哈爾濱工業大學
HARBIN INSTITUTE OF TECHNOLOGY

Software Construction: Developing High-Quality Software Systems

Ming Liu

February 24, 2019

任课教师

- **主讲教师：刘铭**
- 副教授/博士生导师
- 社会计算及信息检索研究中心
  - 办公地点：　　哈工大奥校6楼
  - 电子邮件：　　liuming1981@hit.edu.cn
  - 手　　机：　　18646078196

Goals of this Course

- **Goal: understanding both the building blocks and the design principles for construction of software systems 构造原理？**
  - 在高级语言程序设计的基础上，认识软件构造的质量标准与目标，学习软件构造的基本过程，从而具备面向质量目标的复杂软件构造方法与能力
  - 深入学习抽象数据类型 ADT 与面向对象编程 OOP
  - 初步掌握面向关键质量目标（可理解性、可维护性、可复用性、健壮性、时空性能）的软件构造基本技术
  - 了解软件代码重构和面向更复杂软件系统的高级构造技术
- **For each desired program behavior there are infinitely many programs 多种不同的软件构造方案，有什么差异？如何选择？**
  - What are the differences between the variants?
  - Which variant should we choose?
  - How can we synthesize a variant with desired properties?

Goals of this Course

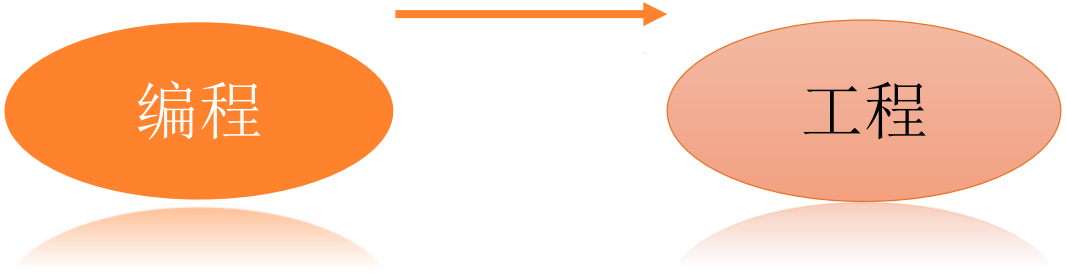
程序设计与实现能力

- 了解软件开发过程中应考虑哪些质量目标
- 掌握面向关键质量目标的软件基本构造技术
- 形成面向质量目标的软件开发思维模式

系统设计与实现能力

- 掌握“面向抽象编程”的核心思想和面向对象软件开发的基本过程
- 能够对实际应用问题进行抽象和建模
- 利用模型与开发者和用户进行有效表达和沟通

Goals of this Course



系统分析与评价能力

- 从个人编程到团队合作的转换从关注单一开发环节到关注全开发过程的转换
- 根据用户期望质量特性进行全生命周期**系统分析与评价**
- 发现系统设计的缺陷并做出优化和改进

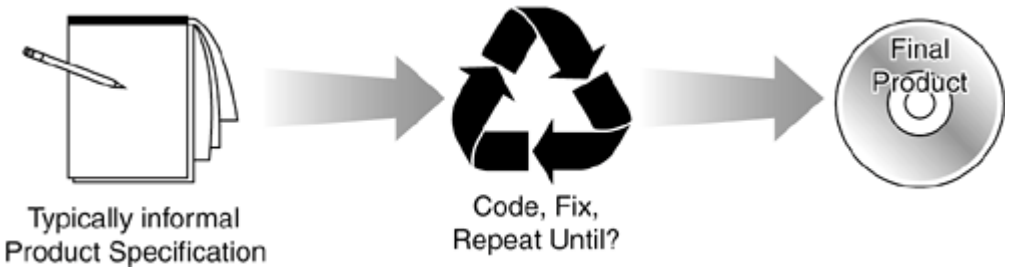
利用现代软件构造工具的能力

- 了解复杂软件系统相对于简单程序的本质差异
- 初步掌握利用各类软件开发工具进行编码、测试和质量保障
- **利用现代软件构造工具**进行高质量和高效率软件开发

A typical software design process

1. **Discuss software that needs to be written**
2. **Write some code**
3. **Test the code to identify the defects**
4. **Debug to find causes of defects**
5. **Fix the defects**
6. **If not done, return to step 1**

写代码----试错----改错，如此循环



Better software design

- **Think before coding** 未雨绸缪
- **Consider non-functional quality attributes** 考虑非功能质量属性
  - **Maintainability, extensibility, performance, …**
- **Propose, consider design alternatives** 考虑多种设计选择
- **Make explicit design decisions** 把设计决策明确写下来

- **Using a design process…**
  - A design process organizes your work
  - A design process structures your understanding
  - A design process facilitates communication

Design goals, principles, and patterns

- **Design goals** enable evaluation of designs
  - e.g. maintainability, reusability, scalability
- **Design principles** are heuristics that describe best practices
  - e.g. high correspondence to real-world concepts
- **Design patterns** codify repeated experiences, common solutions
  - e.g. template method pattern

设计目标：编程的“视野”

设计原则：编程的“标尺”

设计模式：编程的“经验”

课程简介

- **授课对象：** 计算机学院2018级本科
- **课程分类：** 核心基础课
- **学时：** 80 (52+28)
- **先修课程：** C/C++/Java高级语言程序设计；
  计算机系统；数据结构与算法；
- **上课时间/地点：**
  - 1-14周　周一/周三1-2节　正心楼42
  - 1-15周　周三7-8节　格物楼机房 208/213
- **考试时间：**
  - 17周 ？

课程网站

- 软件构造

  http://cms.hit.edu.cn/course/view.php?id=341

  选课密码：SC2019

- 课件、实验与作业要求、各类通知均在此网站发布，实验/作业完成之后也需通过此网站提交。
- 可通过CMS网站提出问题，与教师和同学进行交流。

推荐学习资料

- **MIT Course 6.031: Software Construction**

  http://web.mit.edu/6.031/www/sp17/

6.031: Software Construction

Spring 2017 · Course Staff · MWF11-12:30 (34-101)

General

General Information
Collaboration and Public Sharing
Code Reviewing
Nanoquiz Grading and Makeups
I have a question, who do I ask?

Calendar: classes, assignments, OH/lab

Getting Started

Getting Started: Java, Eclipse, & Git
Getting Started: Java Tutor
Getting Started: Eclipse FAQ

Problem Sets

0: Turtle Graphics
1: Around the World
2: Poetic Walks
3: Calculus
4: Multiplayer Minesweeper

Project

Phase 1: Norn Mailing List System
Phase 2: WebNorn Mailing List System

Quizzes

Quiz 1 and Quiz 1 solutions
Quiz 2 and Quiz 2 solutions

Quiz Archive

Course Archive

Previous semesters

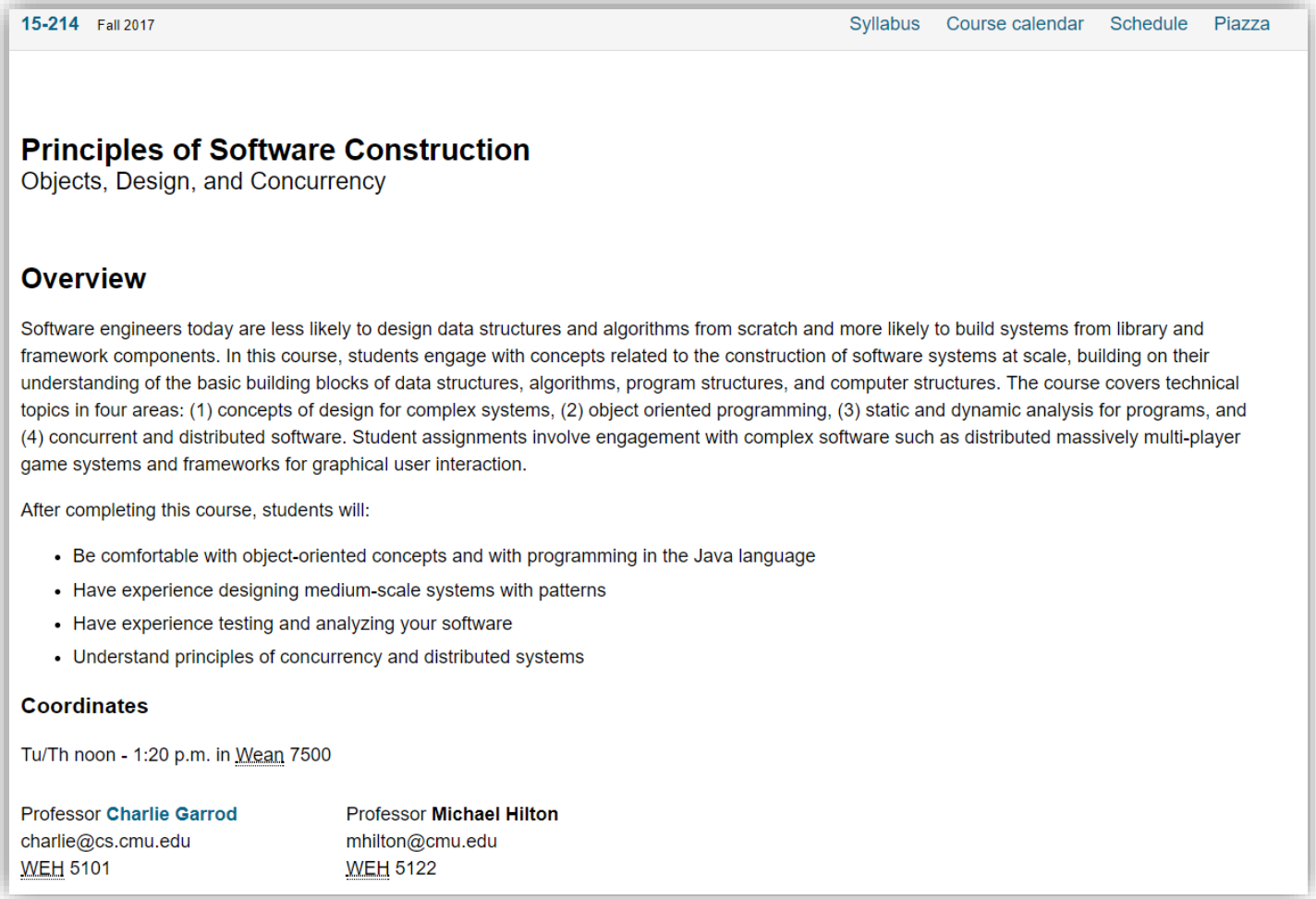
Readings

01: Static Checking
02: Basic Java
03: Testing
04: Code Review
05: Version Control
06: Specifications
07: Designing Specifications
08: Avoiding Debugging
09: Mutability & Immutability
10: Recursion
11: Debugging
12: Abstract Data Types
13: Abstraction Functions & Rep Invariants
14: Interfaces & Enumerations
15: Equality
16: Recursive Data Types
17: Regular Expressions & Grammars
18: Parsers
19: Concurrency
20: Thread Safety
21: Locks & Synchronization
22: Queues & Message-Passing
23: Team Version Control I
24: Sockets & Networking
25: Callbacks
26: Map, Filter, Reduce
27: Little Languages I
28: Little Languages II

推荐学习资料

http://www.cs.cmu.edu/~charlie/courses/15-214/2018-fall

- **CMU 15-214 Principles of Software Construction: Objects, Design, and Concurrency**



15-214 Fall 2017 Syllabus Course calendar Schedule Piazza

Principles of Software Construction
Objects, Design, and Concurrency

Overview

Software engineers today are less likely to design data structures and algorithms from scratch and more likely to build systems from library and framework components. In this course, students engage with concepts related to the construction of software systems at scale, building on their understanding of the basic building blocks of data structures, algorithms, program structures, and computer structures. The course covers technical topics in four areas: (1) concepts of design for complex systems, (2) object oriented programming, (3) static and dynamic analysis for programs, and (4) concurrent and distributed software. Student assignments involve engagement with complex software such as distributed massively multi-player game systems and frameworks for graphical user interaction.

After completing this course, students will:

- Be comfortable with object-oriented concepts and with programming in the Java language
- Have experience designing medium-scale systems with patterns
- Have experience testing and analyzing your software
- Understand principles of concurrency and distributed systems

Coordinates

Tu/Th noon - 1:20 p.m. in Wean 7500

Professor **Charlie Garrod**
charlie@cs.cmu.edu
WEH 5101

Professor **Michael Hilton**
mhilton@cmu.edu
WEH 5122

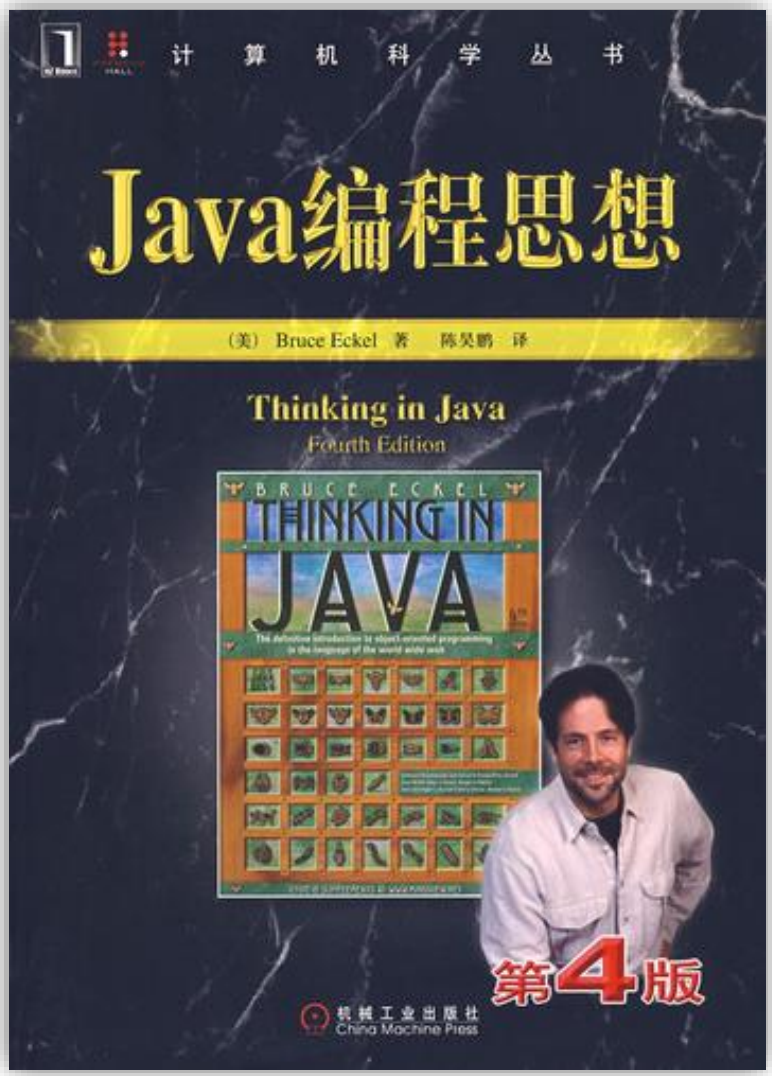
推荐阅读材料

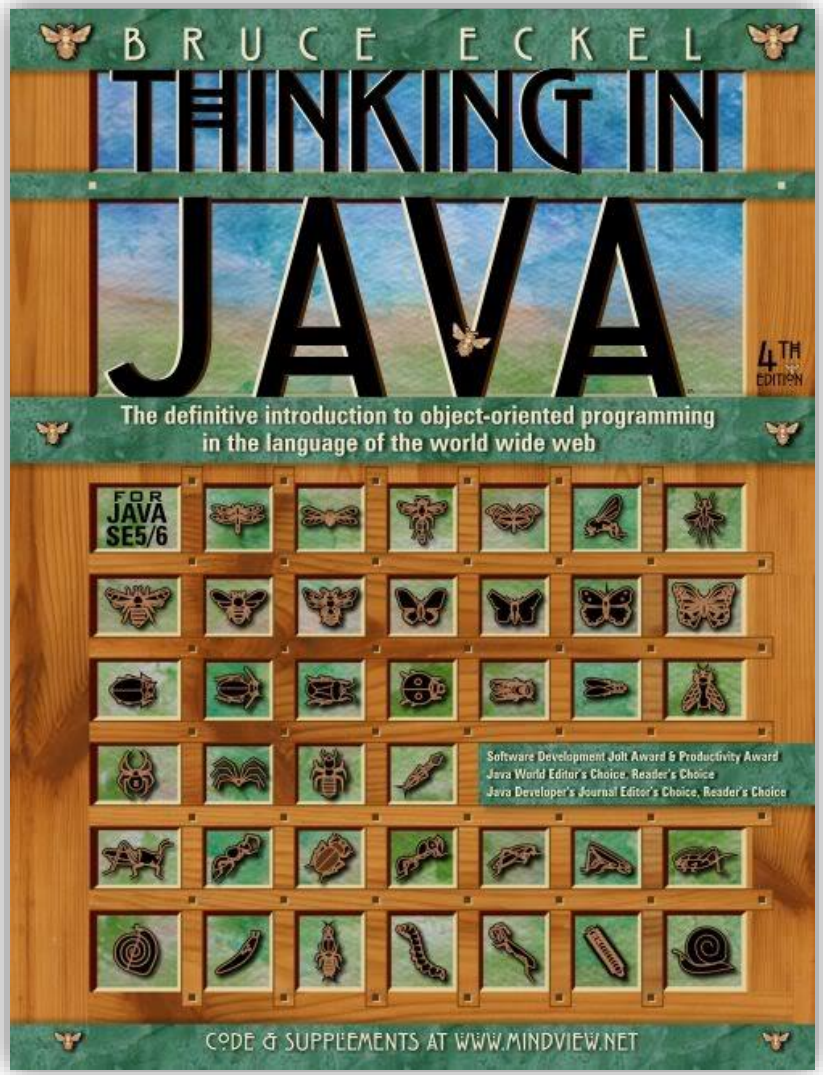
- B. Eckel. Java编程思想 (Thinking in Java), 机械工业出版社, 2016. — 全部
- J. Bloch. Effective Java 中文版, 机械工业出版社, 2009. — 全部
- GoF. 设计模式：可复用面向对象软件的基础 (Elements of Reusable Object-Oriented Software), 机械工业出版社, 2017. — 第5-6-8章
- S. McConnell. 代码大全 (Code Complete), 电子工业出版社, 2006. — 全部
- R. Martin. 代码整洁之道 (Clean Code: A Handbook of Agile Software Craftsmanship), 人民邮电出版社, 2010. — 第4-7-9章
- J. Shirazi. Java Performance Tuning, 2nd Edition, O'Reilly, 2003. — 第8章
- M. Fowler, et al. 重构：改善既有代码的设计 (Refactoring: Improving the Design of Existing Code), 人民邮电出版社, 2010. — 第9章

推荐阅读材料

- R. Pressman. 软件工程--实践者的研究方法 (Software Engineering: A Practitioner's Approach, 7th edition), 机械工业出版社, 2011. 第1-2-7章
- P. Butcher. 软件调试修炼之道 (Debug It: Find, Repair, and Prevent Bugs in Your Code), 人民邮电出版社, 2011. 第7章
- S. Oaks. Java性能权威指南 (Java Performance: The Definitive Guide). 中国工信出版集团, 2017. 第8章
- P. Smith. 深入理解软件构造系统: 原理与最佳实践 (Software Build Systems: Principles and Experiences). 机械工业出版社, 2012. 第2章
- B. Goetz. Java Concurrency in Practice, Addison-Wesley, 2006. 第10章
- A. Oram, G. Wilson. 代码之美 (Beautiful Code), 机械工业出版社, 2009. 课外

If your prefer to have a textbook





Grading policy

- **平时成绩：5%**
  - 阅读相关书籍和论文，思考教师提出的问题，参与课堂测试和交流讨论；
  - 针对教师提出的讨论问题，课后阅读材料，或对实验进展过程遇到的问题和经验教训进行总结思考，以文字形式发表自己的见解，以网上公开博客的形式发表；
- **实验：35%**
  - 共6个，均为个人完成；
  - 现场检查、提交实验报告/实验代码至CMS/GitHub；
- **期末考试：60%**
  - 闭卷

About classroom discussion

- 分为两类：
  - 随堂式讨论：讲课过程中，教师根据所讲内容抛出问题，学生阐述观点；
  - 随堂测试：讲课过程中，教师发布一些问题，学生在手机上作答并提交。
- “翻转课堂”：课堂授课时间较少，学生需要提前阅读教师指定的教材、讲义、论文等，课堂上就其中某些问题进行研讨；
- 根据课堂上提出的讨论问题，事后形成系统化的思考，**撰写个人博客**。
- （分享博客地址）

Using “Rain Classroom” for in-class tests

- 学生带手机到教室，微信里关注“雨课堂”公众号或打开“雨课堂”小程序；
- 扫描二维码加入“软件构造”课程；
- 上课过程中，教师在讲完某些知识点后会通过雨课堂发布小测验，以了解学生的学习效果。

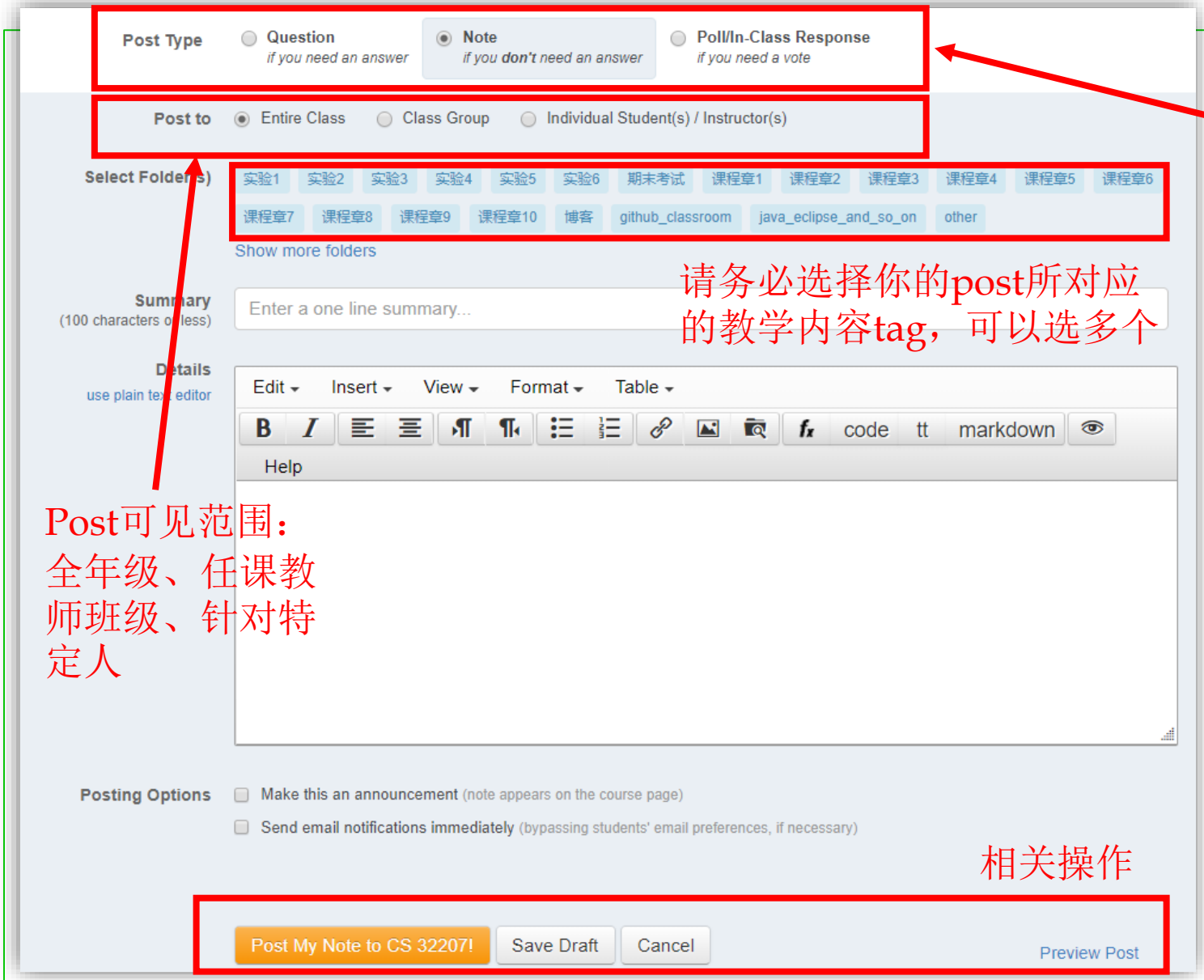
Using Piazza for after-class Q&A and discussion

- 到piazza.com注册账号；
- 访问以下地址加入课程：

  http://piazza.com/harbin_institute_of_technology/spring2019/cs32207
- **Class access code: cs32207**
- 根据自己所在班级的任课教师选择group

- 关于各章节内容、各实验内容、个人博客、期末考试等的任何问题，请在Piazza提出。
- 欢迎其他学生回答技术类问题。

Using Piazza for after-class Q&A and discussion



三种类型Post：

- 提问
- 笔记（分享经验体会）
- 投票（获取其他人的看法）

请务必选择你的post所对应的教学内容tag，可以选多个

Post可见范围：全年级、任课教师班级、针对特定人

相关操作

About labs

- 共6个实验，均为单人完成；
- 28学时实验课，课上+课后完成；
- 按照提交时间、代码/模型的质量、实验报告的质量、TA现场验收（可选）的质量进行打分；
- 6次成绩加权求和，得到总成绩。
- **Deadline: 各截止周的周日夜间23:55 (CMS+GitHub)**

序号	实验内容	覆盖章节	实验课时间	提交截止日期	分数
1	Java编程基础/测试/构建基础	第1/2/7章	1-2周	第3周	10%
2	ADT/OOP	第3章	3-5周	第6周	20%
3	代码可维护性/扩展/复用	第5/6章	6-8周	第10周	30%
4	健壮性/调试	第7章	9、11周	第12周	12%
5	代码静态/动态评审与优化	第4/8/9章	12-13周	第14周	12%
6	多线程编程	第10章	14-15周	第16周	16%

About labs

- **在Java+Eclipse+Git环境下进行，通过GitHub Classroom提交：**
  - https://classroom.github.com，具体参见实验指导手册Lab Guidelines
  - 实验提前2周开放，学生可提前准备好相应的开发环境，熟悉开发任务，实验课上以开发+Q&A为主；
  - 代码只需要提交至GitHub仓库，请确保完整的开发历史都在仓库里，而不仅仅只是最终结果。
  - 实验报告(Word或PDF格式)需同时提交至CMS作业下、GitHub仓库的doc目录下；
  - 不接收通过Email、微信等其他方式的提交，TA无权私下接收实验结果；
  - 进行抄袭检测，若有抄袭出现，双方均无成绩。
- **TA在实验课上抽查学生代码，学生口头阐述实验思路(代码、实验步骤、实验数据等)，若无法解释清楚，视为抄袭他人；**
- **TA课后阅读学生提交的实验报告，结合现场抽查结果、自动测试、人工评判代码，进行打分。**

Late day policy for labs

- **请尽可能在deadline之前提交。但是——**
  - 每人共有5天的free late days，以“天”为单位计算（迟交1秒和24小时均耗费1个free late day）；
  - 如果free late days已用完，后续实验每延迟1天，分数扣除30%；
  - 不管有没有剩余free late days，每次实验最多延迟2天，意即：超过deadline 48小时，分数=0。
- **以下是一个例子：**

实验	Deadline	实际提交日期	剩余free late days	分数计算
Lab1	3月17日 23:55	3月17日 23:56	5-1 (4)	100%
Lab2	4月7日 23:55	4月9日 23:54	4-2 (2)	100%
Lab3	5月5日 23:55	5月7日 23:56	(2)	0 （迟交3天）
Lab4	5月19日 23:55	5月21日 23:54	2-2 (0)	100%
Lab5	6月2日 23:55	6月2日 23:56	(0)	70%（迟交1天）
Lab6	6月16日 23:55	6月18日 23:54	(0)	40%（迟交2天）

Don't get puzzled by CMS

作业: Lab 1: Fundamental Java Programming and Testing
截止时间: 2019年03月19日 星期二 23:55

作业: Lab 2: ADT and OOP
截止时间: 2019年04月09日 星期二 23:55

作业: Lab 3: Reusability and Maintainability Oriented Pr
截止时间: 2019年05月07日 星期二 23:55

作业: Lab 4: Debugging, Exception Handling, and Defensive Program
截止时间: 2019年05月21日 星期二 23:55

作业: Lab 5: Static and Dynamic Code Analysis and Code Optimization
截止时间: 2019年06月04日 星期二 23:55

作业: Lab 6: Multi-Thread Concurrent Programming
截止时间: 2019年06月18日 星期二 23:55

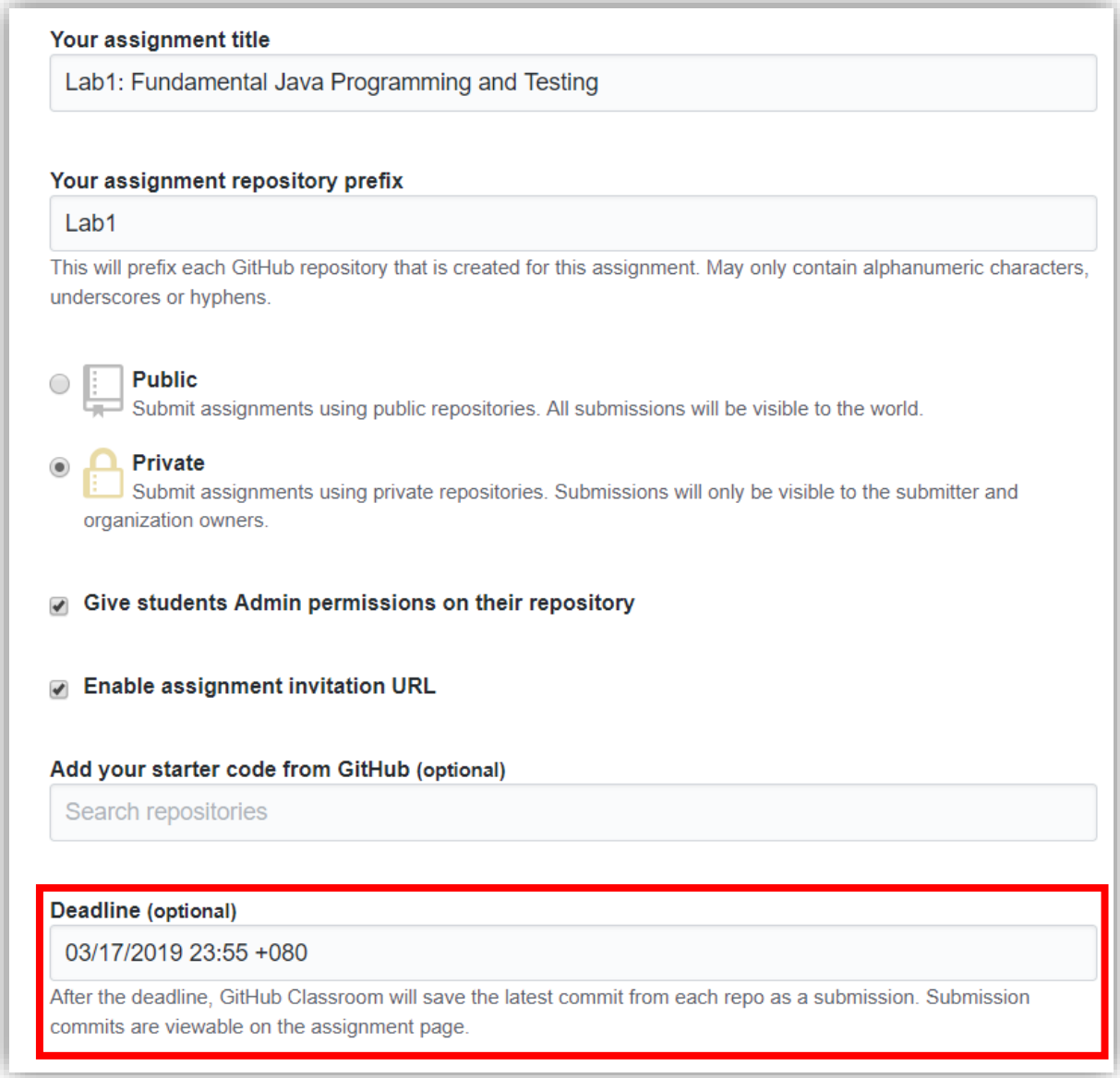
Don't follow this deadline: they're two days late considering you have unused late days !!!

Submit your homework before 23:55 Sunday night as far as possible !!

Pay attention to the system clock of CMS !!

GitHub Classroom

- 在GitHub Classroom里，作业deadline都设定为周日晚上23:55。
- 超时仍可继续向GitHub仓库commit，但Classroom自动将deadline前的最后一次commit作为作业结果交付给教师/TA。
- 如果你使用了free late days，请在提交实验报告至CMS的时候同时提交你的完整项目代码(压缩为一个文件)至CMS (!!!)



关于实验

- 在Java+Eclipse+Git环境下进行；
- 若使用其他编程语言和编程环境，无法得到TA的有效指导，但实验要求必须全部完成，故需要自己搭建相应的开发环境与工具；
- 实验要求提前2周开放，学生可提前准备好相应的开发环境，实验课上以开发+Q&A为主；
- 代码与实验报告需在截止日期前提交至CMS/GitHub，延期不接收通过Email等其他方式的提交；
- 进行抄袭检测，若有抄袭出现，双方均无成绩；
- TA课后阅读学生提交的实验报告，结合现场抽查结果、自动测试报告进行打分；
- 每次实验满分100，6次实验分数加权平均的35%计入最终成绩。

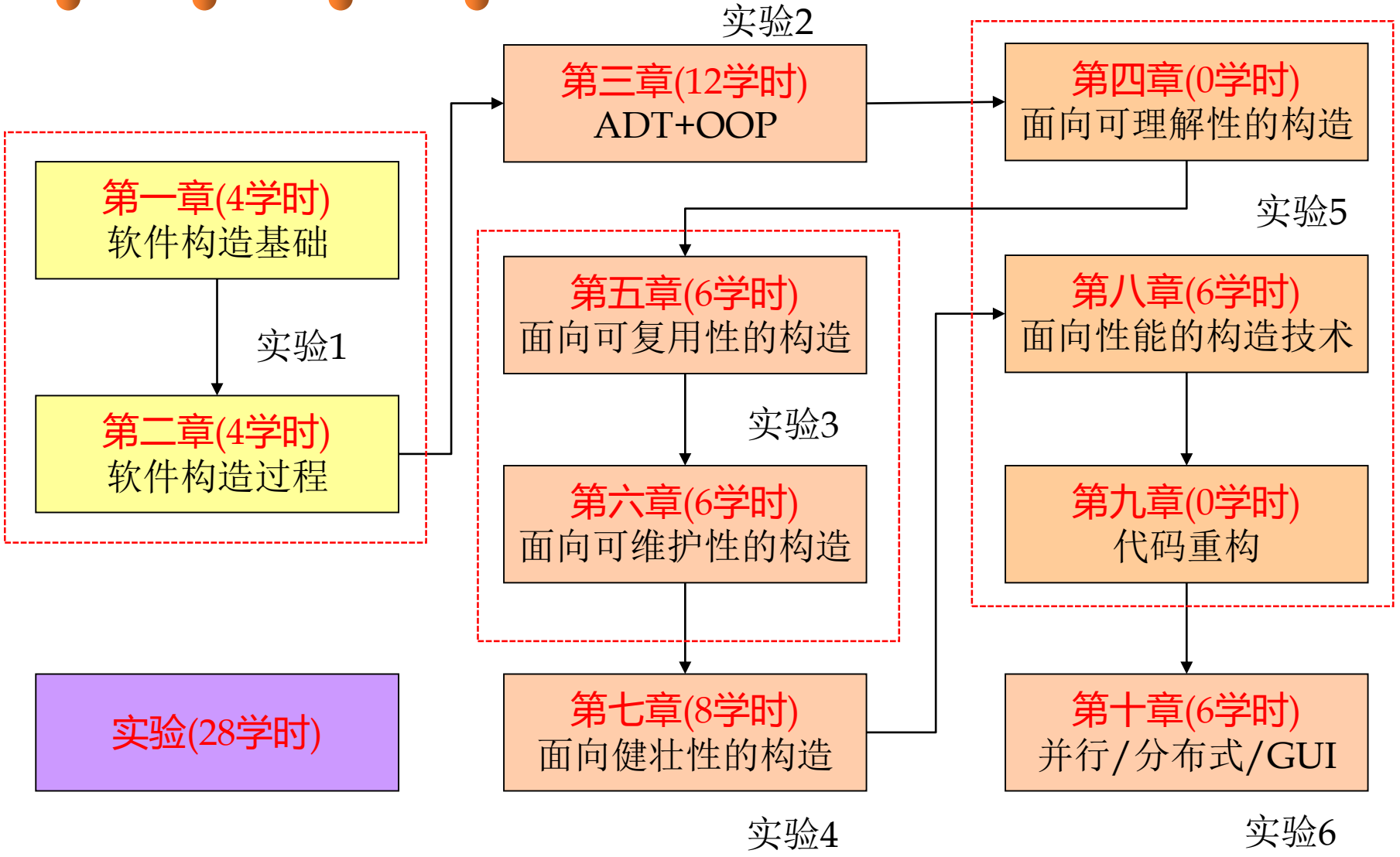
A summary of tools used in this course

- **CMS：**获取讲义、实验要求；提交实验报告（用于教务处备份）和代码（仅针对deadline之后的提交）
- **GitHub Classroom：**获取实验链接，创建私人Git仓库，提交代码至仓库；TA从各人的Git仓库获取deadline之前的最后一次commit进行评价打分
- **Piazza：**与本年级其他班级学生构成大的Q&A论坛，教师、TA、学生均在其中
- **雨课堂：**微信公众号/小程序；教师讲课期间发布随堂测试题，学生现场做答，便于教师了解学习情况
- **微信群：**日常交流

关于TA

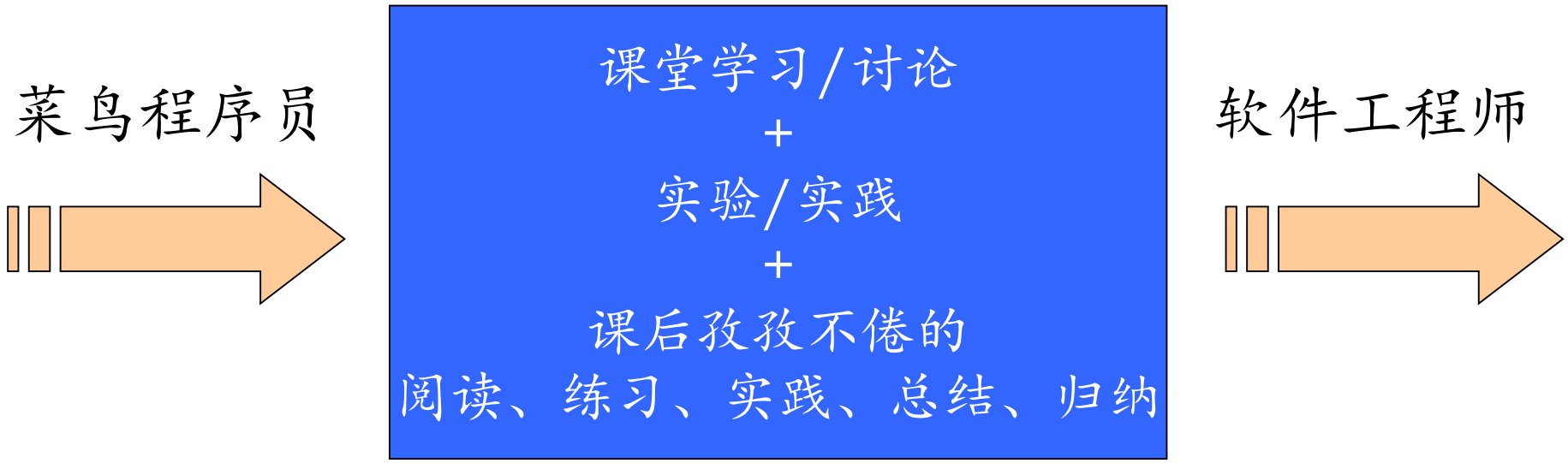
- **三名TA（研一、大三）**
  - 实验课上指导实验
  - 回答编程方面的问题

课程章节安排


第一章(4学时)
软件构造基础
实验1
第二章(4学时)
软件构造过程
实验(28学时)
实验2
第三章(12学时)
ADT+OOP
第五章(6学时)
面向可复用性的构造
实验3
第六章(6学时)
面向可维护性的构造
第七章(8学时)
面向健壮性的构造
实验4
第四章(0学时)
面向可理解性的构造
实验5
第八章(6学时)
面向性能的构造技术
第九章(0学时)
代码重构
第十章(6学时)
并行/分布式/GUI
实验6

总结属于自己的“最佳实践”

- 多动手、多实践，方可成为合格的“程序员”；
- 实践越多、写的代码越多、参与的项目越大，积累经验越多；
- 首先遵循他人提出的“最佳实践”，进而创造自己的“最佳实践”；
- 从“菜鸟程序员”成长为“软件工程师”。



如何学习该课程

- **时刻关注课程日历，了解课程的整体进度安排，尤其是各实验的上课时间、现场检查时间、提交时间；**
  - 建议：提前搭建好实验环境，学习实验所用的工具，提前开始实验，实验课上用于与TA的交流，答疑解惑，并接受验收。
  - 单纯使用2-3学时的实验课无法完成实验。
- **提前阅读下一次课程的待讲授内容，阅读教材相关章节，进行预习；**
  - “需要我学习的知识，老师一定会在课堂上去讲”
  - 课堂上讲思想和难点，仅靠听课无法获得全部的考试点
  - 需要阅读大量的辅助教材
- **对下一次课要讲的内容，提前阅读资料做好准备。**

The end

March 16, 2017