**《软件工程》课程教学大纲**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 英文课程名 | Software Engineering | | 总 学 时 | | 48 | 学 分 | 3 |
| 课程编码 | G126067 | | 理论教学学时 | | 32 | 适用专业 | 软件工程 |
| 课程类别（请在课程所属类别栏注明选修或必修） | 通识课程 |  | 实践  教学  学时 | 实验学时 |  | 先修课程 | 数据结构、程序设计、数据库原理  及应用 |
| 大类基础课程 |  | 上机学时 | 16 | 开课学院（部） | 计算机科学与技术学院、软件学院 |
| 专业课程 | 必修课 | 其它 |  | 基层教学组织 | 软件工程系列课程教学团队 |

一、课程简介

该课程是一门针对软件工程专业的本科生专业基础必修课，是高等学校计算机教学计划中的核心课程之一。它具有较强的理论性和综合性，不仅需要软件开发过程和管理的理论和方法等知识，还需要问题建模、程序设计、文档撰写等工程实践知识，同时还和实际软件开发相结合。从课程地位上来说，该课程是一门综合运用先修基础课程所学知识解决实际问题的重要课程。

# 二、教学目标

2.1 课程教学目标

1. 掌握软件工程和软件开发模型的基本概念，软件过程的管理和质量控制，以及软件项目的计划、可行性分析和成本周期估算等知识，并了解软件开发相关技术标准。
2. 掌握软件需求分析、设计概念和一般性过程，能够运用这些知识对复杂软件系统进行需求分析、模块划分和详细设计，并掌握相关文档的编写；掌握程序设计方法，增强编程实践；掌握软件测试技术与维护方法。
3. 掌握UML建模技术，能够在软件工程过程中使用常用建模工具对复杂软件系统进行需求分析、模块分解和设计。

2.2 课程目标与毕业要求（指标点）对应关系

该课程支撑以下毕业要求和具体细分指标点：

【毕业要求3】设计/开发解决方案：能够设计针对复杂软件工程问题的解决方案，设计满足特定需求的系统、单元（部件）或工艺流程，并能够在设计环节中体现创新意识，考虑社会、健康、安全、法律、文化以及环境等因素。

支撑指标点3.2：能够对复杂软件工程问题进行需求分析，能够运用软件工程思想构建系统总体解决方案，确定设计目标并进行模块分解。

【毕业要求5】使用现代工具：能够针对复杂软件工程问题，开发、选择与使用恰当的技术、资源、现代工程工具和信息技术工具，包括对复杂软件工程问题的预测与模拟，并能够理解其局限性。

支撑指标点5.2：能够开发或选择使用恰当的工具和技术用于解决实际软件工程问题。

【毕业要求6】工程与社会：能够基于软件工程相关背景知识进行合理分析，评价软件专业工程实践和复杂软件工程问题解决方案对社会、健康、安全、法律以及文化的影响，并理解应承担的责任。

支撑指标点6.1：熟悉软件领域相关的技术标准、知识产权、产业政策和法律法规。

【毕业要求10】沟通：能够就复杂软件工程问题与业界同行及社会公众进行有效沟通和交流，包括撰写报告和设计文稿、陈述发言、清晰表达或回应指令。并具备一定的国际视野，能够在跨文化背景下进行沟通和交流。

支撑指标点10.2：具有良好的写作能力，能对软件的需求分析、系统设计和系统测试等软件开发各阶段撰写报告和设计文稿。

【毕业要求11】项目管理：理解并掌握工程管理原理与经济决策方法，并能在多学科环境中应用。

支撑指标点11.1：理解软件工程项目开发的特点，掌握项目管理的概念与方法。

支撑指标点11.2：能对软件工程项目进行经济可行性分析和决策。

本课程目标与毕业要求（指标点）的对应关系如表1所示。

表1 课程目标与毕业要求（指标点）的对应关系

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 课程目标 | 毕业要求指标点 | 教学环节 | | | | |
| 课堂授课 | 实验 | 作业 | 课堂讨论 |
| **目标1**：掌握软件工程和软件开发模型的基本概念，软件过程的管理和质量控制，以及软件项目的计划、可行性分析和成本周期估算等知识，并了解软件开发相关技术标准。 | **指标点6.1**：熟悉软件领域相关的技术标准、知识产权、产业政策和法律法规。  **指标点11.1**：理解软件工程项目开发的特点，掌握项目管理的概念与方法。  **指标点11.2**：能对软件工程项目进行经济可行性分析和决策。 | √ | √ | √ | √ |
| **目标2**：掌握软件需求分析、设计概念和一般性过程，能够运用这些知识对复杂软件系统进行需求分析、模块划分和详细设计，并掌握相关文档的编写；掌握程序设计方法，增强编程实践；掌握软件测试技术与维护方法。 | **指标点3.2**：能够对复杂软件工程问题进行需求分析，能够运用软件工程思想构建系统总体解决方案，确定设计目标并进行模块分解。  **指标点10.2**：具有良好的写作能力，能对软件的需求分析、系统设计和系统测试等软件开发各阶段撰写报告和设计文稿。 | √ | √ | √ | √ |
| **目标3**：掌握UML建模技术，能够在软件工程过程中使用常用建模工具对复杂软件系统进行需求分析、模块分解和设计。 | **指标点5.2**：能够开发或选择使用恰当的工具和技术用于解决实际软件工程问题。 | √ | √ | √ |  |

# 三、课程教学内容及学时分配

## 3.1 理论教学安排

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 章节或知识点(模块) | 教学内容 | 学时分配 | 教学要求  (应明确教学重点、难点和教学方法) | 学生任务 | |
| 作业要求 | 其他要求(自学/讨论） |
| 1 | 软件工程概述 | 1. 软件的概念、特点和分类 2. 软件的发展和软件危机 3. 软件过程和能力成熟度模型 4. 软件生命周期模型 5. 软件小组 | 4 | 教学要求：   1. 掌握软件工程的概念； 2. 了解软件危机面临的问题及解决方法； 3. 熟悉软件过程和成熟度模型； 4. 熟悉软件小组的组织。   教学重点：   1. 讲授软件工程的基本知识； 2. 基于案例让学生更直观地了解相关概念。   教学难点：指标点11.1  教学方法：讲授 + 案例式教学。 |  | 自学：软件工程的发展、软件过程等相关知识。  讨论：软件生命周期模型的常见类型，以及优缺点。 |
| 2 | 软件工程工具 | * + 1. 逐步求精法     2. 成本-效益分析法     3. 软件度量     4. CASE的分类和范围     5. 软件版本     6. 配置控制 | 2 | 教学要求：   1. 掌握软件工程的理论方法； 2. 掌握常用的软件工程的工具。   教学重点：   1. 讲授软件工程的理论方法和工具分类。   教学难点：指标点5.2 ，11.1，11.2  教学方法：讲授。 | 常用的软件工程建模工具练习； | 自学：软件工程建模工具的使用。  讨论：软件工程的理论方法和工具分类，以及应用场合。 |
| 3 | 计划和估算 | 1. 计划和软件过程 2. 周期和成本估算 3. 软件项目管理计划 4. 文档标准 5. 用于计划和估算的CASE工具 | 2 | 教学要求：   1. 掌握软件项目管理计划和可行性分析，周期和成本估算； 2. 了解相关文档编写的技术标准； 3. 了解用于计划和估算的工具。   教学重点：   1. 讲授软件管理相关的基本知识； 2. 学生深入自学技术标准和软件管理工具。   教学难点：指标点5.2，6.1，11.1，11.2  教学方法：讲授 + 自主学习。 | 软件项目管理计划的编写。 | 自学：相关文档编写的技术标准。  讨论：用于的计划和估算的工具有哪些，以及成本估算中经常出现的问题。 |
| 4 | 从模块到对象 | 1. 模块内聚和耦合 2. 数据封装 3. 抽象数据类型 4. 信息隐藏 5. 继承、多态和动态绑定 6. 面向对象范型 | 2 | 教学要求：   1. 掌握模块内聚、耦合、数据封装、抽象数据类型等基本概念； 2. 掌握面向对象范型。   教学重点：   1. 讲授模块和面向对象范型的基本知识。   教学难点：指标点3.2  教学方法：讲授。 |  | 自学：面向对象范型的基本概念，以及在程序开发过程中的应用。  讨论：模块内聚与耦合对模块设计的影响。 |
| 5 | 传统的分析 | 1. 需求阶段概述 2. 结构化系统分析和实例介绍 3. 有限状态机 4. Petri网 5. 传统分析阶段的CASE工具 6. 规格说明文档 | 8 | 教学要求：   1. 了解需求获取； 2. 了解传统分析的基本过程和方法； 3. 掌握用例图、数据流图的表示方法。   教学重点：   1. 讲授需求和传统分析基本知识； 2. 基于实际案例教授需求和传统分析方法； 3. 组织学生基于实际问题运用用例图和数据流图表示方法建模。   教学难点：指标点3.2，5.2，10.2  教学方法：讲授 + 案例式教学 + 练习。 | 用例图、数据流图的练习。 | 自学：结合应用案例，进一步掌握传统分析的基本过程和方法。  讨论：用例图、数据流图，在系统分析建模过程中的作用。 |
| 6 | 面向对象分析 | 1. 分析流 2. 功能建模 3. 实体类建模 4. 动态建模 5. 抽象边界类和控制类 6. 面向对象分析实例研究 7. 面向对象分析阶段CASE工具 8. 面向对象分析阶段文档编写 | 14 | 教学要求：   1. 了解面向对象分析的基本过程和方法； 2. 掌握类图、顺序图、状态图等的建模方法，以及相应文档撰写。   教学重点：   1. 讲授面向对象分析基本知识； 2. 基于实际案例教授面向对象分析方法； 3. 组织学生基于实际问题运用面向对象分析各类模型的表示方法进行建模。   教学难点：指标点3.2，5.2，10.2  教学方法：讲授 + 案例式教学 + 练习。 | 类图、顺序图、状态图等练习。 | 自学：结合应用案例，进一步掌握面向对象分析的基本过程和方法。  讨论：顺序图、类图的主要作用，以及如何体现在系统实现中。 |
| 7 | 设计 | 7.1 软件设计的目标和任务  7.2 软件设计基础与技术  7.3 设计和抽象  7.4 结构化的设计方法  7.5 面向数据流设计方法  7.6 面向对象设计方法  7.7 详细设计的形式化技术 | 4 | 教学要求：  了解设计的基本过程和方法；   1. 掌握概要(总体)设计、详细设计、分解与模块化等方法； 2. 了解软件设计的有关问题及启发式规则。   教学重点：   1. 讲授软件设计基本知识； 2. 基于实际案例教授概要设计和详细设计等方法。   教学难点：指标点3.2，5.2，10.2  教学方法：讲授 + 案例式教学。 | 概要(总体)设计、详细设计等练习。 | 自学：结合应用案例，进一步掌握概要(总体)设计、详细设计、分解与模块化等。  讨论：详细设计的主要表示方式有哪些，及其适用的场合。 |
| 8 | 实现阶段 | 1. 编程语言的选择 2. 编码标准 3. 代码复用 4. 实现阶段中的测试 5. 测试用例选择 6. 黑盒单元测试技术 7. 黑盒测试用例 8. 玻璃盒单元测试技术 9. 代码走查和审查 | 4 | 教学要求:   1. 了解软件实现的基本过程和方法，以及常用的软件测试技术。   教学重点：   1. 讲授软件实现相关的基本知识； 2. 学生深入自学编码标准和测试技术。   教学难点：指标点3.2  教学方法：讲授 + 自主学习。 | 系统的编码与测试练习。 | 自学：结合应用案例，进一步掌握系统的编码、测试等。  讨论：软件测试的方法有哪些，以及良好的编码习惯如何养成。 |
| 9 | 交付后的维护 | 1. 维护的必要性 2. 对程序员的要求 3. 交付后维护的管理 4. 面向对象的软件维护 5. 交付后维护技能与开发技能 6. 逆向工程 7. 交付后维护阶段的测试 | 2 | 教学要求：   1. 了解维护的基本概念，明确维护的重要性。   教学重点：   1. 讲授软件维护相关的基本知识。   教学难点：指标点3.2  教学方法：讲授。 | 软件维护练习。 | 自学：结合应用案例，进一步了解维护的基本过程。  讨论：软件维护的重要性。 |
| 10 | UML建模 | 10.1 类图  10.2 用例图  10.3 交互图  10.4 状态图  10.5 活动图  10.6 组件图  10.7 配置图 | 6 | 教学要求：   1. 掌握多种UML图的概念和建模方法。   教学重点：   1. 讲授UML表示方法； 2. 基于实际案例教授UML建模方法。   教学难点：指标点5.2  教学方法：讲授 + 案例式教学。 | UML模型等练习。 | 自学：UML部署图、构件图等内容。  讨论：UML的静态模型与动态模型的区别和联系。 |

## 3.2 实践教学安排

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 项目名称 | 学时 | 类型 | 每组人数 | 教学要求  (应明确教学重点、难点和教学方法) | 学生任务 | |
| 作业要求 | 其他要求(自学/讨论） |
| 1 | 应用系统传统分析和需求文档编写 | 2 | 设计 | 1 | 教学重点：   1. 掌握结构化分析方法和需求文档编写； 2. 熟悉并掌握软件分析工具。   教学难点：指标点3.2、5.2、10.2  教学方法：上机实践。 |  | 自学：实验预习。  讨论：需求文档编写的要求与相关标准。 |
| 2 | 应用系统面向对象分析和和文档编写 | 4 | 设计 | 1 | 教学重点：   1. 掌握面向对象分析方法和需求文档编写； 2. 熟悉并掌握软件分析工具。   教学难点：指标点3.2、5.2、10.2  教学方法：上机实践。 |  | 自学：实验预习。  讨论：面向对象分析方法的特点。 |
| 3 | 应用系统传统设计 | 2 | 设计 | 1 | 教学重点：   1. 掌握结构化设计方法和设计文档编写； 2. 熟悉并掌握软件设计工具。   教学难点：指标点3.2、5.2、10.2  教学方法：上机实践。 |  | 自学：实验预习。  讨论：结构化设计方法的主要方法。 |
| 4 | 应用系统面向对象设计 | 2 | 设计 | 1 | 教学重点：   1. 掌握面向对象设计方法和设计文档编写； 2. 熟悉并掌握软件设计工具。   教学难点：指标点3.2、5.2、10.2  教学方法：上机实践。 |  | 自学：实验预习。  讨论：面向对象设计与结构化设计方法的差别。 |
| 5 | 自选系统分析、设计和实现 | 2 | 设计 | 4-5 | 教学重点：   1. 全面了解分析、设计、代码实现和测试技术； 2. 熟悉各类软件工具。   教学难点：指标点3.2、5.2、10.2  教学方法：上机实践。 |  | 自学：实验预习。  讨论：软件测试的作用。 |
| 6 | 自选系统UML建模 | 4 | 设计 | 4-5 | 教学重点：   1. 掌握所有UML常用模型，能够进行软件项目分析和设计。   教学难点：指标点3.2、5.2、10.2  教学方法：上机实践。 |  | 自学：实验预习。  讨论：UML的9种模型的特点。 |

四、考核方式及成绩评定方式

该课程的考核强调过程化考核。其总成绩分为进程性成绩和期末成绩两部分，分别占50%。进程性成绩主要考核学生的作业、课程大报告两个方面。各部分所占的考核比例及基本要求如下：

（1）作业：占平时成绩20%-25%。要求：教师针对某些知识模块布置一定数量的作业，以巩固学生软件工程的理论基础知识和基本工作原理。对于作业中的共性问题，教师须在课堂讲解，以帮助学生提高和进步。

（2）课程大报告：占平时成绩的25%-30%。要求：学生分小组自选软件项目，完成软件项目管理计划、可行性分析，以及软件需求分析、设计、开发等环节，并完成报告撰写。通过该实践环节，加强学生复杂软件系统分析、设计和开发能力，软件项目管理能力以及文档编写能力。

期末考试将按照本课程的教学目标全面考核学生课程学习的效果，采用闭卷的形式。

# 五、教材、课程网址及参考书目

教 材：《面向对象与传统软件工程 统一过程的理论与实践》，Stephen R.Schach著，韩松、邓迎春译，机械工业出版社，2011年12月，第8版。

课程在学校网络教学平台的地址(核心课程必填)：<http://i.mooc.chaoxing.com/space/index.shtml>。

参考书：

1. 《软件工程：实践者的研究方法》，[普雷斯曼](http://book.jd.com/writer/%E6%99%AE%E9%9B%B7%E6%96%AF%E6%9B%BC_1.html) 著，[郑人杰](http://book.jd.com/writer/%E9%83%91%E4%BA%BA%E6%9D%B0_1.html)等译，机械工业出版社，2011年5月，第7版。
2. 《软件工程》，钱乐秋、赵文耘、牛军钰 著，清华大学出版社，2016年9月，第3版。
3. 《软件工程入门经典》，Rod [Stephens](http://search.dangdang.com/?key2=Stephens&medium=01&category_path=01.00.00.00.00.00)  著，[明道洋](http://search.dangdang.com/?key2=%C3%F7%B5%C0%D1%F3&medium=01&category_path=01.00.00.00.00.00)、[曾庆红](http://search.dangdang.com/?key2=%D4%F8%C7%EC%BA%EC&medium=01&category_path=01.00.00.00.00.00) 译，清华大学出版社，2016年7月。

……

**执笔者：董天阳，王婷**

**审核者：田贤忠**

**课程教学团队成员：江颉，董天阳，王婷，田贤忠**