CS PRINCIPLES IN CS 202: INTRO TO COMPUTATION

Professor Andrea Arpaci-Dusseau
University of Wisconsin, Madison
Computer Science Associate Chair
Summer 2012

OUTLINE

• CS Principles AP Pilot Program
• CS 202: Introduction to Computation?
  - Fit at UW-Madison
  - Goals and Enrollment
  - Material
  - Assignments
• Suggested Resources

CS AP PRINCIPLES

• New course to broaden participation in computing and computer science
• Collaborative effort College Board and National Science Foundation
• Over 80 colleges and universities have attested to goals
• AP Test to debut in May 2016 or later
• Fall 2010: 5 University Pilots
• Fall 2011: 9 Pairs of University and High School Pilots
  - UW-Madison and West High School (Andy Kuemmel)

7 BIG IDEAS

1. Creativity: Computing is a creative activity
   - Computing fosters creative expression
   - Programming is a creative process
2. Abstraction: Reduces detail to focus on relevant concepts
   - Abstractions on binary sequences can represent all digital data
   - Many levels of abstraction used in computation
   - Models and simulations use abstraction to answer questions
3. Data: Data facilitates creation of knowledge
   - Computing facilitates exploring big data
   - Trade-offs for representing and handing data
7 BIG IDEAS (CONT)

4 Algorithms: Express solutions
   - Evaluate analytically and empirically
5 Programming: Enables problem solving, expression, and creation
   - Execute algorithms, use logical concepts
6 Internet: Pervades modern computing
   - Network of autonomous systems
   - Importance of security
7 Impact: Computing has global impacts
   - Affects communication and cognition
   - Enables innovation in every field
   - Situated within social contexts

COURSE SYLLABUS

• Freedom to incorporate big ideas as desired into syllabus
  • Order of ideas, percentage of time
• Can’t discuss big ideas sequentially or in isolation
• Course different at every pilot site
• No one programming language
  (Scratch, Alice, Excel, Python, Processing, …)

OUTLINE

• CS Principles AP Pilot Program
• CS 202 : Introduction to Computation?
  • Fit at UW Madison
  • Goals and Enrollment
  • Material
    • Assignments
• Suggested Resources

UW-MADISON CS

• Computer Science department
  • About 200 CS Majors (or certificates) each year
  • 40 faculty members
  • Consistently ranked around 10th in nation
  • Strong MS and PhD research programs

13% females over all years
**UW-MADISON CS**

- CS Curriculum
  - CS 302: Intro to Programming (Java)
  - CS 367: Intro to Data Structures
  - Many 500-level courses
    - Theory, OS and Networks, Databases, PL and Compilers, Security, AI, NLP, Graphics, Vision, etc.
  - No overview course for either majors or non-majors

**CS 202: GOALS**

- Positive impression and understanding of CS
  - Not just how to use computers and applications or jargon
  - Interesting and creative
- Computer science is important/useful for everyone
  - Almost every field can benefit from CS
    - Simulations, modeling
    - Problem solving and algorithmic thinking

**GOAL: POSITIVE IMPRESSION**

- Scale 0-4, how interested are you in learning more about area? (45 freshman female in residential dorm for STEM)

**CS 202: WHO TAKES IT?**

- Not geared towards CS majors
  - (but persuades some people – esp CS certificate)
    - Why? Interesting or to fulfill QR-A
    - Not required for any major – wide variety
    - Freshman Interest Groups (FIGs)
- Small section in Fall 2009 (20 students)
  - Approx ½ freshman, ¼ soph, ¼ junior + senior
  - 40% female through Student Center
    - Much lower percentage through advising (Fall freshman)
  - 80% no prior programming experience
WHAT DO STUDENTS LEARN IN CS 202?

- Design and implement creative applications
  - Scratch: programming environment for beginners
- Computation and data are powerful
  - Simple algorithms can solve complex problems quickly
- How modern computers work
  - Hardware: From bits to running anything and storing all information
  - Software: Run multiple apps and send messages
- Interesting applications and frontiers of Computer Science
  - Artificial Intelligence, Robotics, Security, Education
- Similar Big Ideas as AP CS Principles
  - CS 202 has more on “how computers work” less on “Societal Impact”

ADVANTAGES

- Not intimidating
- Visual, fun projects
- No syntax errors
- Easy to obtain freely
- Solid and tested
- Easy to share online
- Variables, Lists
- Most students like it

DISADVANTAGES

- No procedures
- Painful for large projects
- No real file I/O (big data)
- Slow to grade
- Some want “real” language

WHAT IS NOT FOCUS OF CS 202?

- NOT: How to use different applications
  - NOT: Word, ppt, excel, databases, web searching
  - Goal: Design and build own applications
- NOT: How to administrate computers
  - NOT: boot or install OS, new printers, network?
  - Goal: Understand fundamentals of how computers work
- NOT: Implications of technology
  - NOT: Impact of facebook and twitter on society
  - Goal: Understand basics of how technology is designed, Explore limits of current technology

PROGRAMMING LANGUAGE: SCRATCH

Advantages

- Not intimidating
- Visual, fun projects
- No syntax errors
- Easy to obtain freely
- Solid and tested
- Easy to share online
- Variables, Lists
- Most students like it

Disadvantages

- No procedures
- Painful for large projects
- No real file I/O (big data)
- Slow to grade
- Some want “real” language

MOTivating THEMES

- 15 weeks of lectures (3 hours a week)
  - No labs or discussion sections
  - 3 themes, 5 weeks each (exam after each)
- Theme 1: Interacting with humans
  - How can we create games competitive against humans?
- Theme 2: Big Data
  - How does huge amounts of data change the problems we can solve?
- Theme 3: Run any program
  - How do computers work to solve these problems?
- Challenge: Intermix programming (and other big ideas) throughout course
**How do computers...?** | **Answer**
--- | ---
Interact with humans? | Artificial intelligence
Solve problems? | Algorithms
Know what to do? | Programming languages
Make art? | Control flow
Show animated stories? | Flowcharts + Abstraction
Make decisions? | Decision Trees and If’s
Remember? | Variables
Avoid races? | Critical sections
Educational software? | Software structure
Guess what happens? | Probability trials
Win games against you? | Game trees
Interact with humans? | Social robots

**How do computers...?** | **Answer**
--- | ---
Solve societal problems? | Lots of data
Visualize data? | More abstraction
Find goal? | Optimization
Find stuff? | Search
Find stuff faster? | Binary search
Sort data? | Selection and insertion
Sort data faster? | Merge and quick sort
Analyze text? | Histograms
Find web pages? | Search engines
Predict the future? | Simulation
Share secrets? | Cryptography
Reach their limits? | P vs. NP

**How do computers...?** | **Answer**
--- | ---
Work?? | Abstraction
Represent numbers? | Bits and binary
Words, pictures, sound? | Encode and compress
Act logically? | Boolean logic
Calculate? | Gates and circuits
Remember? | Memory
Execute instructions? | CPUs
Run multiple apps? | Operating Systems
Share memory well? | Caching
Communicate ? | Networking
Use other languages? | Compilers
Tolerate errors? | Replication

**How do computers...?** | **Answer**
--- | ---
Solve problems? | Algorithms
Know what to do? | Programming languages
Make art? | Control flow
Show animated stories? | Flowcharts + Abstraction
Make decisions? | Decision Trees and If’s
Remember? | Variables
Avoid races? | Critical sections
Educational software? | Software structure
Guess what happens? | Probability trials
Win games against you? | Game trees
Interact with humans? | Social robots

**How do computers...?** | **Answer**
--- | ---
Interact with humans? | Artificial intelligence
Solve problems? | Algorithms
Know what to do? | Programming languages
Make art? | Control flow
Show animated stories? | Flowcharts + Abstraction
Make decisions? | Decision Trees and If’s
Remember? | Variables
Avoid races? | Critical sections
Educational software? | Software structure
Guess what happens? | Probability trials
Win games against you? | Game trees
Interact with humans? | Social robots

**HOMEWORKS:**
CREATE, EXPLORE, UNDERSTAND

- CS is about creating, not memorizing
- Weekly homework assignments
  - Create open-ended projects in Scratch
    - Art, stories, music, games – not quantitative
    - Large range of skill level
    - Need some specification and guidelines
    - Graded generously – bugs won’t kill us
    - Share work with others in class (extra credit for artistry)
  - Explore: How good are current techniques and applications?
    - Language translation, image recognition, search, visualization
  - Understand: How does computation work?
    - Programming, logic, binary numbers
HOMEWORKS: CREATE, EXPLORE, UNDERSTAND

1. Explore Scratch Website, Explore Games with a Purpose
2. Create a Scratch Drawing and Understand Scratch Code
3. Create Animated Story, Explore Recommendation Systems, Understand Variables
4. Create a Points-based Game, Explore Translating and Conversation
5. Create a Probabilistic Music Composition with Lists and Explore Data Visualization
6. Create Weebles and Understand Searching
7. Explore Google Trends, Explore Education and Technology, Understand Sorting
8. Create Trivia Games in Scratch
9. Play with Binary Numbers and Explore GIFs
10. Explore TED Talks

FINAL PROJECT: Create card game with Partner (Go Fish, 21, Memory, War, Uno)

RESOURCES

  - Stable version, all lecture notes and homeworks
- [http://www.csprinciples.org/](http://www.csprinciples.org/)

"Blown to Bits – Your Life, Liberty, and Happiness after the Digital Explosion" by Abelson, Ledeen, and Lewis

AFTER SCRATCH?

Other appropriate languages:
BYOB – Snap! (Berkeley – Build Your Own Blocks in Scratch)
Python
Processing

QUESTIONS?