

## Final exam: Wednesday, March 20, 2:30pm

- Here in Mary Gates 389, 2:30pm – 4:20pm
- Focus will be on material covered in lecture and in labs during the **second half of the course**
  - Use textbook to supplement / fill in details; if you see something in the textbook that we didn't cover in class, then it's not on the exam
- Questions will be similar to homeworks and past exams
- How to prepare:
  - Study lecture slides (review coursecasts if you missed something)
  - Understand the **homework problems and solutions**
  - Review the practice problems in the book
  - Look at previous exams
- Q&A session: Tuesday, at ??
  - But please post questions to the discussion board before then

## CSE 351 grading

- Your overall grade in the class is calculated from:
  - Homeworks (20%)
  - Labs (40%)
  - Midterm exam (15%)
  - Final exam (25%)

<http://www.cs.washington.edu/education/courses/cse351/13wi/policies.html>

## The Big Theme: Interfaces and Abstractions

- Computing is about abstractions
  - (but we can'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 bugs can they cause and how do you find them?
- How does the hardware (0s and 1s, processor executing instructions) relate to the software (C/Java programs)?
  - Become a better programmer and begin to understand the important concepts that have evolved in building ever more complex computer systems

## Little Theme 1: Representation

- All digital systems represent everything as 0s and 1s
  - The 0 and 1 are really two different voltage ranges in the wires
- “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 stored away in memory
- These encodings are stored throughout a computer system
  - 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
  - Words, phrases and grammars
- 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)

Winter 2013

Wrap-up

5

## Little Theme 3: Control Flow

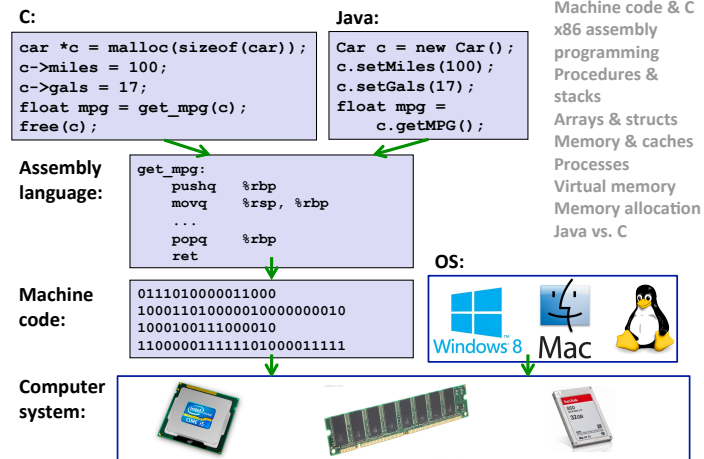
- How do computers orchestrate the many things they are doing – seemingly in parallel
- What do we have to keep track of when we call a method, and then another, and then another, and so on
- How do we know what to do upon “return”
- User programs and operating systems
  - Multiple user programs
  - 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”?

Winter 2013

Wrap-up

6

## Roadmap



Winter 2013

Wrap-up

7

## Course Perspective

- This course will make you a better programmer
  - Purpose is to show how software really works
  - By understanding the underlying system, one can be more effective as a programmer
    - Better debugging
    - Better basis for evaluating performance
    - How multiple activities work in concert (e.g., OS and user programs)
  - Not just a course for dedicated hackers
    - What every CSE major needs to know
    - Job interviewers love to ask questions from 351!
  - Provide a context in which to place the other CSE courses you'll take

Winter 2013

Wrap-up

8

## If you liked this class, then consider...

