AP CS Principles Project & UW's Pilot

Lawrence Snyder
[with Susan Evans and Brandon Blakeley]
University of Washington, Seattle

Motivation

- Stipulate these facts …
  - CS majors have been declining (though maybe …
  - Labor-needs in CS-related fields already exceed supply and it's getting worse fast
  - US Economic Health strongly tied to computing
  - CS has rotten diversity
  - AP CS A Java Programming attracts few takers
  - AP CS A has a rotten reputation
  - Quality CS not available in most high schools
  - HS "Computer Science" doesn't count for NCAA!

The Vision

- A "different" AP exam emphasizing CS fundamentals could help on many fronts …
  - CS can take its place with HS science peers
  - Convey the "joy, beauty and awe" many experience
  - Greatly expand population of people truly familiar with what computing is … maybe improve image!
  - AP is rigorous
  - AP is replicable
  - AP is loved by students, teachers, admins, colleges

A New AP Course

COMPUTER SCIENCE PRINCIPLES

- Designed for general student population
- Another AP CS class; AP Java Program'g remains
- Not an "apps" course, but concepts & capabilities

AP Development Process …

- When NRC told AP to retool it's science exams on fundamentals, they gave a process; we use it
- Jan Cuny funded Owen Astrachan & Amy Briggs
  - Commission of 10 – HS & college teachers
  - Advisory Committee of 20 – college academics
- Formulate list of fundamentals; vet widely
  - 7 big ideas; 6 computational thinking practices
- 2010/11 Pick 5 universities to pilot course
- 2011/12 High school pilots + more U pilots
- Along way, flesh out curriculum, test questions
- Launch course + test in ~2015

Team

7 Big Ideas

1. Creativity → innovation, exploration
2. Abstraction reduces detail to solve problems
3. Data and information create knowledge
4. Algorithms express solutions to comp probs
5. Programming produces comp artifacts
6. Devices, systems, networks, … automate computational solutions
7. Computing enables innovation in other fields

http://csprinciples.org

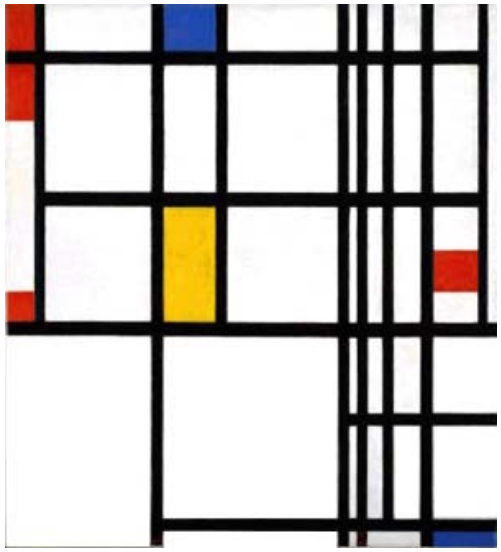
"Vet Widely"

- The Commission "decides," but definitely in consultation with Advisory Committee
- College Curriculum Survey – CS Departments asked to review the learning objectives
  - 116 replies to 136 item survey, taking over 2 hours
- Departments "Attest" to project's worth and say if they'd give credit/placement
  - 100+ departments positive; 80 to give C or P
- College Board was WOWed by CS community

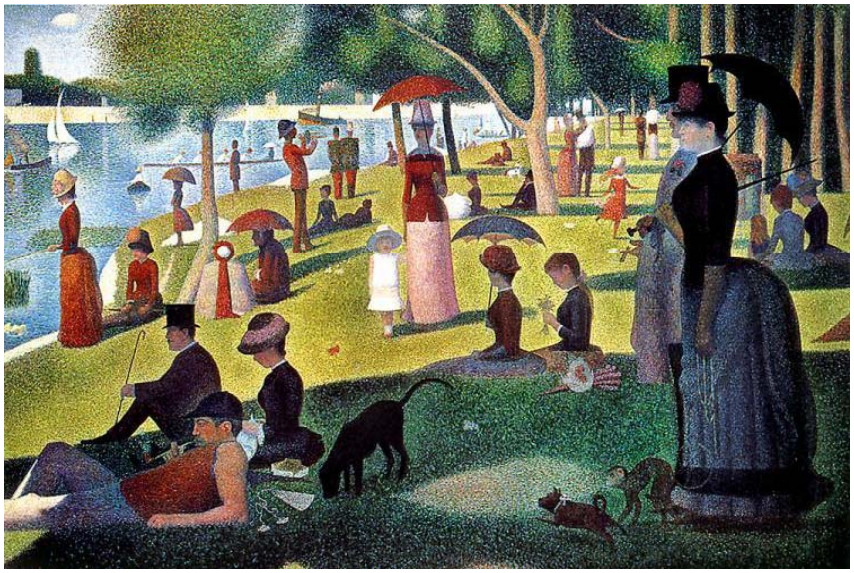
The Task for Piloters…

- The Commission and the Advisory Committee specified the content in broad strokes
- The Pilot's task is to express it in the fine detail of a college class …

Mondrian to Seurat



→



Five Campuses, Five Teachers

The pilot schools and instructors …

- Metropolitan State College, Denver: Jody Paul
- UC Berkeley: Dan Garcia
- UC San Diego: Beth Simon
- UNC at Charlotte: Tiffany Barnes
- U Washington: Larry Snyder

The teachers were chosen from among the Commission and Advisory Committee Members

Cross-Campus Comparison I

	Title	1st	~n
MSCD	Living in a Computing World	F (S)	20
UC B	The Beauty and Joy of Computing	F (S)	80
UC SD	Fluency with Information Technology	F (Q)	650
UNC C	The Beauty and Joy of Computing	S (S)	25
U W	Computer Science Principles	W (Q)	25

Cross-Campus Comparison II

	Programming Language	Lec	Lab	Dis	Wks	Total Contact
MSCD	Scratch, HTML/CSS	4	0	0	15	60
UC B	BYOB Scratch	2	4	1	14	98
UC SD	Alice, Excel	3	2	0	10	50
UNC C	BYOB Scratch	3	0	0	15	45
U W	Processing, XML/XSL	3	2	0	10	50

CS Principles: Catalog Entry

CSE120: **Computer Science Principles**

"Must-know computing knowledge for 21st Century"

Credits: 5

3 Lectures, 2 Labs (Closed)

Satisfies: Quantitative & Symbolic Reasoning Req

Pre-requisites: None

Follow-on Classes: None require it (yet)

Implementation of 7 Big Ideas and 6 Comp Practices

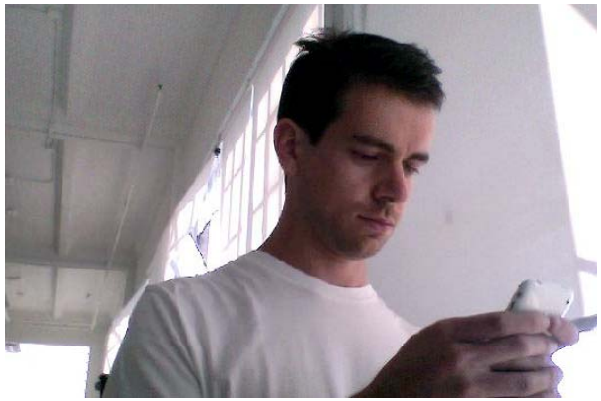
Thread 1: Principles, such as all information encoded in bits

Thread 2: Capabilities, such as CT, abstraction, programm'g

- http://www.cs.washington.edu/cse120/

Computer Science Principles

- Why take this class …?
  - It's not about using computers … you do that now
  - It's about changing your thinking, to envision how computing can work for you
    - Who's this guy???



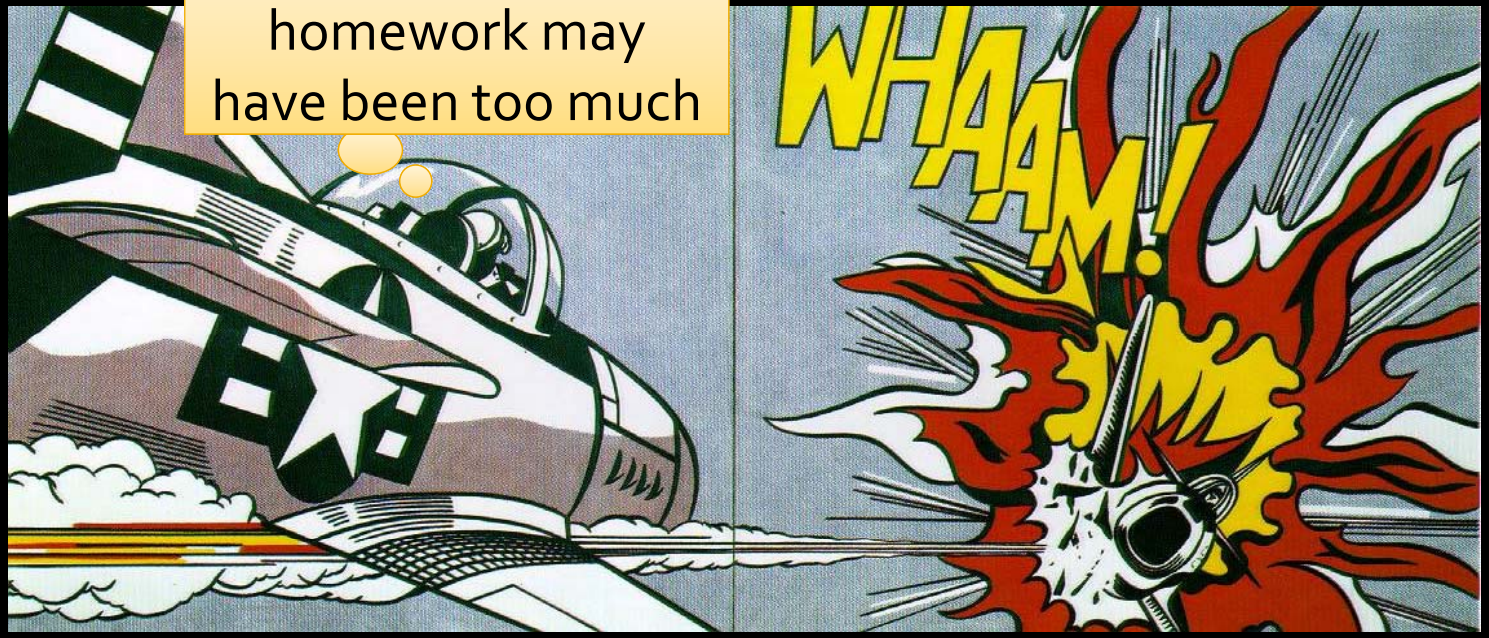
  - *Be able to come up with new idea to use computers*
  - *Understand someone else's new computing idea*
  - *Be a user on alpha-implementation when everything is breaking*

Select Class

- Using "by permission of instructor" I excluded students having taken CS
- The Class
  - 22 students
  - 11 Men, 11 Women
  - 12 "students of color"
    - 5 from under-represented groups
    - 1 Native American
    - 6 Asian (includes 2 international students)
  - Mostly pre-majors; about half intend tech majors

Challenges For Instructor

- Cover items on 7Big & 6CompThink lists
- Must be fundamental; must be doable
- Must be totally fascinating



Thread 1: Principles

- Principles covered in UW CSP –
  - **Bits**: sufficient to encode all information
  - **Binary**: like decimal but with radix 2, not 10
  - **Info**: physically is presence/absence of phenomenon
  - **Functional abstraction**: enables software layering
  - **Meta-data**: enables automatic processing of info
  - **TCP/IP**: like sending novel by postcards
  - **EtherNet**: like cocktail party chat
  - **Privacy**: right of people to decide the extent …
  - … Content people should know, direct from 7BI & 6CP

Thread2: Capabilities

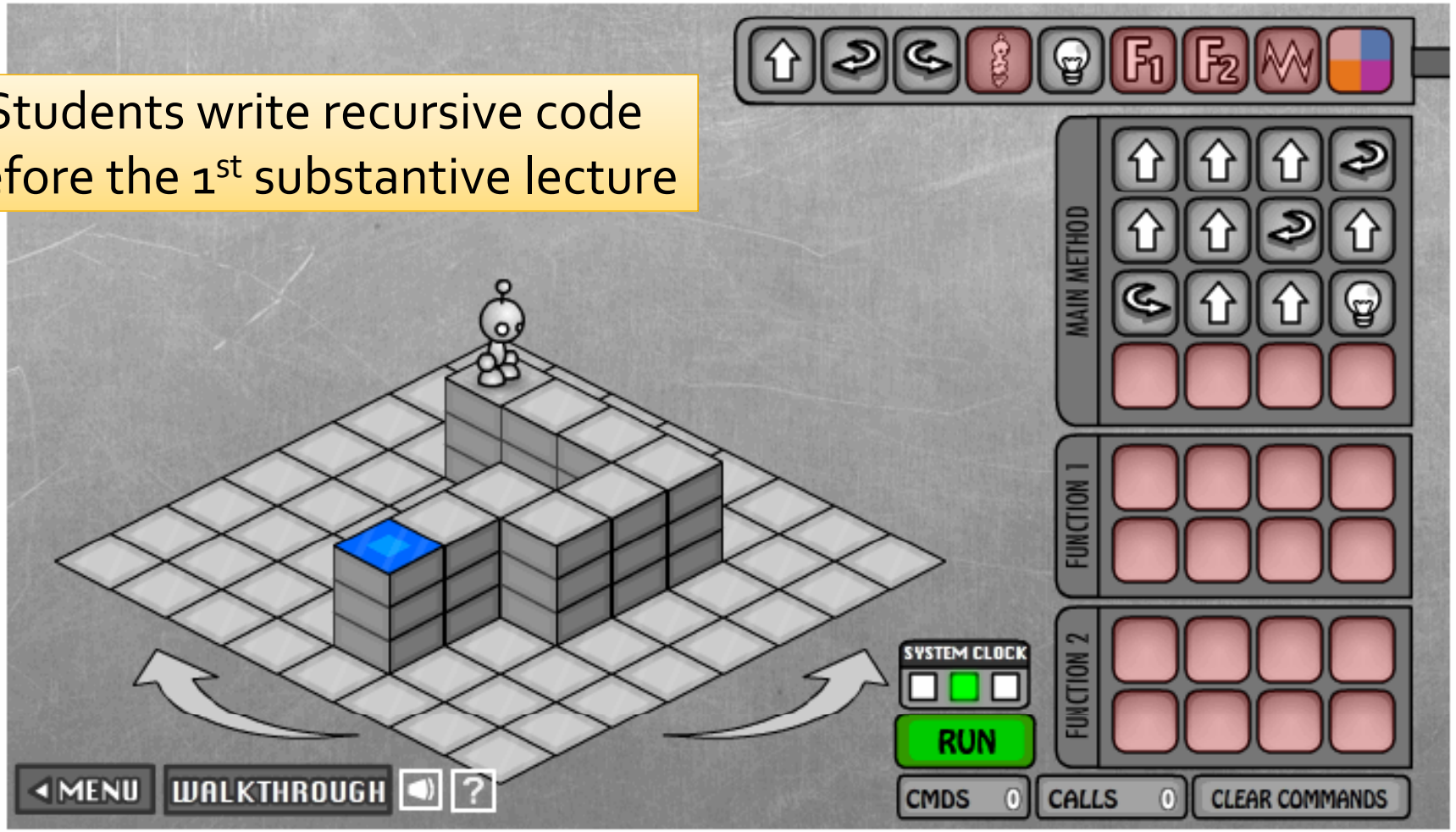
- What we do in UW CSP –
  - Programming in several forms
    - LightBot, an introduction to programming & recursion
    - Processing, graphic-centric design language
    - XML data structuring and personal database design
    - … also some HTML, CSS, Scratch and other software
  - Functional abstraction, recursion
  - Creating artifacts to implement personal intent
  - Repurpose tools for own use; pgmming by analogy
  - …

Practices that reinforce principles … be bold, creative, exploratory

Programming Experience

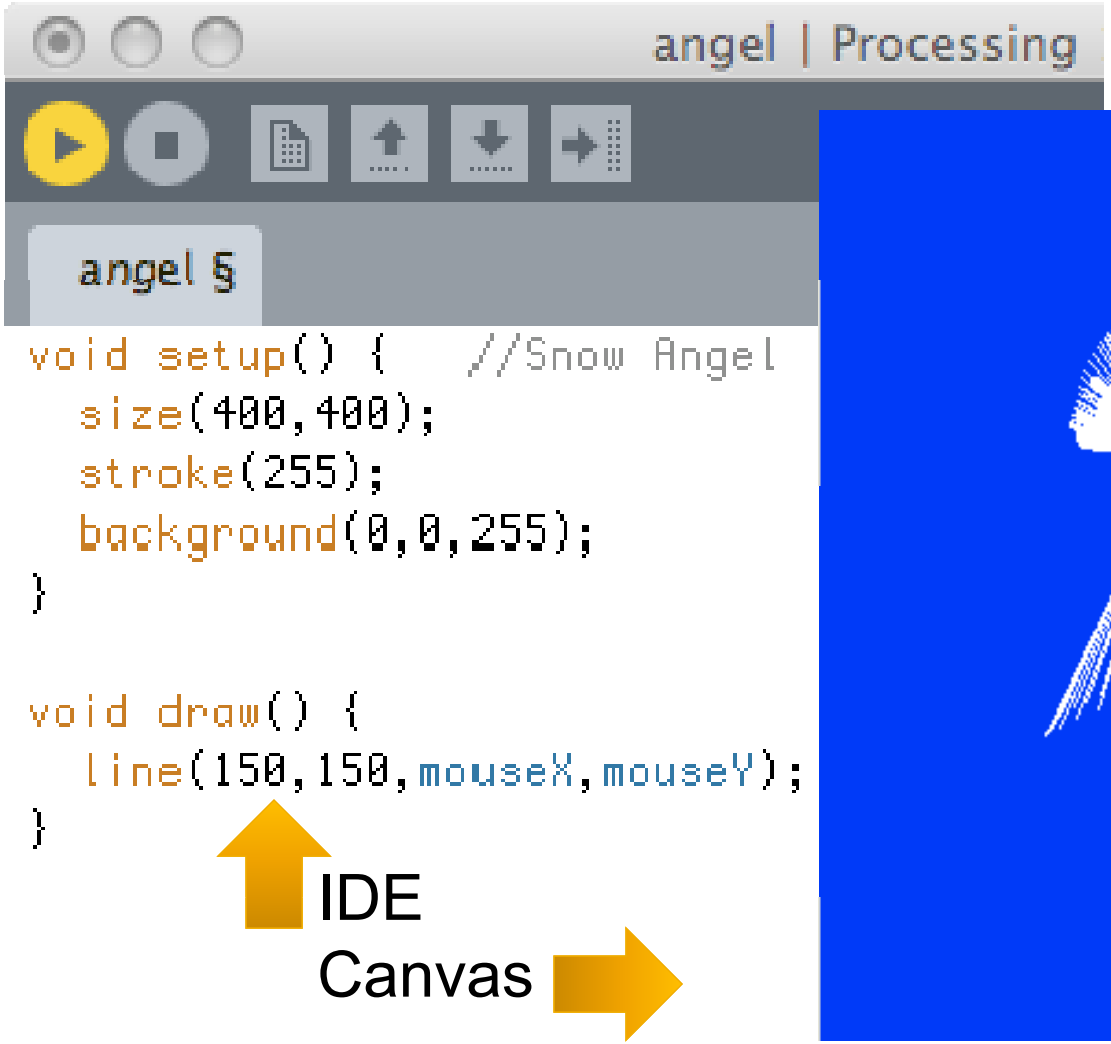
- Week 1: Lightbot … it's a fun game & it's prog'g

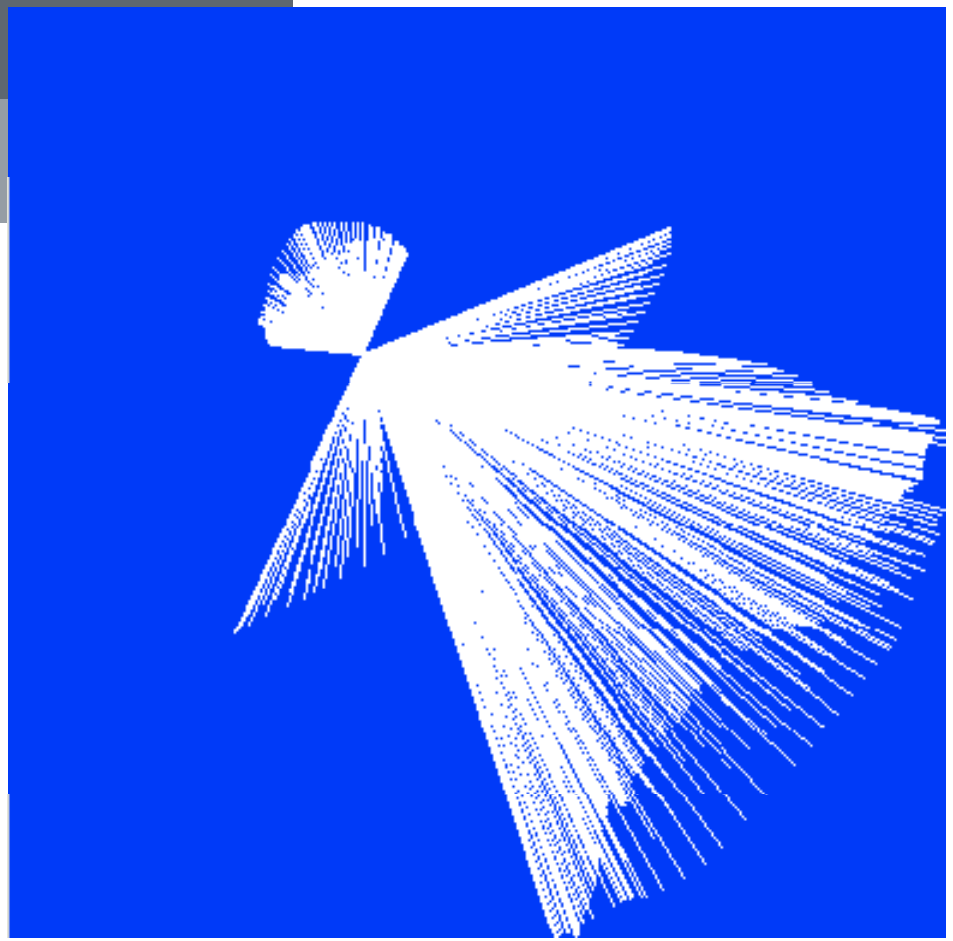
Students write recursive code before the 1st substantive lecture



Processing, the language

- Graphics prototyping language built on Java





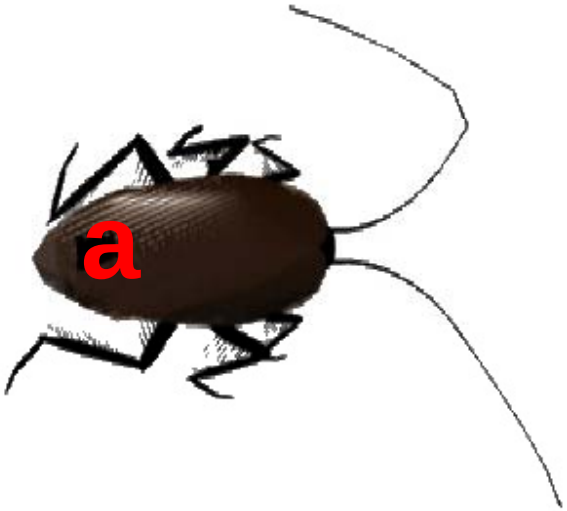
Processing, A Pedagogical Wonder

- Processing is …
  - "Totally fun!"
  - Free and trivial to install
  - Graphics are fun and trivial to do; interaction is trivial; text is actually harder
  - IDE is very forgiving
  - Trivially export a Web-embeddable version of code
  - All standard programming concepts available in standard form

Creativity

- Big Idea #1 is Creativity

– it was #1 in class



Steganography – A Lecture Topic

- The process of hiding information
- Two Greek roots meaning:

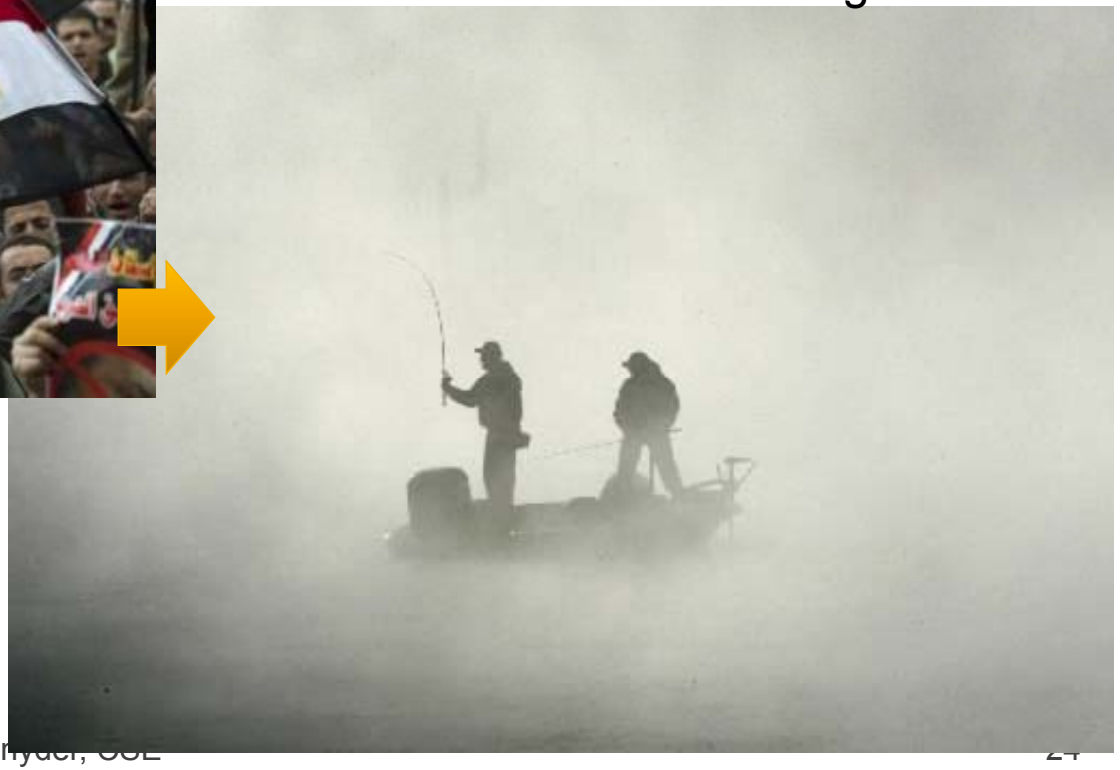
  "stego" == "roof" "stega" == "cover"

Illustrate A Way To Do It

- The Plan …
  - hide "subversive" protest picture in "calendar art"

Guest Image

Host Image



Step 1: Reduce Bits of Guest

- We don't need all of the bits in RGB to get a decent picture

All bits

1011 0100 1101 0011 0001 1100

Left 2 bits of each color

1011 0100 1101 0011 0001 1100

Step 2: Replace Bits In Host

- Put guest bits into right 2 bits of host

1111 0100 1101 0011 1011 1101

1000 0000 1100 0000 0000 0000

1111 0110 1101 0011 1011 1100

Compare fog.jpg with stegFog.png

fog.jpg

stegFog.png

Really?
Just Do It!

Teaching Under A Microscope

- Someone watched everything we did
- Each week students are asked to fill out the After Image Survey (AIS) – free form
  - What was engaging?
  - What worked?
  - What didn't work?
- Susan Evans (my HS teacher) summarized and sent me a "report card"
- Probably not scientifically reliable, but it's good to measure the "temperature" of class

Summary Bullets

- Students say: "I didn't expect to like it; I do!"
  "I didn't expect I could do it; I can!"
- Complaint: Assignments too long; "unclear"
- My Prob: Sequencing … too much adv'ed prep
- One Reward: Teaching CS ideas for own sake
  (contrast **Fluency with IT**, CS ideas you can use)
- Challenge: students work harder than they're used to … need to keep it interesting, fresh

Links

- All Class Stuff: http://www.cs.washington.edu/cse120/
- Dev Blog: http://csprinciples.cs.washington.edu/blog/