

## Culturally Responsive Pedagogy and Equity in CSA

### Contents

<b>CSA Pedagogical Threads</b>	<b>1</b>
<i>Overview of the pedagogical threads that make up the CSA curriculum</i>	
1. Identity and Culture	4
2. Personally Relevant Practice	5
3. Processing and Meaning Making	6
4. Software Engineering for All	7
<b>Strategies Used in the Curriculum</b>	<b>8</b>
<i>Overview of key strategies used in the CSA curriculum that incorporate the pedagogical threads</i>	
<b>Digging Deeper</b>	<b>12</b>
<i>Resources for additional information and strategies</i>	

### Our Approach to Equity in CSA

In 2020, underrepresented racial/ethnic groups<sup>1</sup> only made up 15% of AP<sup>®</sup> CSA exam takers. Female students made up 25%<sup>2</sup>.

To address these gaps and needs, the Code.org Computer Science A (CSA) curriculum incorporates culturally responsive strategies to influence attitudes and perceptions towards software engineering by increasing engagement, confidence, and sense of belonging with strategies that ensure equitable learning opportunities and outcomes. We believe it is important to build "the learning capacity of diverse students who have been marginalized educationally"<sup>3</sup> while learning the Java programming language and software engineering skills so that **all** students experience success in the CSA classroom.

In particular, the Code.org CSA curriculum aims to positively impact equity by designing content and activities that support teachers in providing students with multiple avenues to learn and understand the CS content so that all students can be successful in the course. At Code.org, we work to identify and eliminate barriers that have prevented the inclusion and full participation of underrepresented groups in CS.

### CSA Pedagogical Threads

Our approach to implementing culturally responsive pedagogy and addressing equity in CSA consists of four main threads:



- 1. Identity and Culture:** Affirm the identity and culture of students to create a sense of belonging.
- 2. Personally Relevant Practice:** Provide opportunities for students to practice what they are learning in a meaningful way.
- 3. Processing and Meaning Making:** Ensure students interact with content and process new information in a way that works best for them.
- 4. Software Engineering for All:** Expose students to the skills and common practices used in the industry by taking on the role of software engineers.

<sup>1</sup> URG or underrepresented racial/ethnic groups refers to students from marginalized racial/ethnic groups underrepresented in computer science including students who are Black/African American, Hispanic/Latino/Latina/Latinx, Native American/Alaskan, and Native Hawaiian/Pacific Islander

<sup>2</sup> <https://code.org/promote/ap>

<sup>3</sup> Zaretta Hammond, Distinctions of Equity, [https://crtandthebrain.com/wp-content/uploads/Hammond\\_Full-Distinctions-of-Equity-Chart.pdf](https://crtandthebrain.com/wp-content/uploads/Hammond_Full-Distinctions-of-Equity-Chart.pdf)

## 1. Identity and Culture

**What it is:** Stereotypes associated with software engineers and the computer science industry can make students feel like computer science is not for them. We believe that when students feel a strong sense of belonging in the CSA classroom, they experience greater success. Affirming personal and cultural identity builds trust, which in turn helps transition students from dependent learners to independent learners as they grow in the four dimensions of an academic mindset identified by Zaretta Hammond<sup>4</sup>.

As students become independent learners and develop an academic mindset, they then feel supported to put forth effort, share their perspectives, ask questions, and take risks to expand their knowledge and skills.

Effectively affirming students' identity and culture to support their growth as independent learners means focusing on understanding deep culture impacts how students make sense of information and events. It is important to understand the levels of culture and the impact deep culture has on how students learn information<sup>5</sup>.

### Beliefs of Students with an Academic Mindset

from Zaretta Hammond's *Culturally Responsive Teaching & the Brain*

**I belong to this academic community.** A strong sense of academic belonging where students see themselves as members of not only a social community, but an intellectual community.

**I can succeed at this.** The degree to which students believe they are "good" at a particular kind of task or field of study is strongly associated with academic perseverance.

**My ability and competence grow with my effort.** The degree to which students have a growth mindset means they are more likely to interpret academic challenge or mistakes as opportunities to learn and develop their brains.

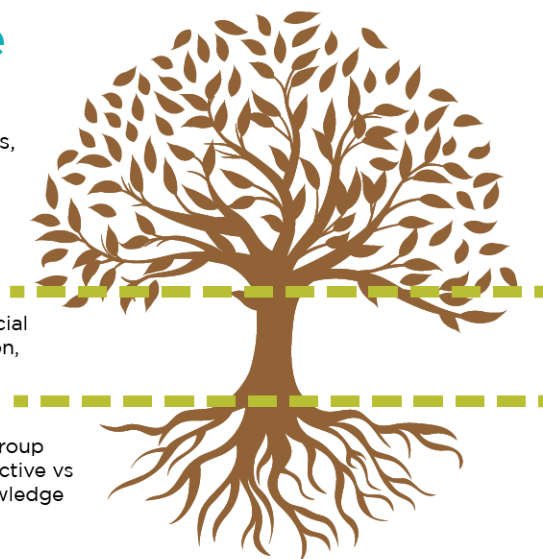
**This work has value for me.** The intrinsic value placed on academic tasks and topics that connect in some way to students' lives, future educational pursuits, or current interests.

## Levels of Culture

**Surface Culture:** observable elements, such as food, music, dress, holidays

**Shallow Culture:** unspoken rules and social norms, such as nonverbal communication, eye contact, personal space

**Deep Culture:** worldview, core beliefs, group values, cultural archetypes such as collective vs individual, mental models, funds of knowledge



<sup>4</sup> Zaretta Hammond, *Culturally Responsive Teaching & the Brain*. (California: Corwin, 2015), p.75-118

<sup>5</sup> Zaretta Hammond, *Culturally Responsive Teaching & the Brain*. (California: Corwin, 2015), p.22-24

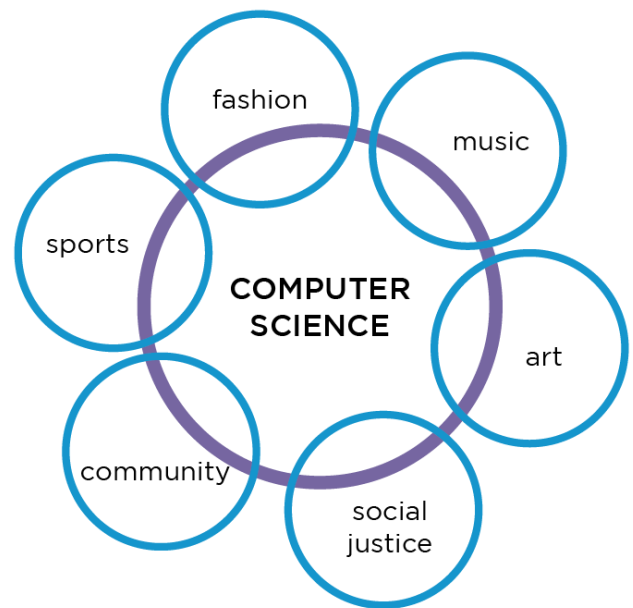
**How we do it:** Our curriculum incorporates strategies and activities to support teachers in connecting with and validating students from diverse backgrounds. We help teachers make authentic connections with their students as they learn about their personal interests and strengths and support their growth. We want students to feel that they belong and their voice is important in the CSA classroom and in the CS community. Through our curriculum, students see representations of themselves in the computer science industry to help them see what is possible while also normalizing doubt. Our curriculum gives students the time and space to work through problems and explore new concepts like a software engineer and focuses on developing their skills in the problem-solving process to encourage persistence, effort, and engagement.

## 2. Personally Relevant Practice

**What it is:** Students need to know why the concepts they are learning matter and how they will be used in the real world. Practice problems should allow students to apply new knowledge in a way that supports their growth while also providing relevance for the work they are doing. As noted in the *Guide to Inclusive Computer Science Education*<sup>6</sup>, activities should "build on the cultural assets, prior knowledge and interests of students to feature CS in contexts that are interesting for students." Practice problems that are meaningful and relevant to students in this way build confidence and interest in learning computer science.

**How we do it:** Meaningful practice is different for each student, so it is important to provide multiple means of engagement to meet the needs of various learning preferences and personal interests that are present in the CSA classroom. We believe that providing choices in achieving lesson and unit objectives helps students feel more self-confident and connected to their learning<sup>7</sup>. Practice that is differentiated by skills or topics helps students build their confidence as they experience success on problems that are at their personal skill levels and increase their personal connection to the content as they use programming to explore their interests.

Additionally, we want students to make connections between the work they are doing in the classroom and the work of software engineers to help them see the authenticity and relevance of the concepts they are learning. The Universal Design for Learning Guidelines notes that students are more engaged when activities are relevant and valuable to their personal interests<sup>8</sup>. We believe practice should not be disconnected from the work of software engineers and should give students opportunities to explore how computer science can be used in various aspects related to their personal interests.



<sup>6</sup> Microsoft, *Guide to Inclusive Computer Science Education: How educators can encourage and engage all students in computer science*, [ncwit.org/resource/csedguide](https://ncwit.org/resource/csedguide), p. 16

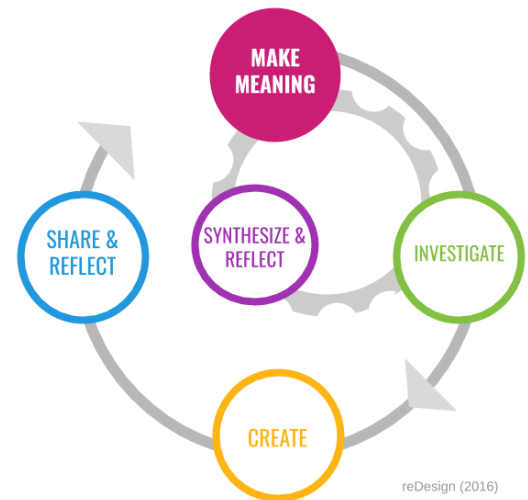
<sup>7</sup> CAST, *The UDL Guidelines*, [Optimize individual choice and autonomy](#)

<sup>8</sup> CAST, *The UDL Guidelines*, [Optimize relevance, value, and authenticity](#)

### 3. Processing and Meaning Making

**What it is:** Students have different ways and preferences for processing and making meaning of new information according to their individual deep cultural roots. At the deep culture level are the mental models and funds of knowledge that students use to make connections between prior knowledge and the new content. Students need opportunities to process and reflect to make these connections through investigation, synthesis and reflection, and meaning-making phases for the new information to become part of working memory. Additionally, reDesign notes the importance of these meaning-making opportunities in the Learning Cycle to<sup>9</sup>:

- Orient learners to the lesson purpose and success criteria
- Activate prior knowledge and cultural references
- Pose a provocation for collaboration and productive struggle
- Introduce new material, while explicitly teaching skills and strategies



Within each phase of the learning cycle during a lesson and throughout a unit, students need opportunities to interact with new content and process new information in a way that uses their prior experience and frames of reference. Effective processing and meaning-making activities support students in using their mental models and funds of knowledge to investigate and synthesize new content through multiple representations of information, scaffolds for information processing, and progressive release of information<sup>10</sup>.

**How we do it:** We believe it is important for students to have opportunities to interact with content and process new information in a way that works best for them. It is difficult for students to engage with new content if it is presented in a way that does not meet their individual needs or connect with their deep cultural roots. We incorporate varied examples and use different formats to provide multiple opportunities to interact with new content. Additionally, we focus on eliminating barriers to learning to help students interact with and process new information to meet the needs of students who need these additional supports while encouraging productive struggle in the problem-solving and learning process.

<sup>9</sup> <https://www.redesignu.org/creating-engagement-and-connectedness-through-making-meaning/>

<sup>10</sup> CAST, *The UDL Guidelines*, [Guide information processing and visualization](#)

## 4. Software Engineering for All

**What it is:** We believe all students can succeed in computer science, and it is foundational for many different career paths. In this curriculum, students take on the role of software engineers and practice skills that are used in the field. The Software Engineering for All narrative helps students visualize themselves as software engineers through representation in our videos and exposure to common practices used in the tech industry.

**How we do it:** We highlight specific characteristics to help students make connections between their personal skills and characteristics and those of software engineers. Throughout the curriculum, students participate in activities that help them practice and develop these characteristics and celebrate their growth and progress. The software engineers featured in our videos share examples about how these characteristics impact their work, and students frequently reflect on how these also impact how they solve problems and develop programs in the CSA classroom.



**Problem Solver**



**Creative**



**Collaborator**

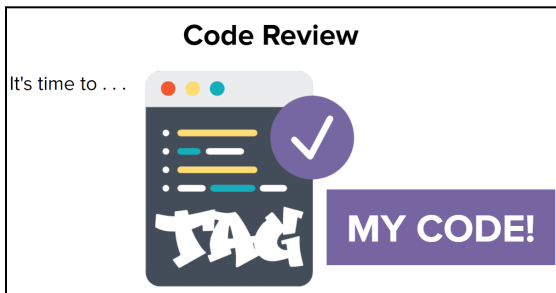


**Learner**

## Strategies Used in the Curriculum

Culturally responsive pedagogy does not mean that a teacher must learn the details of all cultures in the classroom or that it is not applicable in less culturally diverse classrooms. These strategies are effective for students from all backgrounds and needs to support learning and academic success.

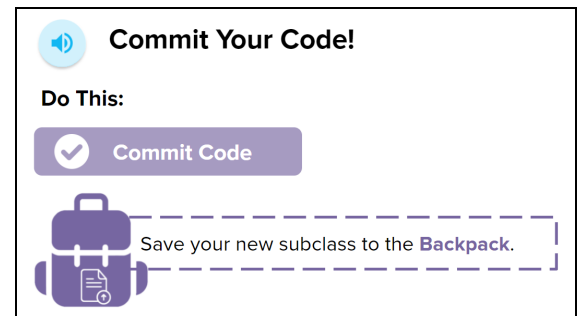
Zaretta Hammond notes that, in many oral cultural traditions, learning and storytelling begin with an attention-getting activity, such as drumming, chanting, music, and hand-clapping. Using culturally-oriented attention-getting strategies is effective in grabbing attention because these are similar to what students experience at home or in their community<sup>11</sup>.



We incorporate **Call and Response** in the curriculum to help build a positive community environment and create routines in the CSA classroom. Call and Responses grab the attention of students to alert them that something is about to happen<sup>4</sup>. For example, students participate in a Call and Response when it is time to initiate a Code Review to bring the class together as a community and ignite excitement around giving and receiving feedback on their work. Teachers start the routine with a call ("It's time to...") to grab students' attention. Students respond to the

call ("TAG my code"), bringing the class together and initiating the code review activity.

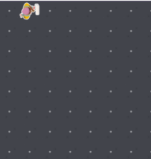
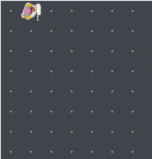
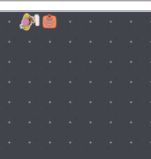

Additionally, we use **musical cues** in the curriculum that are embedded in the unit presentation slides to signal the start of an activity and affirm the musical aspects of many cultures. Musical cues are an effective culturally-oriented attention-getting strategy, as this is similar to experiences students have at home or in the community<sup>4</sup>. For example, teachers are encouraged to play a musical cue on a slide to alert students to commit their code and save new versions of their classes. These musical cues increase excitement and engagement, especially for students from cultures that use music to cue the start of a story or cultural tradition.



We also incorporate **quotes**, **interesting problems**, and **videos** that highlight the achievements of software engineers from underrepresented groups in the computer science industry. Students are encouraged to imagine the possibilities of what they can achieve and explore the intersection of their personal identities and the skills and characteristics of software engineers. Students participate in activities to share their perspectives and thoughts with their peers about these problems and potential career paths to further strengthen their sense of belonging in the CS community.

<sup>11</sup> Zaretta Hammond, *Culturally Responsive Teaching & the Brain*. (California: Corwin, 2015), p.128-129



	<p><b>a isOnBucket()</b></p> <p>Use an if statement and the isOnBucket() method to take all of the paint from the paint bucket if the Painter object is standing on a paint bucket.</p>		<p><b>b isOnPaint()</b></p> <p>Use an if statement and the isOnPaint() method to move forward one square if the Painter object is standing on paint.</p>
	<p><b>c canMove()</b></p> <p>Use an if statement and the canMove() method to move forward one square if the Painter object can move forward in the direction it is facing.</p>		<p><b>d Use them all!</b></p> <p>Use if statements and the isOnBucket(), isOnPaint(), and canMove() methods to navigate The Neighborhood.</p>

Our curriculum incorporates **choice levels** for independent practice. Students can choose problems to practice new content differentiated either by topic or skill. All students should have the opportunity to experience success to build confidence as they practice new skills. Choice levels are designed to meet students where they are and support their learning and growth, allowing them to choose problems based on their individual perceptions of their progress and focus on the specific skills they feel they need the most practice.

Additionally, the **unit projects** are designed to encourage student voice and choice by allowing students to pursue personal interests and express themselves creatively. For example, students create artwork using The Neighborhood in the first unit to illustrate a personal interest, and later create a store management program to represent and manage items or services their business might offer. Students can show what they learned throughout the unit by applying their skills to a problem that is meaningful to them. Supports are provided to help students plan and develop their programs and receive feedback from their peers to improve their work.



Name(s) \_\_\_\_\_ Period \_\_\_\_\_ Date \_\_\_\_\_

### Store Management Project Planning Guide

**Project Description**

You are opening a new business in your community! Businesses often need programs to manage the products and services they offer and track orders and requests from customers. Your goal is to create a store management system for your business.

**Program Requirements**

Use your knowledge of object-oriented programming, user input, the DRY principle, and the Object class to create your store management system:

- Create a new superclass – Develop a new class that represents an object that your business offers
- Create one or more subclasses – Develop one or more subclasses that extend the superclass that represents a more specific type of object that your business offers
- Implement user interaction – Use the Scanner class to obtain user input to interact with the store management system

**Project Steps**

**Day 1 Steps**

- Brainstorm ideas for your business

Use the problem to identify the classes and methods you will need to implement

your superclass that represents an object that your business offers

one or more subclasses that represent more specific types of object that your business offers

user interaction to allow owners or customers to interact with the store management system

in a code review to give and receive feedback

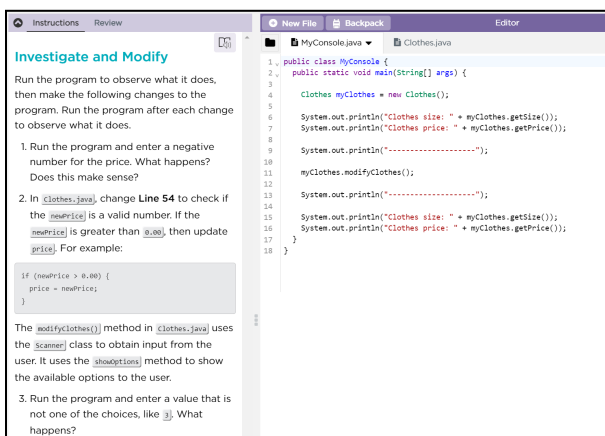
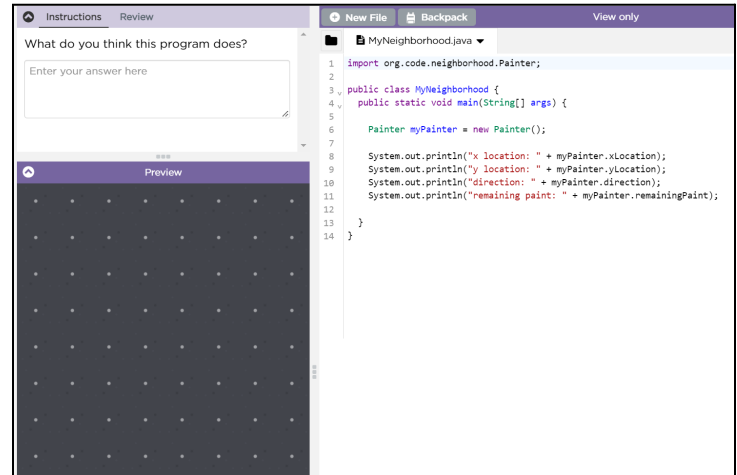
your work using the program requirements and rubric

your program using the feedback you received and your self-assessment

your work!

We believe students need opportunities to interact with existing code to strengthen code reading skills and to encourage collaboration around the meaning-making process. We frequently structure lessons in a format similar to PRIMM<sup>12</sup> to help students make and test predictions, investigate and modify code to explore its structure and functionality, and apply their findings to solve new problems.

For example, students frequently encounter **Predict** levels throughout the curriculum that ask them to make a prediction about the behavior of a program based on what they have already learned and the new components they see in the program. These levels are intended to activate prior learning and use it as an anchor for exploring new concepts.



Additionally, students also encounter **Investigate and Modify** levels to explore an existing program using guiding questions to interact with new concepts. Students discuss their findings as a class after interacting with both Predict and Investigate and Modify levels to share what they learned.

<sup>12</sup> <https://blog.teachcomputing.org/using-primm-to-structure-programming-lessons/>



Zaretta Hammond notes the importance of incorporating unstructured think time and cognitive routines to help students process new information<sup>13</sup>. Students need time to reflect and elaborate on the new information to identify connections and relationships with prior knowledge.

We encourage teachers to provide the **wait time** students need to process information and gather their thoughts during whole-class discussions. We incorporate strategies such as **Retrieve-Pair-Share** and **Hold That Thought** to support students in processing and reflecting in a way that works best for them, whether that is having time to jot down their ideas before verbalizing them or just needing a silent moment to think before sharing with a partner or the class.



<b>Method name:</b> _____	
<b>Writing a Traversal Algorithm</b> <ol style="list-style-type: none"> <li>1. Identify the process and write a method header.</li> <li>2. Identify the information that needs to be used in the next iteration.           <ol style="list-style-type: none"> <li>a. What information is needed in the next loop cycle?</li> </ol> </li> <li>3. Create a variable to store this information.</li> <li>4. Create a for-loop to access every item in the array.</li> <li>5. Perform <i>Test1</i>.</li> <li>6. Inside the loop, perform the required operation and update the variable storing the information to be used in the next iteration.</li> <li>7. After the loop, process and return the result.           <ol style="list-style-type: none"> <li>a. Processing is an optional step that may be needed for algorithms such as one to find the average of a list.</li> </ol> </li> <li>8. Perform <i>Test2</i>.</li> </ol> <b>Tests for Algorithm Correctness</b> <p><i>Test 1 - Are all indices accessed by the loop</i></p> <p>Print the loop variable</p> <p><b>Debugging:</b></p> <ol style="list-style-type: none"> <li>1. Outside the loop, print the length of the list.</li> <li>2. Will there be an off-by-one error?</li> </ol> <p><i>Test 2 - Does the algorithm perform the intended task?</i></p> <p>Call the method on a short array.</p> <p><b>Debugging:</b></p> <ol style="list-style-type: none"> <li>1. Inside the loop, print the value of the array element.</li> <li>2. Inside the loop, print the value of the information that is carried into the next iteration.</li> <li>3. Is the variable containing the information that is carried into the next iteration being updated with the array element as expected?</li> </ol>	<b>Pseudocode</b>

As noted in the *Guide to Inclusive Computer Science Education*<sup>14</sup>, problems in computer science can have multiple solutions. We believe it is important for students to explore different perspectives and approaches to try different solutions and challenge ideas. Students often work in pairs or small groups to brainstorm and test solutions using **pseudocode**, **UML diagrams**, and **flowcharts** to represent processes and algorithms before translating these into Java code. Rather than focusing on finding a single right answer, students learn how to decompose problems to develop and test possible solutions through guided inquiry, collaboration, and self-reflection.

Each unit also includes a **Unit Guide**, which provides an overview of the syntax and vocabulary students learn throughout the unit and guided notes for videos, discussions, and individual reflections. The Unit Guides allow students to draw visuals for vocabulary and create flowcharts and diagrams to represent information to support meaning-making of new content. We believe our Unit Guides help students interact with and process new information while also providing a study tool for preparing for end-of-unit assessments and the AP CSA Exam.

CSA Unit 1 Guide		
<b>Video: Creating Objects</b>		
How is memory in a computer like the land a house is built on?		What is the relationship between a Painter object and the Painter class?
Term	Definition	Example / Picture / Code
state		
	Finding and fixing problems in an algorithm or program.	

<sup>13</sup> Zaretta Hammond, *Culturally Responsive Teaching & the Brain*. (California: Corwin, 2015), p.130-136.

<sup>14</sup> Microsoft, *Guide to Inclusive Computer Science Education: How educators can encourage and engage all students in computer science*, [ncwit.org/resource/csedguide](https://ncwit.org/resource/csedguide), p. 12

## Digging Deeper

These are not the only strategies that can be implemented to affirm the identity and culture of students to create a sense of belonging, provide opportunities for students to practice what they are learning in a meaningful way, and ensure students interact with content and process it in a way that works best for them. We encourage teachers to go beyond these strategies to meet the specific needs of the students in their classrooms. Ultimately, it is up to the teacher to decide how best to set the tone for their classroom and to identify the strategies that help their students feel supported.

### Additional Information and Resources

- Zaretta Hammond, *Culturally Responsive Teaching and the Brain* (California: Corwin, 2015), [crtandthebrain.com](http://crtandthebrain.com)
- Kapor Center, *Culturally Responsive-Sustaining Computer Science Education: A Framework*, [kaporcenter.org/equitablecs](http://kaporcenter.org/equitablecs)
- Microsoft, *Guide to Inclusive Computer Science Education: How educators can encourage and engage all students in computer science*, [ncwit.org/resource/csedguide](http://ncwit.org/resource/csedguide)
- The UDL Guidelines, [udlguidelines.cast.org](http://udlguidelines.cast.org)