

CSSE 220 Day 29

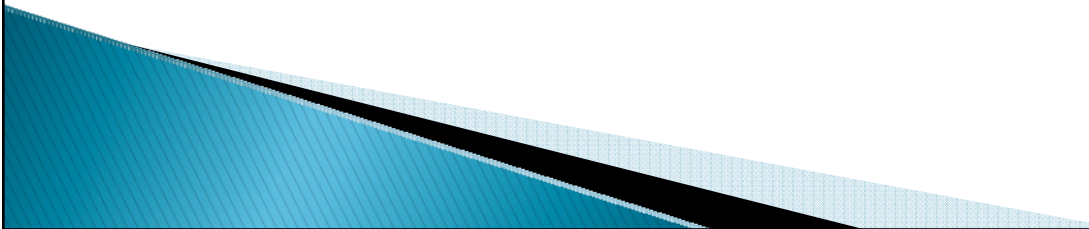
Course evaluations
Final Exam preview
Work on SpellChecker

- ▶ Please commit your outline of your presentation if you haven't done so
- ▶ Tomorrow, I'll have you check those items on grading checklist to help me not miss anything.

- ▶ Questions?

- ▶ Today:
 - Final exam review
 - Course evaluations
 - Project time

Project Presentations

- ▶ Order of team presentations will be randomly chosen.
 - ▶ Person who presents for each team will be randomly chosen.
 - ▶ At most 7 minutes, plus time for questions.
 - ▶ Fill out a form for each team's presentation (except your own)
 - ▶ No one but the presenters should use a computer during the presentations
- 

Final Exam Monday at 6 PM

- ▶ Two parts, like the previous exams
- ▶ My timing aims:
 - "Get it working" programming part:
 - Average (B) student can get 90% of the points in 2 hours.
 - C students can get 80% of the points in the 3 hours.
 - Other part:
 - Most students can complete it in 45 minutes
 - All students can get 95% of the points they can earn in less than an hour.

* By "B student" or "C Student", I mean students who will have that grade at the end of the term; not those who have it going into the exam.


A Comprehensive Exam

- ▶ Covers the entire term
- ▶ But since you have not been tested at all on the material since the last exam, it will count a disproportionate percent (approximately 30–40%) of the exam points.
- ▶ More than on the previous exams, there may be questions that ask you to think and apply course material to a new situations

What you can bring

- ▶ Same as on previous exams
 - First part
 - An 8.5 by 11 inch piece of paper on which you have written anything you want.
 - A calculator
 - Any quantity of blank paper
 - Programming part
 - Books, notes, course web pages and ANGEL pages, SUN's Java pages, any other pages that you have bookmarked before the exam.
 - No Google (or similar) searches, IM, mail, chat, etc.
 - **No use of headphones or earphones**

Effect of the exam on your grade

- ▶ Counts a little more than either of the previous exams (16% vs 12% of the total grade)
 - ▶ If your final exam percentage is significantly higher than your overall average for the course, I will "bump up" your average by a few points.
 - Especially if your average is just below a grade cutoff point.
 - Some people finally "get it all together" at the end of the course.
- 

Exam Provisions in the Syllabus

- ▶ You must earn a C grade on at least one exam in order to earn a C in the course.
 - If you did not earn at least 70 points on the first exam and did not earn at least 60 points on the second exam, you must have a C on the Final in order to get a C for the course.

Exam Provisions in the Syllabus

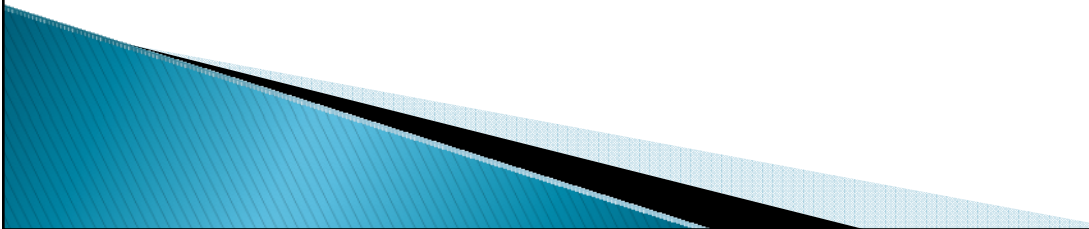
- ▶ You must have a passing average on the exams to pass the course.
 - If you did not earn a total of 110 points on the two exams, you will need a high enough score on the Final to bring your average up to the passing level (55%). **Are you in this situation?**
 - If you are barely above the passing exam average mark, you need to make sure that you do not dip below it as a result of the Final exam. **Are you in this situation?**

Exam Provisions in the Syllabus

- ▶ You must demonstrate that you can individually write and debug simple Java programs at the level of this course in order to pass. The exams will be the usual way to do this.
 - If your total for the programming parts of the two exams is less than 60, you have not yet demonstrated this competence.
 - On the programming part of the Final, you will need to get 50% of the points, or 80% of the class average (whichever is lower) in order to pass the course.

First part

- ▶ Questions will mostly be similar in style to written Homework problems and ANGEL quizzes
- ▶ Some may actually *be* problems from written Homework and ANGEL quizzes
- ▶ Some questions may be like "why does Java ... ?" or "Why would you choose ...?"

- ▶ Possible topics for CSSE 220 Exam 2
 - ▶ UML Class Diagrams
 - ▶ Terminology from chapters 1–4
 - ▶ Dynamic Binding and Polymorphism
Java Generics via Type Parameters (Generic classes, interfaces, methods)
 - ▶ Arrays class and Collections class – providers of static methods for search, sort, max, min, etc.
 - ▶ Measuring runtime efficiency
 - ▶ Big-oh, big-omega, big-theta (and the method of using limits to show relationships)
 - ▶ Inheritance to the max, as in BallWorlds
- 

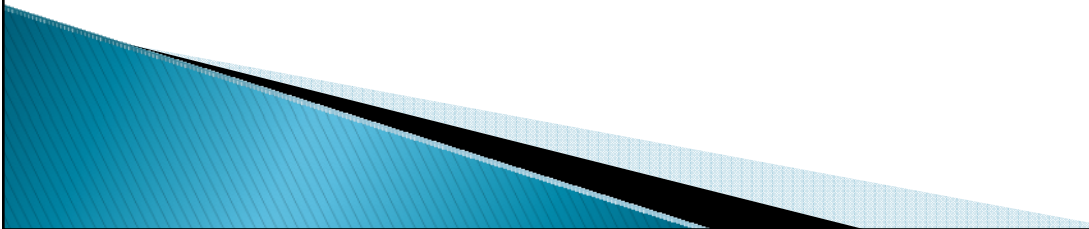
Exam 2 topics continued

- ▶ Function Objects (including Comparator). Compare and contrast Comparator and Comparable interfaces.
- ▶ Algorithms and running time for sequential and Binary search, basic ideas of interpolation search.
- ▶ Abstract data types and Data Structures: specification, implementation, application.
- ▶ Implementation of low-level data structures, such as integers and arrays.
- ▶ Data Structures Grand Tour: Know the basic definitions of each of the structures, along with big-oh running time for their main operations.
- ▶ Collections framework, including the main interfaces (Collection, List, Set, Map)
- ▶ The difference between a Set and a Map, and the practical differences between the "hash" and "Tree" versions of those.
- ▶ Iterator and ListIterator interfaces
- ▶ Implementing a list as an array list
- ▶ Implementing a list as a linked list (with and without a header node)

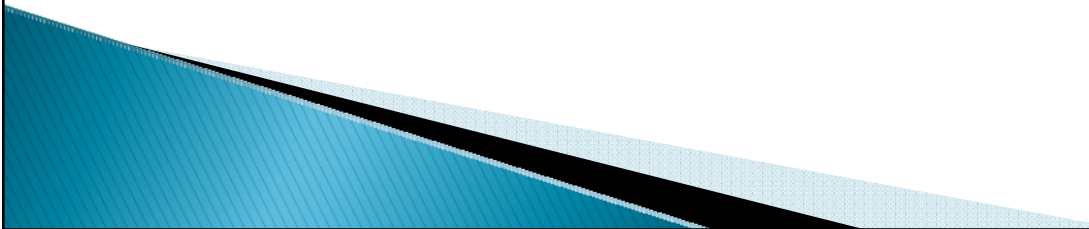
Topics since exam 2

- ▶ Recursion
- ▶ Sorting
 - Know insertion, bubble, selection sorts at the level of being able to write the code
 - Know how Shell Sort and Merge Sort work
 - Know how to analyze all but Shell
- ▶ Binary files, serialization

Programming part: What kinds of questions might you see?

- ▶ Create or modify a simple GUI program
 - Could include buttons, text fields, labels, text areas, drawing, panels, layout (flow, border, grid), listeners (mouse, action, key)
 - ▶ Data Structure use, implementation, type parameters
 - You know I'm going to ask lots on this!
 - ▶ Comparators and other function objects
 - ▶ Searching and sorting
 - ▶ 1D and 2D matrices like HW problem
 - ▶ Non-text IO, reading and writing objects
- 

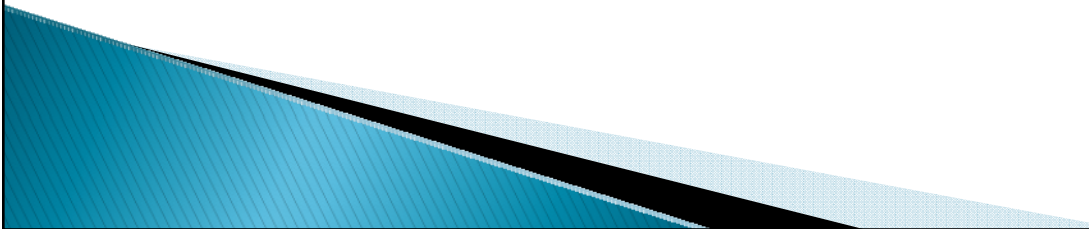
Material that may be covered by the exam

- ▶ Weiss chapters 1–4 (except 4.7.5, 4.7.6)
 - ▶ Sections 5.1, 5.2, 5.4–5.8 (you do not have to know the formal definition of big-oh and its cousins or do any proofs, but you should know the informal definitions)
 - ▶ Chapter 6
 - ▶ Sections 7.1, 7.3, 8.1–8.5 (not 8.4.1)
 - ▶ GUIs and Events
 - ▶ Implementation of Stack, Queue, ArrayList, and especially LinkedList
 - ▶ Object I/O
- 

If you want extra practice problems

- ▶ Consider these
 - 6.14, 6.15, 6.17, 6.19, 6.20, 6.21, 6.22
 - 8.1abc, 8.4abc, 8.5abc, 8.6abc
 - 15.9
 - 16.1, 16.3, 16.6, 16.7, 16.9
 - 17.5, 17.6, 17.12, 17.17, 17.18
 - Written problems from the previous exams

My evaluation of student work in the course

- ▶ Most of you have worked very hard and learned a tremendous amount.
 - ▶ You have moved from needing a lot of hand-holding toward being confident, competent, independent programmers
 - ▶ I think if you can do the written and programming exercises that I have given you, you are ready to compete with students at this level from any college in the country.
- 

Course evaluations

