CSE 142
University of Washington
Spring 2003

Welcome!
Organization & Administrivia

3 handouts today
Syllabus, Calendar, and an Assignment

Outline for Today

- Course Overview
- Administrative details
- Workload and grading
- Resources
- And a brief introduction to computer science & modeling

  - This information (and more) is included in today's handouts, and is on the web – no need to transcribe; just note highlights
  - Some things are new or different this quarter – be sure you're using current information

Introductions

- Instructors
  - Martin Dickey (9:30) & Alon Halevy (11:30)
    cse142-instructors@cs.washington.edu
- TAs
  - Many – see next slide
    cse142-tas@cs.washington.edu
- Course Administrator
  - Pim Lustig
    cse142-admin@cs.washington.edu
- Consultants: Savvy students we've hired to help out in the lab
  cse142-staff@cs.washington.edu reaches entire staff
- Students: You!

Teaching Assistants

- Alexander Cho
- Melissa Garcia
- Ksenia Guertsenberg
- Dennis Kehl-Fie
- Miryung Kim
- Chen-Chun Lin
- Theresa MacDuff
- Jayant Madhavan
- Clint Mumaw
- Rishi Parmar
- Stefan Sigurdsson
- Christopher Thompson
- Amanda Wang
- Albert Wong
- Zuo Yan

Course Organization

- 3 lectures per week (MWF)
- Quiz section once per week (Thursday)
  - Regular quizzes (easy to do if you keep up)
  - Exercises, review, discussions, etc.
    - Groups of 4-5 students will work together on activities throughout the quarter
- Designated quiz sections: more later
  - Regular
  - High-background?
  - Low-background?

Course Goals

- Learn general principles of computer programming
- Develop skills in the context of Computer Science
  - Reading and Analysis
  - Design
  - Implementation
  - Writing and Documentation
  - Testing
  - Debugging
- Develop technical communication skills
  - This is hard – and important to do well

- (And learn some Java in the process)
- (and have some fun)

My Goals for You

- Take you to the next technical step in programming
- Challenge you with material of considerable intellectual content, and with projects of considerable complexity.
- Develop your ability to learn independently
- Develop your ability to learn cooperatively
- Develop your ability to deal with incomplete and ambiguous information
- Increase our awareness of larger issues surrounding the use of information technology in our world
- If possible, make it fun. If possible...

My Goals For Myself

- Top goals for the course:
  - Help all of you learn
  - Keep the course on track
  - Make the homework projects interesting
  - Make lecture and section events you look forward to!
- Plus some more personal goals...
  - Learn some more Java myself
  - Make better use of technology in the classroom
  - Refine some teaching techniques
  - Take lots of pictures
  - And... learn a bunch of names!

Programming

- Both easier and harder than most people make it out to be
  - Easier: Many of the things good programmers do well are things that we already do all the time, but we don't think consciously about it
  - Harder: Programming is in large part a skill or an art
    - Requires a level of design, problem-solving, and precision that is not common in most of the rest of life
    - Very different from using applications or writing simple scripts
- Best learned by practice, trying things out, and reasoning
  - Don't worry – you won't break the computer by trying something new

Java!

A modern approach to programming including

- Objects everywhere; classes, interfaces, polymorphism
- Exceptions
- Streams and networking support
- Garbage collection
- Specifications, design by contract support
- Rich set of standard libraries
- Documentation tools and standards, on-line library documentation

- If none of the above makes sense... don't worry! It will eventually
- We'll use Sun's Java SDK 1.4.1
  - 1.3 will *not* do.
  - J++ (Microsoft) will *not* do
  - Details: *Computing at Home* page

What to Expect

- Homework assignments (almost weekly)
  - Mix of written problems and short programming exercises, some using a computer
  - Done individually
- Longer programming projects
  - 3-4 of these
  - Up to 2 weeks each
  - Work with a partner – pair programming
    - Partners assigned by course staff; different partner for each project
  - Individual written reports for each project
- Discussions and activities in lectures and quiz sections
- Designated textbook sections
- *Reading carefully and following instructions are key to success in this course*

Is it or Isn't it?

- 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 for full descriptions of the Java language
    - We cover the parts of Java that support good programming
  - Many important computer science topics
    - Some related to programming, but broader than Java
    - Data structures, algorithms, complexity analysis, software engineering...
- Fact:: writing programs that work perfectly isn't enough to get a perfect grade (!)

Who is the Course For?

- Course is for beginners, who...
  - Want a serious and rigorous introduction to programming and computer science
  - Can commit to the effort needed to succedd
- Previous programming experience is *not* a prerequisite!
- You should be comfortable with Math, Science, and English through the 12th grade level
- If you have already programmed...
  - In Java or C++? Did pretty well? Consider going right on to CSE143
    - Lecture MWF 2:30 pm Gugg 224 – try it *today*!
  - If you are not a beginner: remember that the course is *not* primarily for you
    - If you stay, please participate and be helpful

Keeping Up

- Course is for beginners, however…
- Material is cumulative
  - *Essential* to keep up
  - Ask for help the moment you need it; don't fall behind
- No late assignments accepted; no makeup exams or quizzes – need to keep on schedule
- Talk to course staff and fellow students
  - We're here to help
  - But ultimately it's up to you
    - "I waited for hours for the consultant" is no excuse – figure it out yourself!!

Communication

- People learn best when they ask questions and discuss material
  - With each other, with course staff, with friends, both in and out of class
    - Ask questions; participate!
- Main discussion channel: EPost Message Board
  - Link on course web page
  - Read this regularly & contribute when you can
  - Course staff will participate and contribute
  - You *must* use the Message Board as the starting point for technical questions
  - You *may not* post code to the Message Board

Resources to Help You Succeed

- Course staff
  - We're all in this together – feel free to talk to *any* TA or instructor and come to *anyone's* office hours
- Main information source: course web pages
  - www.cs.washington.edu/142
  - Start browsing now – be sure you can find your way around
- cse142-announce@cs mailing list for urgent messages from CSE142 staff to everyone
  - Registered students are included on this list automatically
- Staff email addresses for things that are not appropriate for the discussion board – details on the course web

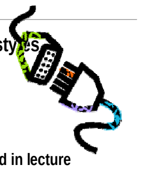
Book and Lecture Slides

- Textbook: *An Introduction to Programming and Object-Oriented Design* by Nino & Hosch
  - See course calendar for readings to do before class
    (latest version on the course web site)
- Updated lecture slides will be posted to the course web, sometime after the topic is completed
  - You should print the preliminary version, look at it before lecture, and bring it with you to take notes
  - Lecture slides are not a substitute for attending class!
    – there will be additional information, explanations, and activities in class that do not appear on the printed slides

Assessment

- Short miniquizzes in class (regularly)
  - Graded on a simple scale
  - cover current readings, whether discussed in class or not
- Midterm exams in lecture
  - Friday, April 25 and Friday, May 12 (tentative, but likely)
- Final exam
  - Wednesday, June 11
  - Time and location will be different than on the regular exam schedule
  - You must take the final exam on Wednesday, June 11– do not plan to leave campus early
    No matter how good a discount airfare you can get on June 10!
- Exams are a mix of multiple choice, written questions, short programming problems, etc.
- Exams and assignments do not necessarily assess the same skills and knowledge!

Disconnect?



- The parts of the course have different goals and styles
  - May seem disconnected from one another
- Tests vs. projects
  - Each measures things that the other can't
  - Tests may seem hard even when homework doesn't!
  - Homework may require learning about topics not covered in lecture
- Lectures vs. homework
  - Lectures may cover topics not practiced in homework
  - Lectures cover concepts and examples; will rarely talk about homework
  - Lectures sometimes mathematical, homework rarely so
- Quiz sections
  - active learning, practice, and review of recent topics

Grading

- Anticipated breakdown
  - 35% Homework and projects
  - 14% + 16% Midterm exams
  - 21% Final exam
  - 10% Quizzes
    weighted equally, regardless of length or difficulty
  - 4% Service and participation
    in-class activities, class participation, assistance to class members and staff, etc.
- Individual assignments and projects may weighted differently
  - depending on difficultly, length, etc.
- Percentage breakdown may change a fraction
  - depending on how the course evolves over the quarter

Collaboration vs Academic Misconduct

- While you should discuss ideas and learn with others, it is academic misconduct to represent someone else's work as your own, even if you have modified it
  - Same standard as in an English or History class – nothing changes because computer code might be involved
- You should acknowledge places where you receive help on homework or projects
  - "Help" means discussing problems, getting suggestions, but not writing up actual solutions or code (except with partner on programming projects)
- We have sophisticated software tools to check for problems, and we follow up when we find them
  - You ***don't*** want to receive an invitation to meet with the Vice Provost

More About Quiz Sections

- Regular: designed for all students – no prior experience
- High-background: designed for students with prior exposure to computing – chance to go into additional technical details, etc.
- All sections have the same assignments, take the same tests, and are graded the same
- On Wednesday, you may be able to request a switch to a different kind of section – we'll do the best we can to accommodate preferences
  - Between now and then, find out which section you're registered for and what kind it is
- Possible to informally switch sections with permission of TAs involved, even after Wednesday – no registration change needed

Computing Facilities

- CSE142 uses the UWired general labs
- Primary lab for CSE142/143 is the Introductory Programming Lab (IPL), 3rd floor Mary Gates Hall (MGH)
  - Pay a visit there today!
  - Course consulting staff available in the IPL
  - Can also use machines in Computing Commons in MGH and Odegaard (OUGL)
- Computing at home
  - Course software and tools are freely available for download
  - Instructions on the CSE 142 web
- Many assignments are submitted via the web
  - Very important to follow *exactly* the instructions for turning in each part of each assginment!
  - You don't follow the instructions – you don't get credit

Can't Get In?

- New slots open up as people drop
- No waiting list
- No entry codes
- Attend lectures and any old quiz section for the time being. But no guarantees – you might not get in.
- If you aren't registered by Wednesday or so – consider making a new plan