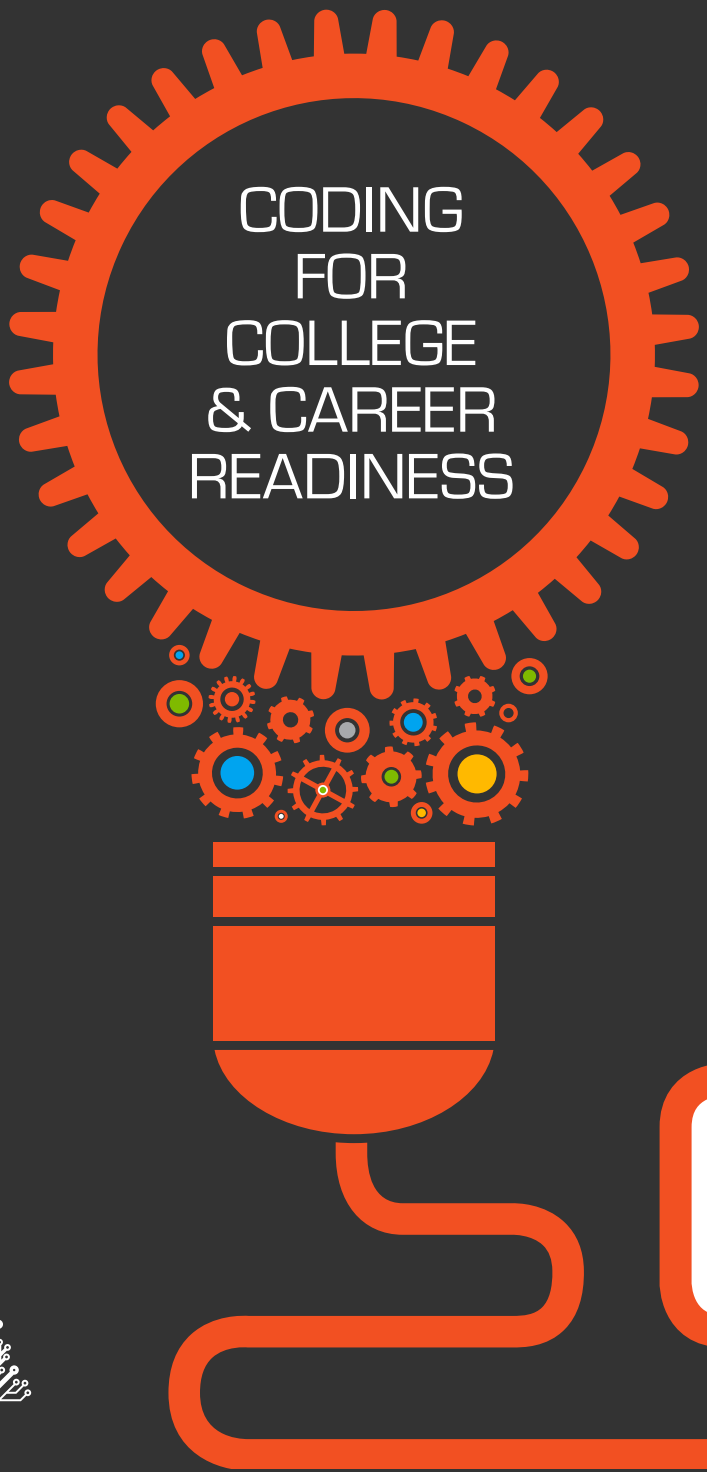
GETTING SMART ON



IN PARTNERSHIP WITH:

Microsoft

PUBLISHED BY:

GETTING SMART

Introduction

Anthony Salcito, Vice President,
Worldwide Public Sector Education at Microsoft

TABLE OF CONTENTS

During the recent Hour of Code, more than 76 million students in over 77,000 events got a "behind-the-screens" peek at the programming languages that power the devices and apps that power their lives. Microsoft is a proud partner of Code.org, which provides firsthand experiences for students and learners of all abilities and backgrounds to the limitless opportunities of computer science.

As my colleague Alison Cunard, General Manager of Microsoft Learning Experiences (LeX), said recently, "Today's students—our future business innovators—will not only depend on technology skills to get work done, collaborate, and communicate; but to navigate their day-to-day tasks and activities in an increasingly complex world."

As this collection of blogs illustrates, there are at least five reasons for expanding access to coding and computer science learning experiences in K-12 education – and beyond:

- **Coding is career prep.** Young people are growing up in a technology rich, data-driven world and will enter a workforce that depends on tech-skills--not just in information technology (IT), but in many traditional career fields including marketing, healthcare, finance, and increasingly education.
- **Coding is critical thinking.** Microsoft co-founder Bill Gates said, "Learning to write programs stretches your mind and helps you to think better, creates a way of thinking about things that I think is helpful in all domains."
- **Coding is engaging.** Coding powers personalized authentic learning. As educator and blogger Greg Garner notes, "Learning should always be a process of identifying and then solving unique and interesting problems that don't yet have answers." Coding provides real challenges, high engagement, and rapid feedback--a model for learning in other subjects.

- **Coding is entrepreneurship.** As educator Adam Renfro notes, there is a long list of self-made youth entrepreneurs on the Internet--and coding opens that door wide open. Making coding and computer science available to all students is a social justice issue--there doesn't have to be a gap between haves and have nots here.
- **Coding is collaboration.** Most programming in the work world is done by a team. Coding teaches collaboration and, like good writing, develops a culture of revision. Again, critical skills for today's work world.

As Mitchel Resnick, Professor of Learning Research at MIT Media Lab, points out, participants in Hour of Code "…are not just learning to code, they are coding to learn." Beyond math and computation, students learn problem solving, communication, collaboration, and project management skills—valuable skills for everyone, at any age.

In addition to providing powerful and easy-to-use coding tools like TouchDevelop, Kodu, and Project Spark, Microsoft has created a set of tutorials that are fun and engaging for students and easy to deploy for teachers -- just follow these step-by-step instructions.

On behalf of Microsoft Education, we hope this collection will inspire to you to introduce your students to computer science and shine a light on the importance of coding knowledge. Enjoy our blog series – and happy coding.

Teaching Kids to Code: An Economic and Social Justice Issue

Authored by Tom Vander Ark, founder and CEO, Getting Smart

Hadi Partovi, CEO of Code.org, wants more kids to learn to code. Bill Gates, Mark Zuckerberg, Sheryl Sandberg and many others agree. Partovi wants all high schools to offer computer science (CS) classes because CS represents a growing cluster of job skills. Unfortunately, few schools teach CS—particularly schools attended by low-income and minority students.

To fix the problem, Partovi launched Code.org in 2013. The site is packed with stats that make the case for coding (as seen in the video below). For example, did you know that coding jobs aren't just in tech? In fact, almost 70 percent of them are in other sectors. Most businesses need people who can code. But there are fewer schools offering computer science, as well as fewer computer science teachers and students, in the U.S. than ten years ago. By contrast, every high school graduate in China must take four credits of computer science, yet in the U.S. it's not even on the menu in most schools.

Code.org is also home to the Hour of Code, a global movement that has impacted tens of millions of kids in over 180 countries. The Hour of Code consists of a series of one-hour tutorials (in over 30 languages) designed to provide students across the world with an experience in coding. Partners like Microsoft believe in Partovi's mission and are dedicated to supporting events such as the Hour of Code in an effort to better prepare students for college and career.

His next step is to find a place for CS on the master schedule of high schools around the country. He'd like to see computer science added to list of math and science classes kids can take to satisfy state graduation requirements.

The Digital One Room Schoolhouse (DORS) is working to create such a program by connecting students from the Silicon Valley with students in Ferguson, Missouri, around math and science project-based learning activities. DORS facilitates workshops and Common Core-aligned science and math projects with students, accredited teachers and "real" scientists and scholars from across the country. Everything is coordinated using the Edmodo platform.

Partovi appreciates folks like Project Lead The Way (PLTW) making computer science a priority. (For more on PLTW, see Getting Smart's feature).

He's also fond of startups like CodeHS with a computer science curriculum for high school. CodeHS ran a successful teacher professional development (PD) course in the summer of 2012 that trained teachers with no prior experience to become computer science teachers.

Partovi is looking for more ways to support PD for math and science teachers that can teach coding. He thinks there's plenty of demand, and "it's a lack of teachers and budget that are holding us back."

Partovi's initial strategy of inspiration and advocacy has brought a greater awareness to coding and expanded opportunities for learning and teaching. Code.org advocacy appears to be working. When his video is shown in a high school, Partovi reports "a three- to four-fold increase in enrollment."

For more on Code.org, watch this video:

I asked Partovi why we couldn't just rely on commercial sites like Udemy or Lynda to learn programs like Ruby. He said, "Learning Ruby may be one of the best vocational things anyone could learn, but I wouldn't recommend putting it in high school curriculum." Partovi would rather "teach basic problem-solving strategies like loops and functions, not specific languages."

"Coding is at the intersection of tech ed and EdTech," said Partovi. "People 'get' online computer science." He believes "it may be easier to 'sell' blended computer science than blended math."

Originally posted on December 8, 2014 GettingSmart.com

The Transition from Cursive to Coding

Authored by Adam Renfro, Educator, North Carolina Virtual Public School

My son missed a day of school last week, and when he returned, his teacher gave him directions for his makeup work that were written in cursive. He returned to that teacher later in the day, and this exchange took place:

Son: "Mrs. So-and-so, excuse me, I'm sorry, but I can't read your writing."

He gave the note back to her. She smirked at him with a disapproving look. (His words.) *Another student who didn't learn to read or write cursive.* She then looked at the note and was a bit puzzled. Something was wrong. Finally…

Mrs. So-and-so: "Why didn't you bring this to me sooner? I can't remember what I wrote!"

She couldn't read it either! *Cursive.* Indecipherable, even to its own writer.

"Learning to write programs stretches your mind and helps you to think better, creates a way of thinking about things that I think is helpful in all domains."

-- Bill Gates

Its usefulness heated up in our house last week (because we don't have real things to keep us up at night).

"It was difficult for doctors to give up leeches for modern medicine, but after the witch uprising was put down, it seemed like the right thing to do. We can shed cursive in the same manner."

That is how I waded into the deep end of the Great Cursive Debate pool, but old-school English teachers and grammarians ("old" implied) on Reddit and other forums fought back with rather vicious attacks, mostly *ad hominem* in nature. I was called an English-language traitor and much worse. Their invective, mudslinging language was right out of *Star Wars* ("wretched hive of scum and villainy") and *True Grit*, the new one, not the old one ("You, sir, are the rodeo clown from Yell County!"). Thank you, dictionary.com and urbandictionary.com for translating some of the insults for me. I was undaunted, though. They were empowered with righteousness. I had rightness on my side.

Their arguments to maintain cursive instruction in the grade school curriculum can be lumped into these airtight, data-driven arguments:

- It's tradition. (Logical fallacy in determining merit.)
- I had to learn it. (What's your position on Pluto as a planet?)
- It lets you write faster! (Myth.)
- Heaven forbid we actually make students do something hard. (You're just saying words now.)
- It's our signature. How will people know it's really us? (You're the reason we have CAPTCHA.)
- It's beautiful. (Don't confuse cursive with calligraphy.)

Meanwhile, in Estonia…
Students are learning to write code in first grade! Estonia, also known as "E-stonia," is not just a technology-driven country (that has given us Skype, among other things), but also ranks above the U.S. in reading, math and science scores. Parmy Olsen, of *Forbes*, writes, "The idea isn't to start churning out app developers of the future, but people who have smarter relationships with technology, computers and the Web."

"Google and Facebook buy entire companies, simply to gain access to their code-literate employees."

-- Doug Rushkoff

If we're looking for areas to replace outdated skills with 21st-century skills in the curriculum, this would be Ground Zero. Estonia's first graders are learning the logic skills necessary for coding at the later grades. That's uber-practical for the 21st century.

Reasons to Code

Coding is the most important language in the world right now. It's the language that enables the machines and software that we use every day. Coding also opens the door up to entrepreneurial opportunities for anyone, regardless of their background. Many coders and those who advocate for teaching students how to code do so because learning to code helps equalize the playing field for all students, making coding a social justice issue. Finally, coding is a future-ready job skill for all disciplines.

Madeline McSherry of *Slate* says it best: "Coding is the hottest skill on the job market, the modern-day language of creativity, and a powerful force in the economy." Here are four reasons why I believe students should be coding:

1. Language Study. As mentioned earlier, coding is the most important language in the world right now because it enables the machines and software we use every day.
2. Cross-Discipline Benefits. This isn't just about technology. All fields—art, music, exercise—have programming needs. If fact, it's the ultimate creative outlet for the arts in digital media.
3. Entrepreneurship. The list is long of self-made youth entrepreneurs on the Internet. Coding opens that door wide. There doesn't have to be a gap between "haves" and "have nots" here.
4. Future-Ready. Coding is a future-ready job skill for all disciplines.

Ian Quillian of MindShift identifies four other reasons to code:

1. Subject Mastery. Students need it before they can code.
2. Systems Thinking. Students must understand the inputs and outcomes of programming.
3. Collaboration. Programming is often done by a team.
4. Passion. Students can take their programming skills into any field that they are passionate about.

My own kids are both learning code through Codecademy. In less than a month, they have gone from having only basic awareness about coding to acquiring rather advanced skills (way over my head). We have a standing house rule for our kids: If they want to use anything that runs on code (computer, smartphone, gaming console), they have to complete at least one Codecademy lesson for the day. It's never an issue, though. At. All. They are intrigued by coding and find it very cool to understand how it works and to watch their own codes work. You know—those same feelings you get when you write in cursive.

During the height of my cursive writing debate, I asked my kids whether it was more important to learn cursive writing or coding. They looked at me like this was some sort of trick question, like I had lost my mind. Like this was a grasshopper-snatch-the-pebble-from-my-hand-and-become-the-master test. Then they fell on the floor laughing.

"Seriously, Dad? You're kidding, right?"

"Cursive is basically just a font, Dad!"

"Can we also learn how to write in Helvetica or Impact or Verdana or Windings?"

"We might need that one day!"

"Is BeDazzled a font?"

"Can we learn to write in BeDazzled?"

"Yeah, but only if we can use the BeDazzler!"

Experts Weigh in on K-12 Coding and CS Resources

Authored by Tom Vander Ark, founder and CEO, Getting Smart

We've had some requests for recommendations for K-12 resources for teaching coding and computer science, so we reached out to some folks in the know. Here's a summary of what we learned:

Hadi Partovi is CEO of Code.org. He thinks teaching kids to code is an economic and social justice issue. Statistics on his site make the case for coding, so it's a good place to start.

Ed Lazowska of the University of Washington has four suggestions. First, he says, "To state the obvious, there is no substitute for a motivated, curious teacher who is empowered to do great things with his/her students. That said, CodeHS has been looking pretty good to us for an intro course." Next, he mentions popular CS principles courses. "These are part of a national effort to revise the Computer Science AP exam. UW has one of these prototype courses, but probably the Berkeley class is getting most traction."

Lazowska also mentions that "David Malan's CS50 intro course at Harvard is available online both directly and through edX. It's a wonderful course, and David is a superb teacher. I know a number of (smart, motivated) middle teenagers who have loved this course online." And finally, he adds, "The CS10K community is a good place to start."

Albert Wenger of Union Square Ventures comments from an investor perspective, saying, "We of course like Codecademy, and they have an after-school kit." He adds, "Khan Academy also has some good courses."

Fred Wilson, Wegner's partner at USV, helped create the Academy for Software Engineering in NYC, which has an extensive curriculum. He mentions that "the National Science Foundation's Exploring Computer Science is becoming a standard for the intro to CS class in most high schools. That is often followed by the new AP Computer Science Principles and then the legacy AP CS class for those who want to really go deep."

Rob Hutter, Managing Partner at Learn Capital, says: "As investors, we're big believers in CodeHS. They use a special JavaScript framework that Stanford University students have been using to learn computer science for years and have tuned it for K-12 to let kids make games and other high-interest interactions. But since it's based on one of the most popular object-oriented languages in the real world, CodeHS skills have high transfer to genuine coding tasks. Also, it's getting great traction in schools because it's built for the K-12 market. Kansas City public schools just did a district-wide deal."

Hutter also likes CodeNow. "They coordinate a volunteer army of software engineers to mobilize and train inner-city students in code learning with a focus on disadvantaged populations. It's not a curriculum but a service, and they work directly on premise with high-need districts. I couldn't commend the team there more highly for their dedication and mission." CodeNow launched its six week Summer Fellowship on July 1, 2013, with a class of 20 outstanding NYC students getting smart at NYU.

Tim Brady, ImagineK12, says that in addition to CodeHS (which he incubated), "You could also include: Scratch from MIT, Codecademy and Khan Academy for offering computer science lessons." He adds, "There is a pretty good list at Code.org."

Nic Borg of Edmodo says, "Two popular apps being used by teachers and students on the Edmodo platform are LearnStreet and ManyLabs' Arduino Programmer." LearnStreet provides teachers with tools, content and analytics for teaching JavaScript, Python and Ruby. Teachers can use the LearnStreet course content as well as a library of over 100 fun practice projects.
ManyLabs provides an Arduino programmer that enables students to write, save and share Arduino programs. A Starter Kit is also available, which includes an Arduino along with a collection of sensors and other devices.

JD Hoye, National Academy Foundation (NAF) comments, "NAF is currently developing a plan to update its Academy of Information Technology theme. We envision moving toward a set of pathways within the field, including computer science. We recently had a meeting with a number of top chief information officers from major high technology companies to get their advice on technology education. Their view was that programming and coding were important skills with excellent career prospects. Their guidance was to design a program that would initially excite students about developing games or other applications with less focus on computer languages. Once students are hooked on design and development, they would be more inclined to learn more about languages and programming."

NAF's research is led by Andy Rothstein. Like Brady, he's a fan of Code.org, Scratch and Khan Academy. Like Wilson, he likes Exploring Computer Science and the new AP Computer Science Principles, which he says "is being piloted now and will be released in a couple of years."

Alison Anderson, Getting Smart teacher blogger, likes Thimble, the Mozilla Webmaker. "Thimble allows anyone to hack websites in a "sandbox" with a split screen—code on one side, what you're creating on the other." She adds, "Webmaker has hacker projects that easily work for students. My students enjoyed it the most because they could see the results of their coding in real time, it allowed the students to interact and get creative with real websites, and seemed to work right at their ability level."

Also, students at Umatilla High School in Oregon now have the chance to enroll in a new elective courses through Treehouse. With an extensive library of step-by-step video courses and training exercises, students earn badges by learning the skills needed to build apps and websites.

Adam Renfro of NC Virtual says, "My kids have to use one of these each day: Code Avengers, Codecademy, Gamestar Mechanic, Scratch or Hackety Hack. "I like Codecademy the best so far," says Renfro. And for the iPad and iPhone Renfro recommends Hakitzu and Cargo-Bot. "Hakitzu and another iPad app called Hopscotch are really great. If the kids don't have access to a computer, they use the iPad." Refro adds Hacker Scouts to his list, noting that "coding is pretty much an after-school event in most places."

Tricia Moore of Reynoldsburg City Schools in Columbus, Ohio, says their STEM elementary runs TopCoder competitions (details here). "Students participating in the FIRST Robotics Competition learned to program from Java for Dummies plus trial and error. No kidding."

Moore adds, "At our STEM middle school, students who had demonstrated mastery in the core subjects before the last week of school were able to choose among several Udacity courses, including computer science, for acceleration.... Job shadowing and other online coursework were also options. My kiddo worked through a bunch of Codecademy modules. He wrote a report explaining the different computer languages and why his teachers should use Codecademy in their core instruction."

Reynoldsburg High School eSTEM academy uses Udacity physics, calculus and computer science. Ninth grade students rotate through a nine-week, double-blocked computer science course using a portion of a Udacity course in a blended presentation with a former computer scientist teaching it. Fifteen students took the Udacity CS course this summer. School head Marcy Raymond comments, "We will also give credit for Coursera through flex credit. Students are able to apply for credit using completion of coursework with Coursera with a performance or project that demonstrates application of what they have learned."

Alex Hernandez at Charter School Growth Fund, agrees with Renfro and adds Move the Turtle and Hopscotch for iPad. Back in browser-land, he likes Khan Academy and Code Racer. He also reminded us not to forget physical computing, for example:

- Arduino: Open-source electronic prototyping platform allowing users to create interactive electronic objects
- LilyPad Arduino toolkit: Allows users to design and create soft, interactive circuits
- ModKit: Helps users make (almost) anything smarter
- Rasberry Pi: "We want to see cheap, accessible, programmable computers everywhere."
- Lego Mindstorms: Version 3.0 is coming out this year
- AgentSheets: A Web-based tool for game design
- LiveCode: Offers a teacher site
- MIT App Inventor: Allows even novice users to build Android apps

Education Week mentions "CodeEd, which focuses on teaching computer science to girls, and CoderDojo, an after-school coding club staffed by volunteers." *EdWeek* also Storified "Computer Coding: A Powerful Learning Tool."

Udemy's Dennis Yang says, "We have a great affordable library of coding/CS resources that come along with native apps on iPad and iPhone. We have private-access Udemy for organizations if they would like to have more control over their experience."

Grechen Huebner of Surfscore notes Kodable, "A tool for parents to help their K-2 grade child learn the basics of problem solving and programming." She adds, "We're already in the app store and are free to download" with a "curriculum full of additional features designed to help kids learn and adults teach programming."

And Kelly Drill of BotLogic tells us, "Our dev team is actually working on a new project to help young kids develop the logic/problem-solving skills they'll need to eventually learn coding. Kids program a robot to navigate through progressively challenging mazes and can even go head-to-head with friends in programming tournaments."

Coding for the Future

Authored by Megan Mead, Project Manager and
Math Contributor, Getting Smart

(Note: please don't let this definition stop you from reading on.)

According to Wikipedia, computer science is "the scientific and practical approach to computation and its applications. It is the systematic study of the feasibility, structure, expression, and mechanization of the methodical procedures (or algorithms) that underlie the acquisition, representation, processing, storage, communication of, and access to information, whether such information is encoded as bits in a computer memory or transcribed in genes and protein structures in a biological cell. A computer scientist specializes in the theory of computation and the design of computational systems."

If you read and understood that entire definition, CONGRATS! For the majority of us who had to read it at least a couple of times to make sense of it all, let's get excited that computer science, once a mystery to most, is now becoming more accessible and digestible than ever, especially when it comes to coding.

There is no question about it—coding is hot right now. Thanks to organizations such as Code.org, Girls Who Code, General Assembly and Microsoft Virtual Academy (and loads of others), coding has become a lot less overwhelming and a lot more accessible and empowering. This brings a new perspective to a subject that once was viewed necessary for only a select few. It is pretty difficult to argue against its value. However, there is still significant work to be done to increase access to computer science education as we look to better prepare students for the jobs of today—and tomorrow.

As educators, it is our job to equip students with the skills needed for success in college, career and life. The kicker is that in today's world we don't necessarily know what jobs we will be preparing them for. Technology changes quickly and constantly presents new opportunities. Many of our students will find themselves settling into jobs that don't even exist today, and others will decide to follow their own path, creating their own companies and careers. Regardless of the unknown future, we must develop students who:

- Think critically;
- Solve problems with perseverance and creativity;
- Anticipate challenges and look for a variety of solutions;
- Are not frightened to make mistakes but rather use the results of each attempt to create a stronger plan of attack.

Students who are respectful, collaborative and understand the importance of routine and procedure will be important contributors to a work team. Computer science is the perfect entry point for such skill development and a critical part of success in an ever-changing job market.

Millions of students around the world have invested an hour of their time to learn the basics of computer science through the Hour of Code, an initiative of Code.org that has increased the visibility, popularity and access to coding content for anyone with the Internet. The annual event receives increasing engagement and a ton of positive feedback, but we also must encourage our schools to be ready to take it to the next level. What will you do after the Hour of Code?

In support of the initiative, Microsoft launched Learn to Code through their Microsoft Virtual Academy. Designed as a focused starting point and learning path to fuel student interest in the creative world of coding, the goal is to provide a range of opportunities for learners. Students can start using the course titled "Hour of Code with TouchDevelop" and continue their coding learning journey through Python and HTML5. With over 15 additional hours of online content already available, Microsoft coders sharing their stories with students, and links for young learners to download free software through DreamSpark, students can immerse themselves in coding skills and learning resources helping them to build games and apps.

Engagement from major technology companies such as Microsoft highlights the industry's recognition of the global economic importance of the computer science movement and its potential as a tool for creating well-prepared future employees.

Early in 2014, I was able to chat with Grant Hosford, CEO and founder of CodeSpark, a game-based coding program designed specifically to engage a younger audience (ages 5-8). I asked "why coding?" and we discussed the unique benefits of coding as a jumping off point for greater interest in the broader world of computer science. We talked about how coding creates schemas for more advanced mathematics topics, and how it lays the foundation for a strong sense of sequencing and problem-solving. And as Hosford noted, "We are preparing our students for a future of careers that will undoubtedly require strong computer skills."

As we look to the future for our students, we may not be able to tell with 100 percent certainty what jobs they will take, but that shouldn't stop us from providing unique opportunities that they will take into their interviews and add to their resumes. Coding and computer science can support the next generation of learners—learners who will tackle even the most challenging of problems with anticipation and drive. The next generation of learning must address a new type of skill development, and a great place to start is with coding. In the fall of 2016, students will even be able to take an advanced placement course on Computer Science. Will your students be ready?

Coding in the Classroom: So Much More Than an Hour

Authored by Greg Garner, Instructional Technology Facilitator,
Chapel Hill-Carrboro City Schools

Last year, 20 million people participated in the Hour of Code. My hope is that you will, too. (Resources can be found here, here, here, and here.) Many educators view this as an opportunity to try something different in their classrooms or perhaps a chance to engage students in a different way. I think the very idea of learning to program is a fundamental change in education as we've known it. In a traditional K-12 setting, students were given problems that already had answers. They were taught how to work through and solve each problem and then verify that their answer matched exactly the answer of the teacher or textbook. Learning to code, however, begins with an idea, a spark. Students have to learn to identify a problem before they can begin working on it. Once they've identified the problem, they have to develop their version of the solution. Necessarily, there cannot be only one right answer.

But the Hour of Code's greatest opportunity is also its greatest threat. It's easy (relatively) to agree to give up an hour of class time for students to "play games," especially when winter break is just around the corner. It's low-commitment. You don't have to agree to transformational changes in pedagogy to participate, you just have to be willing to let your students try something new. But what happens when it goes well and students wonder why learning doesn't look like that more often? What happens when students find their curiosity piqued and want to spend much more than an hour teaching themselves how to solve problems that don't have answers? The traditional classroom doesn't have time for this kind of "playing." And so, you'll probably give your students a list of websites and resources for them to explore on their free time. You may even go so far as to create clubs and extracurricular opportunities for students to join. You might give up your lunch and allow students to practice their coding in your room while you scarf down your Sunday-night-leftovers twice-reheated. These are all noble and good things. But they fall short.

You see, when students experience this kind of authentic learning, it plants in them the notion that learning should always be like this. Learning should always be individualized. Learning should always pique their curiosity. Learning should always be a process of identifying and then solving unique and interesting problems that don't yet have answers. How naive of them! As a result, then, you run the risk of your classroom being a hindrance to their real learning. Classrooms everywhere could be seen not as places of learning but places that prevent learning. Instead of looking forward to the interesting problems they'll solve, students may very well only tolerate your class as a means to an end: the sooner they make it through math class, the sooner they can learn and use math to write an original algorithm during their lunch period to help them create.

It may not happen this year. Or next. But I think we're all at least marginally aware that we can't just stop with a single hour. We have to think about what the second hour will mean for our classrooms and students. Teaching students to code today very well could mean the need to completely re-think the way we've done education. And it's about time.

Originally posted on December 1, 2014 GettingSmart.com

Conclusion

Authored by Tyler Nakatsu, Content Coordinator, Getting Smart

As this collection of blog entries has highlighted, coding is the most important language in the world right now. With the increase in the use of technology, coding powers the machines and software that we use every day. As a language, its universality positions programming as a medium primed for collaboration. Coding also opens the door to entrepreneurial opportunities for anyone, regardless of their background.

The ability to code can level the playing field and provide access to college and careers for every student. With the uncertainty in ROI for traditional four-year postsecondary degrees, coding has become a lot less overwhelming and a lot more accessible and empowering. A Flatiron School jobs report shows that coding academies are quickly becoming an affordable alternative to college. This brings a new perspective to a subject that once was viewed as necessary for only a select few. Computer science is finally transitioning from being seen as a specific job training course to being recognized as a key element to a student's 21st-century education.

The global importance of computer science is recognizable as policymakers and companies such as Microsoft support and act on its necessity in everyday life. Recent local-, regional- and national-level policies implementing computer science, and STEM in general, in schools reflects a growing understanding of its importance.

The integration of coding curricula in schools represents a fundamental shift in the education system as we know it. As Greg Garner said, "When students experience this kind of authentic learning, it plants in them the notion that learning should always be like this. Learning should always be individualized." With coding curricula, personalized learning opportunities and competency-based and project-based learning are at the core of the student experience.

As Anthony Salcito noted in the introduction, this collection of blog entries illustrates that there are at least five important reasons for expanding access to coding and computer science learning experiences in K-12:

- Coding is ***career prep***.
- Coding is ***critical thinking***.
- Coding is ***engaging***.
- Coding is ***entrepreneurship***.
- Coding is ***collaboration***.

Every one of the skills taught by coding is essential for our students' success in the 21st century.

The next step is to find a place on the master schedule of primary and secondary schools around the globe. Computer science needs to be added to the list of math and science classes students can take to satisfy graduation requirements. It needs to be part of postsecondary institution applications, and on the mile markers of pathways to careers.

PUBLISHED BY:

GETTING SMART

IN PARTNERSHIP WITH:

Microsoft

The work of teaching coding continues- and is thriving. In 2014, 60 million students tried the Hour of Code. 60 school districts partnered with Code.org, including all 7 of the largest US school districts.

Students are passionate about coding and as this passion grows and spreads, expanding access to coding and computer science learning experiences in K-12 is vital for a myriad of reasons.

Powerful opportunities exist for young people to learn and create code. Strong partnerships between the business and education sectors help students across the globe realize their passion and potential while boosting readiness for college and careers.