

What we know about teaching CS (Answer: Not all that much)

Mark Guzdial
School of Interactive Computing



1961 MIT Sloan School Symposium



COMPUTERS
and the world

Nothing amazes more human beings than computers. They are applicable to real business or ignorant tell, and believe, in their grip.

1
Scientists
and
Decision Making

Speaker	SIR CHARLES PERCY SNOW Author London, England
Discussants	ELTING E. MORISON Professor of Industrial History Massachusetts Institute of Technology NORBERT WIENER Institute Professor, Emeritus Massachusetts Institute of Technology
Moderator	HOWARD W. JOHNSON Dean and Professor of Industrial Management Massachusetts Institute of Technology

\$2.45

5
The
Computer
in the
University

Speaker	ALAN J. PERLIS Director of the Computation Center Carnegie Institute of Technology
Discussants	PETER ELIAS Head, Department of Electrical Engineering Professor of Electrical Engineering Massachusetts Institute of Technology J. C. R. LICKLIDER Vice President Bolt Beranek & Newman Inc.
Moderator	DONALD G. MARQUIS Professor of Industrial Management Massachusetts Institute of Technology



Learn Programming to Re-Think Process Everywhere

- Alan Perlis argued that computer science should be part of a liberal education.
 - Explicitly, he argued that all students should learn to program.
- Why?
 - Because Computer Science is the study of process.
 - Automated execution of process changes everything
 - Including how we think about things we already know





The Power and Fear of Algorithms

- The Economist (Sept., 2007) spoke to the algorithms that control us, yet we don't understand.
 - Credit Ratings, Adjustable Rate Mortgages, Search Rankings

And this is that decisions which are going to affect a great deal of our lives, indeed whether we live at all, will have to be taken or actually are being taken by extremely small numbers of people, who are normally scientists. The execution of these decisions has to be entrusted to people who do not quite understand what the depth of the argument is. That is one of the consequences of the lapse or gulf in communication between scientists and nonscientists.

There it is. A handful of people, having no relation to the will of society, having no communication with the rest of society, will be taking decisions in secret which are going to affect our lives in the deepest sense.

Economist.com

BUSINESS

Algorithms

Business by numbers

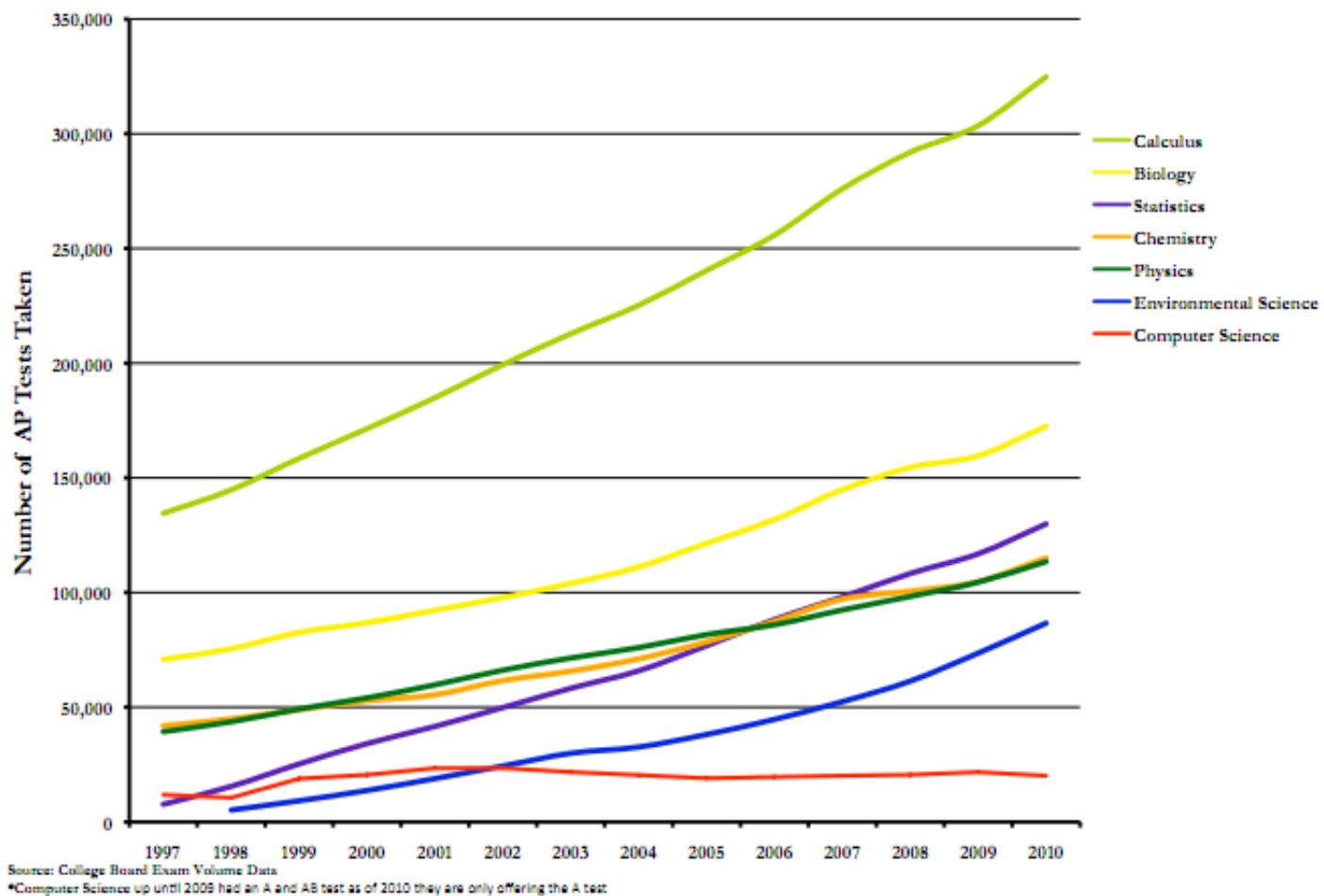
Sep 13th 2007

From The Economist print edition

Consumers and companies increasingly depend on a hidden mathematical world



High School Participation in AP STEM Disciplines



— Chris Stephenson, CSTA, 2010
Georgia Tech College of Computing



Need is greater than capacity

- *Computer Science has a much larger potential audience beyond software developers.*
 - Estimates:
 - ~13 million non-professional programmer/end-user programmers in US by 2012,
vs.
~3 million professional software developers (Scaffidi, Shaw, & Myers, 2005)
- Our track record for the first CS course is poor:
30-50% failure or withdrawal rates (Bennedsen & Caspersen, 2007)
 - Worse with female and URM students (Margolis & Fisher, 2003)



Story

- We want to teach computing to everyone, but *how*?
- Historically: CS is really hard to learn.
- What helps?
 1. Context
 2. Collaboration (Sometimes)
 3. Restructuring learning
 4. Fitting into the learner's life (We hope)



The Rainfall Problem

- Problem: Read in integers that represent daily rainfall, and printout the average daily rainfall.
 - If the input value of rainfall is less than zero is less than zero, prompt the user for a new rainfall.
- When you read in 99999, print out the average of the positive integers that were input other than 99999.



Results at Yale in Pascal in 1983

	% of Students who got it right
Novices (3/4 through first course)	14%
Intermediates (3/4 through second course)	36%
Advanced (Jrs and Srs in Systems Programming)	69%



Not an anomaly

- Elliot Soloway and his students replicated this study many times.
- Others have used this same problem with similar results (Most recently: Venable, Tan, and Lister, 2009)
- Anecdotally, every institution I've been at has attempted this problem, with similar results.



MIMN Studies

- “Is it just Yale? Is it just Yale and Pascal?”
- In 2001, Mike McCracken et al. gathered data from 216 students in 5 institutions in 4 countries: “Build a calculator.”
 - Out of a possible 110 points, average score was 22.89.



Lister's Response

- “Maybe the problem is design. Can they at least *read* code?”
- In 2004, Raymond Lister gathered even more students, from more institutions and countries.
- Overall score: About 40%

Question 5.

Consider the following code fragment:

```
int[ ] x = {0, 1, 2, 3};  
int temp;  
int i = 0;  
int j = x.length-1;  
  
while (i < j)  
{  
    temp = x[i];  
    x[i] = x[j];  
    x[j] = 2*temp;  
    i++;  
    j--;  
}
```

After this code is executed, array “x” contains the values:

- {3, 2, 2, 0}
- {0, 1, 2, 3}
- {3, 2, 1, 0}
- {0, 2, 4, 6}
- {6, 4, 2, 0}



Maybe these are just poor tests?

Table 16: Significant Pearson Correlations between Pseudo-code and CS1 Language Versions of the FCS1 Assessment

- Allison Elliott Tew built the first language-independent validated test of CS1 knowledge.
 - Domain chosen from review of CS1 textbooks.
 - Expert panel reviewed problems
 - Multiple iteration with think-aloud protocols
 - 950 subjects from two countries.

Population	df	<i>r</i>	<i>p</i>
<i>Total</i>	850	.572	<= 0.001
<i>Java</i>	74	.665	<= 0.001
<i>Matlab</i>	491	.547	<= 0.001
<i>Python</i>	285	.415	<= 0.001
<i>CS-Python</i>	43	.615	<= 0.001
<i>Media-Python</i>	242	.372	<= 0.001

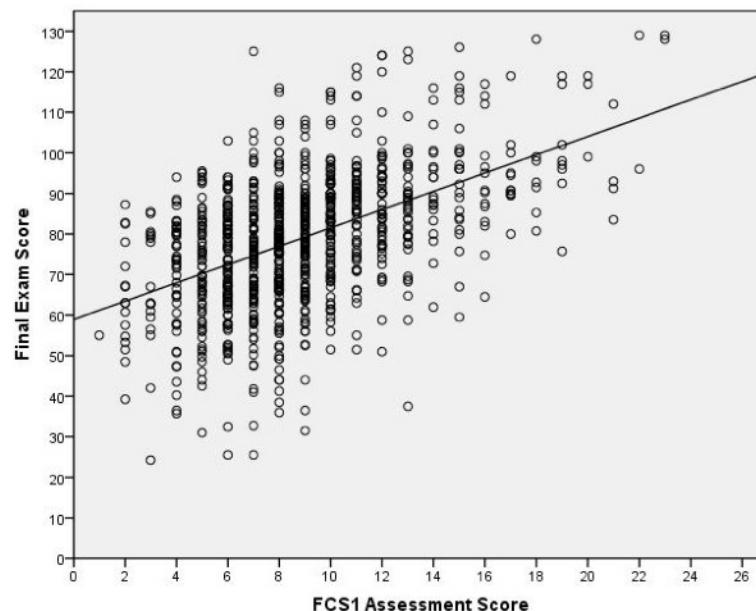


Figure 9: Scatterplot of Scores for Correlation of FCS1 Assessment Score and CS1 Final Exam Score



Performance on FCS1

- The majority of students did not pass.
 - The average score on the pseudo-code test was 33.78%.
 - The average score was 48.61% on the “native” language test.



We don't know why programming is so hard

- Specifying behavior for an alien agent (Resnick's MultiLogo).
 - We can specify instructions for humans (Pane, Commonsense Computing)
- Students are overwhelmed (Clancy & Linn, 1990):
 - “I don’t know where to start.”
 - “You’ve taught me so many details, I don’t know which ones to use.”
- Nothing much is changing (Scratch work from Technion, 2011)



Story #1: Context Matters

- *Fall 1999:*
 - All students at Georgia Tech must take a course in computer science.
 - Considered part of General Education, like mathematics, social science, humanities...
- 1999-2003: Only one course met the requirement.
 - Shackelford's pseudocode approach in 1999
 - Later Scheme: How to Design Programs (MIT Press)

One-class CS1: Pass (A, B, or C) vs.



WDF (Withdrawal, D or F)

Success Rates in CS1 from Fall 1999 to Spring 2002 (Overall: 78%)

Architecture	46.7%	.37%
Biology	64.4%	
Economics	53.5%	
History	46.5%	.63%
Management	48.5%	
Public Policy	47.9%	

Total Fall01 Females Fall01 Males Fall01

Total Sp02

Females Sp02

Males Sp02

Total Fall02

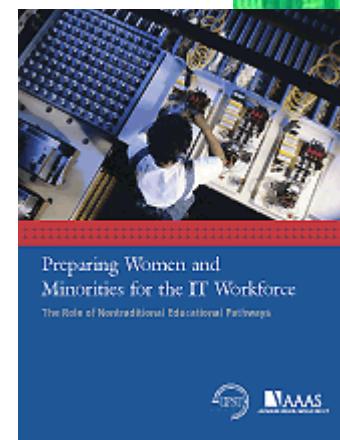
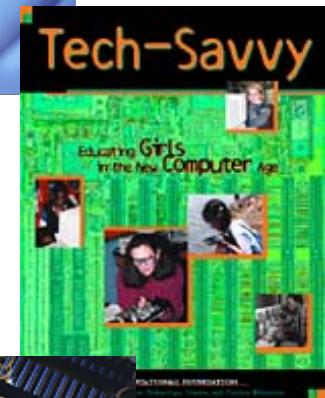
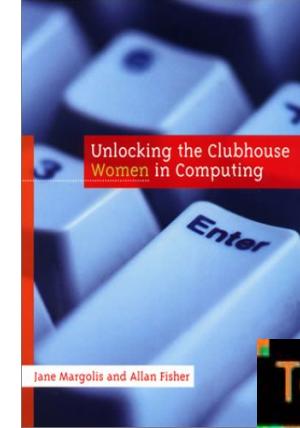
Females Fall02

Males Fall02



Contextualized Computing Education

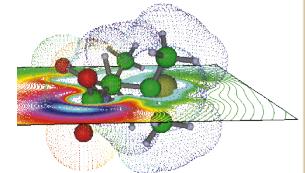
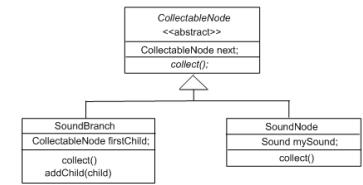
- What's going on?
 - Research results: Computing is “tedious, boring, irrelevant”
- Since Spring 2003, Georgia Tech teaches three introductory CS courses.
 - Based on Margolis and Fisher’s “alternative paths”
- Each course introduces computing using a context (examples, homework assignments, lecture discussion) relevant to majors.
 - Make computing relevant by teaching it in terms of what computers are good for (from the students’ perspective)





Our Three CS1's Today

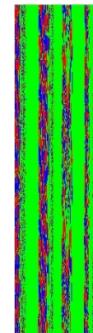
- CS1301/1321 *Introduction to Computing*
Traditional CS1 for our CS majors and Science majors (math, physics, psychology, etc.). Now, uses robots.
- CS1371 *Computing for Engineers*
CS1 for Engineers. Same topics as CS1301, but using MATLAB with Engineering problems.
- CS1315 *Introduction to Media Computation* for Architecture, Management, and Liberal Arts students.



Media Computation: Teaching in a Relevant Context



- Presenting CS topics with media projects and examples
 - Iteration as creating negative and grayscale images
 - Indexing in a range as removing redeye
 - Algorithms for blending both images and sounds
 - Linked lists as song fragments woven to make music
 - Information encodings as sound visualizations



```
def clearRed(picture):  
    for pixel in getPixels(picture):  
        setRed(pixel,0)
```



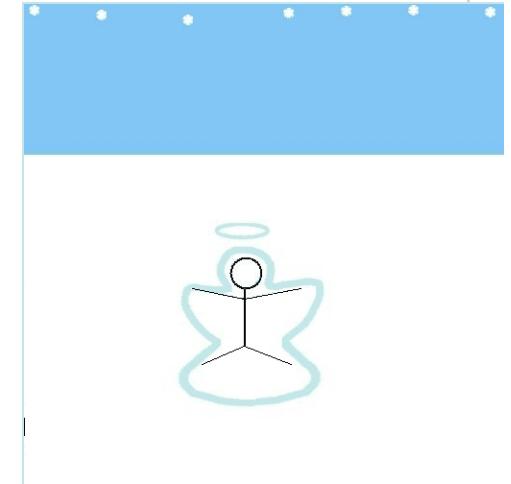
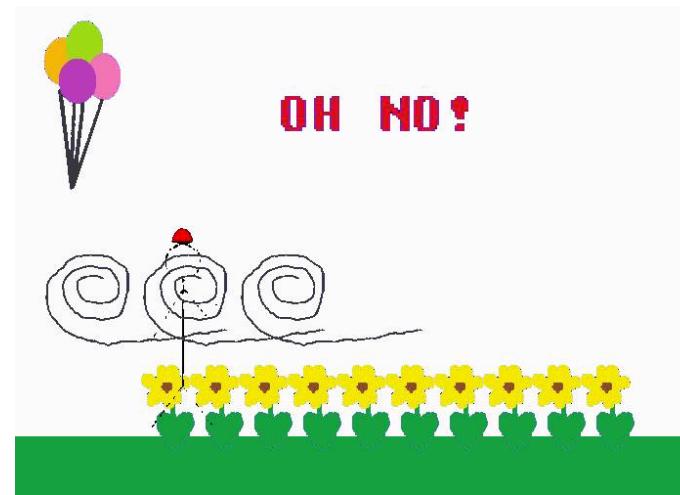
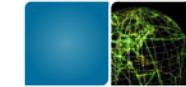
```
def greyscale(picture):  
    for p in getPixels(picture):  
        redness=getRed(p)  
        greenness=getGreen(p)  
        blueness=getBlue(p)  
        luminance=(redness+blueness+greenness)/3  
        setColor(p, makeColor(luminance,luminance,luminance))
```



```
def negative(picture):  
    for px in getPixels(picture):  
        red=getRed(px)  
        green=getGreen(px)  
        blue=getBlue(px)  
        negColor=makeColor(255-red,255-green,255-blue)  
        setColor(px,negColor)
```



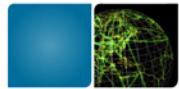
Open-ended, contextualized homework in Media Computation CS1 & CS2



Sound collage

Linked list as
canon





Results:CS1“Media Computation”

**Change in Success rates in CS1 “Media Computation” from Spring 2003 to Fall 2005
(Overall 85%)**

	WDF	Pass
Architecture	46.7%	85.7%
Biology	64.4%	90.4%
Economics	54.5%	92.0%
History	46.5%	67.6%
Management	48.5%	87.8%
Public Policy	47.9%	85.4%

■ WDF
■ Pass



Other examples where context helps

- Brian Dorn's 2010 thesis on teaching graphics designers computer science.
 - By “hiding” CS lessons in their case studies.
- Leahy et al. 2008 study of teaching computer organization with Gameboys.
 - No learning difference, big motivation difference and time-on-task

Script Development

The goal of this project is to automatically disable all of the text layers in a mockup. We'll begin by thinking about how we would accomplish this manually in the Photoshop interface. The process is pretty simple.

1. Consult each layer in the layer palette one at a time.
2. If a given layer is a text layer (i.e., the layer's icon shows the letter "T"), disable it by clicking on the eyeball icon at the left.

To translate this into a program, we need to first find out how to access the layers in our program. Using the Object-Model-Viewer, we can determine that the layers of the current document in Photoshop can be accessed using `app.activeDocument.layers`. This provides a reference to an `array` object containing each layer as an individual element. Consulting the properties for Layer objects in the Object Model Viewer, we can also discover how to disable a layer using the `.visible` property. Assigning a `boolean` value to this property will determine whether the corresponding layer is enabled or disabled in Photoshop.

Below is our first attempt at writing the code. We use a `definite loop` to iterate through each of the layers in the array (see lines 4–7). Then for each layer, we set the `visible` property to `false` in order to disable it (line 6).

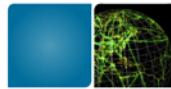
```
Script Version 1 [hide]  
1 #target photoshop  
2  
3 var inputLayers = app.activeDocument.layers;  
4 for (var i = 0; i < inputLayers.length; i++)  
5 {  
6     inputLayers[i].visible = false;  
7 }  
8
```





Story #2: Collaboration helps (sometimes)

- Based on earlier work with Jennifer Turns (2000).
- We showed that “anchored collaboration” resulted in longer, on-topic discussion threads.
 - Compared with course Usenet newsgroups.
- What if anyone could create an “anchor” or a “discussion space”?



Using the first Wiki for undergraduate learning

The screenshot shows two windows side-by-side. The left window is a Microsoft Internet Explorer browser titled "Sandbox - Microsoft Internet Explorer". It displays a simple wiki page with the title "Sandbox" and the text "Here's a place to play with Swikis." Below this, there is a list of links: "A Sample Page", "Naomi was here.", "Let's talk about ITICSE 2001.", "Mark Guzdial was here.", and "Mike Eisenberg was here.". An image of several ants on a leaf is displayed below the links. The right window is a Netscape Communicator browser titled "Netscape: Edit Sandbox". It shows the same wiki page with the title "Sandbox" and the same text. The edit mode is active, showing the raw HTML code and the history of edits. The code includes the original text, followed by a series of edits from users "Naomi", "Mark Guzdial", and "Mike Eisenberg", each adding their name and a small joke or comment.

Sandbox

Here's a place to play with Swikis.

[A Sample Page](#)

[Naomi](#) was here.

Let's talk about [ITICSE 2001](#).

[Mark Guzdial](#) was here.

[Mike Eisenberg](#) was here.



Back Forward Reload Home Search Netscape Images Print Security Shop Stop

Location: http://swiki.cc.gatech.edu:8080/edtech/520.edit What's Related

CS2340 CoWeb CS4660 Education AccessAtlanta - Mark Guzdial In Collaborative S Computer Music

view edit attach history home changes search help

Title: Sandbox

Last edited: 10:31:13 pm on 5 November 2001 by guzdial2.cc.gatech.edu

Here's a place to play with Swikis.

<hr>

Naomi was here.

Let's talk about *ITICSE 2001*.

Mark Guzdial was here.

Mike Eisenberg was here.

ants.jpg

What do you call a drunk rhinoceros?
A wino Rhino. --JimDavies

Elliot Soloway was here.



Uses of the CoWeb in CS Classes

- Galleries of student work
- “Organizational memory” for teams (and graders)
- Midterm and final exam reviews
- Glossaries
- Case libraries



[Hotspots: Readme First | CS2340 Class Administration Pages](#)

Sp2000 Midterm Review: O-O Systems

Review at [Midterm Review - Sp2000](#).

[Michael Emard](#) and [Michelle Burnett](#) bring you the following answer.

(a) A class is a general model of a real world object, such as a chair. An instance is a particular real world object, such as myChair or michellesChair.

(b) A class method is one that can be used without an instance of that particular class. An instance method is one that must be called upon through an instance. A good example would be a calculator class. You could have a class method add(num, num) that would return ... (you know what... if you don't know what this is going to return you are goofy). Anyway, you don't need an instance of calculator to use the add method. One could decide to use several calculators at once. Calculator could also contain items such as color. You could create a new instance of Calculator and set the color using an instance method.

(c) Class variables are not used very often but they can be explained with the calculator example. If you had a few instances of calculator and had color as an instance variable, all calculators could be different colors. If you declared color as a class variable all calculators would have to be the same color. Color could change, but it would change for every instance.

(d) I'm gonna take a guess here and say that prototype systems are like simula and sketchpad where you can have a master prototype and other objects can mimic this master. Class-based object systems bring inheritance and hierarchy subclassing into the picture.

IF EVERYBODY PUT ONE OR TWO ANSWERS ON THE CO-WEB THE MIDTERM WOULD BE MUCH EASIER.

(a) I'd like more detail than a definition, please. What does a class do? What is the relationship between an instance and a class? What does the instance get from the class?
(b) No, that's a Java explanation. What's the Smalltalk explanation?
(c) But what are the strengths and weaknesses?
(I completely agree about one or two answers, but you're going to need multiple answers to some of these.)
[Mark Guzzia](#)



Role of the Homework “Galleries”

Q: What do you think about the homework galleries on the CoWeb?

Student 4: It's nice to see other people, like what they did with it... And there is no better feeling than getting something done and knowing that you've done it right.

Student 3: I don't ever look at it [the homework gallery] until after I'm done. I have a thing about not wanting to copy someone else's ideas. I just wish I had more time to play around with that and make neat effects. But JES [IDE created for this class] *will be on my computer forever*, so... the nice thing about this class is that you could go as deep into the homework as you wanted. So, I'd turn it in, and then *me and my roommate would do more after to see what we could do with it*.

Anecdotes from Engineering and Math



- On a mandatory assignment involving a math class studying results from Engineering students' simulations, 40% of math students accepted a zero rather than collaborate with engineers.
- We provided an Equation Editor in the CoWeb for an Engineering and a Math course to facilitate talking about equations. Not a single student even *tried* the Editor.
- Changed the focus of our research: Why not participate?



Competition

- Student quotes on “Why didn’t you participate in CoWeb?”

“1) didn't want to get railed 2) with the curve it is better when your peers do badly”

“since it is a curved class most people don't want others to do well”

Note: Students claimed that the course grades were “curved” even when there was none.



Learned helplessness

- Student quotes:

“I haven't posted about questions because I am confident that my answers are wrong.”

“I thought I was the only one having problem understanding what was asked in the exam.”

“Who am I to post answers?”

“The overall environment for [this class] isn't a very help-oriented environment.”

Bottom line: Collaboration may not “just work” in Engineering/Math/CS – not without an explicit focus to make it work.



Story #3: Restructuring Instruction

- Educational psychologists really know something about learning & teaching.
- When everyone is demanding your classes, you don't have to sweat it.
- But to teach those that aren't *demanding* to get in, we *do* have to sweat it.



Helping students figure out “how”

- Provide worked examples (Anderson et al., 1984)
 - Label subgoals in worked examples (Catrambone, 1994; 1998)
 - Successful in other domains (e.g., statistics)
 - Hypothesis: Students remember the subgoals, and transfer those.
 - Subgoal labels groups steps of a worked example into a meaningful unit
 - Helps students differentiate between structural and incidental information
 - Provide framework for general mental model of process



Employing Subgoal Labels in CS Ed

- Used these techniques to teach Android App Inventor to programming novices
- 2 groups (N=40, No App Inventor experience):
 - Original/Conventional (created by Barbara Ericson)
 - Subgoal-labeled
- Materials were identical except for the subgoal labels



Experiment Procedure

- 2 hour-long sessions
- Session 1
 - 5 min – demographic questionnaire
 - 40 min – instructional period for first app
 - 15 min – assessment 1
- Session 2
 - 10 min – assessment 2
 - 25 min – instructional period for second app
 - 25 min – assessment 3
 - 15 minutes to answer questions
 - 10 minutes to make a new (third) app to specification



Example of Written Materials

Subgoal

Define Variables from Built-in

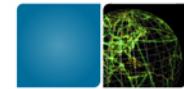
- Click on "Built-In" and "Definition" and pull out a *def variable*.
- Click on the "variable" and replace it with "fortuneList". This creates a variable called "fortuneList".
- Click on "Lists" and drag out a *call make a list*
- Click on "Text" and drag out a *text text* block and drop it next to "item". Click on the rightmost "text" and replace it with your first fortune.

Handle Events from My Blocks

- Click on "My Blocks" and "Button1".
- Drag out a *when Button1.Click*.

Non-subgoal

- Click on "Built-In" and "Definition" and pull out a *def variable*.
- Click on the "variable" and replace it with "fortuneList". This creates a variable called "fortuneList".
- Click on "Lists" and drag out a *call make a list*
- Click on "Text" and drag out a *text text* block and drop it next to "item".
- Click on the rightmost "text" and replace it with your first fortune.
- Click on "My Blocks" and "Button1".
- Drag out a *when Button1.Click*.



Original/Conventional Video





Video with Subgoals

The screenshot shows the App Inventor for Android web interface. At the top, there's a toolbar with various links like Favorites, MOTODEV Documentation, Free Hotmail, Loops, MSN, and Google. Below the toolbar, the main header has the App Inventor logo and navigation links for My Projects, Design, and Learn. On the right, there's a status update message: "App Inventor Status Update: There is a status update on the App Inventor open source transition to MIT. See [this announcement](#) for more." Below the header is a green navigation bar with buttons for New, Delete, Download All Projects, and More Actions. The main content area is titled "Projects" and lists a table of projects. The columns are "Name" and "Date Created". The projects listed are:

Name	Date Created
Fortune_copy	Dec 2, 2011 2:33:23 PM
GroupMeal	Aug 7, 2011 10:03:53 AM
HelloPurr	Apr 2, 2011 3:44:33 PM
Matching	Sep 29, 2011 9:18:14 AM
Mole	Jul 18, 2011 11:21:41 AM
MoleMash	Jun 9, 2011 10:22:57 PM
Old_MusicMaker_copy	Nov 2, 2011 11:53:26 AM
PaintPot	Apr 9, 2011 9:06:03 AM
PressTheButton	Nov 9, 2011 12:20:09 PM
QuizMe	Jul 19, 2011 10:59:05 AM
ShakeDance	Jul 12, 2011 11:21:42 AM
cowbell	Nov 30, 2011 2:47:00 PM
cowbellAu	Nov 18, 2011 6:46:49 PM
cowbell_copy	Nov 12, 2011 2:39:35 PM
cowbell_copy2	Nov 30, 2011 2:31:50 PM
draw	Jul 17, 2011 12:25:01 PM

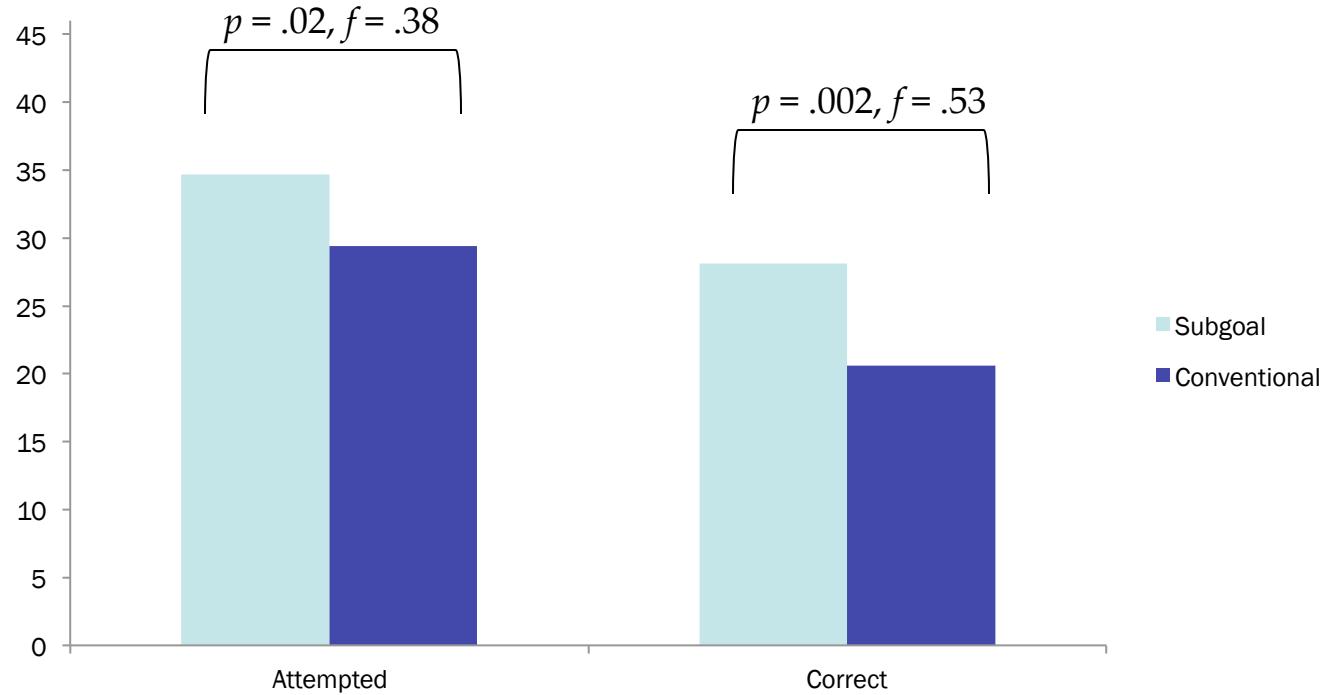


Scoring

- Answers scored on 2 dimensions
 - Subgoals ATTEMPTED
 - Subgoals CORRECT
- 46 subgoals across the 3 assessments
- Interrater reliability
 - $ICC(A) = .97$, Cronbach's alpha = .98, $r = .96$

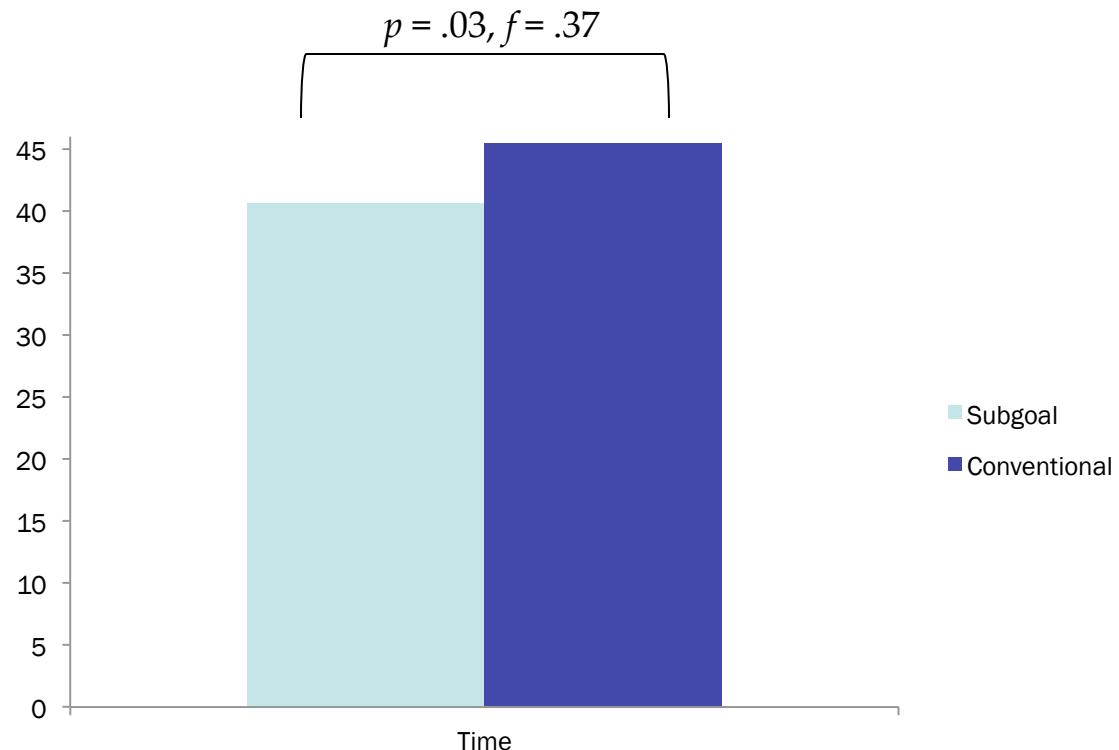


Results: Overall





Results: Time

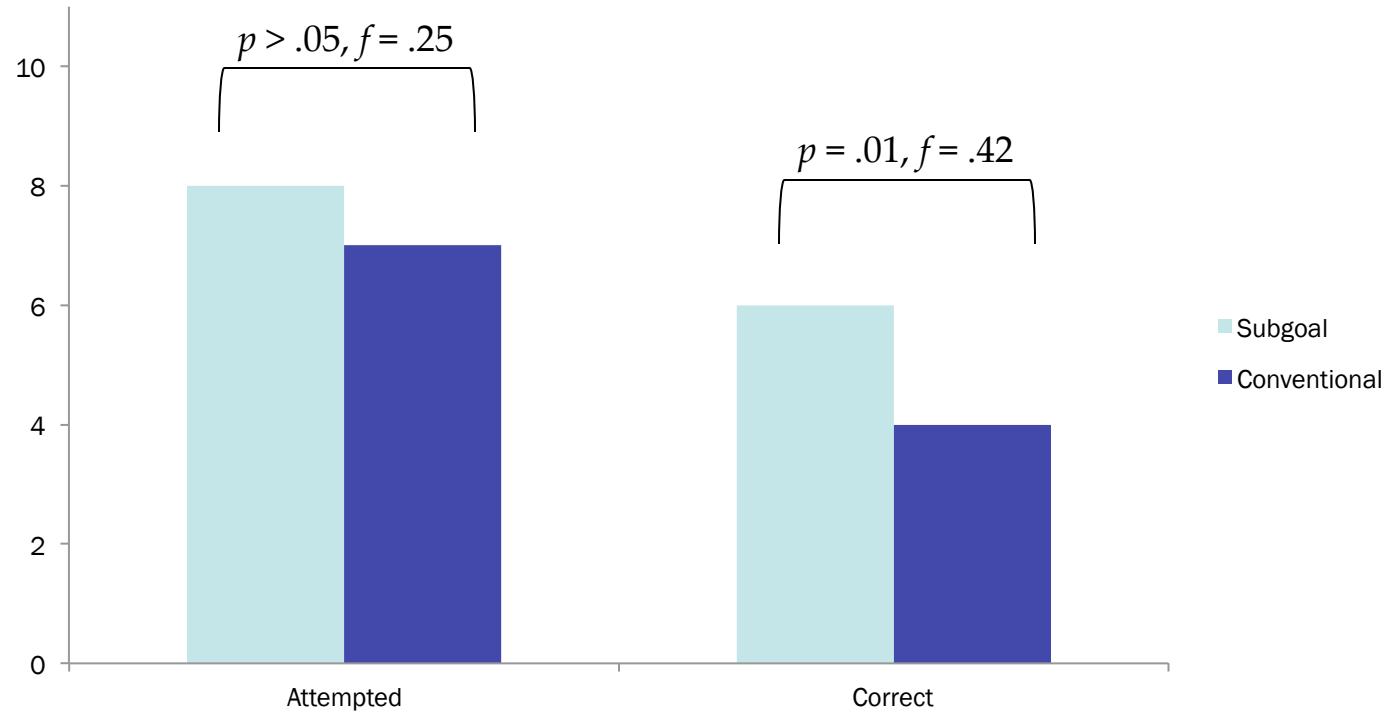


- r between time and correct = .06

What this means: Subgoal group was faster, but faster didn't mean that they did worse.

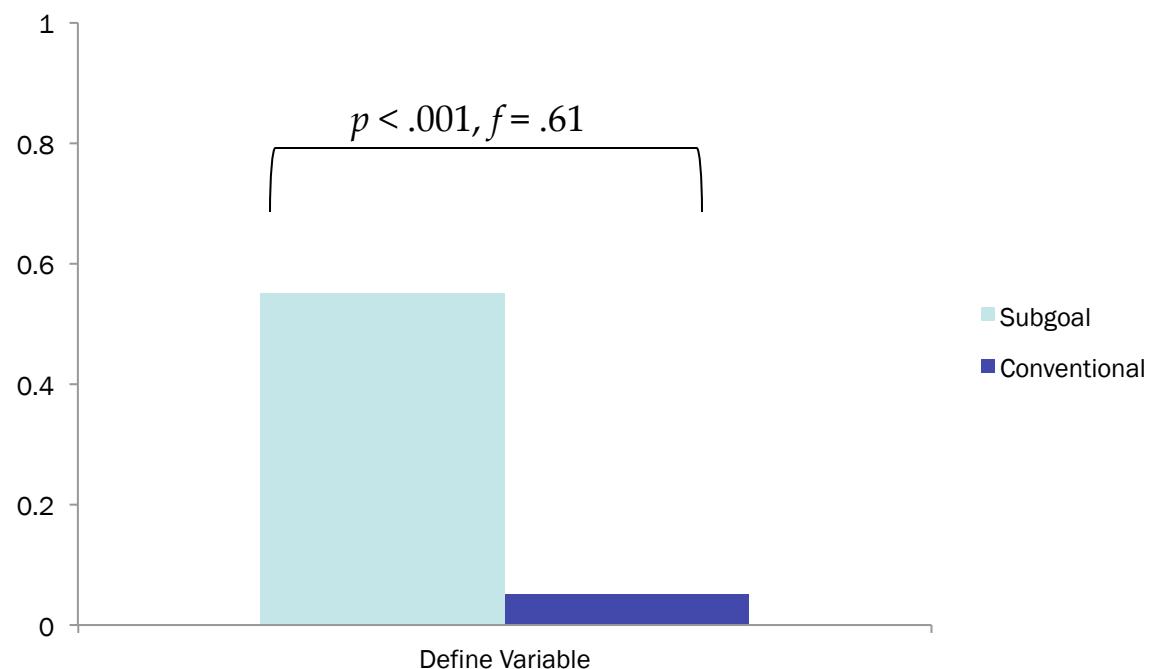


Results: Retention Assessment





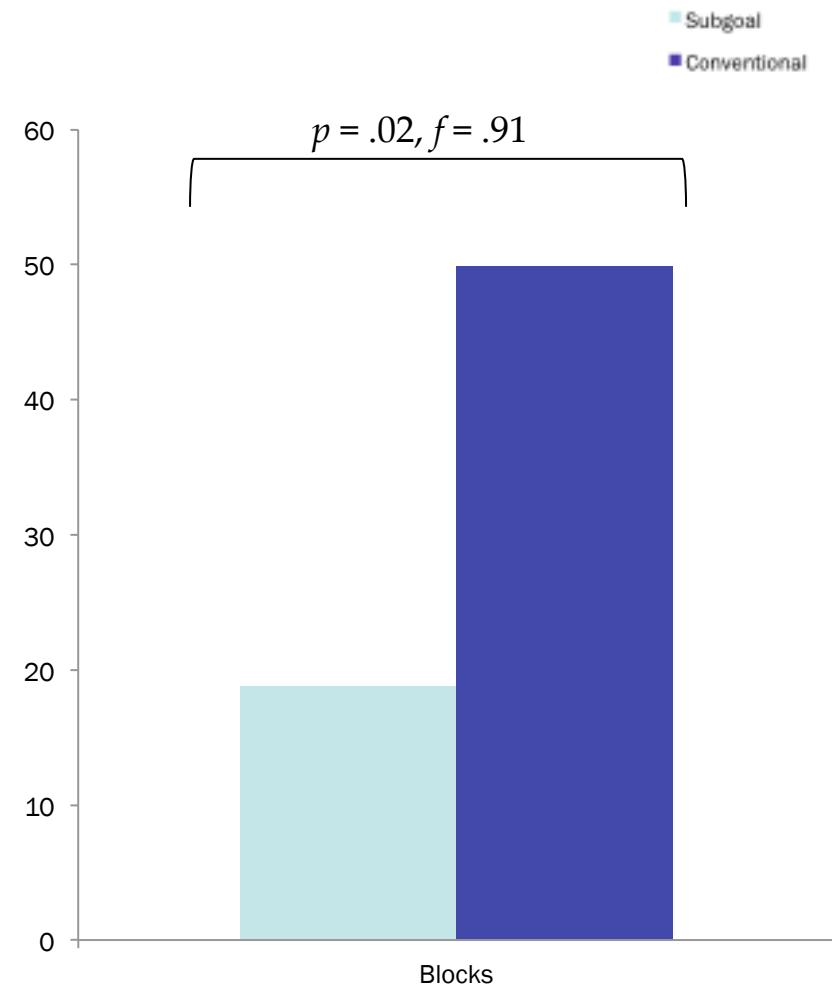
Results: Define Variable Step in Transfer Task

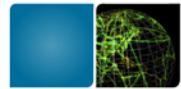




Results: Less thrashing

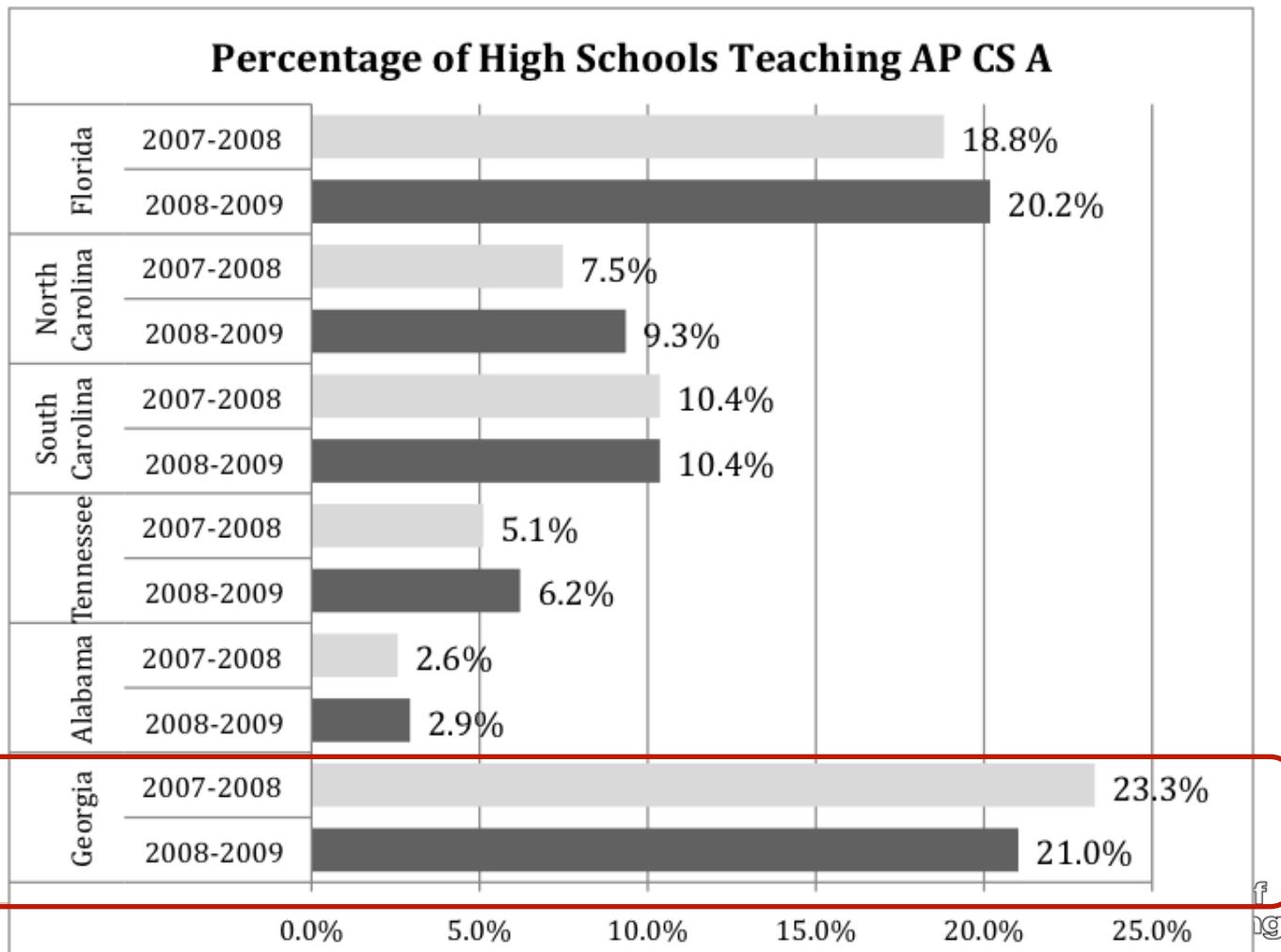
- Side observation:
Less thrashing.
 - Participants in the subgoal group were less likely to drag out blocks while working through assessment tasks ($p < 0.02$)
- r between correctness and #blocks = -.349





Story #4: Fit into their lives

- Where are we going to get enough high school CS teachers?





Teachers feel a need for more training

- [Becky]: “*I struggle with giving everyone the material and being able to explain it to everyone... I struggle with how to be creative with the programming. I have a problem with trying to make the programs have meaning to them... It is hard to teach. It's hard knowing how to teach it, how to give it to them... It's hard to explain. When I look at kid's codes, they think I should know it... They think that I should know it as soon as I look at it. For the longest time I thought I should, but I don't have to. I have to study it just like they do. So, I would like some training.*”

From Lijun Ni's 2011
thesis on CS teacher
identity



Easier with Support

- Confidence: More confident in teaching Math

[John]: “I still think I’m a better Math teacher, just because I’ve had so much support. Whenever I have problems, I can talk with the people that I work with, most of who have taught for many years in Math... With Computer Science, I’ve got nobody to talk to.”



They may find CS inaccessible

- [May]: “I think, computer science is more for really, really smart people. I’m not saying I’m smart, but I’m thinking that if I have to go take this Computer Science degree, that it’s going to be really hard, because it’s going to ask a lot of programming questions, syntax questions. I think computer science is a much higher level... When I say computing, I think of computing as being able to operate the computer, ... I believe that most students can successfully take and complete Computing in the Modern World, but it takes a little higher level of intelligence to complete the Introduction to Programming

How do we teach working high school teachers?



- Study of adult/professional students in CS classes (i.e., apprenticeship-model).
 - They don't have the time to spend hours in front of the IDE.
 - Lacking background, e.g., in mathematics.
 - They get stymied by small errors.

When Life and Learning Do Not Fit: Challenges of Workload and Communication in Introductory Computer Science Online

KLARA BENDA, Georgia Institute of Technology

AMY BRUCKMAN, Georgia Institute of Technology

MARK GUZDIAL, Georgia Institute of Technology



"I had my few afternoon hours that I could work on the stuff, but it all just boiled down to me not having time for my family when I was taking the courses. I think the bottom line was with my family structure, I shouldn't have taken more than one course at once." [...] "sometimes I felt like I wasn't putting enough into one class because I was putting so much into the other class." [...] "Then I had to put more time into the family, because I didn't put in as much as I should have, but I still had to put time in for them."

Andrew - "I said one time that I couldn't get this mathematical problem to work. His response was, "I'm not going to teach you algebra." So if you get one little piece or spacing wrong, it doesn't work."

John – "There were times that it would take me hours to find one comma out of place, or find that one something that was wrong, so I didn't mind sticking with it but it just got to the point where I just didn't get it."



Trying to teach CS more Efficiently

- Using a model of a book, but with live code and Ed Psych principles:

Drag from here

Construct your solution here

```
return curmax
curmax = item
```

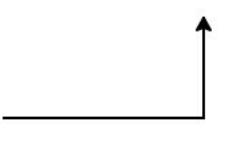
```
def findmax(alist):
    if len(alist) == 0:
        return None

    curmax = alist[0]
    for item in alist:
        if item > curmax:
            curmax = item

    return curmax
```

```
1 import turtle      # allows us to use
2 wn = turtle.Screen()  # creates a graphic
3 alex = turtle.Turtle() # create a turtle named alex
4 alex.forward(150)    # tell alex to move forward by 150
5 alex.left(90)       # turn by 90 degree
6 alex.forward(75)    # complete the second side of a rectangle
7
8
```

Run



Source Code

```
1 for i in range(10):
2     print(i)
```

<< First < Back Step 7 of 22 Forward > Last >>

Global variables:

i	2
---	---

Program output:

```
0
1
2
```

ActiveCode: 1 (ch03_1)

College of
Computing



Conclusions

- We want everyone to learn to program, and it's surprisingly hard.
We don't know why, but we know some things that help and some challenges.
- Story #1: Contexts help.
- Story #2: Collaboration helps, but not always.
- Story #3: Restructuring instruction helps.
- Story #4: We need to figure out how to teach CS in ways that fit into peoples lives.



With thanks to our supporters

- US National Science Foundation
 - Statewide BPC Alliance: Project “Georgia Computes!” <http://www.gacomputes.org>
 - CCLI and CPATH Grants, and now CE21 to produce new media
- Microsoft Research
- Georgia Tech's College of Computing
- Georgia's Department of Education
- GVU Center,
- GT President's Undergraduate Research Award,
- Toyota Foundation

Thank you!

- <http://www.cc.gatech.edu/~mark.guzdial>

<http://home.cc.gatech.edu/csl>

<http://www.georgiacomputes.org>

For more on CSLearning4U:

- <http://home.cc.gatech.edu/csl/CSLearning4U>

For more on MediaComp approach:

- <http://www.mediacomputation.org>

