# California Computer Science Strategic Implementation Plan - DRAFT

September 2018

## Contents

[Introduction 3](#_Toc112331099)

[California’s Commitment to Computer Science Education 5](#_Toc112331100)

[Vision, Mission, and Principles 7](#_Toc112331101)

[Vision Statement 7](#_Toc112331102)

[Mission Statement 7](#_Toc112331103)

[Principles 7](#_Toc112331104)

[1. Expanding Computer Science Course Offerings 9](#_Toc112331105)

[Current State of K-12 Computer Science Course Offerings 9](#_Toc112331106)

[Strategies for Expanding K-12 Computer Science Course Offerings 10](#_Toc112331107)

[2. Improving Access to Computer Science Education for All Students 14](#_Toc112331108)

[Current State of Computer Science Education Access 14](#_Toc112331109)

[Strategies for Improving Access to Computer Science Education for All Students 16](#_Toc112331110)

[3. Supporting Educators to Teach Computer Science 19](#_Toc112331111)

[Current State of K-12 Computer Science Educators 19](#_Toc112331112)

[Strategies for Supporting Educators to Teach Computer Science 22](#_Toc112331113)

[4. Making Systemic Improvements in Computer Science Education 26](#_Toc112331114)

[Strategies for Building Information Technology infrastructure 26](#_Toc112331115)

[Strategies for Accountability and Evaluation 26](#_Toc112331116)

[Strategies for Funding and Institutional Support 27](#_Toc112331117)

[Strategies for Dissemination 27](#_Toc112331118)

[Summary of Recommended Strategies for Making Systemic Improvements in Computer Science Education 28](#_Toc112331119)

[Conclusion 29](#_Toc112331120)

[Appendices 30](#_Toc112331121)

[Appendix A: Introductory High School Computer Science Courses 30](#_Toc112331122)

[Appendix B: Resources for Teaching and Teachers 31](#_Toc112331123)

[Appendix C: California Computing-related Course Codes 35](#_Toc112331124)

[Appendix D: Abbreviations Used in this Document 45](#_Toc112331125)

[Appendix E: Figure Long Descriptions 46](#_Toc112331126)

## Introduction

Our society has witnessed profound changes in the ways we live and work due to advances in computing. California’s digital natives are growing up using technology to communicate with family and friends, to partake in educational opportunities, and to express themselves creatively. Many advocates of computer science (CS) education have cautioned that unless students are taught how computing works, they will remain merely passive consumers of these technologies. The integration of computer technology into every aspect of daily life necessitates a foundation in CS for a well-rounded general education. Today’s students need to understand how their digital world works in the same way they study science to learn how the natural world works or study history to understand cultures and politics.

Although computer science as a discipline of study is well established at the post-secondary level, it is relatively new to the K-12 system. While the public is becoming more aware of computer science and recognizing its value for all students, there is still some confusion over what the discipline constitutes. CS is often misconstrued with other technological terminology such as computer literacy, educational technology, digital citizenship, and information technology. These areas focus more on the use of computing systems (e.g., learning to use word processing software). In contrast, computer science calls upon students to understand why and how computing technologies work, and then to build upon that conceptual knowledge by creating computational artifacts.

Despite growing recognition of the importance of computer science education, the majority of high schools in California do not offer any computer science courses. Even some of the largest districts in the state do not offer computer science courses. These limited opportunities for CS education lead to issues of access and equity for students. In schools that do offer CS, student participation in these courses too often does not reflect the demographic makeup of California. There is no single-subject credential in CS, requiring educators to meet other unrelated requirements to teach CS. Improving computer science education throughout all of California is vital for preparing our children for tomorrow’s challenges.

California’s economy and workforce needs also depend on ensuring more students have access to computer science education. Computer and mathematical occupations are among the fastest-growing and most lucrative sectors in the modern economy, particularly in California. The United States Bureau of Labor Statistics’ Occupational Outlook Handbook projects that computer and information technology occupations will grow 13 percent between 2016 and 2026, faster than the average growth for all occupations[[1]](#footnote-1). In 2016, California had the highest employment in computer and mathematical occupations of all 50 states, with over 500,000 jobs earning an annual mean wage of $102,970[[2]](#footnote-2). As of this writing, there are 74,000 open computing jobs in California. Computing jobs exist across the state and in every industry sector and can provide students with important social mobility opportunities.

Undoubtedly, there is greater access to K-12 computer science education in California today due to the numerous initiatives spurred by various stakeholders in education, industry, and the nonprofit sector. For example, the *Exploring Computer Science* (ECS) course, developed by researchers at the University of California Los Angeles and the University of Oregon, has reached more than 14,000 high school students across 43 schools in Los Angeles since its launch in 2008[[3]](#footnote-3). As another example, the new *AP Computer Science Principles* (AP CSP) course, launched in 2016-17, has more than doubled the number of students taking AP computer science courses in the past three years. AP CSP is specifically designed to reach a larger student population. The College Board approved ten different curriculum providers[[4]](#footnote-4) – including many familiar to California such as Project Lead the Way and Code.org – to provide professional development on the course. As a result, California saw 16,315 public school students take an AP CS exam in 2017. However, more needs to be done to reach the over six million public school students enrolled at over 10,000 schools across the state.

California is committed to supporting equity of access to CS education “by ensuring that all student groups are visible in accountability and improvement efforts and setting goals for closing gaps”[[5]](#footnote-5). This document presents four overarching strategies to guide stakeholders in implementing CS education across the state: (i) expanding CS course offerings, (ii) improving access to CS education for all students, (iii) supporting educators to teach CS, and (iv) making systemic improvements in CS education. It is hoped that this implementation plan will catalyze efforts to transform K-12 computer science education so that all of California’s students will be better prepared to contribute to our digital world.

[As] computer science education is becoming increasingly accepted in our schools...K-12 educators may find it useful to also focus on Papert’s reasons: Teach students computer programming to help them learn everything else.

Sheena Vaidyanathan, Computer Science Integration Specialist, Los Altos School District, Los Altos, CA

## California’s Commitment to Computer Science Education

On or before July 31, 2019, the Instructional Quality Commission shall consider developing and recommending to the state board computer science content standards for kindergarten and grades 1 to 12, inclusive, pursuant to recommendations developed by a group of computer science experts. (California Education Code § 60605.4)

On September 30, 2014, Governor Brown signed Assembly Bill 1539 into law, which added section 60605.4 to the California *Education Code (EC)* directing the Instructional Quality Commission (IQC) to consider developing and recommending to the State Board of Education (SBE) computer science content standards on or before July 31, 2019.

California’s computer science standards are based on the revised international Computer Science Teachers Association (CSTA) standards, which were released in July 2017 and align with the national K-12 CS Framework[[6]](#footnote-6) released in November 2016. The California Computer Science Standards Advisory Committee (CS SAC) retained the core concept and subconcept areas and the computer science practices from the CSTA standards, and they added, removed, and modified the CSTA standards to fit California’s educational context. The CS SAC also drafted introductory material and appendices to introduce and explain the standards to educators, identify areas of alignment between the computer science standards and other content standards, and provide recommendations for implementation.

As with all California content standards, the computer science standards are model standards that define the knowledge, concepts, and skills that students should acquire in each grade band and encourage school districts to provide opportunities for computer science education for all students. As of this writing, there are no plans to develop a computer science curriculum framework or a state summative CS exam to accompany the standards. However, local education agencies (LEAs) will have the freedom to determine their own curriculum and implementation needs. They can choose to develop their own curricula as well as select curricula and professional development from many existing vendors (see Appendix B) in addition to the College Board approved providers of the two AP computer science courses (see Appendix A). Additional detail on the standards development process is available on the CDE website: <https://www.cde.ca.gov/be/st/ss/computerscicontentstds.asp>.

On or before March 1, 2018, the Superintendent shall convene a computer science strategic implementation advisory panel to develop recommendations for a computer science strategic implementation plan…The Superintendent shall develop, and the state board shall consider adopting, a computer science strategic implementation plan on or before July 15, 2019. (California Education Code § 53310-53315)

Early into the computer science content standards development process, the CDE, SBE, and Legislature began the process of developing the California Computer Science Strategic Implementation Plan per California *EC* Section [53310–53315]. Per the requirements of *EC* Section 53310 (b), the Governor, the State Board of Education President, the Senate Committee on Rules, the Speaker of the Assembly, and the State Superintendent of Public Instruction appointed 23 members to the Computer Science Strategic Implementation Plan Panel (CSSIPP) to develop recommendations to inform this implementation plan. Panel members were selected based on their expertise in computer science education, experience in standards-based interdisciplinary instruction and differentiated instruction for a diverse student population, other areas of expertise and leadership, and their previous committee experience.

The CDE convened public meetings of the CSSIPP in Sacramento between March and June of 2018. During these meetings, the panel participated in small and whole group discussions to develop recommendations consistent with the requirements in *EC* Section 53311. CDE held additional meetings with the CSSIPP co-chairs in July 2018 to finalize the panel’s recommendations. The recommendations include timeframes for implementation, which are incorporated in this document. The panel recognized that districts across the state vary in their readiness to implement CS education and that schools face many other priorities that may compete with CS education efforts. As such, the timeframes are offered as suggestions, with the understanding that stakeholders may differ in their ability to follow these timeframes exactly. The panel was also tasked with defining computer science education principles that meet the needs of pupils in kindergarten and grades 1 to 12. Additionally, the CSSIPP created mission and vision statements to guide computer science education in California.

The work of the CSSIPP informed this strategic implementation plan. Their recommendations identified specific entities responsible for the strategies they proposed. In this plan, responsible entities are condensed into four groups: schools and LEAs (LEAs); county offices of education (COEs); institutions of higher education in the University of California and California State University systems and community colleges (IHEs); and state level entities including CDE, CCTC, Legislature, and the State Board of Education (State). A detailed list of the CSSIPP recommendations and additional information about the CSSIP is available on the CDE web page: <https://www.cde.ca.gov/pd/ca/sc/cssip.asp>.

## Vision, Mission, and Principles

When the CSSIPP met in 2018, they decided it was important to communicate the larger goals of K-12 computer science education in California. This strategic implementation plan is guided by their vision of computer science education, mission for stakeholders involved in implementing computer science education, and set of principles supporting California’s commitment to equity in computer science education.

### Vision Statement

*California’s vision is to ensure that all students develop foundational knowledge and skills in computer science to prepare them for college, careers, and civic engagement.*

### Mission Statement

*All schools offer rigorous and relevant computer science education equitably and sustainably throughout grades K-12.*

*All teachers are adequately prepared to teach rigorous and relevant computer science aligned with California’s K-12 computer science.*

### Principles

These principles apply to all California schools (K-12) and the students they serve.

1. Every student and every teacher is capable of learning computer science. Access to and achievement in computer science should not be predicated on the basis of race, ethnicity, gender identity, socioeconomic status, language, religion, sexual orientation, cultural affiliation, learning differences, or special needs.
2. Every student in California should have equitable access to high-quality computer science curriculum and instruction aligned to California’s K-12 computer science standards.
3. Every student should have continuous opportunities and multiple entry points to engage in computer science education, including articulated pathways toward college, careers, and community engagement.
4. Computer science instruction should involve real-world, engaging, meaningful, and personally relevant activities for students that focus on problem-solving, critical thinking, and creativity while emphasizing the ethical impacts of computing.
5. Computer science should align with California’s K-12 Computer Science Standards and be integrated, as appropriate, into other subject areas in grade bands K-2, 3-5, 6-8, and 9-12; computer science should be offered as standalone courses from introductory to more advanced in middle and high school.
6. All California schools should have the infrastructure to support computer science education (including hardware, software, and personnel).
7. Computer science content knowledge and relevant pedagogical practices should be included in all California teacher preparation programs, differentiated by multiple subject and single subject teaching credentials.
8. The state budget should allocate funding for teachers to participate in on-going, high-quality, and differentiated professional learning and support to assist them in implementing and integrating computer science education in their classrooms.
9. California should engage stakeholders, including, but not limited to, members from K-12 education, higher education, industry, local communities, parent organizations, and policy makers, to implement computer science statewide.

## Expanding Computer Science Course Offerings

### Current State of K-12 Computer Science Course Offerings

While CS education opportunities exist in California, measuring their availability is challenging for a variety of reasons. For example, at the elementary and middle school levels, CS is often integrated into existing curricula and information about the prevalence or quality of these offerings is not tracked. LEAs report student course enrollment and completion through course codes as part of California’s Longitudinal Pupil Achievement Data System (CALPADS; see Appendix C). However, these course codes are self-reported, making it possible to miscode a course (e.g., keyboarding) as containing CS content. Miscoded course information can lead to false assumptions about the spread of CS education in California and either exaggerate or attenuate issues of access and equity.

At the secondary level, very few high schools offer CS courses (see Table 1). A recent analysis of secondary CS courses offerings in California, conducted by The Level Playing Field Institute[[7]](#footnote-7) in 2015, revealed that ten of the 20 largest districts in the state did not offer Advanced Placement (AP) Computer Science A and five of these districts did not offer any CS courses at all.

**Table 1.** Number of California High Schools Offering CS Courses (2016-17)

| Course Name | Course Code | Number of Schools | % of Schools |
| --- | --- | --- | --- |
| Exploring Computer Science | 4634 | 284 | 12% |
| Computer Science | 2453 | 298 | 12% |
| AP Computer Science Principles | 2472 | 71 | 3% |
| AP Computer Science A | 2470 | 235 | 10% |
| Robotic Technologies | 4647 | 323 | 13% |

**Source:** Kapor Center/ACCESS (2018). Computer Science in California’s Schools Fact Sheet: Course Availability and Enrollment (Draft v. 4/11/18); **Note:** AP Computer Science Principles was in its pilot year in 2016-17.

The College Board provides the most reliable data source of current CS education opportunities through its yearly reports of AP CS A and CS Principles exams. California is one of the top ten states with the highest per capita AP CS A participation.[[8]](#footnote-8) Yet, compared to other disciplines, the number of California students who take AP CS exams is extremely low. In 2017, 8,679 California public school students took the AP CS A exam and 7,636 California public school students took the AP CS Principles exam[[9]](#footnote-9). These numbers are on par with the number of students who took the AP Spanish Literature exam (8,515) and far less than the number of students who took exams for AP Biology (34,984), AP Calculus AB (48,473), AP Chemistry (20,008), AP Statistics (30,196), AP English Language & Composition (84,960), and AP US History (69,729).

Further, AP exam reports are not without their limitations. First, AP courses only reflect a limited number of high school course offerings. Second, as AP courses target college-bound students, exam reports may not accurately capture information of students who enroll in CS courses but do not complete the corresponding AP exams. Third, students can sit for an AP exam without enrolling in the related course that year or without taking the AP course at a high school. As indicated in Table 2, the number of California students enrolled in the AP CS Principles and AP CS A courses in the 2016-2017 academic year differed from the number of students who took the related AP exams.

**Table 2.** Number of California Students in AP CS Courses and Exams (2016-17)

| Course Name | Course Code | Total Course Enrollment | Total AP Exam Test Takers |
| --- | --- | --- | --- |
| AP Computer Science Principles | 2472 | 3,146 | 7,636 |
| AP Computer Science A | 2470 | 10,821 | 8,679 |

**Sources:** Kapor Center/ACCESS (2018). Computer Science in California’s Schools Fact Sheet: Course Availability and Enrollment (Draft v. 4/11/18); College Board’s AP Program Participation and Performance Data 2017

### Strategies for Expanding K-12 Computer Science Course Offerings

Expanding computer science course offerings will require alignment across the K-8, 9-12, and post-secondary levels, so that students are prepared and motivated to pursue computer science opportunities at each stage of their education.

#### Strategies for expanding at the K-8 level

Introducing students to computer science at the K-8 level will prepare them to become computational thinkers who understand why and how computing technologies work and who use that knowledge to create computational artifacts. A foundation in computer science at the elementary and middle school levels will also prepare students for future computer science course work in high school. An increasing number of teachers in California already offer K-8 students computer science opportunities. Their initiatives should be leveraged to support other teachers in incorporating computer science into their classrooms. A first step in expanding computer science course offerings would be identifying these opportunities, sharing this work across the state, and aligning these efforts with the new state standards.

With many competing initiatives, including the need to address student success in English language arts and mathematics, adding a computer science course to students’ daily schedule may be a daunting task. In response to this concern, the California Computer Science Standards describe how computer science instruction can be integrated into multiple subject classrooms or taught as a discrete, independent course. The standards also include interdisciplinary connection examples in grade bands K-2, 3-5, and 6-8 to help educators identify ways in which computer science fits into existing coursework.

#### Strategies for expanding at the high school level

At the high school level, computer science courses can offer students a deeper understanding of the discipline, motivation to pursue future computer science studies, and exposure to related careers. However, high school students have a myriad of requirements that impact their already full schedules, making it difficult to enroll in computer science courses. Adopting a computer science graduation requirement would ensure that all students at the high school level are exposed to computer science instruction.

If schools and districts are unable to adopt a high school graduation requirement, ideally they would work in the next five years to expand their course offerings to offer A–G approved computer science course sequences. These sequences would begin with an introductory level course and end in an Advanced Placement or college-level course. For students who enter high school with advanced computer science skills, schools should consider developing flexible programming that would allow them to demonstrate their knowledge of computer science concepts and practices through tests and portfolios. Thus, allowing them to move through the course sequences more fluidly based on their proficiency levels.

Also in an effort to expand and incentivize students participating in computer science coursework, the Board of Admissions and Relations with Schools (BOARS) is working to designate computer science courses in other areas besides “g” to allow students more opportunity to participate in computer science coursework.

#### Funding strategies for expanding course offerings

Expanding CS course offerings is likely to be challenging for many districts in the absence of new funding. The Local Control Funding Formula (LCFF), enacted in 2013, allows for LEA flexibility in the use of funding, but this often means that districts must make difficult decisions between implementing new standards and programs and maintaining existing programs and staffing needs. LCFF includes targeted funding especially for schools with low income, English Learner (EL), and foster youth; this is called supplemental and concentration funding. This funding is specifically designed for schools to implement new and improved instructional services for their most struggling students. Many districts already implementing computer science have been using their LCFF funding for this purpose. Other districts may have constraints that do not allow them to make use of this LCFF funding, so an infusion of dedicated categorical funding to CS would be a boon to helping get this new K-12 discipline off the ground across California.

Dedicated funding for the development, improvement, and implementation of integrated CS courses through the University of California Course Integration (UCCI) program could also increase learning opportunities at the high school level. This additional funding would allow more students to pursue computer science in CTE pathways while also allowing them to fulfill A–G subject requirements for admission into the University of California and California State University systems.

#### Summary of Recommended Strategies for Expanding K-12 Computer Science Course Offerings

| Strategy | Timeframe | Responsible Entity |
| --- | --- | --- |
| Identify current K-8 computer science instruction and align it to the California computer science standards | Within 4 years | LEAs  COEs |
| Develop pathways to expose K-8 students to computer science across all grade bands (K-2, 3-5, 6-8) | Within 4 years | LEAs  COEs |
| Adopt a graduation requirement, course sequences and/or alternative methods (tests, portfolios etc.) to ensure computer science proficiency | Within 6 years | LEAs |
| Offer A-G approved computer science courses in areas other than “G” | Within 5 years | LEAs  COEs |
| Create computer science UCCI Courses | 2-5 years | LEAs |
| Dedicate funding for the creation and implementation of computer science courses and services | 2-5 years | State |

## Improving Access to Computer Science Education for All Students

### Current State of Computer Science Education Access

As noted above, access to CS courses is profoundly inequitable across the state. This is particularly evident at the secondary level where it is easier to track CS offerings. The Level Playing Field Institute7 found that:

* High schools with the highest percentages of underrepresented students of color were half as likely to offer any CS course and 12 times less likely to offer AP CS compared to schools with the lowest percentages of underrepresented students of color.
* High schools with the highest percentages of low-income students were less than half as likely to offer any CS course and 11 times less likely to offer AP CS as schools with the lowest percentages of low-income students.

As can be seen in Figure 1, students taking either of the AP CS exams were disproportionately male and did not reflect the ethnic makeup of California.

**Figure 1.** Percent of AP CS Exams Taken by Course, Gender, and Ethnicity in California in 2017 ([Figure 1 Long Description](#_Figure_1._Percent))

Figure 1. Percent of AP CS Exams Taken by Course, Gender, and Ethnicity in California in 2017. Link to long description above image. 

**Source:** College Board’s AP Program Participation and Performance Data 2017

Table 3 highlights some of these disparities by comparing the percentage of AP test takers to the percentage Hispanic/Latinx, Black or African American, and male students in California public high schools during the 2016-2017 academic year. These patterns are not unique to California, but are also seen in the national sample of AP CS exam takers.

**Table 3.** Percentage of California High School Students Compared to AP Test Takers (2016-17)

| Student Group | State of California | AP CS A | AP CS Principles |
| --- | --- | --- | --- |
| Hispanic/Latinx | 53% | 14% | 31% |
| Black or African American | 6% | 1% | 2% |
| Male | 51% | 73% | 68% |

**Sources:** College Board’s AP Program Participation and Performance Data 2017; California Department of Education, DataQuest: 2016-17

In addition to the College Board data, the Kapor Center for Social Impact and ACCESS recently compiled data on secondary CS course enrollment in both AP and non-AP CS courses (see Table 4). Similar to the AP CS exam data, few students are enrolled in secondary CS courses in California and students studying CS are disproportionately male.

**Table 4.** Number of California High Schools Offering CS Courses (2016-17)

| Course Name | Course Code | Total Enrollment | % of Total HS Enrollment | % Male Enrollment | % Female Enrollment |
| --- | --- | --- | --- | --- | --- |
| Exploring Computer Science | 4634 | 18,741 | 1.0% | 69% | 31% |
| Computer Science | 2453 | 15,350 | 0.8% | 64% | 36% |
| AP CSP | 2472 | 3,146 | 0.2% | 70% | 30% |
| AP CS A | 2470 | 10,821 | 0.5% | 72% | 29% |
| Robotic Technologies | 4647 | 13,197 | 0.7% | 71% | 29% |

**Source:** Kapor Center/ACCESS (2018). Computer Science in California’s Schools Fact Sheet: Course Availability and Enrollment (Draft v. 4/11/18).

Inequities in computer science opportunities also exist for students with disabilities. In California, approximately 700,000 students (or 11%) have a disability. While there are no specific reports on the computer science opportunities for these students, they likely encounter issues of access and equity similar to other students with disabilities across the nation. For example, many programming tools designed to support K-12 CS education may not be accessible to students who are blind or students with mobility-related disabilities[[10]](#footnote-10).

These inequities are also reflected in the demographic makeup of technology sector employees. The Bureau of Labor Statistics estimates that in 2017 only 26 percent of people employed in computer and mathematical occupations were women, 9 percent were Black or African American, and 7 percent were Hispanic or Latinx[[11]](#footnote-11). Only 2.3 percent of all employed persons with a disability were employed in computer and mathematical occupations[[12]](#footnote-12).

Large inequities in computer science education and workforce participation reinforce long-standing injustices in society, including wage gaps. A highly inequitable computing workforce can lead to bias in the way tools are designed and function. For example, original health apps allowed users to track a large number of health indicators but did not include a feature for women to track menstrual cycles[[13]](#footnote-13). Addressing inequities in CS education can help ensure that tomorrow’s technology works for all people.

### Strategies for Improving Access to Computer Science Education for All Students

Access to computer science education for all students can be improved when families, community members, teachers and counselors are equipped to engage in advocacy, implement equity strategies, and create expanded learning opportunities.

#### Improving access through family and community support

Families and communities are fundamental to a student’s educational success. Their support is needed to ensure more students have access to computer science. Families and community members can encourage students to explore computer science, assist them in their course work, and provide them with learning opportunities outside of school. Their voices can also influence local schools to offer more computer science courses. An outreach toolkit should be developed to help these stakeholders improve access to computer science education for their own students. This toolkit should contain information about the field of computer science, strategies for supporting students, and tips for advocating for computer science opportunities. To ensure the toolkit reaches a broad audience, materials should be translated into multiple languages and differentiated for families of underrepresented students and community organizations that serve them (e.g., after school clubs). Toolkits could also be used to organize events focused on family engagement around computer science.

#### Improving access through teacher and counselor support

A variety of environmental factors contribute to students pursuing computer science such as early exposure, social support, and a sense of belonging. In the school environment, teachers and counselors can attract and retain more students in computer science courses by implementing practices focused on equity, cultural responsiveness, and the elimination of bias. Educational equity is a cornerstone of California’s education system, and there are teachers and counselors already attending to these issues across the state. Exemplary teachers and counselors committed to improving access and equity in computer science should be highlighted so they can share their strategies with other educators.

Professional learning materials and evidence-based professional development should also be created to inform teachers and counselors of best practices in encouraging diverse students to pursue computer science opportunities. Training materials for teachers should introduce varied pedagogical techniques and include project-based activities with strong CS content, collaborative learning, inquiry-based pedagogy, and culturally responsive teaching. Training materials for counselors should provide an awareness of the California computer science standards, describe the various pathways students can follow to pursue computer science, and highlight common barriers that prevent students from enrolling in computer science. Publishing these materials on the California Department of Education website can help in disseminating best practices to teachers and counselors across the state.

#### Improving access through expanded learning opportunities

Access to computer science learning can also be improved through activities that augment the classroom experience.For example, community partners can visit schools to mentor students or provide information about computing careers. These enriching activities need not be restricted to the regular school day. Engaging with activities outside of school can help students make connections between computer science and their daily lives. In collaboration with community partners, opportunities should be created for students, especially underrepresented students, to participate in expanded learning, scholarships, internships and mentorships related to computer science. The roles and responsibilities of these school-community partnerships should be clearly defined in district technology plans.

#### Summary of Recommended Strategies for Improving Access to Computer Science Education for All

| Strategy | Timeframe | Responsible Entity |
| --- | --- | --- |
| Plan outreach advocacy events focused on creating awareness of computer science, especially for groups traditionally underrepresented in computer science courses | Within 3 years | LEAs  State |
| Develop training materials and professional development for counselors and teachers that prioritizes equity, engages diverse learners, utilizes project-based activities, and introduces varied pedagogical techniques | Within 3 years | LEAs  COEs |
| Partner with community organizations to provide students with expanded learning and mentoring opportunities | Within 3 years | LEAs  COEs |

## 3. Supporting Educators to Teach Computer Science

### Current State of K-12 Computer Science Educators

Ensuring there is a sufficient number of prepared teachers for CS courses for the over 6 million public school students in California is a daunting task. The California CS standards suggest implementing stand-alone courses at the high school level, but the state does not currently have a single-subject credential in CS. Those who wish to teach CS must first obtain a single-subject credential in another area. Secondary teachers with single-subject credentials in Math, Business, or Industrial and Technology Education (ITE) are currently authorized to teach CS courses that are coded as a core academic course. When a CS course is coded as Career Technical Education (CTE), then those with a Designated Subject CTE teaching credential in Information and Communication Technology are authorized to teach the course. The CALPADS codes for computing related courses in California are listed in Appendix C.

Teachers who hold a single-subject credential in another subject area are eligible to receive a Supplementary Authorization in CS after completing a college major in CS, 20 semester units in CS, or ten upper division semester units in CS:

* An **Introductory Supplementary Authorization** authorizes the holder to teach CS content in grade 9 and below; teachers must complete coursework in computational thinking; computing practice and programming; computer and communication devices; and impacts of computing.
* **Specific Supplementary Authorizations** authorize the holder to teach core academic CS courses for all grades; teachers must complete coursework in computer programming; data structures and algorithms; digital devices, systems, and networks; software design; and impacts of computing; their coursework may also include a CS teaching methods/pedagogy course.

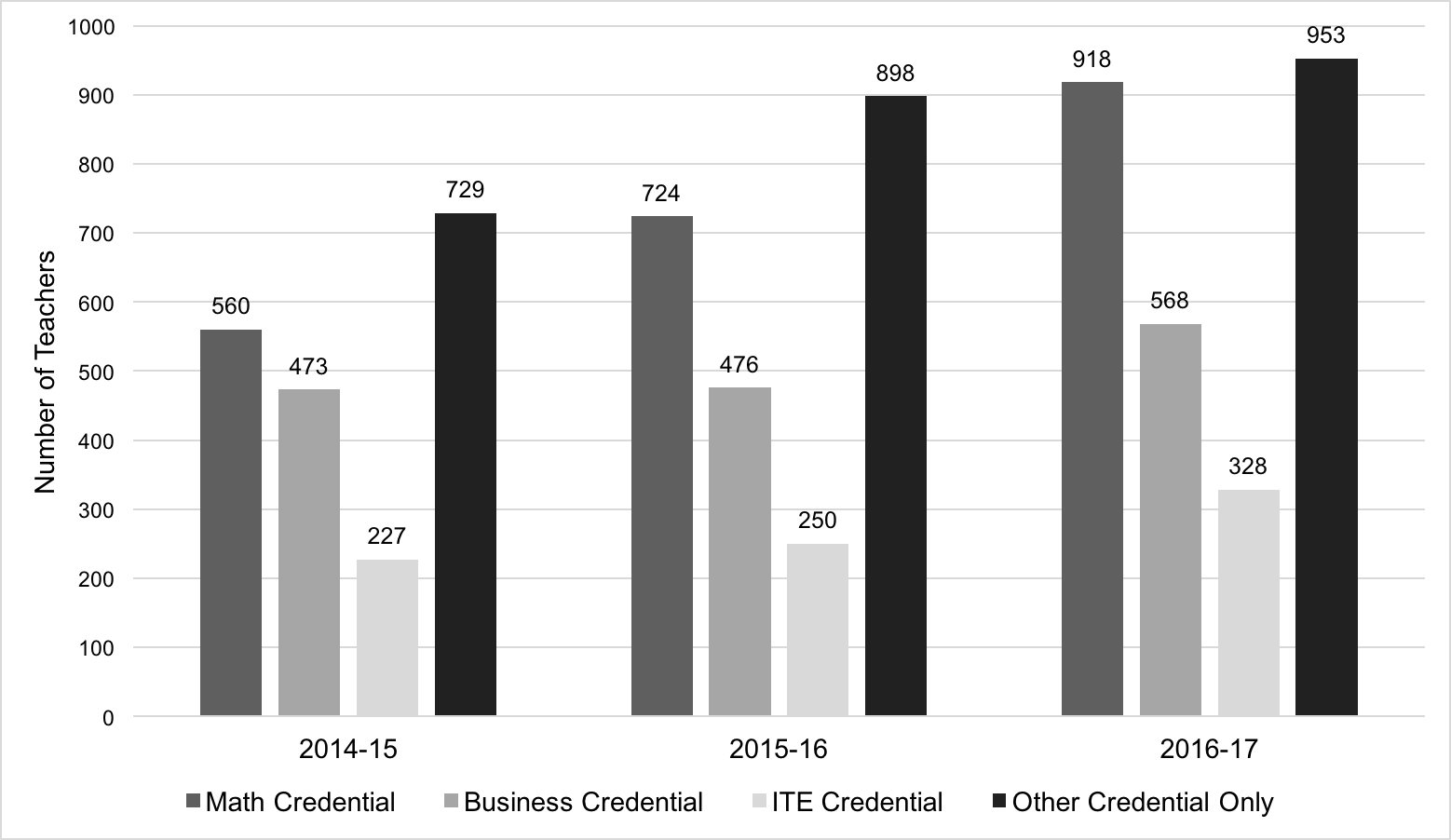
For more detail on the required coursework for both authorizations, see the California Commission on Teacher Credentialing (CCTC) Coded Correspondence 16-05[[14]](#footnote-14),[[15]](#footnote-15).

A bachelor’s degree in Computer Science also qualifies a credentialed teacher to get the Supplementary Authorization in Computer Science. Degrees in Educational Technology or Information Technology do not automatically meet requirements for supplementary authorizations in CS--applicants must submit official transcripts to verify that the complete coursework covers all required content.

#### Who is teaching Computer Science in California?

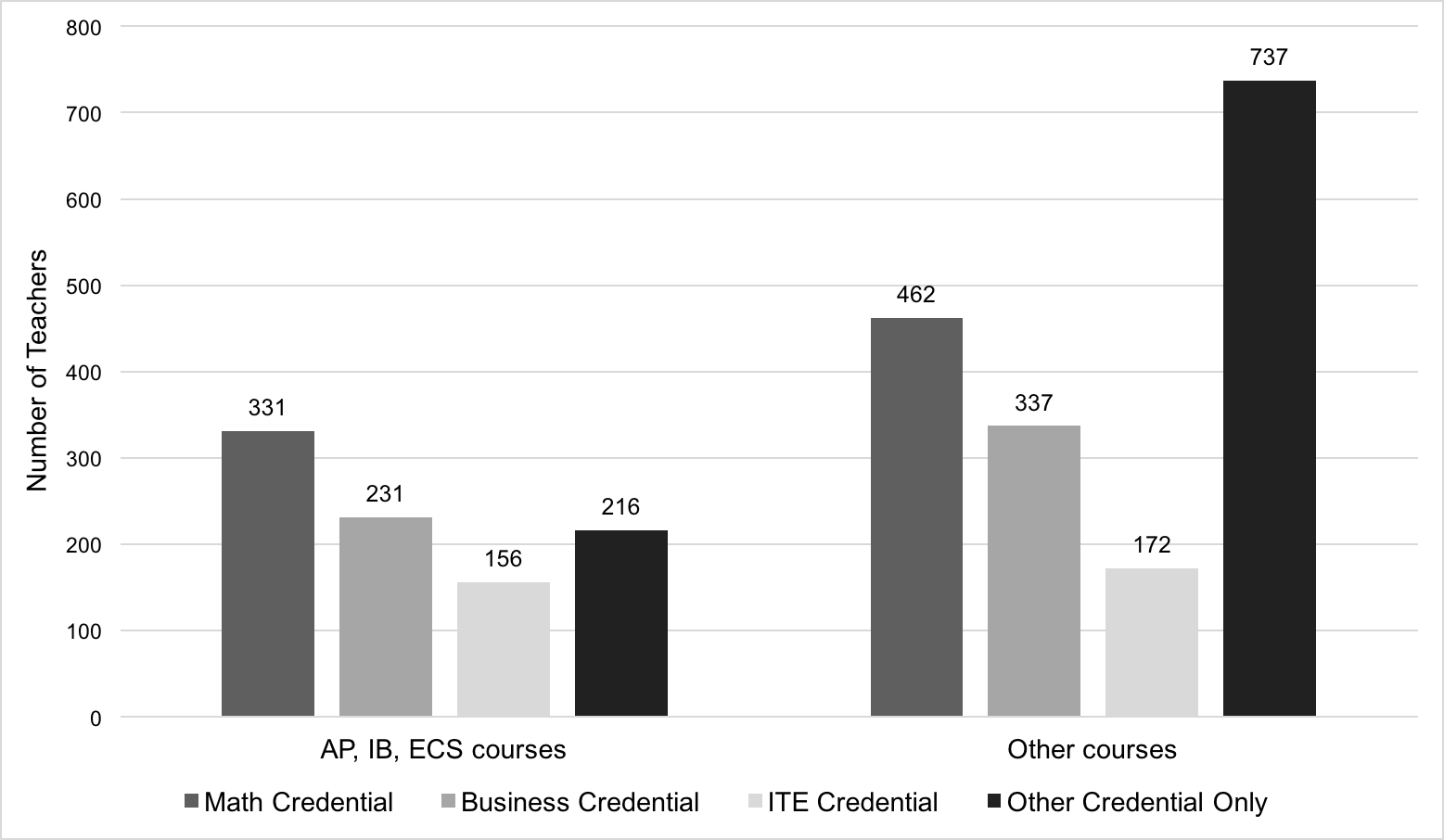
In the 2016-2017 academic year, approximately 2,273 teachers in California taught core academic computer science courses[[16]](#footnote-16). This number has grown steadily over the past few years from 1,609 teachers in 2014-2015 and 1,996 teachers in 2015-2016. Most teachers leading core academic CS courses are credentialed in subjects other than mathematics, business, or ITE (see Figure 2). Aside from these other credentialed teachers, teachers credentialed in mathematics comprise the largest number of core academic CS course educators.

**Figure 2.** Number of Teachers of Core Academic CS Courses by Credential (2014-2017) ([Figure 2 Long Description](#_Figure_2._Number))

**Source:** CDE; **Note:** A teacher credentialed in math, business, or ITE might also hold other credentials.; **Note:** Core academic CS courses reported are 2451, 2453, 2458, 2465, 2470, 2471, 2472, 4619, 4634

These trends generally hold when looking at the number of teachers by CS courses taught in California that are also taught across the country (i.e., AP courses, International Baccalaureate® [IB] courses, and ECS) compared to other CS courses (see Figure 3). One noticeable difference is a lower number of other credentialed teachers teaching AP, IB, or ECS courses.

**Figure 3.** Number of Teachers of Core Academic CS Courses by Course Group (2016-2017) ([Figure 3 Long Description](#_Figure_3._Number))

**Source:** CDE; **Note:** A teacher credentialed in math, business, or ITE might also hold other credentials.; **Note:** AP, IB, ECS courses are coded as 2465, 2470, 2471, 2472, and 4634. Other courses are 2451, 2453, 2458, and 4619.

California has not yet created pre-service CS teacher preparation programs; although, content on computational thinking and CS have been added to some preparation programs in other disciplines, such as math and science. Some universities have created programs for in-service teachers to satisfy the course requirements for the Supplementary Authorization in CS. As of this writing, these programs are offered:

* University of California--Irvine[[17]](#footnote-17): a certificate program consisting of four hybrid (face-to-face and online) courses (15 graduate level quarter units) supplemented by a Professional Learning Community;
* University of California--Riverside[[18]](#footnote-18): five online courses (19 quarter units).

Stanislaus State also provides a list of suggested courses that can be used to meet the requirements for the supplementary authorization[[19]](#footnote-19).

Currently, no additional credential or authorization is required for a teacher to integrate computational thinking and/or CS content into a non-CS course. However, schools and teachers would be well served to participate in professional development activities to better understand ways of integrating CS into existing courses and to create local communities of practice (COPs). A large number of organizations provide free lesson plans, online resources, and other material for K-12 CS education. For a list of resources, see Appendix B.

A major weakness of the existing situation is that single-subject credentialed teachers authorized to teach CS (i.e., Math, Business, or ITE) do not have subject matter requirements that cover basic CS content. Furthermore, they are not trained in pedagogical knowledge relevant to CS, which is different from their core subject. The supplementary authorizations in CS, on the other hand, do require courses that cover CS content knowledge. Yet, there are very few opportunities for credentialed teachers to enroll in such programs and these teachers will not necessarily have had practice teaching in a CS classroom.

### Strategies for Supporting Educators to Teach Computer Science

K-12 computer science education cannot grow in California without increasing the number of teachers qualified to teach CS. Supporting more educators to teach computer science will require a multi-pronged approach that attends to credentialing, the recruitment of new teachers, teacher training, teacher communities, administrator and counselor training, awareness of California’s computer science standards and institutional and financial support.

#### Strategies for Credentialing

A first step to supporting teachers in pursuing CS authorization is to make it easier to identify institutions where they can complete coursework required for existing authorization. For example, CCTC could host a public portal that institutions of higher education (IHEs) populate with their coursework that fulfills supplementary authorization. As another strategy, CCTC could reevaluate subject matter requirements for computer science to determine if existing credentials or supplementary authorizations could also authorize teachers to teach CS. For example, the Industrial and Technology Education (ITE) single-subject credential already authorizes holders to teach CS. However, the credential is quite broad and could benefit from the addition of a distinct strand for CS. Also, CCTC could update the Teaching Performance Expectations (TPEs) for teacher and specialist credentials to include foundational understanding of the computer science standards and related pedagogy. While related TPEs already exist (e.g., TPEs 3.6, 3.6, and 3.8), they focus more on technology and digital literacy and not on computer science. Updating the TPEs could ensure that more multiple-subject and single-subject teachers are prepared to include CS content and pedagogy in their teaching.

Due to the current authorization pathways for computer science teaching in California, all teachers of computer science must first receive authorization to teach some other subject. Passing legislation to develop a CS single-subject credential could encourage more aspiring teachers to pursue CS teaching. To support this strategy, CCTC could develop a California Subject Examination for Teachers (CSET) or consider adopting the forthcoming CS PRAXIS® exam from the Educational Testing Service[[20]](#footnote-20).

#### Strategies for Recruitment

Relying on in-service teachers to fill current and future courses will be insufficient to meeting California’s needs for more CS teachers. By providing funding to incentivize partnerships between computer science and education departments at IHEs, programs for computer science teacher preparation can be developed. Some college students, including CS majors, might not consider teaching as a viable career path when lured by the high-paying salaries of the tech industry. So, the value of CS teaching should be highlighted in these preparation programs. For example, professors could recognize the work of K-12 CS teachers alongside other CS professionals when they aim to recruit students into teaching.

Some of California’s current CS teachers are career changers who previously spent many years working in industry. Programs could be created to provide a bridge for more computer scientists in industry to enter into CS teaching. For example, the EnCorps[[21]](#footnote-21) program provides industry professionals with the pedagogical training and classroom experience needed to pursue certification for STEM subjects. As another alternative, individuals in industry can be encouraged to pursue a Business and Industry Partnership Teacher Authorization (BIPT) allowing them to teach for a year and then apply for a Career Technical Education (CTE) credential.

#### Strategies for Teacher Training

Many teachers who will help computer science spread across California will be new tothe discipline and will need to learn how to teach its concepts and practices. For teachers already experienced in computer science, the dynamic nature of the subject will require them to update their teaching knowledge as new tools and curricula emerge. Professional learning opportunities can offer these teachers the training they need to implement computer science confidently and effectively. While some of these programs already exist, many more opportunities are needed.

Funding should be provided to incentivize IHEs to offer credit-bearing courses or teacher preparation programs that satisfy the CS supplementary authorization and future CS teaching credentials. These opportunities should help teachers learn how to teach concepts and practices aligned to the California computer science standards, differentiated for grade and skill levels. The University of California, Office of the President (UCOP) should also be funded to expand the California Subject Matter Projects with the addition of a new project focused on CS. As the standards recommend that computer science be integrated into multiple subject classrooms, all teacher preparation programs for multiple-subject and single-subject credentials should include computer science content and pedagogy.

Industry can also support educator training by hosting experiences that provide teachers with a deeper understanding of the computing discipline. These partnerships can also help industry build awareness of the computer science teacher shortage and how that interacts with their need for computing talent. Through educator-industry exchanges, industry might become more invested stakeholders in developing their future workforce and continue to contribute to efforts to build sustainable teacher capacity.

To ensure teachers are receiving the professional learning needed to prepare them for computer science, the state should evaluate these programs to ensure they are of high quality. Instructional leaders will also need to be prepared to offer equity-minded professional development. IHEs will need to work with CCTC to establish course articulation agreements with CCTC-approved teacher preparation programs.

Training opportunities should be engaging, providing teachers with excitement-generating, hands-on experiences so they can see examples of activities they may want to do with students. Funding should be provided for the creation of materials to support teachers when they return to their classrooms. It is also important that all teachers, including current CTE/ICT teachers, have access to these learning opportunities regardless of their locale. This can be accomplished by offering both face-to-face and online course offerings and making teaching materials available at central locations.

#### Strategies for Teacher Communities

Providing teachers with training in computer science pedagogy and content is just one step in the path to supporting educators. Once in the classroom, teachers will benefit from opportunities to continue developing their understanding of how to teach CS and their identities as teachers of CS. This is particularly important as many CS teachers are often the only, or one of the few, people in their schools teaching the subject and can feel isolated in their roles. Computer science teachers and teachers interested in integrating computer science into other courses should have opportunities to connect with professional colleagues with whom they can share experiences. For example, teachers could be given regular collaboration time to develop and participate in COPs, especially at the elementary level. In smaller districts, county offices can encourage COPs that invite teachers from multiple schools.

#### Strategies for Administrator and Counselor Training

Administrators and counselors will play a vital role in the spread of K-12 computer science education. Administrators make decisions related to course offerings and teacher assignments and they support teachers in their schools. Counselors inform students of computer science educational opportunities and possible career paths. Ongoing professional development for site and central office administrators and counselors should be offered to boost knowledge of what computer science is and to provide strategies they can use to support computer science education. Key topics to include in these trainings are: the importance of computer science, career and workforce demand, effective computer science instruction, teacher certification requirements for computer science, and equitable implementation practices.

#### Summary of Recommended Strategies for Supporting Educators to Teach Computer Science

| Strategy | Timeframe | Responsible Entity |
| --- | --- | --- |
| Pass legislation to authorize CCTC to develop a single subject credential in CS | Within 3 years | State |
| Recruit more teachers from colleges and from industry by working with institutions of higher education and promoting the BIPT | 2-4 years | IHEs  State |
| Develop more professional learning opportunities aligned to the CS standards that lead towards credit for authorization to teach CS | 2-5 years | LEAs  COEs  IHEs |
| Support teachers in organizing COPs to continue to develop their understanding of how to teach CS | 2-4 years | LEAs  COEs |
| Provide administrators and counselors with training to learn more about CS and ways they can support CS education | 2-5 years | LEAs  COEs |

## 4. Making Systemic Improvements in Computer Science Education

The previous sections of this document describe strategies for expanding CS course offerings, improving access to CS for all students, and supporting educators to teach CS. However, implementing CS across the state will require attention to issues that span across these levels of the education system. This section provides strategies related to information technology (IT) infrastructure, accountability and evaluation of initiatives, funding and institutional support, and dissemination.

### Strategies for Building Information Technology infrastructure

Sufficient IT infrastructure is required for schools to have consistent access to the tools needed for CS education. Defining standards for networking, hardware, and software is a first step to ensuring consistency in IT across a district. Another step is to identify hardware and software needs for CS education, including identifying areas where bandwidth or Wi-Fi may need to be upgraded. Simply installing IT infrastructure will not suffice to support CS education; personnel are also needed to maintain and update the infrastructure. Standards should be defined for the number and qualifications of IT personnel to ensure each district has adequate maintenance support. Multi-tiered support plans should also be created that include dedicated staff to address advanced IT issues that may arise at schools.

### Strategies for Accountability and Evaluation

In order to determine how well the strategies presented in this document are meeting the state’s goal of implementing CS education for all students, accountability systems are needed to track and evaluate progress. At the state level, CS should be included into a future system for collecting data on enrollment and achievement in CS education. While tracking such data on high school students will be facilitated by the use of course codes specific to CS, determining how to track participation at the K-8 level where CS will likely be integrated into existing courses will require creative strategies.

At the local level, CS education can be added as a college and career readiness indicator on Local Control Accountability Plans (LCAPs) under Priority 7: Course Access. Districts can create 4-year implementation and evaluation plans to help all students achieve the K-12 CS standards. Plans should be educator-driven and educator-focused, leveraging interest among teachers to pilot materials and disseminate information to colleagues. The state could assist districts in their implementation efforts by developing criteria to evaluate CS instructional materials.

Data gathered from these accountability and evaluation activities will provide useful information on the effectiveness of implementation initiatives. Also, stakeholders will likely identify areas of improvement from their experiences. As the new California CS standards will shape California’s CS education in the coming years, they can serve as a vehicle for transmitting these improvements. CDE should convene stakeholders to review the CS standards every seven years to evaluate whether they should be refreshed. If revision is recommended, legislative authority to update the standards should be sought.

### Strategies for Funding and Institutional Support

Implementation plans will not go far without sustained, dedicated funding and staff at the state and local levels to support CS education efforts. Provided with funding, CDE could ensure adequate staff are available to support implementation activities and to provide technical assistance to LEAs. A new Curriculum and Instruction Steering Committee (CISC) sub-committee specific to computer science could be created to support stakeholders in addressing the needs of teachers and to provide a network for professional development activities. And LEAs could have sufficient resources to support the professional development of teachers and school leaders.

Financial support could also help teachers overcome some of the barriers they face when completing coursework towards authorization to teach computer science. For example, a grant program could be established to support teachers in completing course work for the computer science supplementary authorization, with additional incentive for teachers who work in low-income and underserved school districts and rural and urban school districts. Also, a loan forgiveness program could be established to incentivize clear credentialed teachers to teach computer science in these schools.

### Strategies for Dissemination

The state alone cannot implement CS education across California; support from students and families, community members, teachers, administrators, schools, LEAs, IHEs, and industry will be required. To garner buy-in for CS education, funding should be designated to launch a multi-faceted campaign that communicates the California K-12 computer science standards and implementation plan. The campaign should emphasize the future of work, labor demand, and career opportunities requiring computer science.

Awareness roll-out presentations and workshops about the California CS standards should also be developed. These presentations should discuss how computer science aligns with other content standards and with requirements for graduation and university admission. To support these workshops, computer science foundation toolkits that describe the standards and how they can be integrated into the work schools are already doing should be developed for each grade band (i.e., K-2, 3-5, 6-8, 9-12).

Lastly, stakeholders should be recognized for the tremendous efforts they contribute to improving CS education for students across California. The state should identify model teachers, schools, and districts that successfully implement the CS standards during the early implementation stages. By sharing their experiences and best practices with the larger education community, other stakeholders may find motivation to continue with and innovate on implementation efforts in their own regions.

### Summary of Recommended Strategies for Making Systemic Improvements in Computer Science Education

| Strategy | Timeframe | Responsible Entity |
| --- | --- | --- |
| Ensure consistency in IT infrastructure through standards and dedicated personnel | Within 3 years | LEAs  COEs |
| Design accountability systems to evaluate the progress of implementation efforts | Within 2 years | State |
| Include support for CS education in district LCAPs under Priority 7: Course Access | Within 2 years | LEAs  COEs |
| Provide sustained, dedicated funding and staff at the state level and local level to support CS education efforts | 1-2 years | State |
| Provide financial support to help teachers complete coursework towards CS authorization and to work in low-income and underserved school districts and rural and urban school districts | 2-3 years | State |
| Build awareness of CS standards and the implementation plan through roll-out workshops and CS foundation toolkits | Within 2 years | State |
| Recognize successful early adopters as a model for other teachers, schools, and LEAs | Within 3 years | State |

## Conclusion

This plan, in conjunction with the California computer science standards, represents an important initial step in California’s efforts to expand computer science education to all students. Embarking on the strategies recommended in this plan can lead to a future where:

* K-8 students are prepared for secondary level CS coursework, CS course offerings are expanded so that all high schools teach at least one CS course, and students have access to CS in both college and career pathways;
* Students are lifelong learners, informed citizens with a conceptual knowledge of how computing technology works, and productive contributors to society as a whole;
* Students traditionally underrepresented in computer science education have greater access to CS opportunities and stakeholders (i.e., counselors, teachers, families, and community partners) are better equipped with materials and strategies to use in supporting students in pursuing computer science opportunities;
* More individuals are recruited into CS teaching, multiple pathways exist allowing educators to teach CS, and educators have access to resources and peers that support effective CS teaching at all stages of their careers.

Much work remains to be done to guarantee our vision that all students develop foundational knowledge and skills in computer science to prepare them for college, careers, and civic engagement. Successful implementation of equitable K-12 CS education in California will require the collaboration and creative effort of multiple stakeholders across all levels of the education system. This plan should be viewed as the beginning, not the end, of the state’s efforts to achieve equitable, K-12 computer science education for all students.

## Appendices

### Appendix A: Introductory High School Computer Science Courses

Currently, there are three common year-long, stand-alone, introductory high school CS courses nationwide, though none of these courses address all the 9-12 Core CS standards:

* *Exploring CS* (ECS). This is an introductory level CS course that focuses on broadening participation. The course contains six units: Human-Computer Interaction, Problem Solving, Web Design, Programming, Computing & Data Analysis, and Robotics. The ECS curriculum is accompanied by a teacher professional development program that provides ongoing implementation support for teachers and administrators. For more information on ECS, visit <http://www.exploringcs.org/>.
* *Advanced Placement CS Principles* (AP CSP). This is a new AP CS course that aims to broaden participation in CS. It emphasizes the big ideas of CS: creativity, abstraction, data and information, algorithms, programming (in a language of the teacher’s choice), the internet, and global impact. Students in AP CSP also engage in six computational thinking practices: connecting computing, creating computational artifacts, abstracting, analyzing problems and artifacts, communication, and collaboration. The College Board recommends that students have completed a first-year high school algebra course prior to AP CSP. The College Board has endorsed several curricula that come with pre-approved syllabi, lesson plans, and formative assessments, delivered by expert education organizations. These curricula include professional development to prepare teachers, whether experienced or new to computer science. More information about AP CSP is available on the College Board web site: <https://apcentral.collegeboard.org/courses/ap-computer-science-principles>.
* *Advanced Placement CS A* (AP CS A). AP CS A is an introductory course in CS comparable to college-level introductory CS courses. The course focuses on the fundamentals of programming and problem-solving using the Java programming language. The College Board recommends that students have a foundation in English and algebra prior to enrolling in AP CS A. More information about AP CS A is available on the college board web site: <https://apcentral.collegeboard.org/courses/ap-computer-science-a>.

Advanced CS courses also exist for students who move beyond the introductory courses such as: Python Game Programming, Beginning Data Structures and Algorithms, Databases, and Programming for Mobile Devices.

### Appendix B: Resources for Teaching and Teachers

#### California CS Content Standards

<https://www.cde.ca.gov/be/st/ss/computerscicontentstds.asp>

California’s CS standards are based on the revised International Computer Science Teachers Association (CSTA) standards. The CSTA standards were modified to fit California’s educational context and include introductory material and appendices to explain the standards to educators, identify areas of alignment between the CS standards and other content standards, and provide recommendations for implementation.

#### Curricula, Learning Environments, Pedagogical Strategies

##### AccessCSForAll

<https://www.washington.edu/accesscomputing/accesscsforall>

<https://www.washington.edu/accesscomputing/accesscsforall/resources>

A project that develops resources and professional learning for computer science teachers to help them include students with disabilities in their courses.

##### Bootstrap

<http://www.bootstrapworld.org/>

Bootstrap includes research-based curricular modules for grades 6-12. Materials reinforce core concepts from mainstream subjects like Math, Physics and Data Science, enabling non-CS teachers to adopt introductory materials while delivering rigorous and engaging computing content.

##### Code Studio

<http://studio.code.org/>

Code Studio is a combined set of tools and guided lessons to get students in kindergarten through high school interested in the underlying concepts behind coding, with an interface for teachers to monitor where their students are in the lesson progression.

##### Code.org 3rd Party Educator Resources

<https://code.org/educate/curriculum/3rd-party>

A repository of CS curricula and professional development providers curated by Code.org that a school or school district can access to provide an in-school offering for their students.

##### CS Teaching Tips

<http://csteachingtips.org/>

A repository of tips to help CS teachers anticipate students’ difficulties and build upon students’ strengths.

##### CS Unplugged

<https://classic.csunplugged.org/>

<https://classic.csunplugged.org/teachers/>

CS Unplugged is a collection of free learning activities that teach Computer Science without computers through engaging games and puzzles that use cards, string, crayons and lots of running around. They also provide links to other resources related to teaching CS and CS outreach.

##### CS4AllKIDS

~~http://cs4allkids.org/~~ [preceding link is no longer available]

~~http://cs4allkids.org/index.php/lessons-for-teaching-integrated-computational-thinking/~~

[preceding link is no longer available]

A repository of integrated computational thinking lessons for grades K-8 curated by educators in Washington state. Tips from teachers are also provided.

##### EngageCSEdu

<https://www.engage-csedu.org/>

A repository of assignments, tutorials, labs, assessments, lecture notes, exercises and projects. Materials focus on introductory-level college or upper-level high school CS and on engaging a diverse population of students.

##### Family Code Night

<http://www.familycodenight.org/>

A family engagement event where parents and their children do their first hour coding together. An event kit is provided to support organizations in hosting a Family Code Night.

##### Nifty Assignments

<http://nifty.stanford.edu/>

A repository of CS assignments gathered at the annual SIGCSE meeting. Descriptions of assignments are provided along with related materials. Materials focus on introductory level CS.

##### Project Quantum

<http://bit.ly/projectquantum>

A crowd-sourced repository of multiple choice assessment items. Questions cover programming, computational thinking, information technology and digital literacy.

##### Scratch and Scratch Jr.

<https://scratch.mit.edu/>

<http://www.scratchjr.org/>

A block-based programming environment and online community students can use to code their own interactive stories, animations, and games. Scratch targets learners aged 8 and older. Scratch Jr. targets learners ages 5 to 7. To support teachers, Scratch provides a community for educators, in-person gatherings, and guides and tutorials. Scratch is designed and maintained by the Lifelong Kindergarten group at the MIT Media Lab. A list of other block-based programming environments is provided on the Scratch Wiki.

#### Educator Communities

##### Computer Science Teachers Association

<https://www.csteachers.org/>

A membership organization that supports and promotes the teaching of computer science at the K-12 grade levels.

##### CS for All Teachers

<https://csforallteachers.org/>

A virtual community of practice, welcoming all teachers from PreK through high school who are interested in teaching computer science.

##### CSforAll Consortium

<https://www.csforall.org/>

A hub for the national Computer Science for All movement. Provides information on providers, schools, funders, and researchers focused on the goal of providing quality CS education to every child in the U.S.

### Appendix C: California Computing-related Course Codes

Course codes currently collected in CALPADS related to computer science. Source: <https://www.cde.ca.gov/ds/sp/cl/systemdocs.asp>

#### General Education Courses

|  |  |  |
| --- | --- | --- |
| Code | Name | Definition |
| 2323 | Computers and electronics/digital music | This course is devoted to music composed or realized through electronic media or computer applications related to music, such as creative work, music writing and printing, and performances of contemporary compositions. The course is inclusive of music standards instruction, the study of appropriate physics of sound, and students learn to use the equipment through performance of music from other sources and explore personal means of self-expression. |
| 2450 | Computer literacy | This introductory course in computers is designed to acquaint the students with techniques for using computers. Students learn key entry skills in order to use simple word processors, mathematical or database applications, and simple graphics programming. When school resources allow, teachers may introduce students to the Internet, where they learn about different search engines, e-mail, and the variety of educational resources on the Internet. |
| 2451 | Computer programming | This course covers the principles and programming styles used in the design and implementation of contemporary programming languages. Students are introduced to the history of programming languages, language syntax and formal grammars, language processors such as compilers and interpreters, and generalized parsing strategies. The course focuses on particular language constructs and their realization in a variety of programming languages. A particular language such as C, BASIC, or Pascal is used to provide students with practical illustrations of various programming principles. |
| 2453 | Computer science | A generalized computer course that acquaints students with problem-solving methods, algorithm development, structured programming, and modular system design. Students are taught about abstract data structures, techniques for data manipulation and other fundamental concepts, such as recursion. Computer coding and program structure are often introduced using BASIC or another computer language such as C or Pascal. The course may provide opportunities to apply the learned skills to relevant applications, such as modeling, data management, graphics, and text processing. Students learn about computer organization, from digital logic and microprogramming through machine and assembly language. |
| 2454 | Computer lab | This course introduces the student, through hands-on operation, to the use of microcomputers. The class may include basic word processing operations, such as terminology and screen formats, text editing, saving and retrieving, and printing text. |
| 2458 | Other computer education course | This designation is for any computer education course not identified in the series of courses outlined in the CALPADS Course Group State codes. |
| 2465 | IB Computer science | Offered at two levels, Computer Science explores the operation of computer systems and the principle underlying problem solving by using computers. In addition to problem solving and programming, students study the structure and design of computer architecture; data representation and logic; and the connecting processes involving operating systems, interfacing, and networking. At both levels, the course demands both logical discipline and imaginative creativity in the selection and design of algorithms and in the writing, testing, and debugging of programs using a high level, block structured language. |
| 2466 | IB Information technology in a global society | Offered at the subsidiary/standard level only, Information Technology in a Global Society (ITGS) examines the interaction between information, technology, and society. The course is designed to help students develop a systematic, problem solving approach to processing and analyzing information using a range of information tools. The impact of modern information technology on individuals, on relationships between people, and on institutions and societies is also discussed and evaluated. The course focuses on six themes: individuals and machines (comparisons and interactions); information systems in today's society; the system life cycle (problem solving); responsible use of information tools; the social impacts of information technology; and evolution from the past and insight into the future. |
| 2470 | AP Computer science A | Computer Science A emphasizes object oriented programming methodology with a concentration on problem solving and algorithm development and is meant to be the equivalent of a first semester college level course in Computer Science. It also includes the study of data structures, design, and abstraction, but these topics are not covered to the extent that they are in Computer Science AB. |
| 2471 | AP Computer science AB | Computer Science AB includes all the topics of Computer Science A, as well as a more formal and in depth study of algorithms, data structures, design, and abstraction. |
| 2472 | AP Computer Science Principles | The AP Computer Science Principles course is designed to be equivalent to a first-semester introductory college computing course. In this course, students will develop computational thinking vital for success across all disciplines, such as using computational tools to analyze and study data and working with large data sets to analyze, visualize, and draw conclusions from trends. The course is unique in its focus on fostering student creativity. Students are encouraged to apply creative processes when developing computational artifacts and to think creatively while using simulations to explore questions that interest them. They will also develop effective communication and collaboration skills, working individually and collaboratively to solve problems, and discussing and writing about the importance of these problems and the impacts to their community, society, and the world. |
| 2479 | MYP Computer Technology | MYP Computer Technology aims at establishing the foundations for technological literacy and know-how and providing a balance among three key areas: systems, information, and materials. The course allows students to display ingenuity and creativity and to devise practical solutions to given tasks by following the design cycle of investigation, planning, creation and evaluation. This subject area offers great potential for reinforcing and integrating skills learned in other disciplines, especially in the presentation and handling of data and the processes involved in the design and manufacture of a product. |
| 2820 | Digital Art/Computer Art/Artistic Graphics | This course helps students develop aesthetic criteria in order to create graphic art imagery using a microcomputer. The course emphasizes the knowledge and application of the art elements and principles of design as used in visual communication. Students use the computer and digital tools to apply or formulate programs to communicate creative visual ideas, including animation, game production, and artistic aspects of Web site design. This course may also include aspects of the Cinematography/Artistic Videos course. Students study artists who practice these art forms and learn and practice critiques of these art forms. |
| 4619 | Computer Programming for Solving Applied Problems | This course introduces students how a computer programming controls robots in an integrated development environment through computer programming. To control the robot, students use critical thinking, problem solving, effective communication, and team work. This course emphasizes object-oriented programming methodology with a concentration on problem solving and algorithm development. |
| 4634 | Exploring Computer Science | This course provides students with foundational knowledge of computer science. Students will explore topics of human computer interaction, problem solving, web design, computer programming, data modeling, and robotics. Throughout the course, students will understand the algorithmic underpinnings of computer applications and gain technical expertise using computational tools. Course titles may also include: Introduction to Computer Science A; Computer Principles. |
| 4647 | Robotic Technologies | This course introduces students the working principles and foundational knowledge of robotics. Students learn to control a single robot and multiple robots by graphical user interface, pose teaching, and object-oriented programs. Students write robotics programs to perform various tasks based on the sensory information of the robot. Students also learn math and physical sciences with robotics. Through hands-on problem-based projects, students develop critical thinking, problem solving, effective communication, and team work skills. Robots are used as platforms to engage students on collaborative learning of science, technology, engineering, and math. This course emphasizes hands-on robotics activities with a concentration on algorithm development for solving problems in math and science. Course titles may also include: Introduction to Robotics. |

#### CTE Courses

|  |  |  |
| --- | --- | --- |
| Code | Name | Definition |
| 8100 | Introduction to Information and Communication Technologies | This course provides students with foundational knowledge of programming and computer science. Students will explore topics of human computer interaction, problem solving, web design, computer programming, data modeling, and robotics. Throughout the course, students will understand the algorithmic underpinnings of computer applications and gain technical expertise using computational tools. Other areas of the sector could be introduced such as software design, networking, game design, information support, and topics like artificial intelligence, and robotics. Social issues in ICT will be introduced such as hacking and cybersecurity, social media usage and protecting personal information, the digital divide, open government data, and ethical computing. Course titles may also include: Exploring Computer Science, Introduction to Computer Science A; Computer Principles; Introduction to Information and Communication Technologies; Introduction to Computer Technologies. |
| 8110 | Introduction to Information Support Services | This course will introduce students to computer usage and functionality, operating systems, the main system components, network connectivity, software installation, data backup, trouble-shooting, and system administration. The role of ICT in organizations and business processes using tools such as organizational charts, flowcharts, and timelines will be discussed. Client relations and communications will be introduced along with information gathering techniques, and critical thinking and listening as part of problem solving. |
| 8111 | Intermediate Information Support Services (Concentrator) | This course is project-based and designed to provide students with hands-on use of software applications while studying computer concepts such as accessing and transmitting information in a networked environment. Students will learn the processes associated with system administration and planning, acquiring, installing, and implementing of software and systems. Learning to work in teams in order to understand client needs, evaluate different possible solutions, pricing systems within budget constraints, and understanding continual improvement cycles are goals. |
| 8112 | Advanced Information Support Services (Capstone) | This course allows students to demonstrate management and implementation of various information, technology, and communication projects. Projects could take the form of enterprise information security plans that include identifying vulnerabilities and deciding what methods to use to achieve cybersecurity. Other projects include developing user training programs to enable simple problem resolutions, help-desk programs, system life-cycle programs that include planning, purchasing, implementing, and integrating of systems for continual improvement. |
| 8120 | Introduction to Networking | This course introduces students to networking terminology and concepts as well as the principles of networking and various technologies, models, and protocols used in networking. Various types of network media and topologies will be introduced as well as network devices and their functions. Concepts covered include network standards of recognized organizations, Open System Interconnect (OSI) network layers, and transmission-control/internet protocol (TCP/IP) and the various network environments. |
| 8121 | Intermediate Networking (Concentrator) | This course allows students to perform hands-on tasking and planning for implementing network systems and/or subsystems after receiving safety training handling network hardware and power supplies. Students will evaluate networking tasks and select network components, media, and protocols to solve networking challenges. Other skills learned include network addressing, configuring, troubleshooting, security, monitoring network traffic and reading system indicators to troubleshoot problems, network administration and accounts, and system backup. |
| 8122 | Advanced Networking (Capstone) | This course allows students to demonstrate network administration and management skills in various networking projects. Possible projects include implementing network security tools to identify network vulnerabilities and performing network security penetration testing, assessment, proposing resolutions, and forming security plans for implementation. Other projects include identifying network threats to cyber security and plans for disaster prevention and recovery, analyzing client networking needs and requirements and developing possible alternatives to meet client needs. |
| 8130 | Introduction to Systems Programming | This course will introduce the systems development process to students. Topics covered include the development life cycle, development models, specifications and requirements, working in development teams, use of versions, and diagramming processes using flowcharts and Unified Modeling Language. |
| 8131 | Intermediate Systems Programming (Concentrator) | This course provides students with the fundamental knowledge of computer programming for solving applied problems. Topics covered include using various programming languages, protocols, language syntax, data structures, object oriented concepts, interfaces, sorting and searching algorithms, and developing reports. Also covered, software testing, debugging, and improvement, integrated development using object-oriented programming and sensory information from robots to solve problems and meet challenges integrating STEM subjects. |
| 8132 | Advanced Systems Programming (Capstone) | This course emphasizes object-oriented programming methodology with a concentration on problem solving and algorithm development. Students apply discrete programming skills to make a video game, a virtual pet, a sound editor, etc. and will explore careers in programming, including wireless applications for iPhone, Android, and applications. It also includes the study of data structures, design, and abstraction. Other topics might include developing databases and data modeling. |
| 8133 | Introduction to Web and Social Media Programming and Design | This course will introduce students to the integration of various media into programming assignments including Web assignments. Students will also be introduced to the basic design elements used in interactive media. Other topics include encoding methods, using media design and editing software, animation and drawing software as well as image editors and three-dimensional design. Online content delivery process will be outlined as well as establishing online presence and e-commerce capabilities and Web security. Simple coding assignments using HTML and Cascading Style Sheets are used to teach basic Web design including the use of images, hyperlinks, tables, forms, video and audio. |
| 8134 | Intermediate Web and Social Media Programming and Design (Concentrator) | This course allows students to work on various projects such as Web-based businesses, e-portfolios, and mobile apps. Students will learn to create multimedia productions and presentations, implement standard graphical programming techniques for object movement, create graphical user interfaces, and apply graphic design principles and visual communication techniques. Other topics include capturing images by developing camera skills and making choices lenses, depth, motion and lighting, developing digitally generated and enhanced media, modeling, simulation, animation and image retouching. |
| 8135 | Advanced Web and Social Media Programming and Design (Capstone) | This course allows students to produce multimedia projects from concept (content gathering/research) to project completion (authoring/transmission) taking into account media that would be used by enterprises or mobile apps for marketing, corporate communication, and public use. As part of these projects students produce professional-level media, images, documents, audio, and video clips and software. Some projects might include Artificial Intelligence methods and behaviors to create web robots (bots) such as chat bots, etc. |
| 8140 | Introduction to Games and Simulation | This course introduces students to the history, art, and science of game development and the unique differences between automated versus non-automated gaming. Students will be introduced to game and simulation analysis, design, documentation, and development tools. |
| 8141 | Intermediate Games and Simulation (Concentrator) | This course allows students to work in teams to develop games or simulations. Students will learn skills such as storyboarding, plot, flow, and using functions. Learning how to implement standard game/simulation strategy and rules of play as well as integrating mixed media appropriate to the game design/simulation will be included. Other topics include design specifications, delivery, rules of play, navigation functionality, scoring, and other special features. |
| 8142 | Advanced Games and Simulation (Capstone) | This course allows students to learn and display mastery of advanced game design/simulation skills in projects they design individually or in teams. Advanced skills include applying programming skills for rendering single player or multiuser games or simulation projects, program control, branching, memory management, strategies, and implementation. Using Artificial Intelligence techniques such as finite state machines in nonplayer characters could be included. |

### Appendix D: Abbreviations Used in this Document

| Abbreviation | Term |
| --- | --- |
| BOARS | Board of Admissions and Relations with Schools |
| COE | County Office of Education |
| CCTC | California Commission on Teacher Credentialing |
| CDE | California Department of Education |
| CISC | Curriculum and Instruction Steering Committee |
| COP | Communities of Practice |
| CSET | California Subject Examination for Teachers |
| CSU | California State University |
| CTE | Career Technical Education |
| ICT | Information and Communication Technologies |
| IHE | Institutions of Higher Education |
| ITE | Industrial and Technology Education |
| LCAP | Local Control Accountability Plan |
| LEA | Local Education Agency |
| SBE | State Board of Education |
| SSPI | State Superintendent of Public Instruction |
| TPEs | Teaching Performance Expectations |
| UC | University of California |
| UCCI | University of California Curriculum Integration |
| UCOP | University of California, Office of the President |

### Appendix E: Figure Long Descriptions

#### Figure 1. Percent of AP CS Exams Taken by Course, Gender, and Ethnicity in California in 2017

Male students made up 73% of AP CS A examinees and 68% of AP CS Principles examinees. Asian students made up 49% of the AP CS A examinees and 34% of the AP CS Principles examinees. Hispanic/Latinx students made up 14% of the AP CS A examinees and 31% of the AP CS Principles examinees. White students made up 27% of AP CS A examinees and 25% of AP CS Principles examinees. Students from other ethnic groups accounted for less than 10% of examinees. [Return to Figure 1](#Figure1).

#### Figure 2. Number of Teachers of Core Academic CS Courses by Credential (2014-2017)

The figure displays the number of teachers teaching CS courses by their credential (i.e., math, business, ITE, or other credential only) for three academic years starting in 2014-2015. In 2014-2015, 729 teachers of CS courses were credentialed in subjects other than math, business, or ITE. 560 teachers were credentialed in math. 473 teachers were credentialed in business. 227 teachers were credentialed in ITE. This pattern repeats in the 2015-16 and 2016-17 academic years. [Return to Figure 2](#Figure2).

#### Figure 3. Number of Teachers of Core Academic CS Courses by Course Group (2016-2017)

The figure displays the number of teachers teaching CS courses by their credential (i.e., math, business, ITE, or other credential only) and by course group (i.e., AP, IB, and ECS courses, or other courses) for the 2016-2017 academic year. For AP, IB, and ECS courses, 331 teachers were credentialed in math. 231 teachers were credentialed in business. 156 teachers were credentialed in ITE. 216 teachers were credentialed in subjects other than math, business, or ITE. For other courses, this pattern repeats, except there were significantly more teachers credentialed in subjects other than math, business, or ITE. [Return to Figure 3](#Figure3).

California Department of Education, September 2018

1. <https://www.bls.gov/ooh/computer-and-information-technology/home.htm> [↑](#footnote-ref-1)
2. <https://www.bls.gov/oes/2016/may/oes150000.htm> [↑](#footnote-ref-2)
3. <http://www.exploringcs.org/for-researchers-policymakers/reports/ecs-enrollment-data> [↑](#footnote-ref-3)
4. <https://apcentral.collegeboard.org/courses/ap-computer-science-principles/classroom-resources/curricula-pedagogical-support> [↑](#footnote-ref-4)
5. Superintendent’s Advisory Task Force on Accountability and Continuous Improvement (2016). Preparing All Students for College, Career, Life and Leadership in the 21st Century. Sacramento, CA: California Department of Education. <https://www.cde.ca.gov/ta/ac/ar/documents/account-report-2016.pdf> [↑](#footnote-ref-5)
6. A national consensus document that outlines a progression of concepts and practices by grade span in computer science that all students could learn in elementary and secondary school. (<https://k12cs.org/>) [↑](#footnote-ref-6)
7. <https://www.kaporcenter.org/wp-content/uploads/2017/05/lpfi_path_not_found_report.pdf> [↑](#footnote-ref-7)
8. <https://www.kaporcenter.org/computer-science-in-californias-schools-2016-ap-cs-results-and-implications> [↑](#footnote-ref-8)
9. The AP Computer Science Principles exam was offered for the first time in 2017. [↑](#footnote-ref-9)
10. <https://www.washington.edu/accesscomputing/what-aspects-exploring-computer-science-ecs-or-computer-science-principles-csp-curriculum-might-present-accessibility-challenges-students-disabilities> [↑](#footnote-ref-10)
11. <https://www.bls.gov/cps/cpsaat11.htm> [↑](#footnote-ref-11)
12. <https://www.bls.gov/news.release/disabl.t03.htm> [↑](#footnote-ref-12)
13. <https://techcrunch.com/2016/11/16/when-bias-in-product-design-means-life-or-death/> [↑](#footnote-ref-13)
14. <https://www.ctc.ca.gov/docs/default-source/commission/coded/2016/1605.pdf> [↑](#footnote-ref-14)
15. Teachers with a Supplementary Authorization in Computer Concepts and Applications awarded prior to April 2016 are also authorized to teach CS. [↑](#footnote-ref-15)
16. This figure includes teachers who taught a computer science course coded as one of the following: 2451, 2453, 2458, 2465, 2470, 2471, 2472, 4619, 4634. See Appendix C for a description of the courses. [↑](#footnote-ref-16)
17. <http://sites.uci.edu/cs1c/cs1catoc-teacher-certificate-program/> [↑](#footnote-ref-17)
18. <https://www.extension.ucr.edu/certificates/19013519/educationandcredentials/subjectmatterspecialization/computerscienceeducation> [↑](#footnote-ref-18)
19. <https://www.csustan.edu/advanced-studies/educational-technology-edma/supplementary-authorization-computer-science> [↑](#footnote-ref-19)
20. <http://www.ets.org/praxis/prepare/materials/5652> [↑](#footnote-ref-20)
21. <https://encorps.org/> [↑](#footnote-ref-21)