



Technology for Teaching the Rest of Us

Mark Guzdial
School of Interactive Computing

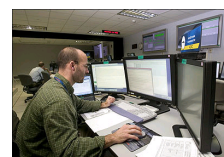
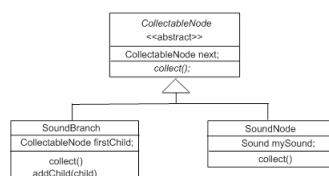
Georgia Tech College of Computing

Story

- Computing is important for more than just those who choose to major in computing.
- Who are “they” (“the rest of us”)? What do they want from CS, and why aren’t they in our classes? How do they learn CS?
- Teaching those who do *not* want to become software engineers or computer scientists
 - The Story of Computing for All at Georgia Tech
- Three Roles for EAAI researchers, developers, teachers:
 1. Matching or tailoring the context to the person.
 2. How to find what they *really* want and need.
 3. Teaching CS concepts without skills.

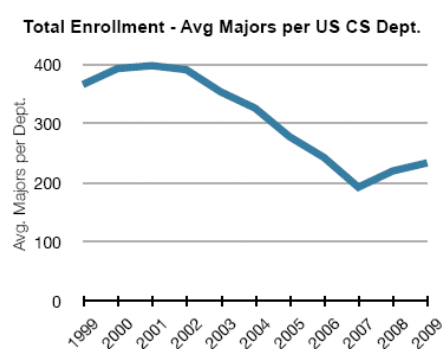
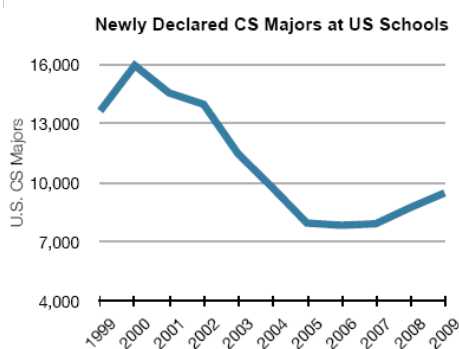
The typical CS student: Future Software Engineer

- To produce reliable, robust, secure software.
- To work in interdisciplinary teams.
- To use appropriate design notations, such as UML.
- To work in multiple programming languages.



Georgia Tech College of Computing

Latest Taulbee Numbers



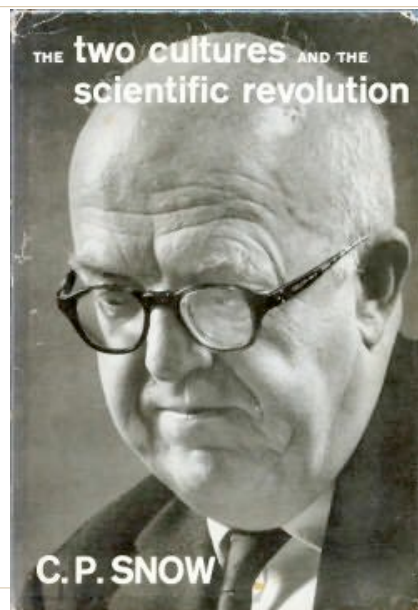
2010 CRA Taulbee Survey of PhD-granting institutions

Georgia Tech College of Computing

Who wants what CS has to offer?

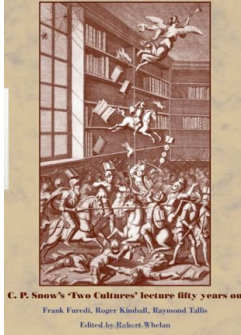
- Computing is at the core of the modern society and modern economy.
- Many people recognize the value of education in computing.
 - Including many of our students.
- However, only a few of them want to become professional software engineers.
 - Why? Complex question.
- ***Computer Science has a much larger potential audience elsewhere.***
 - Estimates: ~13 million non-professional programmer/end-user programmers in US by 2012, vs. ~3 million professional software developers (Scaffidi, Shaw, & Myers, 2005)

The Two Cultures

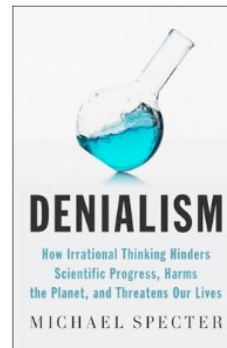


Two Cultures, Continued

From *TWO CULTURES* To *NO CULTURE*

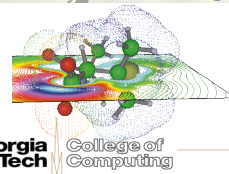
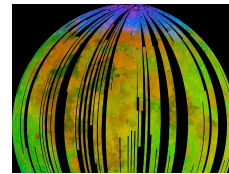


As late as my own childhood in the Sixties, the bright boys were expected to read classics at Oxford, and the less bright steered towards the labs. The Nobel Prize-winning scientist Sir Andrew Huxley recounts that, when he switched from classics to physics, the headmaster of Westminster School accused him of "forsaking virtue for pleasure".



An atypical CS student: Future computational scientist or engineer

- To use computation as a tool to enhance understanding.
- To write programs of (at most) 100 lines (most often, 10 lines) for themselves.
 - They care about the products of the programs, not the programs.
- To learn as few languages as are needed for their tasks.
- To work in interdisciplinary teams, including software engineers.



An atypical CS student: Future high school CS teacher

- To use code to explore and understand ideas of computation.
- To learn what languages are necessary to meet standards and engage students.
- To work with students with a wide range of interests.
 - Probably won't work with professional software engineers



Georgia
Tech

Co
Co

An atypical CS student: Future graphics designer

- To write programs to improve their efficiency, and to implement their dynamic (e.g., Web) designs.
- To do as little coding as possible.
- To learn about computing ideas in order to improve their process, but with a focus on people and creativity.
 - Probably won't work with professional software engineers



Georgia
Tech

College of
Computing

How do meet this need?

- Our track record for the first CS course is poor.
 - 30-50% failure or withdrawal rates (Bennedsen & Caspersen, 2007)
- Other majors tend to be more female and more ethnically diverse than the typical computing student.
 - Our track record with these audiences is particularly poor (Margolis & Fisher, 2003)

Three Pieces, Three Roles for EAAI

- First, the role of context in teaching computing.
 - Role #1: Help students find the context that makes sense to them.
- Second, why aren't the "rest of us" looking to our classes in CS? Where *are* they looking?
 - Role #2: Help amateur programmers find the help they need.
- Third, creating the new AP CS, without the burden of programming.
 - Role #3: Create new kinds of educational content.

Piece 1: Teaching Computing to Everyone



- **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%
Biology	64.4%
Economics	53.5%
History	46.5%
Management	48.5%
Public Policy	47.9%
Total	78%
	Fall02

Contextualized Computing Education

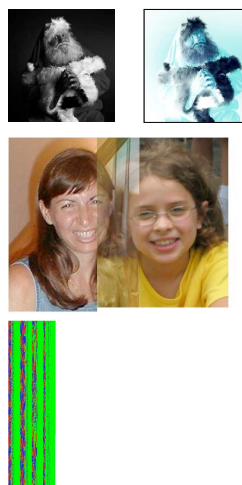
- What's going on?
 - Research results: Computing is “tedious, boring, irrelevant”
- Since Spring 2003, Georgia Tech teaches three introductory CS courses.
 - Based on Margolis and Fisher’s “alternative paths”
- Each course introduces computing using a context (examples, homework assignments, lecture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



Georgia Tech College of Computing

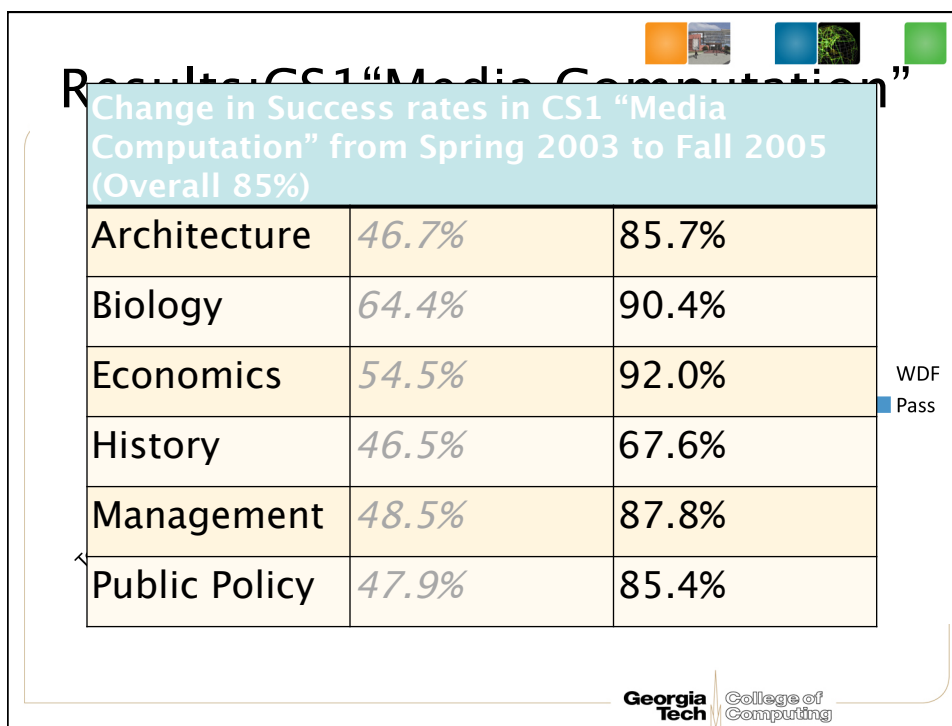
Media Computation: Teaching in a Relevant Context

- Presenting CS topics with media projects and examples
 - Iteration as creating negative and grayscale images
 - Indexing in a range as removing reye
 - Algorithms for blending both images and sounds (weighted sums) and for scaling (sampling).
 - Information encodings as sound visualizations



Georgia Tech College of Computing

16



Voices from Media Computation Students

- Intl Affairs student (female): "I just wish I had more time to play around with that and make neat effects. But JES [IDE for class] will be on my computer forever, so... that's 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."
- "I dreaded CS, but ALL of the topics thus far have been applicable to my future career (& personal) plans—there isn't anything I don't like about this class!!!"
- "Media Computation is a CS class but with less severity. The media part of the class is extremely visually interesting. I would only take another CS class if it were Media Computation."

Georgia Tech College of Computing

Survey One Year Later

- 19% of respondents had programmed since class ended

"Did the class change how you interact with computers?"

- 80% say "Yes"
- "Definitely makes me think of what is going on behind the scenes of such programs like Photoshop and Illustrator."
- 'I understand technological concepts more easily now; I am more willing and able to experience new things with computers now'
- 'I have learned more about the big picture behind computer science and programming. This has helped me to figure out how to use programs that I've never used before.'

Results at Gainesville College

- Similar results at a 2 year public college.
- What is *relevance*?
 - Not useful for *degree*.
 - Somewhat useful for *career*.
 - Mostly useful for *life*.
- Would you like more CS?
 - GT 15.2% "Strongly Disagree." <25% agree.
 - More MediaComp? GT and Gainesville over 40% agree.

Table 2. Success rates at Gainesville College before and with Media Computation class.

	ENROLLMENT	SUCCESS RATE
Gainesville's CSCI 1100		
Average 2000 - 2003	28	70.2%
Media Computation		
Summer 2003	9	77.8%
Fall 2003	39	84.6%
Spring 2004	22	77.3%
Summer 2004	11	90.9%

Table 11. Skills from this Class will be Useful in

	GEORGIA TECH		GAINESVILLE	
	Life	Career	Life	Career
Strongly Agree	12.6%	6.8%	6.2%	0.0%
Agree	47.3%	38.7%	50.0%	37.5%
Neutral	23.9%	31.5%	12.5%	25.0%
Disagree	13.1%	16.2%	25.0%	37.5%
Strongly Disagree	3.2%	6.8%	6.2%	0.0%

(Tew, Fowler, Guzdial, SIGCSE 2005)

Results at U. Illinois–Chicago

(Sloan and Troy,
SIGCSE 2008)

- A CS 0.5 (for CS majors not ready for CS1) using MediaComp Python.
- Improvements in success rates.
 - Even with a more diverse population.

Our school's CS 0.5	Enrollment	Success Rate
Fall 2002	61	74.8%
Spring 2003	38	76.7%
Fall 2003	51	68.6%
Spring 2004	22	82.9%
Fall 2004	15	93.3%
Average "Old"	37	75.9%
New Spring 2005	18	94.4%
New Fall 2005	29	90.0%
New Fall 2006	42	76.2%
New Spring 2007	24	83.3%
Average "New"	28.3	84.1%

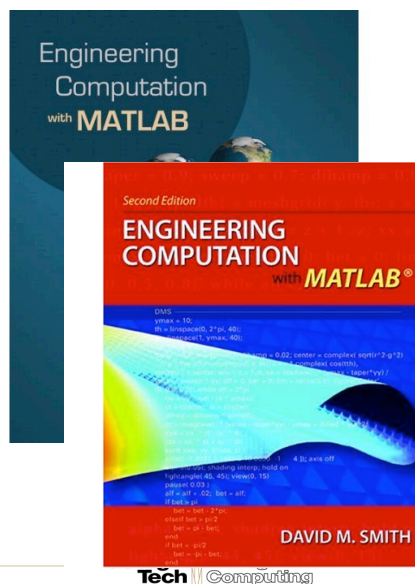
Table 1: Success rate with CS 0.5 before and with new approach. Averages weighted by enrollment.

Table 3: Ethnicity of Survey Participants in Media Computation. Data is from 2003 at Gainesville and Georgia Tech, and from Spring 2005 at our school.

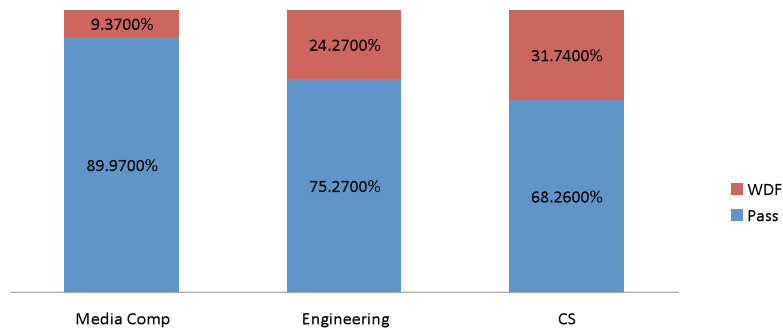
	Georgia Tech	Gainesville	Our school
African-Am.	6.4%	0	11.1%
Asian	0	7.0%	27.8%
Caucasian	80.8%	96.2%	44.4%
Hispanic	0.3%	0	16.7%
Other	5.4%	3.8%	0

Introducing Computing in an Engineering Context

- Developed in collaboration with Civil, Mechanical, and Aerospace Engineering.
- Uses Engineering problems and MATLAB
- Covers traditional CS1 topics
- Among our 3 CS1's, these are the first students to program outside of class.
- The success rate in this class also rose compared to all-in-one.



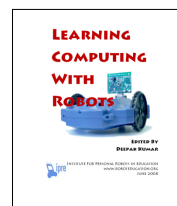
Comparing Success Rates Spring 2004



Georgia Tech College of Computing

A Context for CS1 for CS majors: Robotics

- Microsoft Research has funded the Institute for Personal Robotics in Education
 - Tucker Balch, Deepak Kumar, Doug Blank
 - Joint between Bryn Mawr and Georgia Tech
 - <http://www.roboteducation.org>
- Developed a CS1 with robotics as the context.
 - Includes a camera and media computation functions



Georgia Tech College of Computing

Results at UCSD

- Using Java Media Computation as normal CS1 for CS majors at a research university.
- Did extensive data collection last semester before switching to Media Computation.
- Been following two cohorts of CS1 students for comparison.

Experience Report:
CS1 for Majors with Media Computation
 Beth Simon, Päävi Kinnunen, Leo Porter
 Computer Science and Engr. Dept
 University of California, San Diego
 La Jolla, CA 92093-0404
 +01 858 534 5419
 [bsimon, pkinnunen, leporter]@cs.ucsd.edu

Dov Zakris
 Math and Science Education
 San Diego State University/
 University of California, San Diego
 San Diego, CA 92120
 +01 619 594 4996
 dzakris@ucsd.edu

ITICSE 2010 (not yet in ACM DL)

Findings:

- MediaComp has more focus on problem-solving, less on language.
- MediaComp students have higher pass rates and retention rates one year later

Georgia Tech College of Computing

Simon et al. Key Findings Summarized

Table 1. Course Grade Comparison

	Traditional	Media Computation
Winter Term Course Grades	\bar{x} = 79.9% s.d. = 12.1	\bar{x} = 83.4% s.d. = 14.0

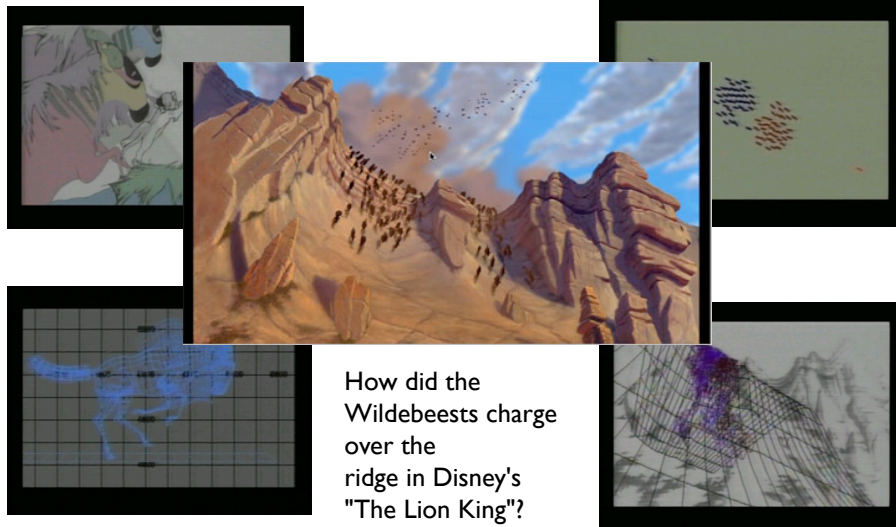
Table 6. Student Success Measures

	Traditional	Media Computation
Major Course Pass Rates	86.8% (99/114)	92.9% (159/171)
Course Pass Rates (majors and non-majors)	80.4% (131/163)	90.1% (237/263)
Retention Rate of Majors 1 year later	62% (67/108)	71.1% (27/38)**

**Only reported on Winter 2009 term, Fall 2009 term data is not yet available.

Georgia Tech College of Computing

A Contextualized CS2: MediaComp Data Structures



How did the
Wildebeests charge
over the
ridge in Disney's
"The Lion King"?

Georgia
Tech | College of
Computing

Research Question: Is context still useful in a second course?



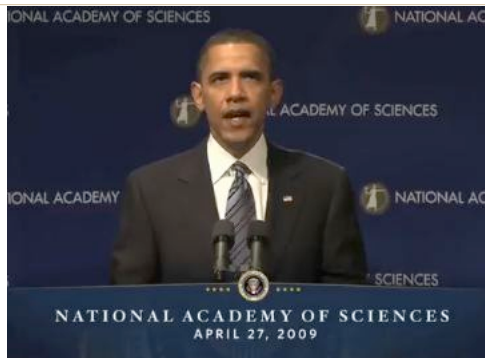
- Mixed model research design
 - Interviews informing whole-class survey
- 11% agreed with “Working with media is a waste of time that could be used to learn the material in greater depth.”
 - “I didn’t take this class to learn how to make pretty pictures.”
- A majority of the class (70%) agreed or strongly agreed that working with media makes the class more interesting.
- 67% of the students agreed or strongly agreed that they were really excited by at least one class project
- 66% reported doing extra work on projects to make the outcome look “cool.”

(Yarosh and Guzdial, JERIC, Jan 2008)

Georgia
Tech | College of
Computing

Software as Personal Tutor

This represents the largest commitment to scientific research and innovation in American history. Just think what this will allow us to accomplish: solar cells as cheap as paint; green buildings that produce all the energy they consume; [and] **learning software as effective as a personal tutor.**



Role 1: Finding or Making Context

- Context *is* important.
 - It works to support success and engage students.
 - But, there is no universally successful context.
 - At Georgia Tech, there are continued demands for finer granularity.

Role for EAAI:

Help students and teachers to match interests and abilities to contextualized educational material, *or*

From a general form, instantiate template into a given context (e.g., sampling, sorting, searching)

Matching or Tailoring

- **Matching** students to existing contextualized content.
 - What variables are important? Level, major, desired degree, previous background.
 - More than a search engine.
- **Tailoring** content from descriptions of context-specific features to learning objectives.
 - Pixels in pictures and samples in sounds: Sampling
 - Discrete event simulation queues: Sorting
 - Format of template? Do context-specific features extend beyond one learning objective?

Piece 2: Do they want what we have to offer?

- Brian Dorn has been studying graphics designers who program.
- Conducted a series of interviews and assessment activities.
- Found that these subjects want more computer science, but don't find courses (and most other resources) adequate (Dorn & Guzdial, to appear ICER 2010)
- *P10:..A lot of people can just fly airplanes...But if you put them in an unusual situation, and they don't understand it...
I think programming's probably the same way. I've written classes and thought, wow I've just created a binary tree here. If I knew what I was doing when I was doing it because I had the academic understanding, then I'd probably look for a base class that's already been optimized, and I wouldn't have to rewrite it. So, that was a really long way of saying **yes. I think that an academic study would make me a better programmer, but not by a whole lot.***

What do software engineers do?

Answer: The Boring Stuff.

- P2: I was able to take different samples from different places and instead of just being let's say an MIS major, or computer science major, you know it's—you're not going to be front-end anything with computer science. You're going to be back-end everything.
- P4: I think as a **front-end developer**, you focus more on the design and the usability, and you're focusing more on the audience. And then on the **back-end** I think you're focused on more, these are like the software developers. And they're programming something, and they don't really see what it's gonna look like; they're just making it work.

Why don't they take CS classes?

- P7: **I started out in computer science, but didn't like it at all.** The fact that I wasn't learning anything new. I took an intro to programming course, and then I talked to some other people in the program and it was all repetition and I guess there wasn't any really new. So you weren't really learning any concepts. You were learning the languages, and I didn't like that at all. So that's why I left...
- P7: **I'm looking for more concepts instead of examples I guess.** I think the, my problem with books was the same thing. They're teaching more of a language than the concepts, and so I just want a place where I can learn the concepts and that's it. And I really can't find that, you know? It's either you learn a language and you hope to find out about the concepts, but you never really do.



They are not afraid of coding

- “What interests you about web design?”
- P12: The coding! I don't like to code. But the things that the code can do is amazing, like you can come up with this and voila, you know, it's there. Javascript for one. The plugins and stuff. I think that's very interesting, intriguing and stuff. Because I mean like the code is just, there's so much you can do with code and stuff. It's just like wow.



They're Lost without Initial Knowledge

- They learn from FAQs, lots of code spelunking, books where they can, friends. Never courses.
- Brian's experiment: Given a case library with conceptual information vs. a code repository alone, what gets learned?
- Surprising tidbit: Less than they might because of a lack of deep knowledge.
 - For example: Understanding code by searching Google for function (not API) and variable names.
 - Not really surprising (Simon and Larkin on deep/shallow knowledge)

Role 2: Interpreting Code for Search

Role for EAAI:

Suggest appropriate search terms for the code that the amateur programming is trying to interpret, use, or reuse.

- Patterns to recognize (simple) recursion, sorting, linked lists, API use.
- Tagging of examples?
- Machine learning approach: Given code X in IDLE or Eclipse, what search terms led to fruitful (defined?) results?


Piece 3: The new APCS

- Big effort to create the Advanced Placement exam “Computer Science: Principles”
 - Explicitly not focused on programming.
- High school is critical to drive interest (prevent disinterest?) in CS at undergrad (and beyond).
- Surprisingly little AP CS now.
 - 2,000 AP CS teachers today. (Goal: 10K by 2015)
 - CS is behind in student enrollment:
 - # Calculus exams given in 2009: 292,000, # Biology exams: 154,000, # Economics exams: 67,000, #CS exams: 20,000.



Learning objectives for APCS

- Draft evidence statements:
 - Explanation of how different levels of abstraction, including data, information and knowledge, are used in computational models.
 - Evaluation of the usability of a computational artifact that has facilitated exploration and creation of new connections in information
 - Summarization of the behavior of a computational artifact and the characteristics of the computational artifact.
 - Identification of unintended consequences that result from modeling complex natural and human systems
 - Explanation of an analysis of the ways in which computing-enabled innovations have affected communication and cognition.
 - Use of simulation to investigate posed/existing questions and develop new questions about complex natural and human systems



Role 3: Supporting learning CS Concepts without Programming

- Some of these AI “owns” (Computing impacts on cognition? Modeling complexity? Representation of knowledge?)
- Others AI can contribute to (e.g., an agent for testing usability designs?)
- Challenge: Not rote memorization, not boring, and usable by teacher with little knowledge.

Role for AI:

Create materials that support the learning of computing concepts (especially AI concepts) without necessitating programming.

Conclusions



- Computing is important for everyone.
- Students who want computing succeed with contexts.
 - Role for EAAI: Helping with the match.
- End-user programmers want what CS has to offer, and there are more of them than there are professional software developers. But they don't know CS.
 - Role for EAAI: Supporting development of base knowledge.
- The new AP CS aims to introduce computing contexts with a minimum of programming.
 - Role for EAAI: Providing materials to support learning of concepts without requiring programming skill.

With thanks to our supporters



- US National Science Foundation
 - Statewide BPC Alliance: Project "Georgia Computes!" <http://www.gacomputes.org>
 - CCLI and CPATH Grants
- Microsoft Research
- Georgia Tech's College of Computing
- Georgia's Department of Education
- GVI Center,
- President's Undergraduate Research Award,
- Toyota Foundation

Thank you!

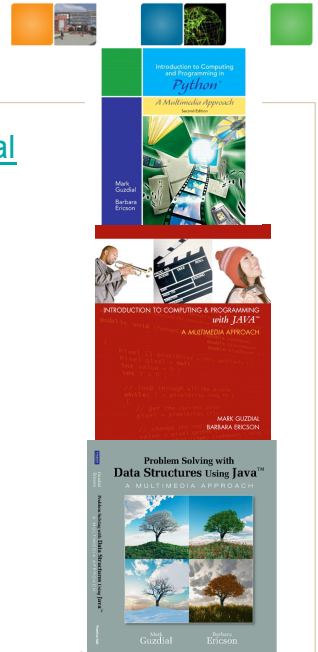
- <http://www.cc.gatech.edu/~mark.guzdial>
- <http://home.cc.gatech.edu/csl>
- <http://www.georgiacomputes.org>

For more on the new APCS.

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

For more on MediaComp approach:

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



Georgia
Tech

College of
Computing