BROADENING PARTICIPATION IN COMPUTING

CS FOR ALL
Story

- Computer science was originally invented to be taught to everyone
  - To help them learn other subjects
  - To support democratic ideals

- Our definition of computer science has become more narrow.
  - Computing education is now an *alternative endpoint*.

- Meeting the original goals of CS addresses both BPC and CSforAll
The computer is a necessary tool for learning science, mathematics, or engineering
“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.”
“Once informatics has become well established in general education, the mystery surrounding computers in many people’s perceptions will vanish. This must be regarded as perhaps the most important reason for promoting the understanding of informatics. This is a necessary condition for humankind’s supremacy over computers and for ensuring that their use do not become a matter for a small group of experts, but become a usual democratic matter, and thus through the democratic system will lie where it should, with all of us.”

Thanks to Michael Caspersen
Programming changes how we understand
First published definition of Computer Science

“The study of computers and all the phenomena surrounding them.”

*Science*, 1967

This is broader than how most people define computer science today.
Let’s call this *Computing*
Definitions of Computer Science

- Computer Science is the study of computers and computational systems. (Encyclopedia Brittanica)

- Computer science is the study of computers and algorithmic processes, including their principles, design, implementation, and impact on society. (Tucker, 2006 - K-12 CS Framework)

- Computer science is the foundational discipline with an emphasis on discovery related to programming, algorithms, and data structures. (ACM/IEEE Computing Curriculum 2021)
Definitions of Computer Science

“The study of computers and all the phenomena surrounding them.”
(Perlis, Newell, & Simon, 1967)

▪ Computer Science is the study of computers and computational systems. (Encyclopedia Brittanica)

▪ Computer science is the study of computers and algorithmic processes, including their principles, design, implementation, and impact on society. (Tucker, 2006 - K-12 CS Framework)

▪ Computer science is the foundational discipline with an emphasis on discovery related to programming, algorithms, and data structures. (ACM/IEEE Computing Curriculum 2021)
Seymour Papert claimed “that children can learn to program and learning to program can affect the way that they learn everything else.”
President Obama “CS for All”

Computer science (CS) is a “new basic” skill necessary for economic opportunity and social mobility.
When did this become about “economic opportunity”?

- Forsythe, Perlis, Snow, Naur, Simon, Papert, and Newell were all arguing for computing education decades before Silicon Valley.
- They weren’t about preparing students for software development jobs.

- “The Case for Alternative Endpoints” (BJET, 2021)
  Mike Tissenbaum, David Weintrop, Nathan Holbert, Tamara Clegg

- What are the goals of Computing Education, if not job skills?
Who gets access to these powerful ideas?

WHO GETS COMPUTING EDUCATION TODAY
5.6%

https://advocacy.code.org/stateofcs
Texas dashboard accessed through the ECEP State Dashboards page

5.6%

By Race/Ethnicity

IN THE 2021-22 SCHOOL YEAR:

4.4% OF ALL HIGH SCHOOL STUDENTS TOOK A CS COURSE
5.6% OF STUDENTS AT SCHOOLS OFFERING CS TOOK A CS COURSE

https://advocacy.code.org/stateofcs
Log (# Female / # Male) for Advanced Placement Exams in 2021

Sum of Log (Fem/Male) for each Exam. Color shows sum of Log (Fem/Male). Size shows sum of Total. The marks are labeled by sum of Total.

Data and Visualization from Barbara Ericson and Willa Hua
If we want to teach **Computer Science for All**, we have to teach where “All” are.

And that’s not CS classes.
Broadening access and participation in computing

WHAT WE’RE TRYING AT MICHIGAN
Program in Computing for the Arts and Sciences

Mark Guzdial, mjguz@umich.edu
What does LSA need in Computing Education?

Dean Anne Curzan and Associate Dean for Undergrad Ed Tim McKay charged the Computing Education Task Force 2020-2021

- What do LSA students need to know about computing?
- What classes and programs already exist?
- Where should we be going?

- Conducted dozens of interviews, reviewed hundreds of courses, surveyed over 100 LSA faculty.

- Final report is available:
3 Themes for Computing Education in LSA

- **Computing for Discovery**: Computational science enables new discoveries across natural and physical sciences.

- **Computing for Expression**: Computing has changed how we communicate and engage with others, from social media to Pixar to AR/VR.

- **Critical Computing, or Computing for Justice**: Computers and applications are pervasive in our daily lives, and thus have immense cultural, social, and political influence. Who is supported by computing, who is oppressed, and how can we create better models?
Program in Computing for the Arts and Sciences

Launched Summer 2022 - me and Gus Evrard, a first-generation computational cosmologist.

Our lecturer, Brian Miller, has a PhD in Music and worked as a data scientist. Academic Program Manager, Tyrone Stewart, has a PhD in American Culture. Faculty teaching in PCAS come from Linguistics and Anthropology.

Goals:
- To meet the needs of all LSA students to learn about computing, especially programming.
- To create new computing courses around the themes of justice, expression, and discovery.
- To create new credentials to enhance majors and provide computing-centric minors in all divisions.
Developing Courses and Minors

Course code for PCAS: **COMPFOR** – COMPuting FOR…

First courses were taught last year:

- **COMPFOR 111** “Computing’s Impact on Justice: From Text to the Web”
- **COMPFOR 121** “Computing for Creative Expression”

Courses introduced in Fall 2023:

- **COMPFOR 131** “Introduction to Python for the Sciences.” Worked with faculty across the Natural Sciences to develop the new course.
- **COMPFOR 101** “The Transistor Disruption: How a Tiny Tool Transforms Society and Science”


Developing minors in **Expression** and **Discovery**.
Formed advisory groups of faculty who self-identified as being about Computing for Expression or Computing for Justice.

Created two sets of shared whiteboards of:
1. Learning objectives that were identified by the computing education task force
2. Student activities to support reaching those learning objectives.

• Instructions: “Please move to the right those that you think are important for the course, and move the left those that you think are less or unimportant for the course.”
Endstate 2/2 - LSA Computing Learning Goals for Justice
What do we want students to know/do? Put important ones to the right, and less useful to the left.

- Write a program to algorithmically generate a sentence, a picture, or a sound.
- Write secure, safe, and robust code.
- Write the program "Hello World!" in a textual language (like C++ or Python)
- Explain the difference between MIDI and MP3

- Build a game in Gamemaker.
- Build a web page in HTML and CSS.
- Use a Web service API in JavaScript
- Build an iOS or Android app.

- Know the difference between Twine and Unreal Game Engines
- Know the difference between C++, Python, and Snap!
- Build an image filter in some programming language
- Compute statistics on a spreadsheet and make a graph

- Write the program "Hello World!" in a block-based language (like Snap! or Scratch)
- Explain the difference between digital and analogue, using the example of Spotify and vinyl records.

Be able to talk to programmers about their processes, including references to Github and IDEs.

- Use a Jupyter Notebook
- Explain why programming languages are a barrier to non-English language speakers.
- Explain what an API is for a website or library
- Explain why social engineering is a great cybersecurity risk.

- Explain the impact of bitcoin mining on the environment.
- Know what a GPU is and what it has to do with making virtual reality work.
- Explain what blockchain is and how it's related to NFTs
- Critique a website for its accessibility.

- Explain how a database is used to generate HTML pages through a template
- Understand how user behavior data can be analyzed and inferences made
- Explain how the Internet works at the level of servers, domain names, and IP addresses.
- Explain how and why facial recognition systems can be biased.

- Describe how sounds and pictures can be represented in numbers or bits.
- Download Facebook or Twitter data to analyze it for keyword trends or sentiment.
- Scrape a website for data and put the data into a spreadsheet for analysis.
Be able to talk to programmers about their processes, including references to Github and IDEs.

Explain the impact of bitcoin mining on the environment. Know what a GPU is and what it has to do with making virtual reality work. Explain what blockchain is and

Use a Jupyter Notebook

Explain why programming languages are a barrier to non-English language speakers.

Explain what an API is for a website or library.

Explain how the Internet works at the level of servers, domain names, and IP addresses.

Explain how a database is used to generate HTML pages through a template.

Understand how user behavior data can be analyzed and inferences made.

Explain why social engineering is a great cybersecurity risk.

Critique a website for its accessibility.

Explain how and why facial sounds and pictures
Write secure, safe, and robust code.

Write the program "Hello World!" in a textual language (like C++ or Python).

Build a game in Gamedeck.

Build a web page in HTML and CSS.

Use a Web service API in JavaScript.

Build an iOS or Android app.

Know the difference between C++, Python, and Snap!

Build an image filter in some programming language.

Explain the impact of blockchain on the environment.

Build a model of phenomena, simulation of data, and the data to dataset.

Explain what a GPU is and what it has to do with making virtual reality work.

Learn to create statistics and spreadsheet a graph.
Endstate 2/4 - LSA Computing Learning Goals for Expression
What do we want students to know/do? Put important ones to the right, and less useful to the left.

- Explain why social engineering is a great cybersecurity risk.
- Write secure, safe, and robust code.
- Explain the difference between MIDI and MP3
- Know what a GPU is and what it has to do with making virtual reality work.
- Critique a website for its accessibility.
- Explain what blockchain is and how it's related to NFTs
- Explain why programming languages are a barrier to non-English language speakers.
- Explain the impact of bitcoin mining on the environment.
- Explain the difference between digital and analogue, using the example of Spotify and vinyl records.

- Build an image filter in some programming language
- Know the difference between C++, Python, and Snap!
- Write the program "Hello World!" in a block-based language (like Snap! or Scratch)
- Describe how sounds and pictures can be represented in numbers or bits.
- Use a Jupyter Notebook
- Explain what an API is for a website or library
- Download Facebook or Twitter data to analyze it for keyword trends or sentiment.
- Know the difference between Twine and Unreal Game Engines
- Use a Web service API in JavaScript
- Build a game in GameMaker.
- Explain how a database is used to generate HTML pages through a template
- Build a web page in HTML and CSS.

Create a model of some phenomena, run the simulation of the model, and compare the data to another dataset.
Compute statistics on a spreadsheet and make a graph.
Talk to programmers about their processes, including references to Github and IDEs.
An Alternative Path
What do Humanities Scholars want Students to Know about the Internet?

History Professor, LaKisha Simmons gave me a list:

1. There are things called databases.

2. That databases, if they are designed well, are easy to index and to find information in.

3. Databases could be used to automatically generate Web pages.
These are “advanced” topics in undergraduate CS

- No CS program *starts* there.
  - Everyone starts with introductory programming, then data structures and algorithms, then…

- Do we have to?

- Alternative Paths for Alternative Endpoints
Supporting CSV files as databases

SELECT COLUMN 3 FROM
sort show items from billionaires when column 0 = 2014 by column b
increasing

SORT ▼ BY COLUMN b ▼ decreasing

show column 3 from billionaires

SELECT FROM billionaires WHERE COLUMN 1 = Bill-Gates

GROUP (billionaires) BY COLUMN e

item 1 of billionaires

show column 3 from (billionaires) when column 3 = 2014

sum show column 3 from (billionaires) when column 5 = 2014

SELECT COLUMN 3 FROM billionaires WHERE COLUMN 3 = 2014

pipe titanic →

show items from  when column 0 = female

split show column c from  by word

GROUP ▼ BY COLUMN b

SORT ▼ BY COLUMN b ▼ decreasing

Thanks to Fuchun Wang
But how are we going to do Web pages? Custom Snap Blocks!
Connecting to SQL in EBook

Chapter 3: Tables of Data

In this class, we started using tools to analyze data on the computer, and how that choice (e.g., between ASCII and Unicode) can be a barrier to people’s access to computing. We considered how computers might recognize and generate sentences, and about the challenges of understanding different dialects of English with the same language model. We then moved to the Internet and the Web, tracing its history and how grounded it was in English and male culture (e.g., the history of BASIC at Dartmouth). We wrote programs to pull data out of the Web (e.g., pulling out URLs and web scraping) and to generate HTML for the Web. In this latest section of the course, we have written programs to analyze data from the Web.

Section 1: Billionaires

In this class, we have been writing Snap programs to take apart and analyze CSV files. In this ebook, we’ll be using the databases and blocks found in the Database Microworld with Titanic project. For example, here’s a script that selects the billionaires from 2014, sorts those billionaires in terms of their rank (in column 8), then selects just the names.

```
SELECT COLUMN 8 FROM billionaires when column 5 = 2014 by column 8 decreasing
```

The list is long — 1,6543 names. We can get Snap to show us all of them. Here’s what the top of that list looks like.

We can also describe the same script as a pipe where the billionaires are first selected, then sorted, and then the names selected out.

```
Open Research Questions
Research Questions We’re Exploring

● How can we support making the notional machines of arts, humanities, and sciences faculty real?

● Are arts and humanities students and faculty getting what they need from these classes?

● What is the process that students follow when programming in these classes, and how does that interact with the unusual structure (e.g., multiple languages, worked examples)?
Research Questions We’re NOT Exploring

- Do these students major in computer science or information?
- How difficult is it for these students to learn C++?
- Are they taking jobs in the computing industry?
- Are they learning how to write safe, secure, and robust code?
Summary: Who should teach Computing?

- Computing Education is different than CS Education.
- Computing Education for everyone is why CS was invented.
- If you want to reach everyone, they won’t be in CS classes.
- Maybe CS departments shouldn’t teach everyone computing. But if not, then who?
Programming can be a Tool for Learning Anything

WE NEED TO MAKE COMPUTING ACCESSIBLE TO EVERYONE

Some of the Collaborators on This Work

- Barbara Ericson, Gus Evrard, Kelly Campbell, Miranda Parker, Kathryn Cunningham, Amber Solomon, Bahare Naimipour, Tamara Nelson-Fromm, Emma Dodoo, Tammy Shreiner, Elise Lockwood, Adaline de Chenne.
- Undergraduate researchers: Aryan Bannerjee, Alexandra Rostkowycz, Erin Shi, Brandon Geng, Jessica Zhang, Ben Steinig, Fuchun Yang, Aoife Harte, Chloe Nguyen, Kashmira Reddy, Kristen Taurence, Angela Li, Derrick White, Jessie Houghton.

- https://lsa.umich.edu/computingfor
- http://computinged.wordpress.com
- http://guzdial.engin.umich.edu

Thank you!