入：授课计划

输出：教学任务书

加工逻辑：根任课教师的授课计划生成有效的教学任务书

加工名：产生课表

编号：4

输入：教学任务书

输出：课表

加工逻辑：将教学任务书生成有效的学生课表

加工名：审核

编号：1.1

输入：教学计划

输出：教学任务

加工逻辑：将审核后教学计划放入教务系统的教学任务当中

加工名：下发教学计划

编号：1.2

输入：教学任务

输出：有效教学计划

加工逻辑：将审核后教学计划公示

加工名：学院教学主管人员给出限制

编号：1.3

输入：有效教学计划

输出：相关限制

加工逻辑：将教学计划限定在相关限制当中

加工名：查询教师信息

编号：2.1

输入：教师信息

输出：是否存在

加工逻辑：判断教师是否存在

加工名：判断教学事故

编号：2.2

输入：教师存在

输出：是否合格

加工逻辑：

If无教学事故

then 输出合格教师

else 输出未通过的人员

endif

加工名：判断工作量

编号：2.3

输入：合格后的教师，工作量

输出：工作量，是否合格

加工逻辑：

If合格

then 输出合格教师

else 输出未通过的人员

endif

加工名：自报课程计划

编号：2.4

输入：合格后的教师

输出：授课计划

加工逻辑：将合格的教师的授课计划上报教务系统

加工名：教研室协调认可

编号：3.1

输入：授课计划

输出：教学计划

加工逻辑：将有效授课计划实施

加工名：学院主管教学人员批准

编号：3.2

输入：教学计划

输出：教学任务书

加工逻辑：

if 满足条件

then 输出已通过的教学任务书

else 未输出

endif

加工名：全院教师查看

编号：4.1

输入：教学任务书

输出：教学任务书

加工逻辑：通知教师查看

加工名：产生课表

编号：4.2

输入：教学任务书

输出：课表

加工逻辑：生成课表并下达学生

**四、概要设计：**

用结构化设计方法从需求分析得到的数据流图导出系统初始结构图：

分成四大部分：

1. 下发教学计划部分：

分为审核，下发，录入三个模块；

1. 学院公示部分：

分为获取，查询，判断，输出，公示五个模块；

1. 生成教学任务书部分：

分为：协调，批准，下发三个模块；

1. 产生课表部分：

分为查看，排课，下达三个模块；

根据上述得系统初始结构图：

|  |
| --- |
| 第一部分数据流图： |
| 第一部分结构图： |
| 第二部分数据流图： |
| 第二部分结构图： |
| 第三部分数据流图： |
| 第三部分结构图： |
| 第四部分数据流图： |
| 第四部分结构图： |

**五、详细设计**

用流程图对于各模块进行详细设计：

|  |
| --- |
|  |

用盒图对于第一和第三部分进行详细设计：

|  |
| --- |
|  |

## N-S图

**特点**：只能描述结构化程序所允许的标准结构

**优点**：

1. 功能域表达明确，功能域从盒式图上可以明显地看出来
2. 很容易确定局部和全局数据的作用域
3. 不可能随意转移控制
4. 很容易表达模块的层次结构，并列或嵌套关系
5. 使得软件设计人员遵守结构化程序设计的规定，自然地养成良好的程序设计风格

**3种基本程序结构的表现方法**：

C

A

A

while C

F

B

A

A

until C

A

B

（a）顺序结构 （b）选择结构 （c）循环结构

**PAD图**

**特点**：使程序一目了然，根据PAD图编出的程序，不管由谁来编写，都会得到风格相同的源程序。

**优点**：

1. 用PAD图设计出来的程序必然是结构化程序
2. PAD图所描绘的程序结构十分清晰，图中最左面的竖线就是程序的主线，即第一层结构。随着程序的层次增加，PAD图逐渐向右延伸，每增加一个层次，图形向右扩展一条竖线。PAD图中竖线的总条数就是程序的层次数。  
   （3）PAD图的符号支持自顶向下，逐步求精的方法，左边层次中的内容可以抽象，然后利用def从左向右逐步细化。  
   （4）用PAD图表示的程序逻辑易读、易懂、易记，使用方便。  
   （5）既可表示程序逻辑，也可用于描绘数据结构。  
   （6）自动生成程序。利用软件工具自动完成，省去人工编码工作，有利于提高软件可靠性和软件生产率。

**PDL**

**特点**：

1. 所有关键字都有固定语法，以便提供结构化的控制结构、数据说明和模块化的特征。为了使结构清晰和可读性好，通常在所有可能嵌套使用的控制结构的头和尾部都有关键字。  
   （2）描述处理过程的说明性语言没有严格的语法限制。  
   （3）具有数据说明机制，既包括简单的数据结构(例如，简单变量和数组)又包括复杂的数据结构(例如，链表或层次的数据结构)。
2. 具有模块定义和调用机制，因此，开发人员应根据系统编程所用的语种，说明过程设计语言表示的有关程序结构。

**优点**:

（1）提供的机制比图形全面，有利于保证软件详细设计与编码的质量。  
（2）可以作为注释直接插在源程序中间作为程序的文档，并可以同高级程序语言一样进行编辑、修改，有利于软件的维护并保证文档和程序的一致性，提高了文档的质量。  
（3）可自动生成程序代码，提高软件生产率。目前已有多种PDL版本，为自动生成相应代码提供了便利条件。

**六、系统测试：**

测试用例设计的基本目的是确定一组最有可能发现某个错误或某类错误的测试数据。好的测试用例可以在测试过程中重复使用，但不可能测试程序的每一条路径，也不可能把所有的输入数据都试一次。因此，测试用例的设计人员必须努力以最少量的测试用例来发现最大量的可能错误。

第一部分流程图： 第二部分流程图： 第三部分流程图：

|  |
| --- |
|  |

#### 1.白盒测试：

把测试对象看做一个透明盒子，允许利用程序内部逻辑结构及有关信息，进行测试。通过在不同点检查程序的状态，确定实际的状态是否与预期的状态一致。又称为结构测试或逻辑驱动测试。

完全测试的困难性：对于一个具有多重选择和循环嵌套的程序，不同的路径数目可能是天文数字。

1. 语句覆盖：就是设计若干测试用例，运行被测试程序，使得每一可执行语句至少执行一次。
2. 分支覆盖：就是设计若干个测试用例，运行被测试程序，使得程序中每个判断的取真分支和取假分支至少经历一次，分支覆盖又称为判定覆盖。
3. 条件覆盖：设计若干个测试用例，运行被测程序，使得程序中每个判断的每个条件的可能取值至少执行一次。
4. 条件组合覆盖：设计足够的测试用例，运行被测程序，使得每个判断的所有可能的条件取值组合至少执行一次。
5. 控制流图覆盖测试：将代码变为控制流图（CFG），基于其进行测试的技术。
6. 单条件嵌套：如果判断中的条件表达式是由一个或多个逻辑运算符连接的复合条件表达式，则需要改为一系列只有单个条件的嵌套的判断。
7. 节点、边覆盖（语句覆盖、分支覆盖）。
8. 路径覆盖。

**第一部分：**

主要对于第一部分进行分析（第二、第三、第四部分同理）

第一部分测试代码（C语言）:

//部分测试代码：

While（text==true）｛

Printf（“请输入教学计划：\n”）;

scanf(“%f”,text);

｝

Printf（“形成教学任务并下发”）；

Printf（“存储教学计划”）；

**对于第一部分进行路径覆盖：**

L:第一次教学计划没有通过审核

**对于第一部分进行语句覆盖：**

与路径覆盖相同

**对于第二部分基本路径覆盖：**

**2.黑盒测试：**

测试对象看做一个黑盒子，测试人员完全不考虑程序内部的逻辑结构和内部特性。只依据程序的需求规格说明书，检查程序的功能是否符合它的功能说明。又叫做功能测试或数据驱动测试。

完全测试的困难性：如果考虑所有可能的输入条件和输出条件中，黑盒测试同样可能是天文数字。

1. 等价类划分：把所有可能的输入数据，即程序的输入域划分成若干部分，然后从每一部分中选取少数有代表性的数据作为测试用例。
2. 边界值分析：对等价类划分方法的补充，大量的错误是发生在输入或输出范围边界上，边界测试可以查出更多的错误。
3. 状态测试：在黑盒测试阶段，通过对状态的测试间接的加以验证功能。

等价类划分：

|  |  |  |
| --- | --- | --- |
| 输入条件 | 有效等价类 | 无效等价类 |
| 教师姓名 | 汉字或字母（1） | 非汉子或字母（2） |
| 课程名称 | 汉字或字母（1） | 非汉子或字母（2） |
| 总课程数 | 0<=课程数<=100（3） | 课程数<0,课程数>100（4） |
| 周一上午课程数 | 1——4（5） | 不是1——4（6） |
| 周一下午课程数 | 1——4（5） | 不是1——4（6） |
| …… | 1——4（5） | 不是1——4（6） |

为有效等价类设计测试用例：（尽可能多的覆盖，直到全部覆盖）

|  |  |  |
| --- | --- | --- |
| 测试数据 | 期望结果 | 覆盖用例 |
| 张三 | 输入有效 | （1） |
| 数据库 | 输入有效 | （1） |
| 30 | 输入有效 | (3) |
| 2 | 输入有效 | (5) |

为每一个无效等价类测试用例：

|  |  |  |
| --- | --- | --- |
| 测试数据 | 期望结果 | 覆盖用例 |
| 123 | 输入无效 | （2） |
| 456 | 输入无效 | （2） |
| \*\*\* | 输入无效 | (4) |
| ++++ | 输入无效 | (6) |

边界值分析：

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | 输入等价类 | 测试用例说明 | 测试数据 | 期望结果 | 选取理由 | | 教师姓名 | 汉子或字母 | 刘3 | 输入无效 | 包含数字 | | 课程名称 | 汉字或字母 | 导论R | 输入有效 | 既有汉子又有字母 | | 总课程数 | 0<=课程数<=100 | 101 | 输入无效 | 刚好超出最大值 | | 周一上午课程数 | 1——4 | -1 | 输入无效 | 刚好小于最小值 | |

**第二部分：**

主要对于第二部分进行分析（第一、第三、第四部分同理）

第二部分测试代码（C语言）:

//部分测试代码：

printf("请输入相关限制条件：\n");

scanf("%s",x);

printf("请输入将要查询的教师信息：\n");

scanf("%s",y);

if(bool1==false){

printf("请输入工作量：\n");

scanf("%f",t);

if(t<=MAX){

printf("准备输出教学计划：\n");

}

}

**对于第二部分进行路径覆盖：**

L1:

（没有教学事故）and（超时）

L2：

（没有教学事故）and（没有超时）

L3：

（有教学事故）

**对于第二部分进行语句覆盖：**

If (没有教学事故)

then (输出工作量)

endif

if (判断是否超时)

then（公示授课计划）

endif

**对于第二部分进行分支覆盖：**

可选择L1和L2和L3进行分支覆盖。

**对于第二部分进行分支覆盖：**

则分别选择L1或者L2均可。

**条件组合覆盖则需要全部执行一次。**

**对于第二部分基本路径覆盖：**

|  |  |
| --- | --- |
|  |  |

**基本路径有：**

基本路径1：1，2，3，4，8；

基本路径2：1，2，3，4，5，6，8；

基本路径3：1，2，3，4，5，6，7，8。

#### 2.黑盒测试：

测试对象看做一个黑盒子，测试人员完全不考虑程序内部的逻辑结构和内部特性。只依据程序的需求规格说明书，检查程序的功能是否符合它的功能说明。又叫做功能测试或数据驱动测试。

完全测试的困难性：如果考虑所有可能的输入条件和输出条件中，黑盒测试同样可能是天文数字。

1. 等价类划分：把所有可能的输入数据，即程序的输入域划分成若干部分，然后从每一部分中选取少数有代表性的数据作为测试用例。
2. 边界值分析：对等价类划分方法的补充，大量的错误是发生在输入或输出范围边界上，边界测试可以查出更多的错误。
3. 状态测试：在黑盒测试阶段，通过对状态的测试间接的加以验证功能。

等价类划分：

|  |
| --- |
|  |

为有效等价类设计测试用例：（尽可能多的覆盖，直到全部覆盖）

|  |
| --- |
|  |

为每一个无效等价类测试用例：

|  |
| --- |
|  |

边界值分析：

|  |
| --- |
|  |