Guide to inclusive computer science education

How educators can encourage and engage all students in computer science

Microsoft TEALS Program CSforALL national center for women & information technology CSTA Computer Science Teachers Association CODE

Table of contents

Introduction

Knowledge of computer science (CS) is fundamental to students' future careers. Today, 58 percent of all new jobs in STEM (science, technology, engineering and math) are in computing.[1] By 2026, there will be an estimated 3.5 million such jobs open in the United States.[2]

Computing and technology hold the promise of solving problems big and small and shaping the world around us. But if the people who work in computing-related jobs don't represent our communities and our populations, the solutions they develop won't be representative either. Technologies will emerge with unintentional bias and limited insight into the diversity of people who will use and depend on them.

This is why we need computer science classes to be inclusive. We need to support students and show them at even the youngest levels that they too can embrace the art and science of computers to solve today's challenges. And then they can ensure that solutions look like the world they live in — or even the world they want to live in.

But we know that girls and students of color are underrepresented in CS learning opportunities. Together, these groups represent 65 percent of the entire U.S. population. Yet only 28 percent of high school students who take the Advanced Placement Computer Science exam are girls, and only 22 percent are students of color.[3]

Further inequities impact access for students with disabilities and students in rural communities. To prepare young people for jobs in computing and technology, we need to help them see themselves as builders, creators, problem-solvers and computer scientists. Developing this vision requires educators to provide a clear and welcoming pathway of CS educational coursework for students.

"If we are ensuring that there are diverse teams and diverse folks at the table at every step of the pipeline, it creates the opportunity to have tech look like the world that it represents, which benefits us in a million different ways."

— Dr. JeffriAnne Wilder, NCWIT

Educators have the opportunity to teach and shape CS studies in ways that let all students know their different backgrounds, perspectives and abilities have value in the work they do and the solutions they create. This includes exposing students to a variety of role models, ensuring the classroom's physical space is comfortable and accommodating, and providing culturally relevant lessons and activities that stimulate different types of students.

This guide provides educators with context and concrete steps to build and expand inclusivity in CS education. By actively engaging students in CS, educators can build an even stronger pipeline of creativity and innovation to tackle the world's challenges and help ensure students have the skills needed to thrive today and tomorrow.

1. Employment Projections (table 1.2). U.S. Department of Labor, Bureau of Labor Statistics, 2016. https://www.bls.gov/emp/tables/emp-by-detailed-occupation.htm
2. By the Numbers. National Center for Women & Information Technology, 2018. www.ncwit.org/bythenumbers.
3. The College Board, AP Program Participation and Performance Data 2018. https://research.collegeboard.org/programs/ap/data/participation/ap-2018

Glossary

The words we use to explain inclusion in computer science (CS) are nuanced and can have varied meanings. For the purposes of this guide, here is how we define the following terms.

Access:

The right and opportunity for all students to learn and experience computer science.

Diversity:

Ensuring CS courses and programs have student enrollment rates that reflect the demographics of the larger school or community population, particularly in terms of race, ethnicity, gender and disability status.

Equity:

Equitable education ensures that learning experiences are accessible and inclusive for all learners. This means that every student has what they uniquely need to succeed.

Culturally responsive teaching and learning:

Instructional practices and learning experiences that actively take into account the context of youth in terms of interests, identities, cultural and linguistic practices, and histories. Culturally responsive education maintains and sustains students' cultural integrity while supporting academic achievement.

Inclusion:

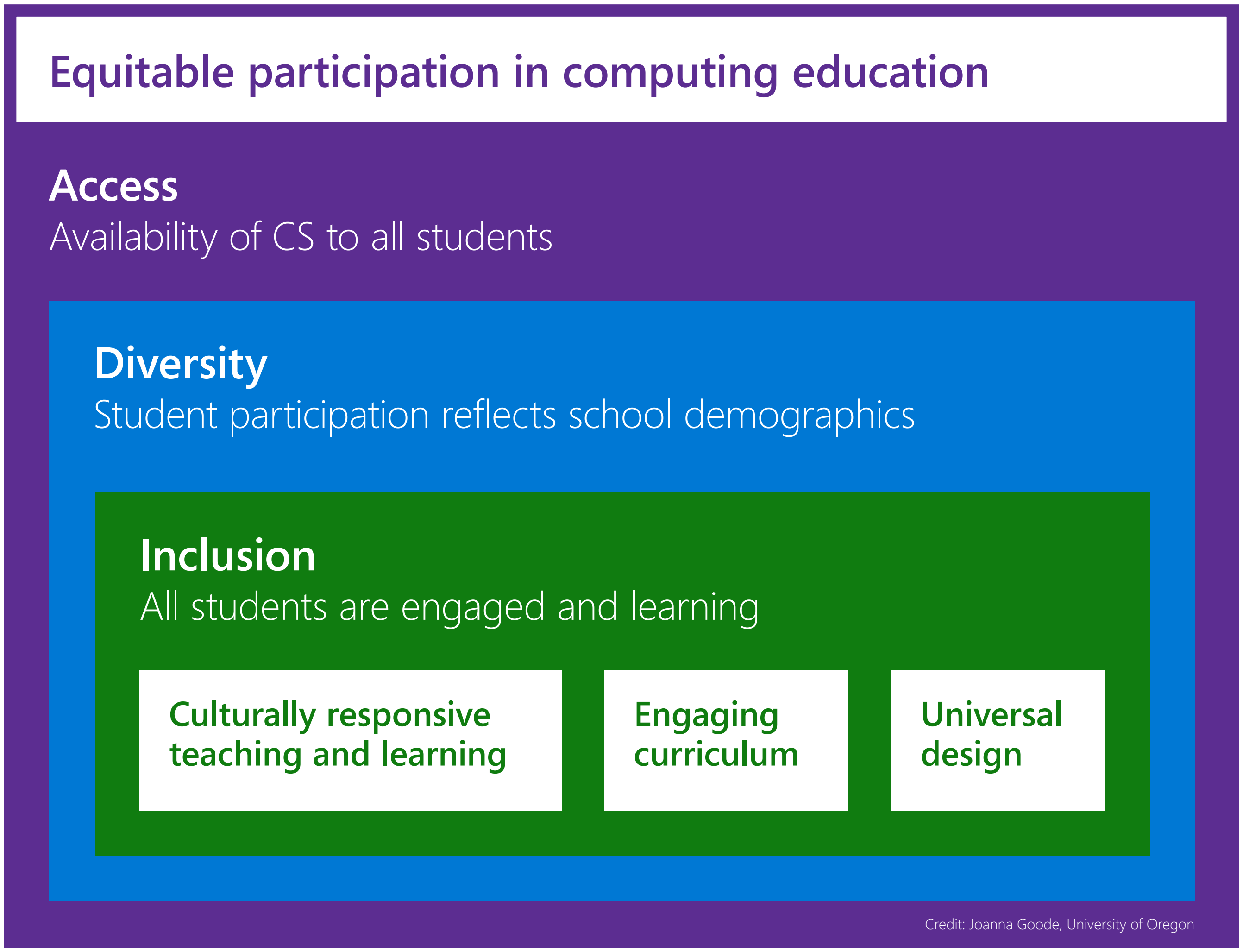
Creating learning environments that are accessible and welcoming of students' identities, backgrounds, differences and perspectives without barriers or judgment. This means actively attending to gender, race, ethnicity, ability or socioeconomic status.

Intersectionality:

The ways that a person's race, gender identity and other identity factors overlap and compound to create particular forms of advantage or disadvantage. This term is a way to describe how different marginalized groups experience equity, inclusion and power differently.

Universal design:

There are two applications of this term. In physical space, it means supporting accessibility for all types of people. This may include wheelchair-accessible desks, collaborative spaces, voice recognition software, etc. In curriculum, it refers to instruction that accounts for students' varied interests, abilities and prior knowledge.


Equitable participation in computing education
Access
Availability of CS to all students
Diversity
Student participation reflects school demographics
Inclusion
All students are engaged and learning
Culturally responsive teaching and learning
Engaging curriculum
Universal design

Credit: Joanna Goode, University of Oregon

Access

Evaluating students' learning opportunities for CS starts with understanding how each student can participate in CS learning.

The first step for any school is to offer CS! Though creating new courses and school-wide support often depend on the leadership of school principals, teachers can play an important role in advocating for CS education for students and planning for equitable access to bring CS to all students.

"When computer science courses are required, then CS classes represent the whole student body."

— Dr. Leigh Ann DeLyser, CSforALL

Considerations

Offer CS learning opportunities in elementary school

- Elementary teachers report that CS can be effectively integrated alongside other subjects' learning standards for younger students. Consider how you might teach CS concepts within math, science, art, language arts or other content areas.
- Provide students with developmentally appropriate CS learning experiences as early as kindergarten. This will ensure they have exposure to the field and can build their confidence so they will keep pursuing opportunities as they arise. See an example of K-5 CS in action at **Frederick Douglass Elementary**.

Access (continued)

Offer CS learning opportunities in middle and high school

- Work to make CS a required course at your school so that there are enough course sections for all students.
- If there is limited availability for CS course enrollment, make sure that the student selection process is not based on assumptions about which students are interested or will do well in CS. Provide teachers, counselors and parents with information about why it is important for all students to take computer science. The **CS Is for Everyone Student Recruitment** toolkit offers helpful ready-to-use resources.
- Leslie Aaronson of NCWIT says that one of students' biggest obstacles to CS education is assuming they need to already be good at it before they try it. Be clear to students that they don't need to already know CS or be a "math whiz" before they enroll.
- Give students an authentic, meaningful chance to grow in their CS learning. According to Dr. Allison Scott, chief research officer at the Kapor Center, it's common to assume that courses like office skills or digital media are teaching computer science — but they aren't. Work with your administrators to understand the value of CS and how it differs from other technology-related courses. Also, Scott recommends offering increasingly advanced courses in CS, so that students who want to pursue it have plenty of runway to continue their learning.
- Consider innovative models like the **Microsoft TEALS** (Technology Education And Literacy in Schools) program which uses technology industry volunteers to co-teach computer science in high schools across the U.S. and British Columbia, Canada.

How do you define CS?

The study of computers and the processes and principles used to make them do useful things for society.

Ensure CS classes are open to all abilities

- Consider other barriers that could be keeping students from accessing CS classes. If students are visually impaired, for example, are there tools that will support them in the classroom? **AccessCSforAll**, a program led by the University of Washington and University of Nevada, Las Vegas, provides resources, tools and materials for making CS instruction inclusive for students of all abilities.
- Look at when CS lessons are offered at your school. Are they at times when some students leave the classroom for other activities, such as to receive special education services? If a child is being pulled out of the classroom for other interventions during CS instruction, talk with an administrator to ensure that child can stay and participate.
- Consider a co-teaching arrangement to offer different teacher perspectives and better help different types of students relate to or engage with their instructors. For example, a CS teacher can partner with a special education teacher, bilingual teacher or instructional coach. The **TACTICal Teaching Brief** offers tips for effective CS co-teaching.

Diversity

To ensure CS classes represent your student population, conduct targeted recruitment. Look at your existing CS classes and learning opportunities. Chances are, unless your CS courses are core requirements or integrated into the curriculum of elementary classrooms, they are leaving some populations out.

As you'll see below, increasing diversity is a team effort, with teachers, counselors and even other students helping spread the word and setting an example.

Considerations

Build support for CS throughout your school system

- Enlist the support of your school ecosystem — administrators, teachers, guidance counselors, families and students — to enhance communication and understanding around CS opportunities. Microsoft Philanthropies provides two free resources — the **Girls in STEM Action Guide** and **Computer Science Is for Everyone** recruitment toolkit — that will help all of you be inclusive ambassadors for CS.
- See the **Computer Science Professional Development Guide** for tips.
- Teachers can earn a badge when they take the **Closing the STEM Gap Microsoft Educator Community (MEC) Course** to learn how to help close the gender gap in computer science.

"If you change the way that guidance counselors think about who is right for computer science, that changes who they recommend for the course. And then you have students giving it a try who never would have done that before."

— Leslie Aaronson, NCWIT

Diversity (continued)

Include guidance counselors in your efforts

- School counselors can be excellent champions for CS courses. But they can also unknowingly filter out students before they have a chance to try CS. Work with your counselors to help them understand what CS is about and who's a good fit for the courses (hint: everyone!).
- **NCWIT Counselors for Computing** offers valuable resources and information for counselors to support all students in exploring CS, including a **Top 10 Guide** for how teachers can engage counselors as allies.

Enlist students to promote CS

- Ask current CS students to promote CS education and share with prospective students what they might learn and create. Focus their "peer presentation" energy around course enrollment time, and coach them to be inclusive and sensitive to all kinds of students when reaching out.
- Generate excitement with a steady narrative about how **CS careers are creative and critical** to solving real-world problems students care about. Put up **posters and displays** that highlight the creative possibilities in CS.

"I tell students the people behind the cool technologies we use should reflect the same types of people who use it."

— Doug Bergman, Porter-Gaud School

Introduce students to diverse role models in CS

- Role models matter for CS inclusion. Think about inviting in a diverse group of educators, **guest speakers** and other role models who can connect with students in different ways. Teachers and other adults sharing their backgrounds and personal stories can be **very valuable** in helping students make links with computing.
- Address the intersectional differences of girls of color by introducing students to female role models of various races and ethnicities. Visit **FabFems** to learn more about facilitating these experiences.

Learning space

The learning environment of the classroom itself is also very important to making students feel included. This is true not only for the students who are enrolled but also for those who are coming in for a tour or first exposure to CS.

As Leslie Aaronson of NCWIT says, "Just looking at a typical computer lab could discourage some people from stepping in, if it feels like a place they don't belong."

Considerations

Incorporate visuals that will appeal to a wide range of student interests and backgrounds

- In order for us to feel like we belong somewhere, it helps to see depictions of people we relate to. Hang **posters** showing a diverse range of people engaging in CS.
- Make sure that the visual displays are relevant in terms of gender, race and age, as well as time period. Technology evolves so rapidly that the way we pictured CS a few years ago might seem dated and irrelevant to students today. Current images, grounded in the world that students know, will help them feel welcome.
- Don't forget examples of CS students and professionals with disabilities. The Alliance for Access to Computing Careers has a great **list of real-life profiles** you can draw from.

"Have representations of a diverse range of figures in computing, and people who have different types of roles in STEM as well. Students are in that classroom every day, so those signals can be very powerful reminders to them."

— Frieda McAlear, Kapor Center

Feature examples of real-world applications of CS in your classroom

- We know that students — girls in particular — are drawn to subjects and careers that have a positive impact in the world. So find ways to represent the **cool and important innovations that depend on CS knowledge**.
- To help students connect CS to the other academic areas they're drawn to, provide examples of how CS is impacting and influencing **everything from the sciences to the humanities and the arts**.

Learning space (continued)

Display student projects and contributions

- Many CS projects have **tangible outputs** — like a visual design, electronic textiles or even something robotic. Putting students' work on display helps sustain interest and pride among current students. It also provides further proof to visitors and newcomers that CS is creative and dynamic.

Arrange the learning space to promote collaboration and hands-on activities

- The classic "desks in rows" setup can imply that CS is a rote subject that everyone should learn individually, through direct instruction from the teacher. If possible, design the room so learning can be guided rather than merely broadcast, and so students can share and try things out together rather than simply listening along.
- Consider having multiple types of spaces in your room, or changing the formation of the room depending on what type of lesson you're focusing on. For example, you can create spaces for both **"plugged"** and **"unplugged"** work, or for both programming assignments and more interactive labs.

Design learning spaces that are accessible to students with diverse abilities

- Arrange aisles and work stations so that students with wheelchairs or other physical mobility aids can get to all the areas they need to access in order to participate fully.

Make sure that technology resources support the needs of students

- Be sure that technology requirements or instructional tools you choose to use in the curriculum are consistently available and accessible for students. If curricular materials require regular or daily use of the internet or technology that is not consistently reliable, you might want to select resources available offline.

Instruction

Just as you consider the physical space of your computer lab or CS classroom, remember to also think about how to teach in a way that includes all students. Building a community of support, inquiry and collaboration is especially important for retaining all students.

Research shows that inclusive CS instruction is guided and that its success depends on teachers' skilled facilitation for active learning. Use the effective practices and approaches you already have for teaching other subjects and apply them with your CS students. Teachers without a CS background have found that **CS professional development programs** with a focus on equity and inclusion are important for developing the instructional skills necessary to teach CS and broaden participation in computing.

Considerations

Differentiate learning

- Differentiate learning for students who might need different types of support, such as English-language learners or students with disabilities.
- Use the **Universal Design for Learning framework** to give students multiple pathways to access information and demonstrate their understanding.

Encourage students to focus on the problem-solving process

- In CS, there can be multiple solutions to a problem. In fact, that's why we need an increasingly diverse cadre of computer scientists in our world. When we have all different kinds of knowledge and perspectives working to solve problems, we will get all different kinds of solutions. So focus your instruction — and encouragement — on solving problems rather than finding a single right answer.
- Emphasize guided inquiry. Design learning opportunities where students can ask questions, explore, try different approaches, and challenge their own and each other's ideas.
- Encourage students to take ownership over their own learning and reflect on their problem-solving process. Ask students about the CS strategies they are using.
- Highlight that mistakes in computer science are called "bugs," a term first coined by CS pioneer Dr. Grace Hopper, and that "debugging" is a central and defining practice in the field of CS. To celebrate debugging, share a (willing) student's error with the rest of the class each day as a learning opportunity — "My favorite bug of the day! Let's figure it out together."

Instruction (continued)

Support students in taking risks

- Build a community. Some students may initially feel out of place in your CS class. Just showing up might be a risk. You can put them at ease by promoting collaboration, peer-to-peer learning and small-group work to help them build relationships and more comfortably share ideas.
- Encourage exploration and creativity by resisting giving students the answers and immediate solutions. Support a "growth mindset" — let students struggle a bit, and reward their ideas and effort rather than their final product.
- Incorporate journal writing as an activity that allows students to gather their thoughts and ideas before sharing out to peer-partners, small groups or a larger class discussion. This will give time for all students to think about their own ideas around a particular topic before listening to others speak in a larger discussion.
- Help students scaffold to more independent thinking. Students with learning disabilities, for example, may struggle with complex, multistep problem solving. Give them additional supports in the beginning, then slowly remove the supports once students build their skills and confidence.
- Create opportunities to encourage students to learn outside the classroom, such as through internships, community college courses and summer programs.

"I saw one team of girls who wanted to create an app that would warn people with asthma about poor air quality, because it was something directly relevant to their lives."

— Dr. Allison Scott, Kapor Center

Use culturally responsive teaching practices

- Dr. Scott, of the Kapor Center, suggests focusing on the foundational ideas of culturally relevant pedagogies: "The idea of the three R's of **rigor**, **relationships** and **relevance** are a great way to capture what needs to happen in computer science."
- Maintain high expectations for all students to counter stereotypes about who should excel in CS. We know that one hurdle to more diversity in CS is students' own belief systems about who can succeed in computer science. Encourage students to reflect on their perspectives and potential biases, and challenge yourself to do the same.
- Build relationships with students to identify opportunities to connect learning to their personal experience. Share your own story and bring in other CS instructors or professionals. Look for stories and experiences about using CS that will be meaningful and relatable to your students.

Instruction (continued)

- Grounding concepts in the real world is good practice in any classroom. Ensure your CS lessons take into account students' cultural experiences and realities. Look for connections to current sociopolitical issues, such as climate change or voting security, or relate computing to pop culture.
- Acknowledge how issues of power and privilege, particularly in the CS realm, have a history of marginalizing groups of people through innovations and normative professional practices. Encourage students to identify how policies and collective agency might disrupt these forces.

Expand your own knowledge and skills

- Expand your lens through professional learning networks, continuing education and online communities. Join the **Computer Science Teachers Association**, read its **newsletter**, explore the **CS for All Teachers** community of practice, or follow blogs or social media feeds relevant to CS education.
- Consider online courses such as **Strategies for Effective and Inclusive CS Teaching** where teachers can gain the insights, skills and strategies to create equitable CS programs.

Case study: Ariana Grande and linked lists

What do pop singers and Python have in common? Ariana Grande's hit song "Thank U, Next" turns out to be a great tool for teaching students about linked lists. The structure of the song's lyrics, as Grande talks about her ex-partners, reflects how a computer programmer would write a linear data set — giving teachers an extremely relatable way to illustrate a tough CS concept!

Frieda McAlear, a researcher at the Kapor Center, encourages CS teachers to look for current and culturally responsive examples like these to help students understand and feel comfortable with CS.

Curricular materials

The final ingredient of building an inclusive classroom is the coursework itself. Decisions around the adoption of curricula are often made by administrators, so it's important that they work together with teachers to make the best decisions for their school and students.

Considerations

Give students variety and choice in their learning experiences

- Allow students to incorporate a range of features and aesthetics into their work, while ensuring all students meet the learning objectives.
- Look for opportunities to engage students in project-based learning around CS. More extensive projects can allow opportunities for students to draw on their cultural assets and community knowledge while developing their CS knowledge and skills.
- Prioritize hands-on learning experiences, such as the **MakeCode for MicroBit Curriculum**, to highlight the creative and problem-solving aspects of CS.

Exploring Computer Science (ECS) curriculum

Exploring Computer Science is an introductory high school curriculum that provides a progressive sequence of lessons, includes validated assessments, and supports inclusion and culturally responsive pedagogy throughout the instructional materials. Visit **exploringcs.org** to learn more about professional development opportunities and incorporating the curriculum in your school.

Sequence lessons and assessments in a cohesive progression of learning

- Because CS is still quite new to many schools and teachers, curricula often consist of a series of discrete skills and activities. But just like math, history and other subjects, CS is best taught as a cumulative subject where each lesson builds on the one that comes before it. To find progressive CS lessons:
    - For elementary schools, see **CS Fundamentals** and **Coding with Minecraft**.
    - For middle school, explore **CS Discoveries** and **MakeCode**.
    - For high school, consider **Exploring Computer Science**, **TEALS Intro to CS**, and **Advanced Placement CS Principles**.
- Make sure that student performance metrics align with your curriculum — covering the same content and skills and making the same considerations for students' contexts.

Curricular materials (continued)

Select curricular materials that highlight diversity and inclusion in meaningful ways

- As with visual representations in the classroom, instructional material should feature diverse cultures and communities to showcase the historical and cultural integration of computing and diverse people who do computing.
- However, avoid presenting any "essential identity" or experience for any group of people based on race, gender, sexuality or (dis)ability. Students should understand that people in any identity group are individuals first and foremost. So present an intersectional approach to accurately understanding the particular needs and lived experiences of people.
- Include lessons and assignments that build on the cultural assets, prior knowledge and interests of students to feature CS in contexts that are interesting for students.

"Look for a curriculum that has threads of universal design, rather than choosing a narrow curriculum and trying to make it inclusive."

— Dr. Maya Israel, Creative Technology Research Lab

Incorporate learning materials that are accessible for students of all abilities

- Choose curricular materials that follow the **Universal Design for Learning** framework. This framework improves and optimizes teaching and learning for all people, by designing lessons that are conscious of a variety of learning barriers and supports. The alternative is teaching a curriculum that caters to specific kinds of teaching, and making it inclusive as an afterthought.
- Students who can't see, can't use a keyboard or have other physical restrictions should still be included in CS. And many tools and curricula make CS accessible. **Quorum**, **CodeJumper** and **Blocks4All** offer coding opportunities for blind learners. **Web Design and Development** (WebD2) is a free course curriculum that integrates accessibility into lessons on site planning, HTML coding and other aspects of web design.
- It's important to scaffold learning so that students benefit from supports during initial phases of learning. As they build skills and confidence, those supports can be removed.
- View the video **How Can We Include Students with Disabilities in Computing Courses** to meet CS students and professionals with disabilities and see how accommodations, assistive technologies and universal design approaches can make computing courses accessible for students with disabilities.
- There are approximately 7.6 million students with disabilities in the U.S. To learn more about making CS accessible to them, check out the **CSforALL Accessibility Pledge**.

Resources

Additional reading

The Kapor Center's Leaky Tech Pipeline report on the lack of diversity in technology-related professions and fields of study: **https://www.kaporcenter.org/the-leaky-tech-pipeline-a-comprehensive-framework-for-understanding-and-addressing-the-lack-of-diversity-across-the-tech-ecosystem/**

"That Classroom 'Magic'" article about sparking participation and interest from a wide variety of students: **https://cacm.acm.org/magazines/2014/7/176208-that-classroom-magic/abstract**

Access

Computer Science Professional Development Guide: **https://aka.ms/CSPDguide**

Computer Science Is for Everyone Student Recruitment Toolkit: **https://www.microsoft.com/en-us/digital-skills/resources**

AccessCSforALL resources for a range of student abilities: **https://www.washington.edu/accesscomputing/accesscsforall**

Additional CS education research: **https://csedresearch.org/**

Diversity

Girls in STEM Action Guide for education and nonprofit leaders, teachers and parents: **https://aka.ms/stemactionguide**

FabFems: **https://www.fabfems.org/find**

NCWIT Counselors for Computing resources: **https://www.ncwit.org/project/counselors-computing-c4c**

NCWIT Top 10 Guide for engaging counselors as allies: **https://www.ncwit.org/resources/top-10-ways-engage-school-counselors-allies-effort-increase-student-access-computer**

Meet Code Creators video series from Code.org and Skype in the Classroom: **https://aka.ms/codecreators**

Diversity posters and displays from Code.org: **https://hourofcode.com/us/promote/resources#posters**

Find diverse guest speakers in CS through Skype in the Classroom: **https://education.microsoft.com/skype-in-the-classroom/find-guest-speakers**

TACTICal Teaching Brief for effective CS co-teaching: **https://ctrl.education.illinois.edu/TACTICal/coteaching**

Resources (continued)

Learning space

Profiles of CS professionals and students with disabilities from Alliance for Access to Computing Careers: **https://www.washington.edu/accesscomputing/resources/choosecomputing/profiles**

Instruction

Universal Design for Learning framework: **https://ctrl.education.illinois.edu/TACTICal/udl**

Computer Science Teachers Association (CSTA) information and membership: **https://www.csteachers.org/**

CSTA newsletter: **https://www.csteachers.org/page/CSTAVoice**

CSforAll teachers community of practice: **https://csforallteachers.org/**

Strategies for Effective and Inclusive CS Teaching course by the University of Texas at Austin: **https://stemcenter.utexas.edu/strategies-effective-and-inclusive-cs-teaching**

Curricular materials

MakeCode for MicroBit Curriculum for hands-on learning: **https://makecode.microbit.org/courses/csintro**

Code.org CS Fundamentals (elementary school): **https://code.org/educate/curriculum/elementary-school**

Coding with Minecraft (elementary school): **https://education.minecraft.net/class-resources/coding-with-minecraft**

CS Discoveries (middle school): **https://code.org/educate/csd**

Exploring Computer Science (high school): **http://www.exploringcs.org/curriculum**

TEALS Intro to CS (high school): **https://tealsk12.gitbook.io/intro-cs/**

AP CS Principles (high school): **https://apcentral.collegeboard.org/courses/ap-computer-science-principles/course**

Quorum programming language: **https://quorumlanguage.com/**

CodeJumper coding materials for people across the visual spectrum: **https://codejumper.com/**

Blocks4All accessible programming: **https://stemforall2018.videohall.com/presentations/1078**

Web Design and Development (WebD2) course overview: **http://www.washington.edu/accesscomputing/webd2/**

CSforALL Accessibility Pledge: **https://www.csforall.org/projects_and_programs/accessibility-pledge/**

How Can We Include Students with Disabilities in Computing Courses video: **https://www.washington.edu/doit/videos/index.php?vid=64**

Contributors

Information in this guide was developed in collaboration with the following partners. A special thank you to **Dr. Joanna Goode**, University of Oregon, who was a valued lead consultant on this project.

Leslie Aaronson, National Center for Women & Information Technology (NCWIT)

Jake Baskin, Computer Science Teachers Association (CSTA)

Doug Bergman, Porter-Gaud School

Callista Chen, Tech Bridge Girls

Dr. Leigh Ann DeLyser, CSforALL

Lien Diaz, Educational Innovation and Leadership, Georgia Tech College of Computing

Dr. Maya Israel, Creative Technology Research Lab

Dr. Andy Ko, University of Washington

Frieda McAlear, Kapor Center

Brook Osborne, Code.org

Dr. Allison Scott, Kapor Center

Dr. JeffriAnne Wilder, NCWIT

About Microsoft Philanthropies TEALS Program

The TEALS (Technology Education and Literacy in Schools) Program helps high schools develop and grow inclusive and sustainable CS programs. Volunteers support teachers as they learn to teach CS independently over time. Through TEALS, technology professionals share their knowledge with teachers while students benefit from learning how computer science is used in the workplace. TEALS operates in U.S. states, Washington, DC, and British Columbia. Since its inception, TEALS has provided nearly 75,000 students with access to computer science.

Learn more at **microsoft.com/teals**.

Microsoft skills for employability

Microsoft is working with nonprofits to ensure every person has the skills, knowledge, and opportunity they need to succeed in the digital economy.

Learn more **aka.ms/skills-employability**

To know more on Microsoft Philanthropies, visit **microsoft.com/philanthropies**

Microsoft