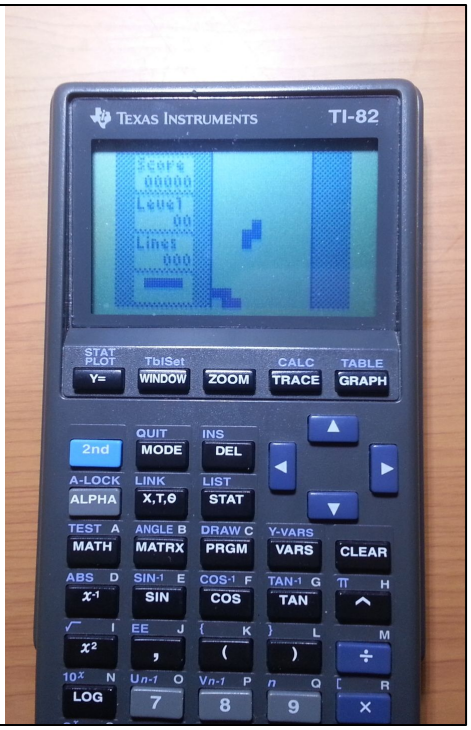
Code, Calculators, Creativity, and the Many Paths to CS Education

Amy J. Ko, Ph.D.
The Information School
University of Washington



I'm so excited to be speaking to you all today. I want to talk about why CS belongs in our schools. Now, I'm a Professor, and so there's a risk that I'd try to do this by making some complicated, abstract, theoretical argument. And I could definitely do that! But today I'd rather just tell a story. In particular, my story, of how my schools and teachers helped catalyze and grow a lifelong interest not just in CS, but in everything that we do with it in society.

After my story, I'll turn to the stories of our state's youth, and why we need to make room for countless other stories for our children to encounter CS.

Let's start at the very beginning.

Sunset Elementary, my primary school.

I was born in the Pacific Northwest in 1980, in a small town called Ontario in Eastern Oregon. But shortly after, my family moved to Portland, and then a suburb of Portland called West Linn, where I went to Sunset Primary School. My Mom was a 5th grade teacher for 30 years in St. Helens, Oregon, just across from the southwest Washington border, and I remember fondly visiting her classroom every single year to set it up before her kids arrived. I adored my teachers throughout K-12—Ms. Wright, Mr. Hardt, Mr. Strictland, Ms. Hudson, Mr. Gonnerman—they're the ones that showed me the world, helped me find my place in it, and helped me get to where I am today.

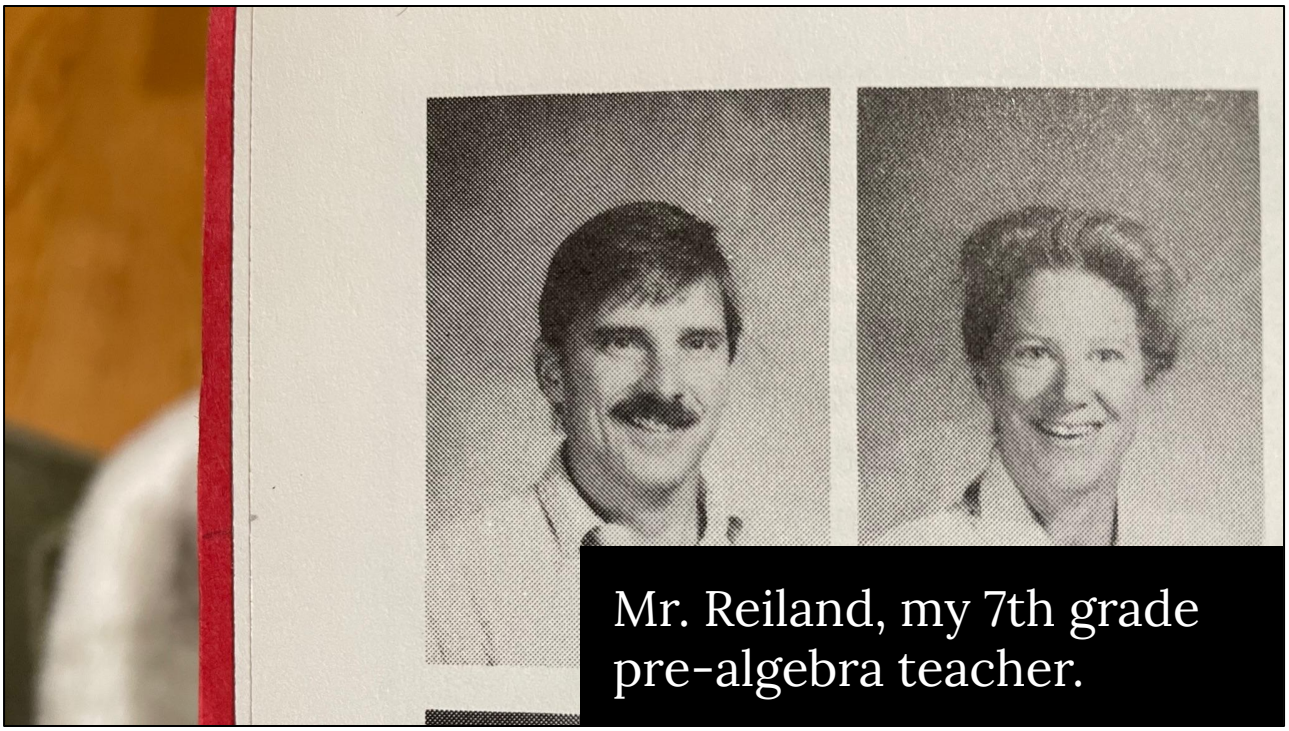
Athey Creek, my middle school

And it's that respect and passion for learning, teaching, and education that's brought me here to speak today. I fell in love with school. When I first started tutoring at my middle school, Athey Creek, I fell in love with teaching. And long after that, when I went to college and discovered research, I fell in love with research about learning and teaching too: my mind raced as an undergraduate with all of the things that made learning possible, and all of the things that got in its way.

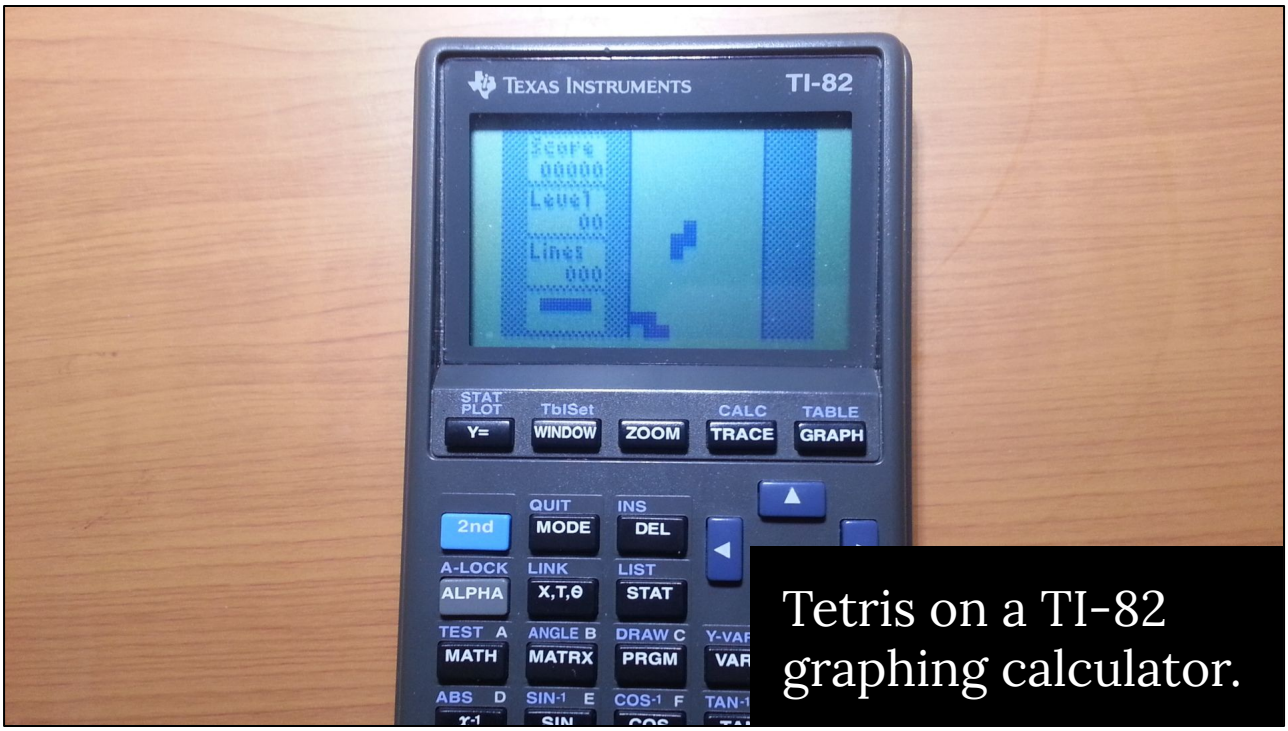
And long after that, when I went to college and discovered research, I fell in love with research about learning and teaching too: I was fascinated by all of the things that made learning possible, and all of the things that got in its way.

Me, at home, making things on our PC while my brother played with a balloon.

But throughout this story, there's something else I fell in love with: computers. But not just using computers, but expressing myself with computers. Back in the 1980's and 90's, I made art, I made music, I wrote stories. Computers for me were these amazing devices that let me explore and communicate my ideas, whether they were ideas from subjects in school, or ideas from my life with friends, family, or comic books.

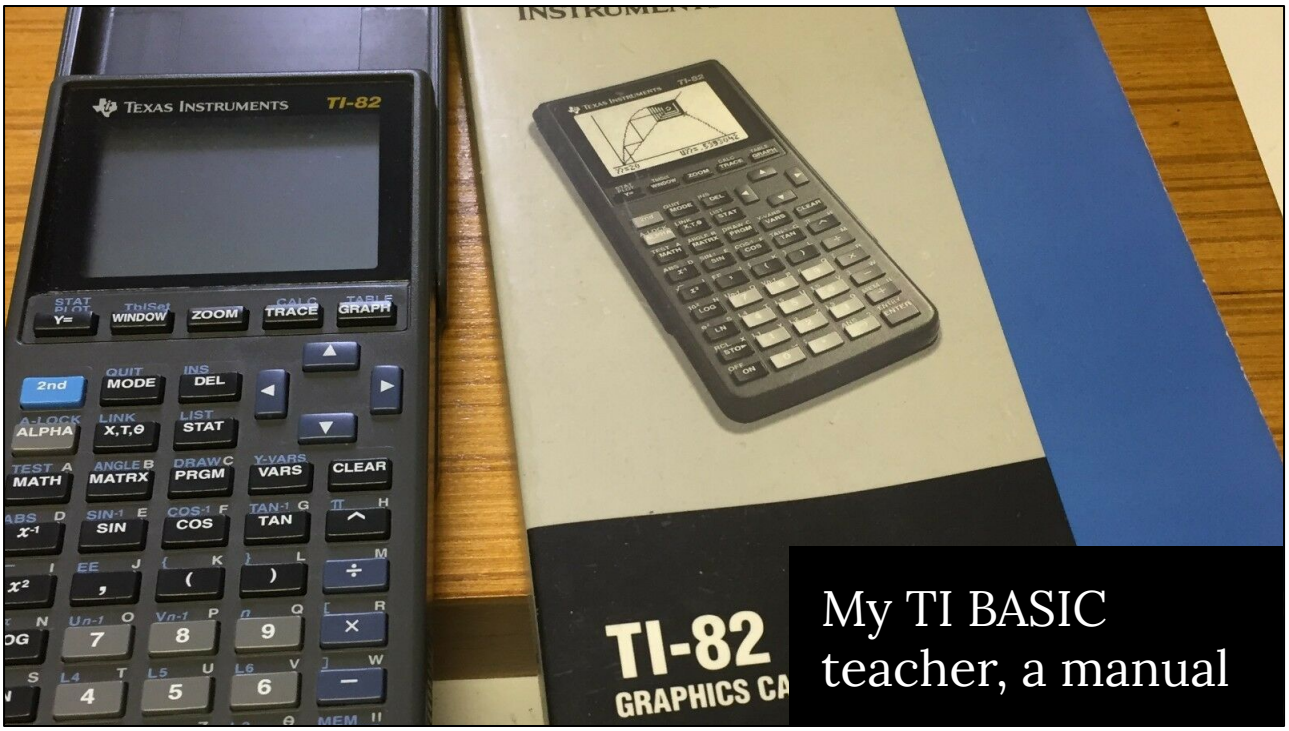


These kinds of expressions were great, but it wasn't until 7th grade that I found my true passion for computers: programming them. My middle school pre-algebra teacher Mr. Reiland spent the first day of 7th grade teaching us how to program a TI-82 graphing calculator to do basic formulas. That itself was incredibly boring: typing cryptic commands to compute numbers was nothing like using painting programs to create digital art, or using my keyboard to write songs.

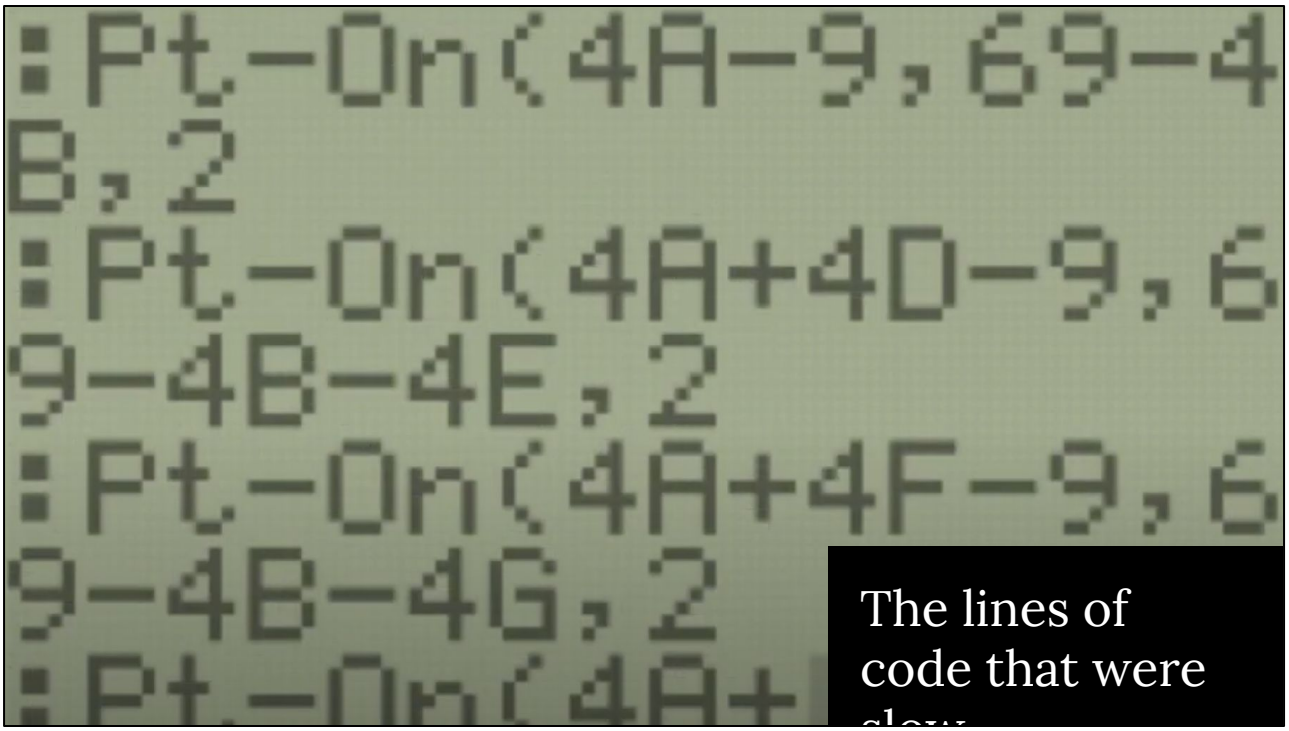


What *was* exciting was what my friend showed me after class: a program he'd gotten from his older brother, a full version of the game Tetris running on a TI-82. As a devoted fan of the Game Boy, Tetris was my favorite game. I could play it for hours. I was mesmerized by the possibility of playing Tetris in class. We didn't have a way to transfer it, so I mail-ordered the link cable and a month later, it came. We plugged our calculators together, fumbled through the transferring commands, watched the program copy bit by bit, and after a few minutes, I had my own copy of Tetris on my TI-82 graphing calculator. It was just as exciting as buying a game from Toys R Us, and perhaps even more exciting: this was a game that I could play in class.

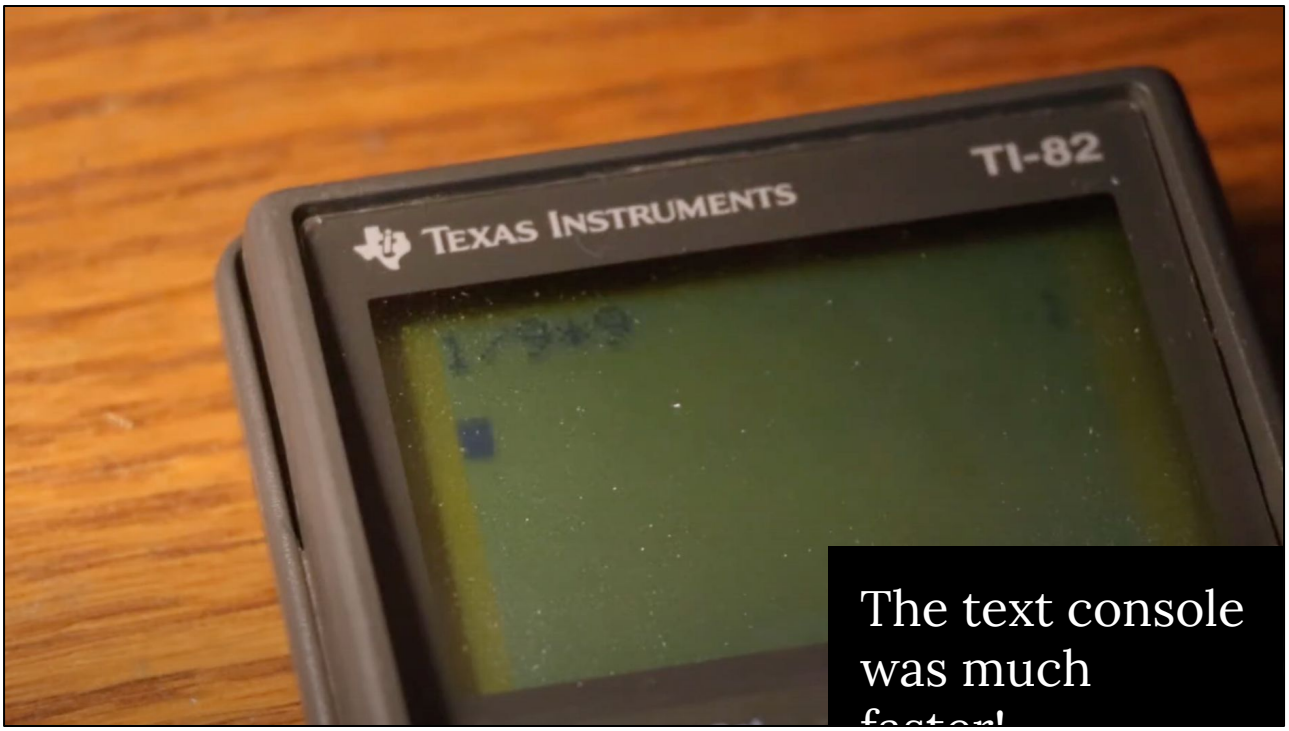
Now, to my great disappointment, the game was incredibly slow. It was Tetris: it followed the rules of the game, it showed the pieces of the game. But it drew the pieces of the game on the 2D scatter plot screen, pixel by pixel, and only a few pixels per second, and maybe a few pieces per minute. Maybe a few render per second. And so while it was Tetris functionally, it was not the Tetris experience. Without speed, there was no challenge, and without challenge, there was no fun.



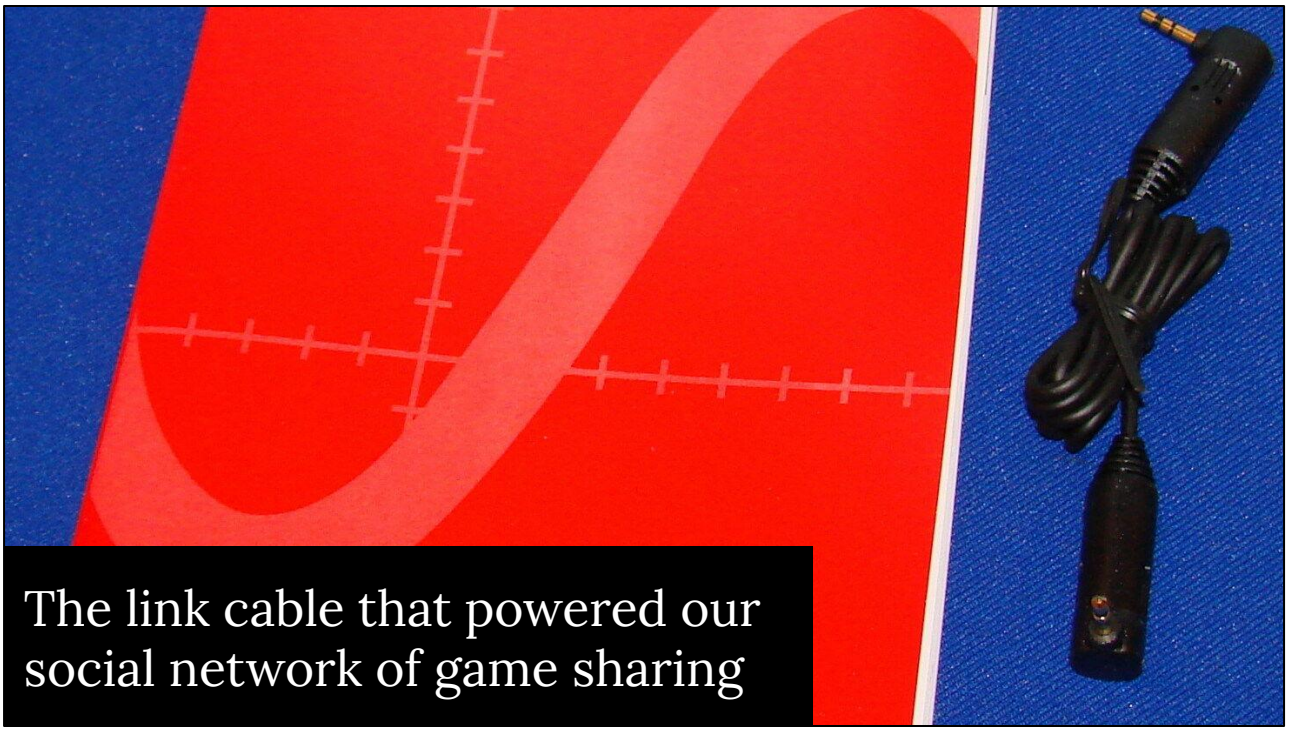
I could have set the game aside, disappointed that I couldn't play at school and in class. But in that moment, something special happened: I remembered what Mr. Reiland showed us on the first day of class: the F5 PRGM key to see the program's code. I went into the code and saw the same cryptic commands he'd described. But I pulled out my manual, and began reading what all of the commands meant. I remember coming home after school each day, translating a few more lines of code into English, trying to understand how the Tetris game was built, line by line.



Eventually I discovered why the game was slow: drawing anything to the plot was slow. I tested it: drawing lines was slow, drawing scatter plots was slow, drawing curves was slow, drawing text was slow, and drawing individual points was slow. But I discovered while reading the manual that there was one other way to draw things to screen: the text console, where you usually entered calculator commands and formulas. It was actually possible to render text at any position on the display by providing a row, column, and a character to display. So I set out to replace the part of the program that drew the pieces, pixel by pixel, with commands that drew to the console instead, using letters to represent the different rotational states of the Tetris pieces



After a month of trying, failing, and trying again, and lots of re-reading of the manual, I had it working: rotate the TI-82 to make a vertical display, use the left and right arrows to move pieces and the rotate button to rotate them. To make it work, I had to use countless other skills and concepts. The logic we'd been so carefully taught in my 6th grade transition math class helped me reason through the conditional logic of the game. The arithmetic I'd been taught throughout primary school helped me compute where pieces should move. The pre-algebra I was learning in the very math class that Mr. Reiland was teaching helped me use variables and algebra to track game points and winning thresholds. The task management and organization skills that my 7th grade homeroom teacher had taught us at the beginning of the year helped me organize my work in my school planner. The careful reading comprehension I was taught throughout 6th grade English helped me extract meaning from the calculator manual. And even the sketching skills I was learning in my 7th grade calligraphy and sculpture classes taught me how to plan a piece, but for ink and clay instead of screen layouts. And the result? The revised game was lightning fast! In fact, I had to insert a command to slow it down, because it was too fast to play.



The next day, I took my link cable, transferred it to all of my classmates' calculators, and for a good portion of the beginning of class that day, all of us were playing Tetris on our calculators instead of learning about algebra. It was a glorious moment in my childhood: my classmates were amazed; my self-efficacy skyrocketed, and suddenly all of that passion for making art and music with computers shifted to programming computers (including programming art, music, stories, and animations, and games). Of course, my teacher wasn't as impressed. In fact, he confiscated all of our calculators for the day. After class, he returned them to us, and had me stay behind to chat. He told me that he was so proud of what I'd created—it just didn't belong in class.

My 13th birthday, around the time I realized I wasn't a boy, and needed something to be proud of.

In that moment, I could have felt deflated, but instead I was proud. I'd made something that earned the respect of all my peers and my teacher, not an easy feat for a young, shy, unpopular, closested trans girl looking for ways to make friends and escape puberty.

And so I persisted at home. I created interactive text adventures, action games, and interactive animations and art. I learned enough geometry to begin making 3D worlds and simulations. My friends made art, music, and sounds and we created a racing game in which we would drive around a block-for-block recreation of our neighborhood, trying to not run into houses or trees. Computing, to my friends and I, was creativity, community, friendship, identity, play, independence, and confidence, all rolled into one. And coding was the engine for that work: it allowed me to create anything we could imagine, as long as I was willing to learn and persist.

My CS education library, Powell's books.

And while it was catalyzed by school, my learning mostly happened outside of school: without teachers, mentors, resources, or guidance. All I had was my calculator manual, the used technical books I bought for $1 at Powell's Books downtown Portland, and my mind. It was slow and isolating, but it was my passion.

And that trend of isolation continued. While I had many of friends interested in art, and many friends interested in computers, I had no one interested in programming. We had no CS classes in my high school. There was a 2-year college student that hung out in our computer labs and let me do his homework, and tutored me on my programming projects. An exchange student from the Netherlands who did have CS classes in the 1990's taught me about variables, arrays, and loops. And when it came time to apply for college, I signed up for the AP CS A exam, but I was the only one, so the proctor locked me in a literal broom closet to take the exam. I only got a 3, partly because I couldn't get out to pee, and partly because I had no idea what would be on it—or that it was in a programming language I didn't know.

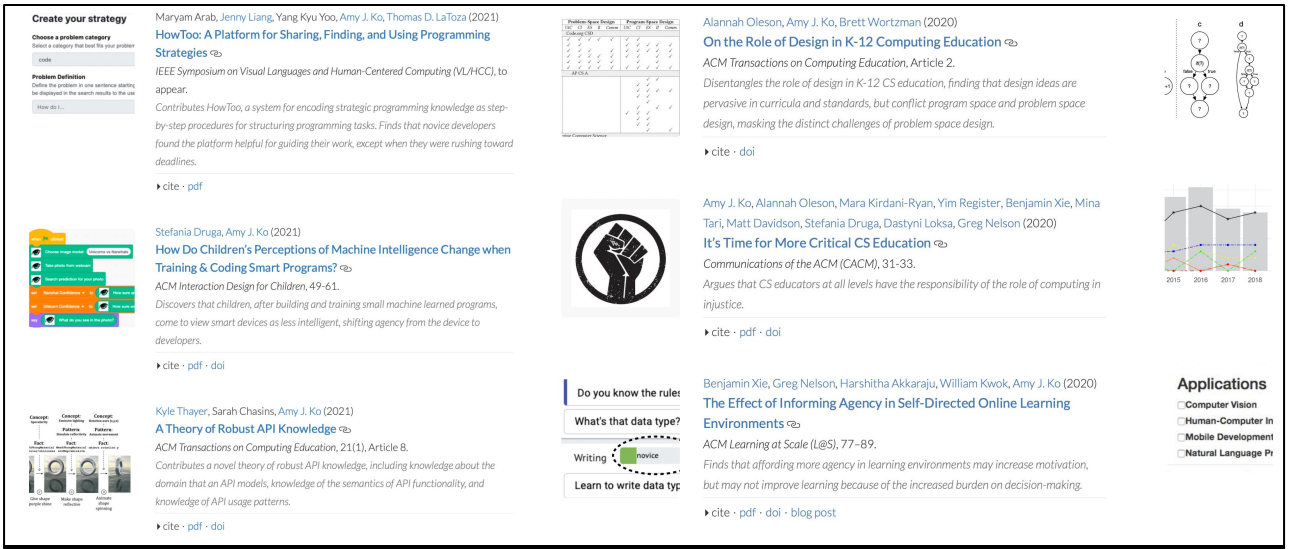
My ex wife and our daughter celebrate graduation.

Despite the isolation, lack of support, and even the stigma and bullying for being someone interested in computers, those years in middle and high school shaped who I am today. I went on to major in Computer Science and Psychology in college, to fall in love with invention and discovery, and eventually got a Ph.D. in human-computer interaction, the study of how people and computing interact. And since 2008, I've been a Professor at the University of Washington, a school I always dreamed of attending as a student, but now help run as a tenured Full Professor.

And over the past 10 years, I've turned my state-subsidized research attention to a question: how can we catalyze the kinds of experiences that I had with computing for all youth, independent of their interests, their identities, and their abilities, not just in Washington state, but everywhere in the world?

Because computing isn't just about games, music, art, animation, algebra, and geometry, like it was for me. In our modern world, in which computing is woven into everything, it's also about science, health, wellness, engineering, sustainability, politics, love, war, culture, the present, the past, and the future. Just as with the invention of the book, there isn't a single discipline that hasn't been radically changed by computing, and so there isn't a single student that doesn't need to understand it.

And so this question of how to catalyze personally meaningful, culturally sustaining CS learning experiences has led me to do several things.



My lab has made hundreds of discoveries about CS teaching and learning.

It inspired research: how can we teach CS equitably, inclusively, and in culturally responsive and sustaining ways? My lab has invented numerous teaching methods that are now part of curriculum used across North America, Europe, and Asia.

This question inspired teaching: I've taught high school CS classes for the past 5 summers through the University of Washington's Upward Bound program. I've also collaborated closely with colleagues in the College of Education to start educating dual endorsed pre-service teacher candidates with Math + CS, Science + CS, Social Studies + CS, and even World Languages + CS.

And this question has inspired service: I've not only organized my local community around Puget Sound, but also helped organize the state, leading CS for All Washington, a group of stakeholders from across the state who help inform the state's K-12 CS Education policy needs, advocate for resources, and organize information about K-12 CS Education in our state.

My hope is that through these efforts, and the countless efforts of everyone in this room, that we can bring CS into public education at scale: as standalone CS classes, integrations of CS into other disciplines, and creative encounters with CS in after school and summer programs. We need all of these opportunities to help all youth see that computing is unavoidably part of everything that happens in our modern world—just like reading, writing, mathematics, science, engineering, and social and human perspectives on society, values, and life.

Computing is part of every profession, but schools aren't yet teaching as if it is.

And this can't come soon enough for our communities, state, country, and planet:

- We need scientists that know how to harness computing to make advances in our understanding of global warming and sustainability.
- We need engineers that know how to harness computing to invent and build a sustainable future.
- We need writers and journalists that know how to report on computing and analyze technology policy to help us make sense of computing's ever-changing role in our everyday lives and our civil rights
- We need politicians that understand the fundamentals of what computing is and how flexibly it can be harnessed to accrue and abuse power.
- We need people in the trades that can help us maintain the ever more computational nature of our homes, workplaces, and communities.
- We need artists and designers who can help us imagine futures of computing that are more just, equitable, inclusive, and even joyful than the narrow consumerist views imagined by Silicon Valley and our very own technology companies across Washington state.

And none of this happens unless we have *teachers* who can help youth see all of these amazing, surprising, and empowering connections between this marvel of digital innovation, and their families, communities, values, and dreams.

So what can you do?

- **Teachers**: find professional development that helps you integrate CS into your teaching, regardless of what you teach, and find others who are doing the same. With enough time, community, and support, you may even find that someday you want to teach a full CS class.
- **School and district leaders**: recognize that computing is no longer something we can ignore. It is part of society and part of our lives, and every child needs to understand more than just how to use computers: they need to understand how it influences their life and communities, and potentially how to code computers, so that they can reshape its impacts to be more just.
- **Policy makers**: recognize that none of the education reform visions I've shared will happen unless we simplify the complex web of policies and regulations that stand in the way of teacher certification, and invest in our public schools, not just for CS, but for all subjects and all teachers, because CS is part of all subjects.
- **Industry representatives**: know that schools aren't here only to create future employees. They're here to inspire the next founders, not only of for profit technology companies, but of for profit, not for profit, and government organizations that will ensure a healthy, well, and sustainable future for all of us, but also the political leaders we need to organize us around these visions.

- You can help by supporting the good work that public schools are trying to do, and recognizing that not everything is about profit.
- And for anyone else, know that there is a role for you too. Check out the csforallwa.org website for suggestions on how to volunteer, organize, contribute, and advocate.

I hope that everyone across our state can come together around these goals and activities, not only at this summit, but in the coming years and decades, to realize this dream. Our children deserve it, our world needs it, and our teachers need all the help they get to make it possible.

Thank you!