HARBIN INSTITUTE OF TECHNOLOGY

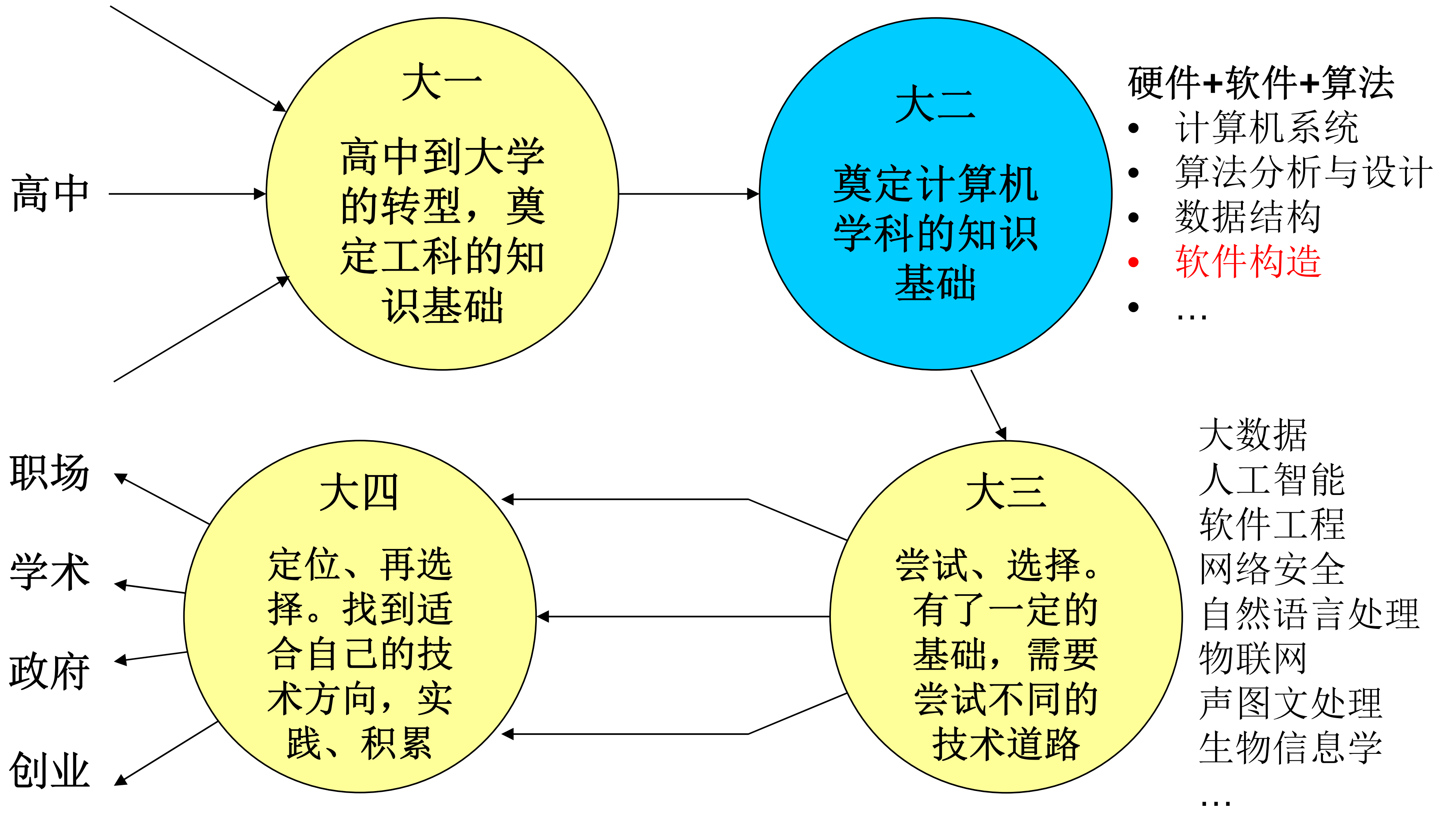
# Software Construction: Developing High-Quality Software Systems

# 软件构造：开发高质量的软件系统

Wang Zhongjie
rainy@hit.edu.cn

May 10, 2021

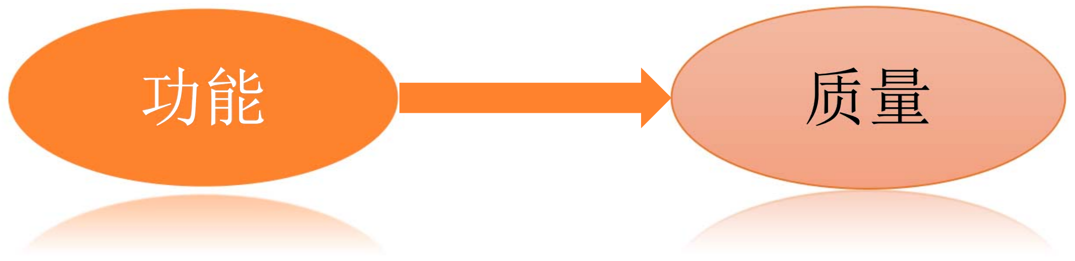
## 打好基础，为“选择”做准备


高中
大一
高中到大学的转型，奠定工科的知识基础
大二
奠定计算机学科的知识基础
硬件+软件+算法
• 计算机系统
• 算法分析与设计
• 数据结构
• 软件构造
• …
大三
尝试、选择。有了一定的基础，需要尝试不同的技术道路
大数据
人工智能
软件工程
网络安全
自然语言处理
物联网
声图文处理
生物信息学
…
大四
定位、再选择。找到适合自己的技术方向，实践、积累
职场
学术
政府
创业

## Goals of this Course

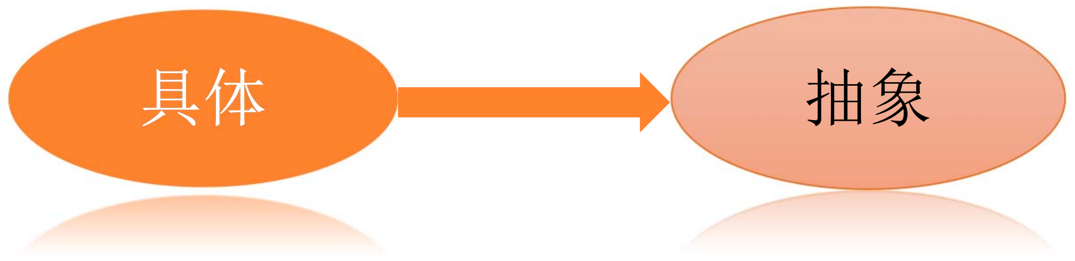
- **Goal: understanding both the building blocks and the design principles for construction of software systems** 构造原理？
  - 在高级语言程序设计的基础上，认识软件构造的质量标准与目标，学习软件构造的基本过程，从而具备面向质量目标的复杂软件构造方法与能力
  - 深入学习抽象数据类型 ADT 与面向对象编程 OOP
  - 初步掌握面向关键质量目标（可理解性、可维护性、可复用性、健壮性、时空性能）的软件构造基本技术
  - 了解软件代码重构和面向更复杂软件系统的高级构造技术
- **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**

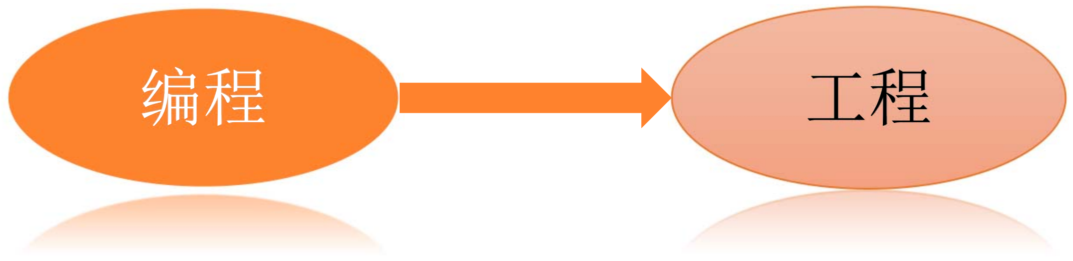
- 了解软件开发过程中应考虑哪些质量目标
- 掌握面向关键质量目标的软件基本构造技术
- 形成面向质量目标的软件开发思维模式



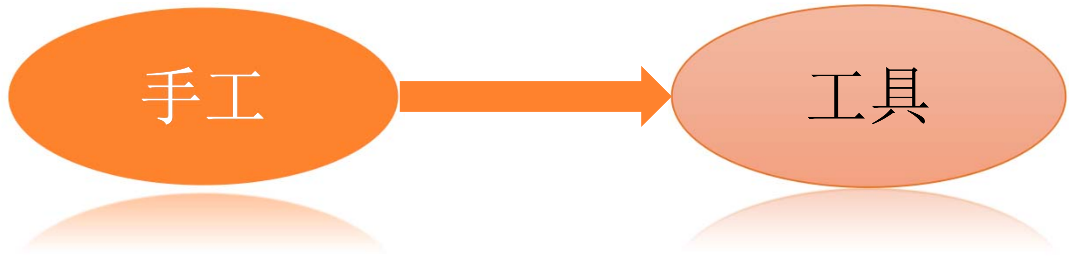
**系统设计与实现能力**

- 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

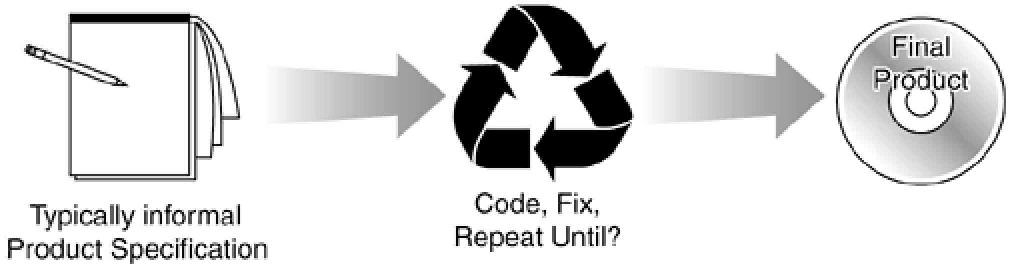


**利用现代软件构造工具的能力**
- 了解复杂软件系统相对于简单程序的本质差异
- 初步掌握利用各类软件开发工具进行编码、测试和质量保障
- **利用现代软件构造工具**进行高质量和高效率软件开发

## 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

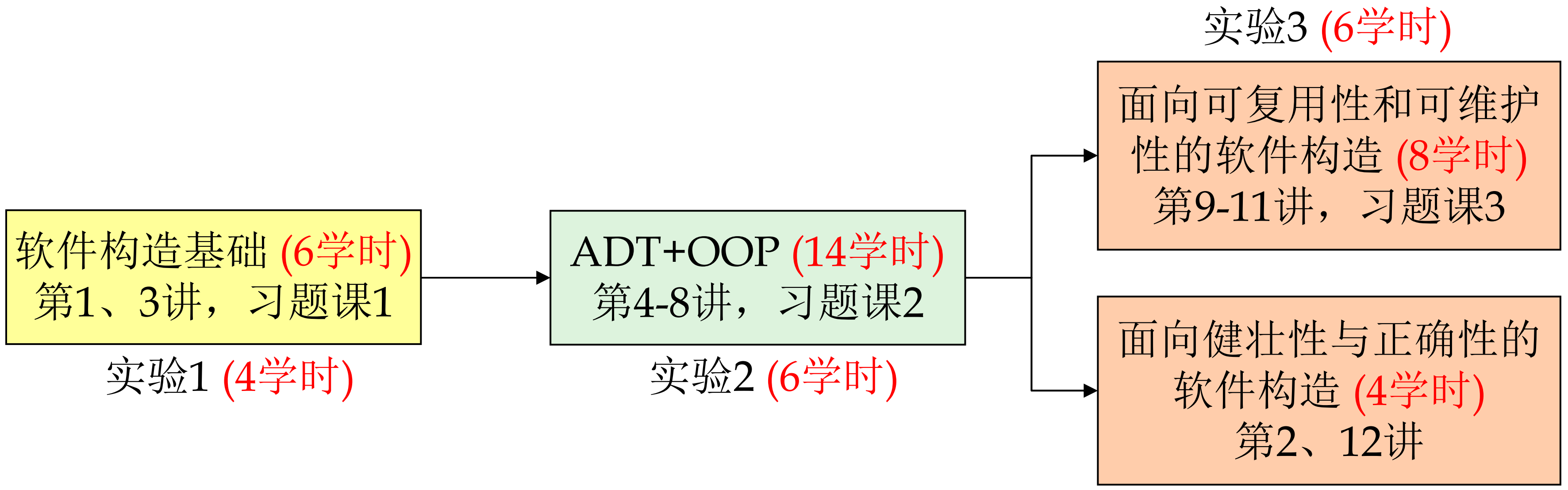
- **Design goals** enable evaluation of designs
  - e.g. maintainability, reusability, scalability
- **Design principles** are heuristics that describe best practices
  - e.g. high correspondence to real-world concepts
- **Design patterns** codify repeated experiences, common solutions
  - e.g. template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

## Chapters and hours of the course

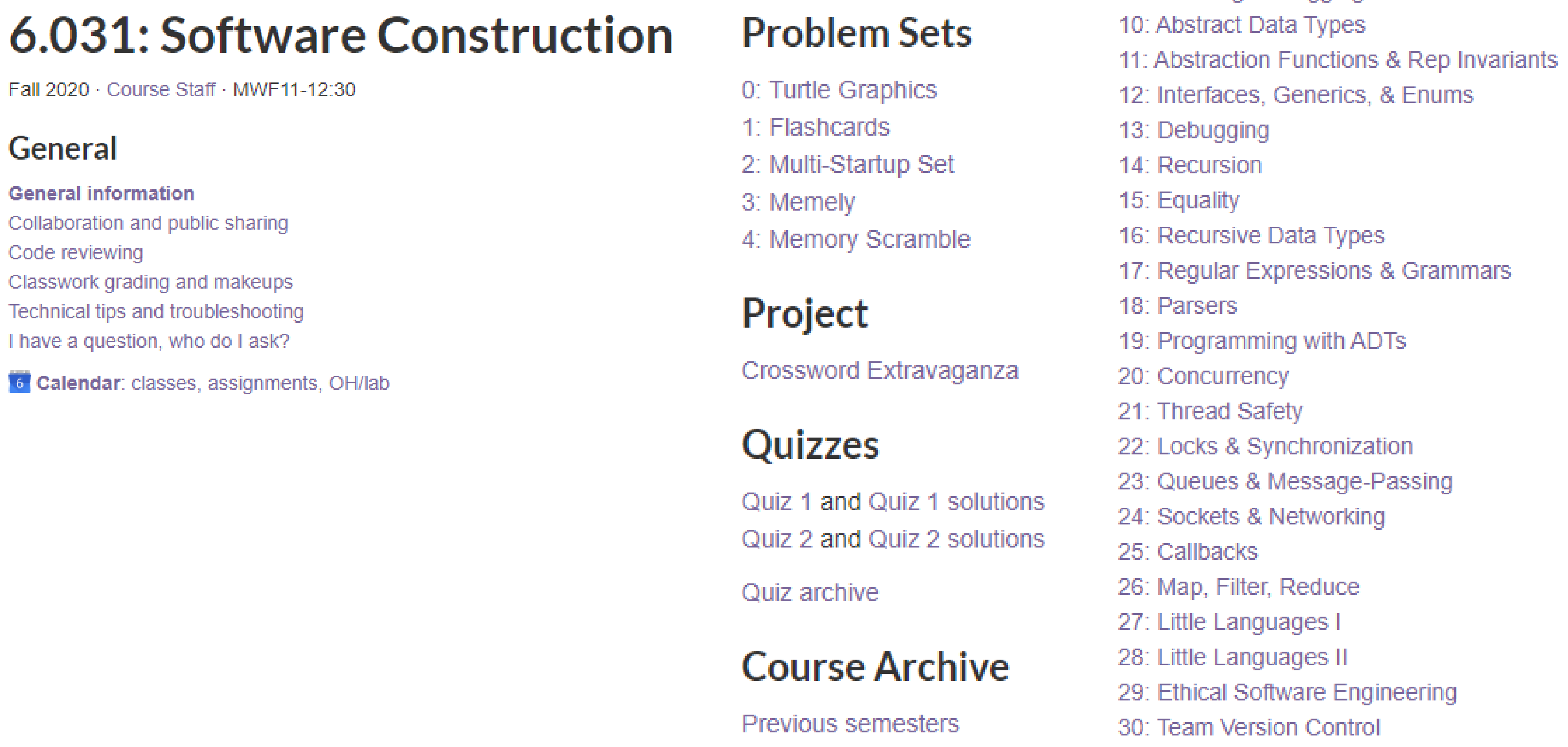


## Basic information

- 授课对象： 计算机学院2019级本科生 (英才)
- 课程分类： 核心基础课
- 学时： 48 (32+16)
- 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：
  - 9-16周 周一1-2节/周三1-2节 正心楼13
  - 10-17周 周二7-8节 格物楼机房207/213
- 考试时间：
  - 18周 待定

## Reading materials (mandatory)

- **MIT Course 6.031: Software Construction**
  [http://web.mit.edu/6.031/www/fa20/](http://web.mit.edu/6.031/www/fa20/)



### 6.031: Software Construction

Fall 2020 · Course Staff · MWF11-12:30

#### General

**General information**
Collaboration and public sharing
Code reviewing
Classwork grading and makeups
Technical tips and troubleshooting
I have a question, who do I ask?

**Calendar**: classes, assignments, OH/lab

#### Problem Sets

0: Turtle Graphics
1: Flashcards
2: Multi-Startup Set
3: Memely
4: Memory Scramble

#### Project

Crossword Extravaganza

#### Quizzes

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

Quiz archive

#### Course Archive

Previous semesters

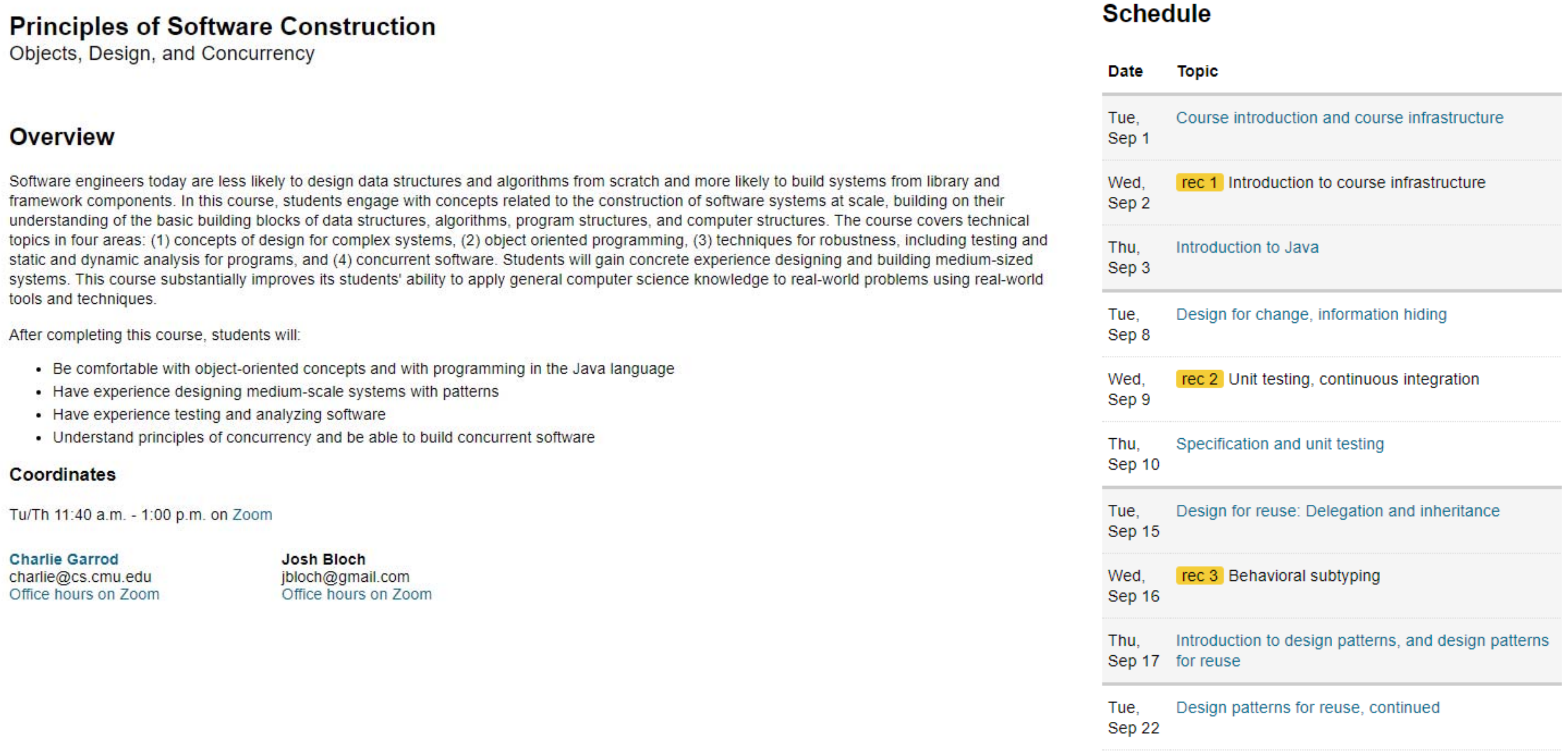
#### Readings

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

## Reading materials (highly recommended)

https://www.cs.cmu.edu/~charlie/courses/17-214/2020-fall/

- **CMU 17-214 Principles of Software Construction: Objects, Design, and Concurrency**



### 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) techniques for robustness, including testing and static and dynamic analysis for programs, and (4) concurrent software. Students will gain concrete experience designing and building medium-sized systems. This course substantially improves its students' ability to apply general computer science knowledge to real-world problems using real-world tools and techniques.

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 software
- Understand principles of concurrency and be able to build concurrent software

### Coordinates

Tu/Th 11:40 a.m. - 1:00 p.m. on Zoom

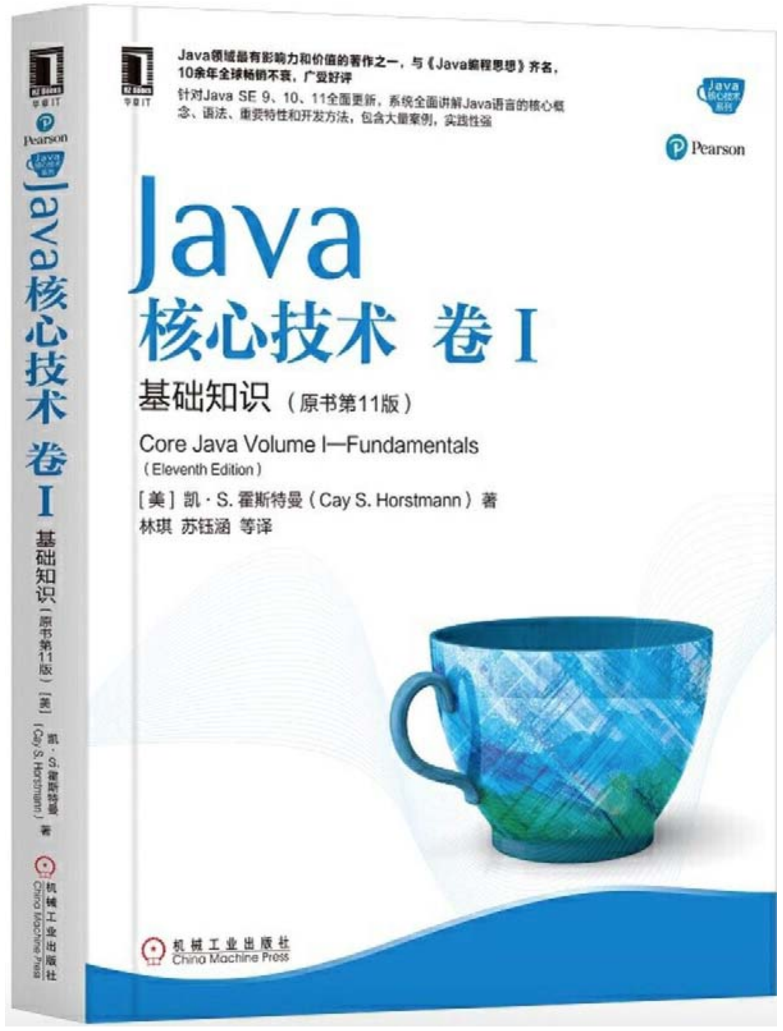
**Charlie Garrod**
charlie@cs.cmu.edu
Office hours on Zoom

**Josh Bloch**
jbloch@gmail.com
Office hours on Zoom

### Schedule

| Date | Topic |
|---|---|
| Tue, Sep 1 | Course introduction and course infrastructure |
| Wed, Sep 2 | rec 1 Introduction to course infrastructure |
| Thu, Sep 3 | Introduction to Java |
| Tue, Sep 8 | Design for change, information hiding |
| Wed, Sep 9 | rec 2 Unit testing, continuous integration |
| Thu, Sep 10 | Specification and unit testing |
| Tue, Sep 15 | Design for reuse: Delegation and inheritance |
| Wed, Sep 16 | rec 3 Behavioral subtyping |
| Thu, Sep 17 | Introduction to design patterns, and design patterns for reuse |
| Tue, Sep 22 | Design patterns for reuse, continued |

## If your prefer to have a textbook



Java领域最有影响力和价值的著作之一，与《Java编程思想》齐名，10余年全球畅销不衰，广受好评

针对Java SE 9、10、11全面更新，系统全面讲解Java语言的核心概念、语法、重要特性和开发方法，包含大量案例，实践性强

Pearson

Java 核心技术 卷 I

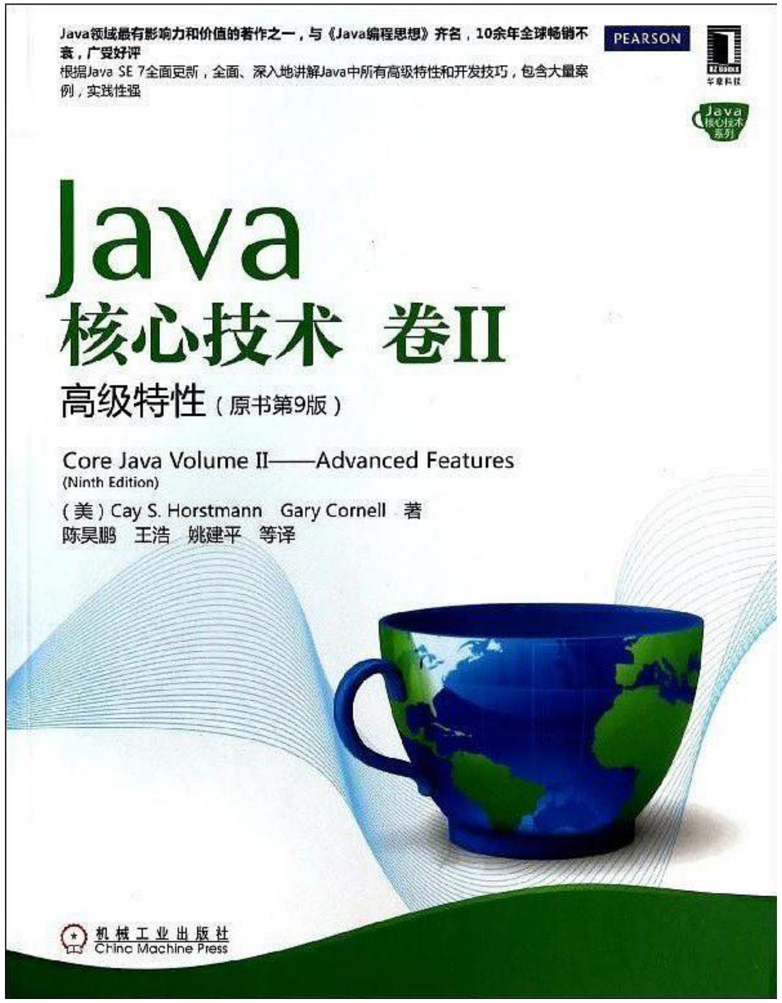
基础知识（原书第11版）

Core Java Volume I—Fundamentals (Eleventh Edition)

[美] 凯·S. 霍斯特曼（Cay S. Horstmann）著

林琪 苏钰涵 等译

机械工业出版社 China Machine Press



Java领域最有影响力和价值的著作之一，与《Java编程思想》齐名，10余年全球畅销不衰，广受好评

根据Java SE 7全面更新，全面、深入地讲解Java中所有高级特性和开发技巧，包含大量案例，实践性强

PEARSON

Java 核心技术 卷II

高级特性（原书第9版）

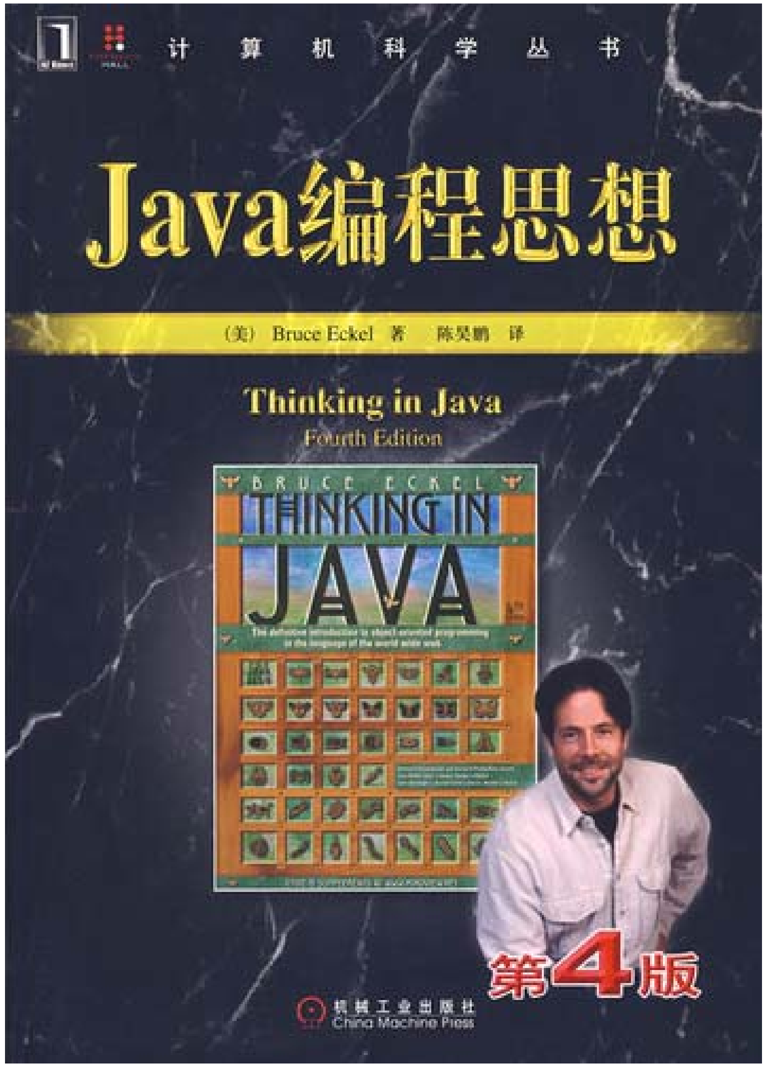
Core Java Volume II——Advanced Features (Ninth Edition)

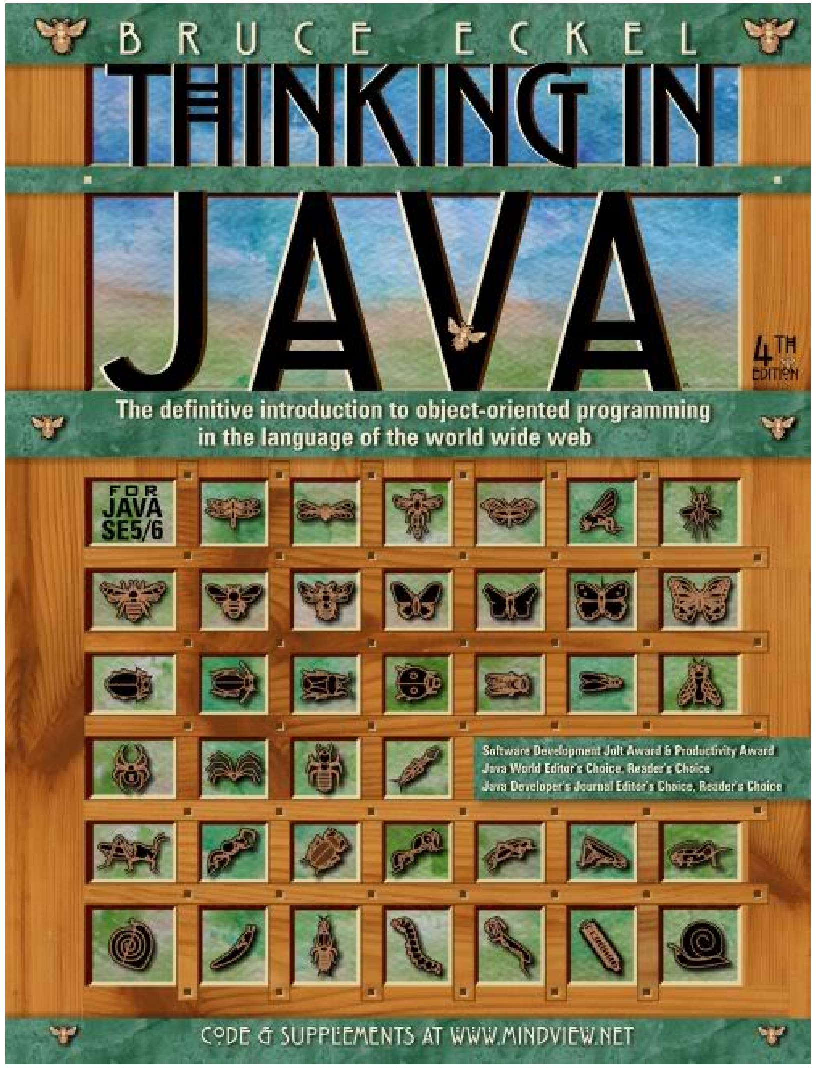
（美）Cay S. Horstmann Gary Cornell 著

陈昊鹏 王浩 姚建平 等译

机械工业出版社 China Machine Press

## Or


计 算 机 科 学 丛 书
Java编程思想
(美) Bruce Eckel 著 陈昊鹏 译
Thinking in Java
Fourth Edition
第4版
机械工业出版社
China Machine Press


BRUCE ECKEL
THINKING IN JAVA
4TH EDITION
The definitive introduction to object-oriented programming in the language of the world wide web
FOR JAVA SE5/6
Software Development Jolt Award & Productivity Award
Java World Editor's Choice, Reader's Choice
Java Developer's Journal Editor's Choice, Reader's Choice
CODE & SUPPLEMENTS AT WWW.MINDVIEW.NET

## Grading policy

- **实验：35%**
  - 共3个，均为个人完成；
  - 提交实验报告/实验代码至GitHub；
- **个人博客：5%**
  - 针对教师提出的讨论问题进行课后阅读，或对实验进展过程遇到的问题和经验教训进行总结思考，以网上公开博客的形式发表见解；
- **期末考试：60%**
  - 闭卷

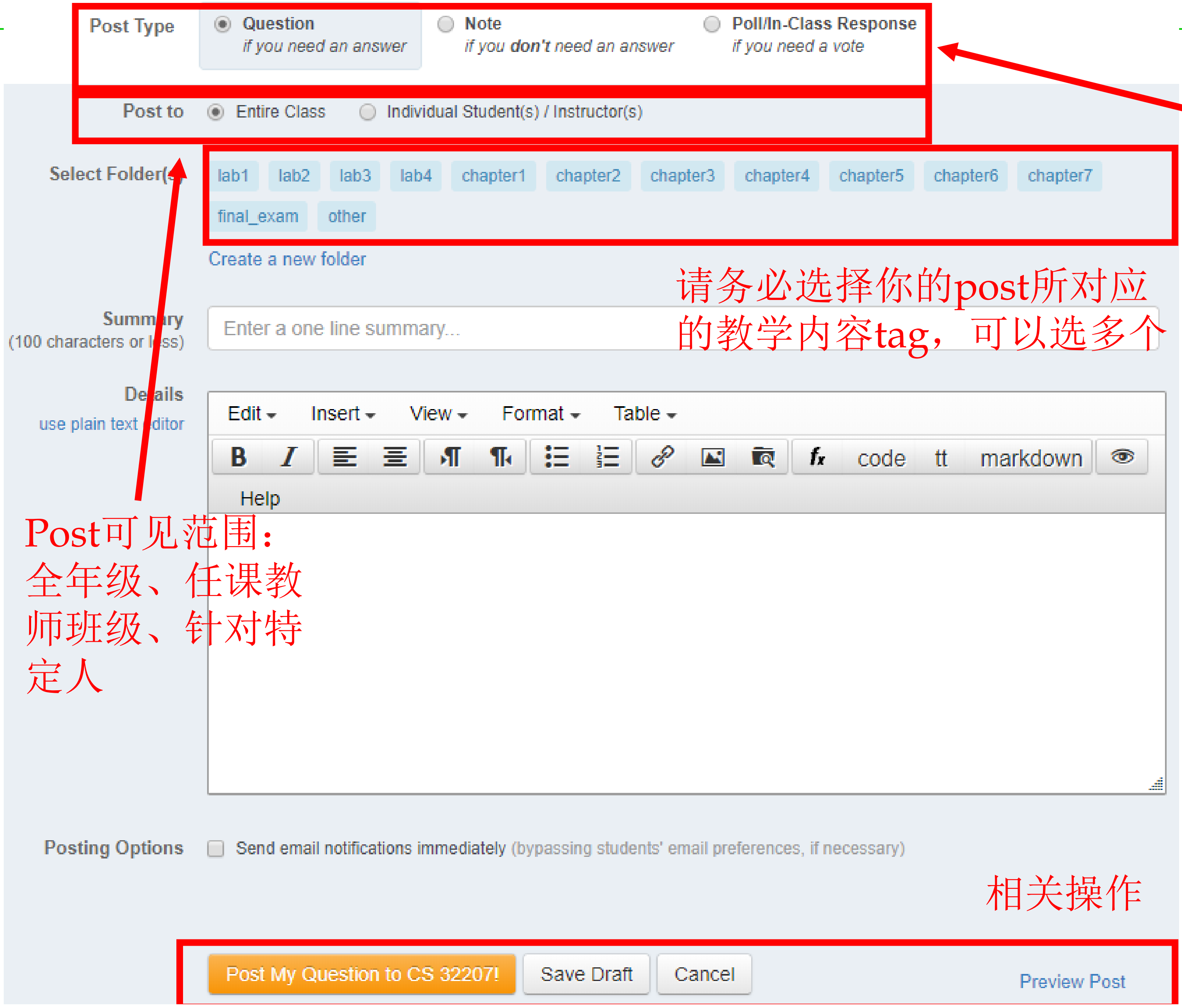
## Using Piazza for after-class Q&A and discussion

- 到piazza.com注册账号，注册后请实名，**TA**不回答非实名的问题；
- 访问以下地址加入课程：

  http://piazza.com/harbin_institute_of_technology/spring2021/cs32207
- **Class access code: cs32207**

- 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



Post Type [✓ Question if you need an answer] Note if you don't need an answer / Poll/In-Class Response if you need a vote

Post to [✓ Entire Class] Individual Student(s) / Instructor(s)

Select Folder(s): lab1 lab2 lab3 lab4 chapter1 chapter2 chapter3 chapter4 chapter5 chapter6 chapter7 final_exam other

Create a new folder

Summary (100 characters or less): Enter a one line summary...

Details — use plain text editor

Posting Options: Send email notifications immediately (bypassing students' email preferences, if necessary)

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 Personal Blog

- **根据课堂上学习的理论方法、教师提出的讨论问题，查阅相关资料，进行系统化的思考；**
- **对实验进展过程遇到的问题和经验教训进行总结思考；**
- **撰写个人博客，公开发表；**
- **评判标准：**
  - 博客数量
  - 与课程内容/实验内容的相关性
  - 博客内容的深度与独特性
  - 个人思考与借鉴网上公开资料的比例
  - 上课期间博客发布的时间均衡程度
- **截止日期：期末考试前一天23:55**

## 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；
- 进入雨课堂后，点击“我的” - “课程”，找到右上角的“我”，点击用户头像编辑个人信息，填写真实姓名、学号、邮箱地址，否则影响成绩。
- 你已经通过身份绑定加入了“**2020春-软件构造2019-2020-2-CS32207-001**”，如果没有，现在马上就开始绑定。
- 课程讲义通过雨课堂发布。疫情期间，通过微信群发布；
- 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。

## About labs

- 共3个实验，均为单人完成；
- 16学时实验课，课上+课后完成；
- 按照提交时间、代码/模型的质量、实验报告的质量进行打分；
- 3次成绩加权平均，得到总成绩。
- **Deadline: 各截止周的周日夜间23:55 (GitHub)**

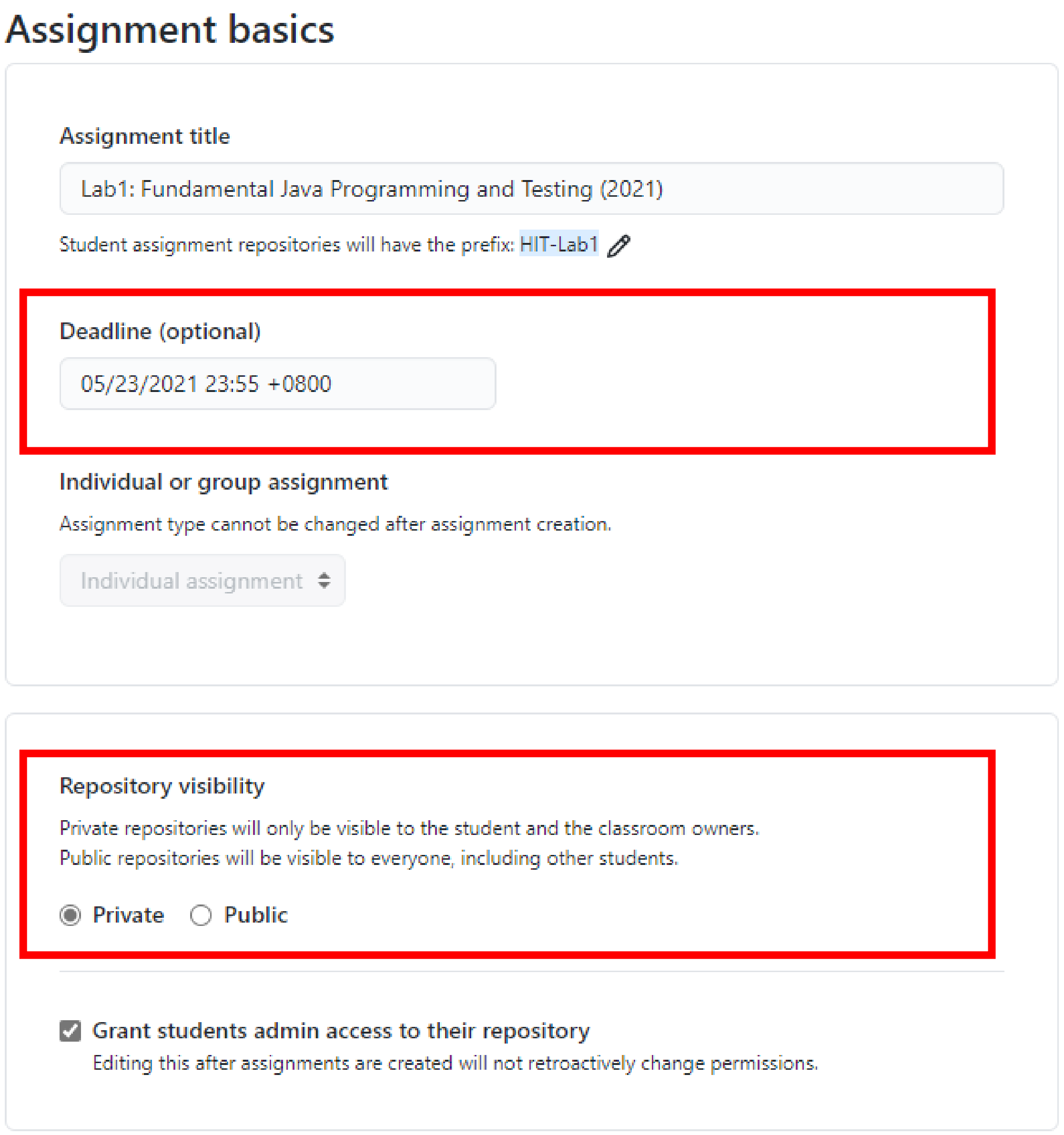
| 序号 | 实验内容 | 实验课时间 | 提交截止日期 | 分数 |
|---|---|---|---|---|
| 1 | Java编程与测试基础 | 10-11周 | 第11周 | 20% |
| 2 | ADT设计与OOP实战 | 12-14周 | 第14周 | 40% |
| 3 | 面向复用和可维护性的构造 | 15-17周 | 第17周 | 40% |

## 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1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；
  - 代码和实验报告 (Word或PDF格式) 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。
  - 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；
  - 超过截止时间提交的作业，无成绩。
- **TA抽查学生代码，若发现雷同代码，视为抄袭他人；**
- **TA课后阅读学生提交的实验报告，结合自动测试、人工评判代码，进行打分。**

## GitHub Classroom

- **在GitHub Classroom里，作业deadline都设定为周日晚上23:55。**
- **超过截止时间之后仍可继续向GitHub仓库commit，但TA以截止时间前的最后一次commit作为评判的对象。**



Assignment basics

Assignment title

Lab1: Fundamental Java Programming and Testing (2021)

Student assignment repositories will have the prefix: HIT-Lab1

Deadline (optional)

05/23/2021 23:55 +0800

Individual or group assignment

Assignment type cannot be changed after assignment creation.

Individual assignment

Repository visibility

Private repositories will only be visible to the student and the classroom owners.
Public repositories will be visible to everyone, including other students.

[✓ Private] Public

[✓] Grant students admin access to their repository

Editing this after assignments are created will not retroactively change permissions.

## TAs for labs

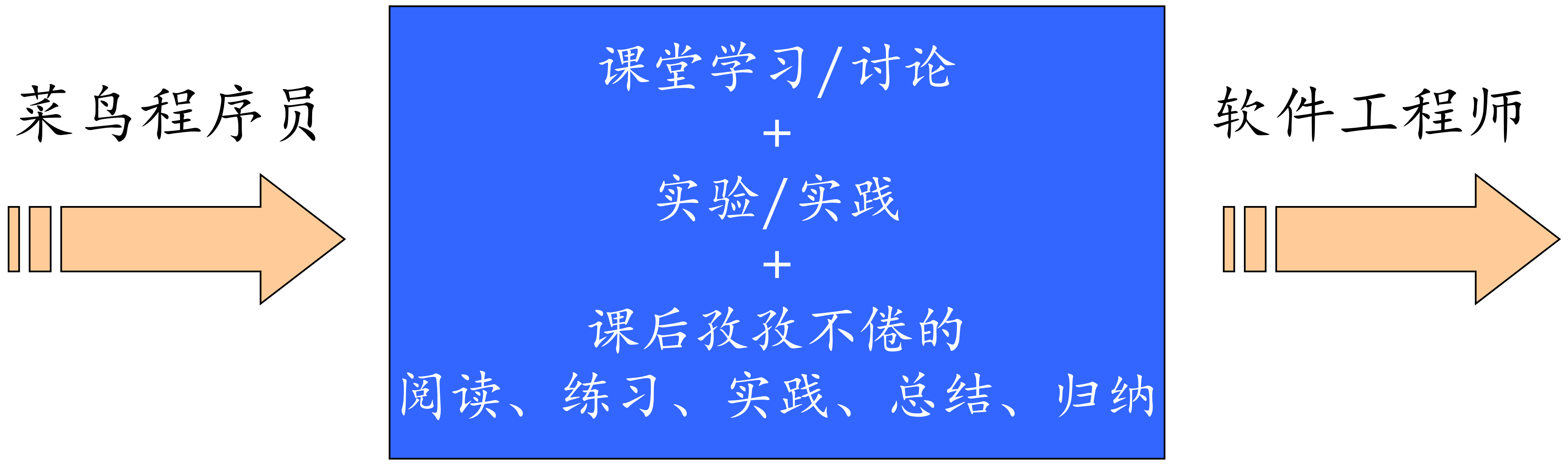
- **TA:**
  - 1
  - 2
  - 3
- 实验课期间，与**TA**面对面交流 **(**疫情期间，教师和**TA**在线答疑**)**
- 其他时间，通过**Piazza**平台进行**Q&A**
- 考虑到**TA**自己具有的学习和研究任务，时间紧张，非实验课时间恕不接待通过微信或**Email**等手段的提问

## A summary of tools used in this course

- **GitHub Classroom：** 获取实验链接，创建私人**Git**仓库，提交代码和实验报告至仓库；**TA**从各人的**Git**仓库获取最后一次**commit**进行评价打分，计算**late days**
- **Piazza：** 与本年级其他班级学生构成**Q&A**论坛，教师、**TA**、学生均可在其中提问和回答
- **微信群：** 日常交流

## To summarize your own best practices

- 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”。



## How to learn and get a high score

- **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；**
  - 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。
  - 单纯使用实验课学时无法完成实验。

课程日历

- **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，老师未必会在课堂上去讲”
  - 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。**

## To be a pragmatic programmer

不断积累自己的代码

持续深入某个技术领域，坚持学习和总结

与其他程序员交流经验和教训

记录你的成绩和经历

# The end

May 10, 2021