EECS 111: Fundamentals of Computer Programming

Winter 2012
Doug Downey
Northwestern University
Goals of the Course

• To teach the fundamentals of designing/writing/testing/maintaining computer programs
Logistics

• Class meets MTuWF: 2-2:50PM, Tech L211
  – Tuesdays: TA-led review session

• Professor: Doug Downey
  – Office hours: Fri 3-4PM, Ford 3-345

• TAs: Bharat Pattabhiram; Tim Rambo
  • Office hours: TBD

• Course web page (also linked from my home page)
Grading, etc.

• 7 Homeworks (50% of grade)
  – Almost entirely programming

• 2-part final project due last two weeks (25% of grade)
  – Game!

• Exam in week 6 (25% of grade)

• Attendance not required
Computer Programming

• Why bother?
  – CS Job Prospects
Job Prospects


Computer Programming

• Why bother?
  – CS Job Prospects
  – Applicability to other fields
    (computational biology, finance, the arts, entertainment, etc.)
  – Fun!
    • ...in this course, we’ll do a ten-minute “moment for CS” every Friday to introduce some fun aspect of the field
A barrage of show-of-hands questions

- How many of you:
  - Are 1st- or 2nd-year students?
  - Have programmed before?
  - Have taken a programming class?
  - Are (or intend to be) a CS major?
  - Are in McCormick?
  - Have programmed in “Scheme”?
  - Have programmed in Java?
  - Have heard of the Fibonacci sequence?
About your instructors

• My research area: AI
  – Specifically machine learning and Web search
  – I have worked as a professional software engineer

• Your TAs
Source Material

• **How to Design Programs, Second Edition** (HTDP/2e) by Felleisen, Findler, Flatt, and Krishnamurthi -- note: this is a work in progress

• **How to Design Programs, First Edition** (HTDP) by Felleisen, Findler, Flatt, and Krishnamurthi

• **The Structure and Interpretation of Computer Programs** (SICP), by Abelson, Sussman and Sussman
This course is about

• The FUNdamentals of programming and computation
  – From specification to implementation
  – Some:
    • Software engineering principles
    • Computational complexity

• “The way to learn to program is by programming”
  – Nathan Myhrvold
This course is not about

Racket: Scheme-like language we use
Dr. Racket: development environment

- Racket, Dr. Racket
- Other programming languages (Java, C++), tools (Visual Studio), APIs, Protocols, etc.

Fundamentals of Computer Programming (this course): how to design, test, implement, and maintain programs
Tiny assignment for today

• Install Dr. Racket
• Skim HtDP Prologue

• ...links for both off course home page
• 1st (real) homework assigned tomorrow