Today

• Final-exam information
• Last few topics in previous lecture
• Course “victory lap”
  – High-level overview of main ideas and goals
  – Connection to homeworks
  – Context
• Also:
  – Thank-yous
  – … and a small surprise
• Last 20 minutes for course evaluations
Final-exam information

- Monday, 8:30-10:20AM
- Very heavily weighted toward second half of course
- See email from me and sample exams
- See email from Alex about Sunday review session
- As usual, “tough but fair and rewarding”
Victory Lap

A victory lap is an extra trip around the track
  – By the exhausted victors
    (that’s us) 😊

Review course goals
  – Slides from Lecture 1
  – What makes CSE331 special
Huge thanks to the folks who made it work

Brandon Dalesandro

Brian Griffith

Karthik Palaniappan

Riley Klingler

Alex Mariakakis

Uldarico Muico
3 slides from Lecture 1…
10 weeks ago: Welcome!

We have 10 weeks to move well beyond novice programmer:

- Larger programs
  - Small programs are easy: “code it up”
  - Complexity changes everything: “design an artifact”
  - Analogy: using hammers and saws vs. making cabinets (but not yet building houses)

- Principled, systematic software: What does “it’s right” mean? How do we know “it’s right”? What are best practices for “getting it right”?

- Effective use of languages and tools: Java, IDEs, debuggers, JUnit, JavaDoc, Subversion, …
  - Principles are ultimately more important than details
    - You will forever learn details of new tools/versions
10 weeks ago: Goals

• CSE 331 will teach you to how to write correct programs

• What does it mean for a program to be correct?
  – Specifications

• What are ways to achieve correctness?
  – Principled design and development
  – Abstraction and modularity
  – Documentation

• What are ways to verify correctness?
  – Testing
  – Reasoning and verification
10 weeks ago: Managing complexity

• Abstraction and specification
  – Procedural, data, and control flow abstractions
  – Why they are useful and how to use them
• Writing, understanding, and reasoning about code
  – Will use Java, but the issues apply in all languages
  – Some focus on object-oriented programming
• Program design and documentation
  – What makes a design good or bad (example: modularity)
  – Design processes and tools
• Pragmatic considerations
  – Testing
  – Debugging and defensive programming
  – [more in CSE403: Managing software projects]
10 weeks (plus 10 days) ago
Divide and conquer: Modularity, abstraction, specs

No one person can understand all of a realistic system

- **Modularity** permits focusing on just one part
- **Abstraction** enables ignoring detail
- **Specifications** (and **documentation**) formally describe behavior
- **Reasoning** relies on all three to understand/fix errors
  - Or avoid them in the first place
  - **Proving, testing, debugging**: all are intellectually challenging
How CSE 331 fits together

Lectures: ideas ⇒ Assignments: get practice

Specifications ⇒ Design classes
Testing ⇒ Write tests
Subtyping ⇒ Write subclasses
Equality & identity ⇒ Override equals, use collections
Generics ⇒ Write generic classes
Design patterns ⇒ Larger designs; MVC
Reasoning, debugging ⇒ Correctness, testing
Events ⇒ GUIs
Systems integration ⇒ N/A
What you have learned in CSE 331

Compare your skills today to 10 weeks ago
  – Theory: abstraction, specification, design
  – Practice: implementation, testing
  – Theory & practice: correctness

Bottom line aspiration: Much of what we’ve done would be 
  easy for you today
  This is a measure of how much you have learned

There is no such thing as a “born” programmer!

Genius is 1% inspiration and 99% perspiration.
  Thomas A. Edison
What you will learn later

• Your next project can be much more ambitious
  – But beware of “second system” effect

• Know your limits
  – Be humble (reality helps you with this)

• You will continue to learn
  – Building interesting systems is never easy
    • Like any worthwhile endeavor
  – Practice is a good teacher
    • Requires thoughtful introspection
    • Don’t learn only by trial and error!
  – Voraciously consume ideas and tools
What comes next?

Classes
- CSE 403 Software Engineering
  - Focuses more on requirements, software lifecycle, teamwork
- Capstone projects
- Any class that requires software design and implementation

Research
- In software engineering & programming systems
- In any topic that involves software

Having an impact on the world
- Jobs (and job interviews)
- Larger programming projects
Last slide

- System building is fun!
  - It’s even more fun when you’re successful

- Pay attention to what matters
  - Take advantage of the techniques and tools you’ve learned (and will learn!)

- On a personal note:
  - I’m pretty proud I made it to class every day 😊
  - Don’t be a stranger: I love to hear how you do in CSE and beyond as alumni