What’s CIS 211 all about?

• Like CIS 210, but more.
• Old topics in greater depth
  – Objects, inheritance, polymorphism
  – Java collections (arrays, ArrayList)
  – Linked lists
  – Recursion
• New topics
  – GUI programming
  – Exceptions and error handling
  – Computational complexity (CIS 313, 315)
  – Searching and sorting (CIS 315)
  – Binary trees (CIS 313)
  – Design patterns

Rough Course Outline

• Week 1: Objects and inheritance (Ch. 8-9)
• Week 2: GUI programming
• Week 3-4: ArrayIntLists (Ch.15)
• Week 5: Linked lists (Ch.16)
• Week 6-ish: Midterm
• Week 7: Searching and sorting (Ch. 13)
• Week 8: Maps, sets, recursion. (Ch. 11, 12)
• Week 9-10: Binary (search) trees (Ch.17)

Primary Resources

Lecture:
  MWF 1pm in Gerlinger 242
Labs:
  026 Klamath
Textbook:
  Building Java Programs, 2nd Edition
  by Reges and Stepp
Web page:
  http://www.cs.uoregon.edu/classes/12W/cis211/
(Tentative) Office Hours

• Daniel: M 4-5pm, W 2-3pm, Th 2-3pm
• Amir: W 10-11:30am, Th 11am-12:30pm
• Emily: W 11:30am-1pm, Th 3-4pm

Supplementary Resources

• Textbook-related-stuff:
  – Practice-It! tool
  – Online video lectures (with book code)
  – PowerPoint slides
    (sometimes different from mine)
• Java API specification
• Other books

Grading

• Similar to CIS 210
• Assignments: 35%
  – 8 assignments total (last one due during Dead Week)
  – Due Thursday at 9:00pm submitted through
    Blackboard
  – No pair programming
• Lab attendance: 10%
  – Skip lab for the week if you complete and turn-in the
    lab problems by 1:00pm on Monday.
• Midterm: 25%
• Final: 30%

Collaboration

You must write your code independently.

Collaboration rule:

You may not discuss homework problems with your classmates when you are near a computer or a source code print-out.

Whiteboard/chalkboard discussions are ok. You can always ask me and the TAs for help in office hours or via email ([211help@cs.uoregon.edu](mailto:211help@cs.uoregon.edu)).

Violations are considered Academic Misconduct, and may be reported to the Office of Student Conduct and Community Standards.
**Tips for Getting Help**

- Be specific. Don’t just say, “my code doesn’t work.”
- Describe the debugging steps you’ve already taken.
  - Did you read the book or look up the API reference?
  - Did you try to come up with a simpler test case?
- If asking by email, allow 24 hours.
- You’ll get a faster response if your email can be answered in 15 seconds.
- Start early and read the instructions carefully!

**Teach Yourself Programming in Ten Years (ideas by Peter Norvig)**

- Many books promise quick success:
  - Mastering a skill takes about 10 years or 10,000 hours; programming is no different.
  - More: [http://norvig.com/21-days.html](http://norvig.com/21-days.html)

**Peter Norvig’s recipe for success:**

- Get interested in programming, and do some because it is fun.
- Talk to other programmers; read other programs.
- Program.
- Get a degree in computer science. (Optional)
- Work on projects with other programmers.
- Work on projects after other programmers.
- Learn 6+ languages.
- Understand the computer running the programs.
- Get involved in language standardization.
- Get out of language standardization.

**What if you don’t want to be an expert?**

- Computer science is prevalent in more and more fields:
  - Biology: Computational biology, bioinformatics
  - Physics: Simulations
  - Math: Theoretical computer science
  - Linguistics: Computational linguistics
  - Sociology: Social network analysis
- Any knowledge of programming is handy.
- CIS 200-series can help with this, too.