Today

- What have we accomplished this quarter?
- How does CSE351 fit into the curriculum?
- What about 400-level courses?

- Evaluation of the course
  - Help us make the CSE351 better in the future
  - How did it help you?
  - What could have been done better?

The Big Theme

- THE HARDWARE/SOFTWARE INTERFACE
- How does the hardware (0s and 1s, processor executing instructions) relate to the software (Java programs)?
- Computing is about abstractions (but don’t forget reality)
- What are the abstractions that we use?
- What do YOU need to know about them?
  - When do they break down and you have to peek under the hood?
  - What assumptions are being made that may or may not hold in a new context or for a new technology?
  - What bugs can they cause and how do you find them?
- Become a better programmer and begin to understand the thought processes that go into building computer systems
Little Theme 1: Representation

- All digital systems represent everything as 0s and 1s
- Everything includes:
  - Numbers – integers and floating point
  - Characters – the building blocks of strings
  - Instructions – the directives to the CPU that make up a program
  - Pointers – addresses of data objects in memory
- These encodings are stored in registers, caches, memories, disks, etc.
- They all need addresses
  - A way to find them
  - Find a new place to put a new item
  - Reclaim the place in memory when data no longer needed

Little Theme 2: Translation

- There is a big gap between how we think about programs and data and the 0s and 1s of computers
- Need languages to describe what we mean
- Languages need to be translated one step at a time
  - Word-by-word
  - Phrase structures
  - Grammar
- We know Java as a programming language
  - Have to work our way down to the 0s and 1s of computers
  - Try not to lose anything in translation!
  - We’ll encounter Java byte-codes, C language, assembly language, and machine code (for the X86 family of CPU architectures)
Little Theme 3: Control Flow

- How do computers orchestrate the many things they are doing – seemingly in parallel
- What do we have to keep track of when we call a method, and then another, and then another, and so on
- How do we know what to do upon “return”
- User programs and operating systems
  - Multiple user programs
  - Operating system has to orchestrate them all
    - Each gets a share of computing cycles
    - They may need to share system resources (memory, I/O, disks)
  - Yielding and taking control of the processor
    - Voluntary or by force?

Course Outcomes

- Foundation: basics of high-level programming (Java)
- Understanding of some of the abstractions that exist between programs and the hardware they run on, why they exist, and how they build upon each other
- Knowledge of some of the details of underlying implementations
- Become more effective programmers
  - More efficient at finding and eliminating bugs
  - Understand the many factors that influence program performance
  - Facility with some of the many languages that we use to describe programs and data
- Prepare for later classes in CSE
Assessment

- How did we do getting these themes across?
- What could have been done better?

- Where the assignments at a good pace and level?
- Did you find the time you spend on the course productive?

- What do you wish we had spent more time on?
- What could have been done more quickly?

CSE351’s role in CSE Curriculum

- Pre-requisites
  - 142 and 143: Intro Programming I and II

- One of 6 core courses
  - 311: Foundations I
  - 312: Foundations II
  - 331: SW Design and Implementation
  - 332: Data Abstractions
  - 351: HW/SW Interface
  - 352: HW Design and Implementation

- 351 sets the context for many follow-on courses
CSE351’s place in new CSE Curriculum

CSE351

CSE352

HW Design

CSE333

Systems Prog

CSE351

Op Systems

CSE401

Compilers

CSE461

Networks

CSE484

Security

CSE466

Emb Systems

Performance

Concurrency

Comp. Arch.

Distributed Systems

Machine Code

Execution Model

Real-Time Control

The HW/SW Interface

Underlying principles linking hardware and software

CS 143
Intro Prog II

Evaluation

- Survey Form – standard questions you’ve seen before
- Additional Questions
  - For ABET accreditation of our Computer Engineering program
  - Specific questions to this course
- Yellow Sheets
  - open format, what you really think of what happened this quarter
  - textbook (readability, denseness, problems, cost, ...)
  - assignments (utility, time commitment, appeal, ...)
  - exams (coverage, fairness, correlation to assignments, ...)
  - topics (remove, add, change coverage, ...)
  - mix of work (reading, programming, problems, section, ...)
  - grading scheme (relative weights of exams, assignments, participation, ...)
  - section (lab or section?, interactive exercises, debugging, topics, ...)
The Hard Things to Evaluate

- What will you remember in going on to next core courses?
- What will you remember in senior year, for later courses?
- Will this have an impact on ability to get internships/jobs?
- Will this enable deeper participation in a range of research?

- This takes years to assess properly
- Continuation of content with follow-on courses
  - e.g., use of X86/Y86 for implementation in 352, same text!
  - e.g., overlap with topics in 333 (C/C++ systems programming)
  - e.g., sufficiency of background from 142/143
  - e.g., 390A (unix tools) as a co-requisite

Acknowledgments

- Thanks for the privilege of being your instructor this quarter
  - You were a fantastic class, great questions, great attitude
- Thanks for your feedback (now and in the future)
  - Your fellow students will appreciate all of your comments/input
- Thanks for the great service of your four TAs
  - Tom (for the sections)
  - Chantal, Chee Wei, Sunjay (for all assignments and grading)
  - And all Fantastic 4 for the support they provided to all of you and to me