CSE 143
Computer Programming II

Welcome!

Course Overview and Administrivia

Pick up a syllabus as you come in

Outline for Today

- Course Overview
- Goals
- Administrative details
- Workload and grading
- Resources
- Background

This information is largely included in the syllabus (handout), and is on the web – no need to transcribe details

Introductions

- Instructor: Hal Perkins
  - cse143-instructor@cs.washington.edu, perkins@cs.washington.edu
  - Allen Center CSE 548, office hours Wed. after class + tba
- TA's: Many
  - cse143-ta@cs.washington.edu (goes to all TAs and the instructor)
- Course administrator: Pim Lustig
  - cse143-admin@cs.washington.edu
  - See Pim for logistical details about the course (enrollments, conflicts, others)
- You!

Are You Ready?

- Course is a direct continuation of CSE 142
- Must have a firm grasp of programming basics
  - including variables, expressions, statements (assignment, conditionals, loops), methods, parameters, arrays
  - Look at old CSE 142 web pages – you should be able to handle those assignments and exams
- What if you took a different version of CSE 142? Or took it elsewhere? Or didn't use Java? Or a long time ago?
  - Sit in on both for a few days if you want
  - Try the first 143 assignment
  - We'll help you switch to 142 if that's your decision

Course Overview

- Basics of data structures and algorithms ("abstract data types")
- Key data structures: stacks, queues, linked lists, binary trees, dictionaries, maps, hashing
  - Mixture of implementation and using/studying off-the-shelf components, particularly classes in the Java Collection Framework
- Programming topics: encapsulation, interfaces, inheritance, recursion, divide-and-conquer, other algorithm design techniques
- Basic complexity theory – comparing algorithms, performance tradeoffs, etc.
- Good software design and effective use of Java

Is This a Java Programming Course or Not?

- This *is* a programming course
  - The key goal is learning to program *well*, not just getting stuff to run
    - Good design, good organization, good style
    - Good algorithms, meaningful efficiency
- This *is not* a programming course
  - Lots of Java features won't be covered
    - See Java reference books & JavaDoc for details about the Java language & libraries
    - We cover the essential parts of Java that support good programming
  - We will cover important computer science topics that are not Java-specific
    - Some related to programming, but broader than Java
    - Data structures, algorithms, complexity analysis, …
- Fact: writing programs that work perfectly isn't enough to get a perfect grade (!)

My Goals for You

- 4 things you should be able to do after CSE143
  - Be able to design and implement abstractions (interfaces and classes) using modern programming language features and techniques
  - Be able to test and systematically locate and remove errors
  - Be able to evaluate tradeoffs between different implementations of an abstraction and pick suitable ones
  - Be able to learn and use new libraries using standard documentation (no training wheels)

My Expectations for You

- Responsibility
  - Keep up, know what's happening
  - Meet deadlines, budget your time, make backups
  - Take responsibility for your own code and debugging
- Respect
  - For others in the class (people sitting around you in lecture, members of your discussion section, …)
  - For the course staff
  - For yourself

My Goals For Myself

- Be an advocate for your learning (credit to Prof. Mary Pat Wenderoth for this notion)
  - Help all of you learn
  - Keep the course on track
  - Make the homework projects interesting
  - Make lectures and sections events you look forward to!
- Keep in touch with what's happening
  - Office hours – please drop by if just to chat
    (you're not being sent to the Principle's office!)
  - Participate in online discussions and other forums, use informal evaluations, etc.

Course Organization

- 3 lectures per week (MWF, 2:30)
- Discussion sections twice per week (various times,T/Th)
  - Exercises, review, discussions, etc.
- Regular (weekly) programming assignments
  - Generally due electronically Thursday nights
- Tests
  - Midterm: Monday, May 8 in class
  - Final: Tuesday, June 6, 2:30-4:20 pm

Late Policy

No Late assignments

- Why?
  - Allows us to review and discuss assignments once they're due
  - Allows timely feedback and grading
  - Helps keep everyone on schedule during a fast-paced quarter
- If circumstances that are truly beyond your control prevent on-time work, contact the instructor
  - If it involves an exam, contact the instructor *before* the exam (at least via email or a phone call)

Grading

- Grade calculation
  - 40% homework assignments (typically 20 pts. per assignment)
  - 20% midterm exam
  - 40% final exam
- These scores are combined to get an overall score between 0-100. Final grades are computed as follows:
  - 90% at least 3.5
  - 80% at least 2.5
  - 70% at least 1.5
  - 60% at least 0.7

Academic (Mis)conduct

- Goal: balance the following
  - *Learning:* each student must do the work to learn effectively
  - *Cooperation:* people learn best when they can learn with others
  - *Fairness* and *honesty:* Nobody should ever represent the work of someone else as their own or try to claim credit for it

Academic (Mis)conduct

- Policy
  - You must do assignments by yourself (unless explicitly stated otherwise in an assignment)
  - You may discuss general approaches and ideas with others, but
  - You *may not ever* give code to or receive code from others
  - You *may not* have another person "walk you through" how to solve an assignment
- We check this and act when trouble is discovered
  - Either informally or through the vice-provost's office
- Use your common sense and ask first if unclear
  - Rule of thumb: *any activity you engage in for the purpose of earning credit while avoiding learning, or to help others do so, is likely to be an act of academic misconduct* (from CSE dept. policy – see link on the web)

Resources

- Handouts, slides, and notes
  - Alert! Not all material is on the handouts!
    (There is at least some reason to attend class and take notes)
  - Handouts will generally be distributed in lectures and sections and will be available on the course web
  - Additional notes will be posted on the web
- No formal textbook
- All e-mail announcements, assignment descriptions, etc. should be considered required reading.

Communicating Electronically

Course web site
- www.cs.washington.edu/143/
- Discussion Board: linked from Web site
  - UWNetID required
  - Open discussion – please contribute!
  - Course staff monitors and contributes as needed
- Email to us for things not appropriate for public discussion
  - Addresses on the web
  - Email works better for some things than other (e.g., very bad for trying to debug code)
- E-mail from us: cse143-announce
  - Sent directly to your UWNetID account
  - We'll try to keep the spam to a minimum, but … you must read and heed what we do send!

Computing Facilities

- Introductory Programming Lab (IPL)
  - Mary Gates Hall 334
  - CSE 143 (& CSE 142) TAs available in the lab
    Hours posted on the web
- Other campus labs
- Computing at home
- Even if you plan to compute at home, learn your way around the UW labs

Java!

- Java version: Java 5 (SDK 1.5)
  - Sun version for Windows/Linux
  - Latest Apple version on Mac OS X 10.4 (Tiger)
- You'll want a programming editor or environment to use on top of the basic Java system
  - Suggestions: TextPad, DrJava, or (for the more adventurous) Eclipse
    We're not particularly religious about which one you use, but your code must be standard Java and not rely on "wizards" or other non-trivial code generated by the programming environment
- Details: *Computing at Home* page on course web