Web site

• Your source for information about this course: http://courses.cs.cornell.edu/cs2112
  – Lecture notes: you are expected to read
    • mostly not slides
    • may not include everything covered in lecture
    • may include extra not covered in lecture
    • often updated after the lecture
  – Assignments
    • may be updated after initial release
  – Pointers to resources

Communicating with staff

• Best way: use Piazza (http://piazza.com/class/spring2014/cs2112) to post questions
  – Answering other questions (well) has good karma.
  – Watch out for violating academic integrity
• Course announcements posted using Piazza (or emailed to all students)
• Consultants–hours online, location TBA
  – “Front line” for answering questions – consulting hours start today
Assignments

- 7 assignments
  - mostly programming but some written problems
  - total to 40% of total score
- First 2-3 assignments done solo; final project done with a partner.
- Assignment late penalties:
  - 1 day late: 10%
  - 2 days late: 20%
  - 3 days late: 40%
  - weekend = ‘1 day’

Exams and more

- Two evening prelims: Mar 11 and Apr 22
  - 10%, 13% of total score resp.
- Final exam: Dec 5, 7pm
  - 30% of total score
- 2% of score:
  - participation (in-class, Piazza, course evals)
  - occasional in-class quizzes
- Mapping from score to grade will be holistic

Meetings

- Lectures: TTh 10:10-11, Phillips 101
- Discussion sections (attend 1 per week)
  - T 12:20-1:10 (Phillips 213)
  - W 1:25-2:15 (Upson 111)
- Labs (attend 1 per week)
  - Monday 7:30-8:20 (Hollister 314)
  - Wednesday 7:30-8:20 (Upson 111)
- 4 hours per week - attendance required

Assignments will be submitted to CMS.
Grades, solutions will be posted on CMS (http://cms.csuglab.cs.cornell.edu)
Regrade requests should be posted to CMS soon after receiving grade.
Labs

- Will do programming exercises, solve problems, learn about tools
- Bring a laptop
- First lab: Monday (set up Eclipse and do some programming)

Textbook

  - Should be available at campus store.
  - On reserve in library
  - Recommended, not required
  - Not heavily used—can share with a friend or two.

Academic integrity

- Work you turn in must be yours
  - You must be able to explain your answers fully
  - Copying code, answers is never okay
  - Letting others copy you also violates the Code
  - Discussions with others are fine if they could have happened in a lightless room.
- We will use highly effective tools for detecting plagiarism.
- Report any discussions on assignments.
- Our goal: spend time on course content.

CS 2112 or ENGRD 2112?

- Does not matter
CS/ENGRD 2110 or CS/ENGRD 2112?

- 2112 is an ‘honors’ version of 2110.
  - harder assignments including a final project
  - more material
    - e.g., more algorithms and their analysis (theory)
    - e.g., more about design, design patterns (practice)
  - more credits (4 vs. 3)
  - aimed at CS majors: smaller

2110 vs. 2112

- Warning: you may get splashed.

What it’s about

An introduction to computer science and software engineering

- Programming language features
  - data abstraction, subtyping, generic programming
  - concurrency and threads

- Object-oriented design — organizing large programs
  - specifications
  - design patterns
  - frameworks and event-driven programming

- Data structures and algorithms
  - recursive algorithms and data structures
  - algorithm analysis and designing for efficiency
  - asymptotic complexity, induction
  - arrays, lists, stacks, queues, trees, graphs, hash tables

Using Java. Not a course on Java!

Sam Loyd’s 8 Puzzle

Goal: Given an initial configuration of tiles, find a sequence of moves that will lead to the sorted configuration.

A particular configuration is called a state of the puzzle.
State Transition Diagram of 8-Puzzle

State Transition Diagram of 8-Puzzle:

A state $Y$ is adjacent to state $X$ if $Y$ can be reached from $X$ in one move.

State Transition Diagram for a 2x2 Puzzle

State Transition Diagram for a 2x2 Puzzle:

*Sorted State*

Solutions for this state:
- SWN
- WSENWSENW
- SWEWN

Graphs

- State transition diagram in previous slide is an example of a graph: a mathematical abstraction
  - nodes (or vertices): the puzzle states
  - edges (or arcs): the transitions, possibly labeled
- Graphs are all around us: airline routes, roadmaps, org charts, pipelines, ...

Graph algorithms

- Large toolbox of efficient algorithms for graphs help us solve problems:
  - searching for best nodes/shortest paths
  - finding maximum flow through graph
  - minimum spanning trees
  - ...
- And known hardness results (e.g., finding Hamiltonian cycles) tell you what you can't solve.
Software design choices

• What operations should puzzle objects have?
• How do we represent states? The initial state?
• How do we present information to the user and support interaction?
• How do we break the coding up into parts that can be coded independently?
• How to structure code so it can be maintained, upgraded?

Why you need CS 2112

• Data structures and algorithms to solve problems efficiently and effectively
• Design techniques to produce code that works quickly and keeps working
• Computer science:
  – algorithms, data structures, programming languages, design principles, knowledge of what is possible and feasible.
• Good programmers have more fun
  – 10X more productive
  – better able to adapt, grow, see opportunities

Next steps

• Get your Piazza account set up ASAP
• Keep an eye on the 2112 website
• Download the first programming assignment, released today, due in one week
• Make sure you have Eclipse downloaded and working. See consultants for help.
• Don’t forget about labs and recitations next week.
• Have fun!