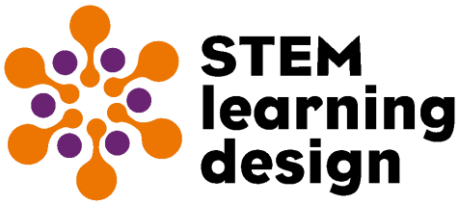
2022 Massachusetts
Digital Literacy and Computer Science
Curriculum Guide

(Updated and expanded from 2021 version)

April 2022
(Updated April 2023)

STEM Learning Design, LLC
Jake Foster
Melissa Zeitz
Lisa Manzi
David Petty



This work was undertaken for the
Massachusetts Department of Elementary and Secondary Education,
under contract awarded for
Digital Literacy and Computer Science Curriculum Evaluator,
RFR #20CISAD2 (2020) and *RFR #21CISPM1* (2021-2022).

www.stemlearningdesign.com

Table of Contents

CONTEXT FOR USING THIS GUIDE 5

Summary of Prior MA DLCS Curriculum Efforts 5
Interpreting Elements of the Curricula Summaries 6
Conducting a DLCS Curriculum Analysis 7
K–12 DLCS Pathways Guidance 12

GRADES K THROUGH 5 15

Considerations for Selecting Elementary School Curricula 16
Action Fractions (3–5) 17
Be Internet Awesome (3–5) 19
Ellipsis Education Computer Science Courses (K–2; 3–5)* 21
Coding and Innovation using micro:bits (3–5) 23
Coding as Another Language (K–2) 25
Computer Science Essentials (K–2, 3–5) 27
Computer Science Fundamentals (K–2; 3–5)* 29
Computing with Minecraft (3–5) 31
Creative Computing Curriculum (3–5) 33
CS First (3–5) 35
Data Handling with micro:bit (3–5) 37
Digital Citizenship (K–2; 3–5) 39
Elementary Computing for ALL (3–5) 41
Growing With KIBO (K–2)* 43
Intro to Computer Science (K–2) 45
Keyboarding Without Tears (K–2; 3–5) 47
Kodable K–2 49
Learn to Code Curriculum (K–2; 3–5) 51
littleBits STEAM+ Coding (3–5) 53
MA STEM+C: Integrated Units (K–2; 3–5) 55
NYC CSforALL: Computer Science Units (K–2; 3–5)* 57
PLTW Launch: Computer Science Units (K–2; 3–5)* 59
SFUSD Creative Computing K–2 Curriculum* 61
Sphero Computer Science Foundations Course 1 (3–5) 63
SPIKE Essential Units (K–2, 3–5) 65
Springfield CS4ALL (K–2; 3–5)* 67
TechnoKids: Selected projects (K–2; 3–5)* 69

GRADES 6 THROUGH 8 71

Considerations for Selecting Middle School Curricula 72
Bootstrap:Algebra 73
Computational Thinking Curriculum 75
Computer Science Applications JavaScript* 77
Computer Science Discoveries* 79
Computer Skills - Digital Savvy* 81
Computing Ideas* 83
Creative Computing Curriculum 85
Developing AI Literacy 87
Digital Citizenship 89
Foundations of Physical Computing: LEGO SPIKE Prime* 91
Intro to CS 93
Introduction to Cybersecurity 95

micro:bit Units 97
NFTE Startup Tech* 99
Ozaria 101
Project GUTS 103
PLTW Gateway: Computer Science Units* 105
Sphero Computer Science Foundations Courses 2 & 3 107

GRADES 9 THROUGH 12 109

Considerations for Selecting High School Curricula 110
AI Foundations* 113
Beauty and Joy of Computing* 115
Bootstrap:Data Science and Bootstrap:Algebra* 117
Code.org Computer Science Principles* 119
CodeHS: Selected Computer Science Courses* 121
Computational Thinking and Problem Solving* 123
Computing with Robotics 125
Cyber.org: Selected Cyber Courses 127
Exploring Computer Science* 129
FIRST Tech Challenge: Robotics Engineering* 131
LocoXtreme 133
Mobile Computer Science Principles* 135
NeuroMaker Modules (9-12) 137
PLTW Computer Science* 139
Science+C 141

APPENDIX 143

Review Process 143
Review Criteria 143
Limitations of this Review 145
MA DLCS Strands, Topics, and Practices 146
Example Application of K–12 DLCS Pathways Guidance 147
Special Topics in DLCS Curricula 155
Programming Languages Used in Curricula 157
Detailed Standards Alignment by Curriculum 160
Contributors 176

An asterisk (*) designates 'comprehensive' curricular materials.

CONTEXT FOR USING THIS GUIDE

This guide provides curricular overviews for schools to engage students in learning of digital literacy and computer science (DLCS) concepts and skills aligned to the standards found in the *2016 Massachusetts DLCS Framework*. Each curriculum overview describes topical alignment and key instructional features. A variety of curricula are included for each grade span to provide schools a range of options to meet differing program needs and conditions. The curricula presented in this guide are not inclusive of all DLCS curricula that may meet the review criteria.

SUMMARY OF PRIOR MA DLCS CURRICULUM EFFORTS

The ability to effectively use and create technology to solve complex problems is an essential literacy skill in the twenty-first century. The Department of Elementary and Secondary Education (DESE) has collaborated with the Massachusetts Computing Attainment Network (MassCAN) and Massachusetts Computer Using Educators (MassCUE) to develop policies and regulations to support this essential literacy. These include, for example, a new DLCS grade 5–12 teacher license and revised Instructional Technology license (2017), and additional K–5 DLCS subject matter knowledge requirements for all pre–service licensure programs (2018).

Since the 2016 DLCS Framework adoption, DESE has partnered with state, federal, and private sector partners to advance DLCS implementation, curriculum, and professional development:

- *Broadening Participation of Elementary Students and Teachers in Computer Science*, a National Science Foundation (NSF)-funded project, that partners DESE, the Education Development Center (EDC), and districts to develop and pilot curriculum modules to focus on computational thinking in grades 1–6. (sites.google.com/site/stemcwithct)
- *Massachusetts K–12 Computer Science Curriculum Guide*, a collaboration of DESE and EDC, helps school districts choose computer science curricula best suited to their communities. (edc.org/sites/default/files/uploads/CurriculumGuide-web.pdf)
- *High School Computation Science Pathway*, another NSF-funded project, is a collaboration with EDC to develop a Computational Science Pathway to College option for Massachusetts high school students.
- *Systemic Change to Improve Equity in Computer Science Student Achievement*, a USED-funded project, partners DESE, EDC, the Massachusetts Association of School Superintendents (M.A.S.S.), and the CSforMA (regional partner for Code.org) to implement a comprehensive district change model that embeds computer science coursework for middle school students in the 7 participating districts.
- *Strategic CS for ALL Resource and Implementation Planning Tool* (SCRIPT) workshops provide a framework to guide district teams through visioning, self-assessment, and goal-setting to implement a K–12 DLCS program.
- *CSforMA (*previously *Broadening Advanced Technological Education Connections*), with funding from Code.org, NSF, and the College Board, offers no cost training and ongoing support for districts and educators to implement computer science programming.
- *Digital Literacy Now Grant*, a state grant designed in partnership with M.A.S.S. to promote K–12 DLCS education in participating districts through developing implementation plans, choosing curricula, and professional development.

This guide builds on these efforts.

INTERPRETING ELEMENTS OF THE CURRICULA SUMMARIES

This guide provides information on a variety of options for district consideration. Each curriculum summary includes information about the nature of the curriculum and alignment to MA DLCS topics, that support district needs (comprehensive coverage vs. strand-specific need).

A detailed overview of the review process and criteria is included in the Appendix. A few summary comments here will help in the interpretation of curriculum summaries to follow.

- An asterisk (*) in a title designates 'comprehensive' curricular materials--resources designed for use with all students in a given class that address multiple DLCS strands and topics--as differentiated from supplemental materials, which are more limited in scope.
- Summary table
    - Integrated vs. Stand-alone: indicates whether the curriculum is designed to be (or has strong potential to be) integrated with other core academic subjects or offered on its own.
    - Indications of whether the curriculum includes Assessment, Differentiation, English Learner, Teacher content knowledge, and culturally responsive supports: uses "Yes," "Limited," or "No" labels, with explanations included in the summary paragraph on the second page.
    - Web-based: indicates whether the curriculum can be done entirely on the web or a cloud-based device.
- Coding for topic alignments
    - Substantial: 2/3 or more of the standards in the topic are addressed by the curriculum.
    - Partial: between 1/3 and 2/3 of the standards in the topic are addressed by the curriculum.
    - No alignment: between 0 and 1/3 of standards in the topic are addressed by the curriculum.

Finally, detailed standard-by-standard alignments are provided in the Appendix.

CONDUCTING A DLCS CURRICULUM ANALYSIS

The purpose of this guidance is to assist schools and districts in analyzing current DLCS programming to identify curricular needs for a comprehensive DLCS program.

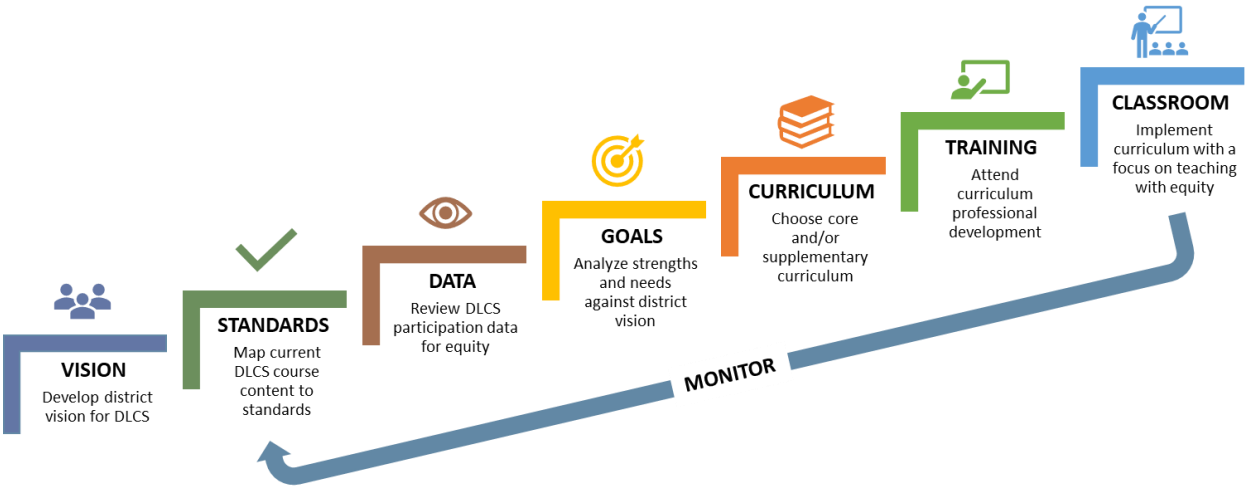
Set up for success

Before getting started, it is important to put several foundations in place:

- Build a team for this work. To ensure success, this team should include district and school decision makers who can support the implementation and move it forward. DLCS standards are often taught across the curriculum, particularly digital literacy skills, and student learning experiences across grade levels is often widely variable. It is recommended that your team consists of at least: one district administrator, one building administrator, one DLCS teacher, and one DLCS curriculum leader, if this role exists in your district. Your work will be much more effective with team members who are knowledgeable about current DLCS practices, who may be key in implementing future DLCS curriculum, and/or who can support decision making and implementation.
- Know where to get data on DLCS enrollment with demographics on enrollment such as gender, race/ethnicity, SWD, ELL, and Economically Disadvantaged (www.doe.mass.edu/infoservices/data/ed.html). Identify who in the district can provide course and enrollment data, or identify the district process to request data. Often a registrar or guidance department can provide this data.
- Consider attending a facilitated SCRIPT workshop (www.csforall.org/projects_and_programs/script/) to be attended by your team.

Overview of the process

The graphic below outlines the general process for building a strong K-12 DLCS implementation in which conducting a DLCS curriculum analysis is an integral step. To find more on this process and associated resources, see *DLCS District Implementation Process* (bit.ly/DLCS_Imp_Doc)



The steps below can be used to complete a thoughtful DLCS curriculum analysis. These are intended to be a guideline for your work; your district context may call for changes to these steps in some way.

DISTRICT and SCHOOL VISION

It is important that your team is clear about the vision and goals of a district and school DLCS program. Throughout this process you will reflect on how your work and choices advance the vision and goals. Your vision and goals may be adjusted as you progress through the process. The goal of this *DLCS Curriculum Guide* is to assist in the implementation of a comprehensive DLCS program that addresses the range of topics in the *2016 Massachusetts DLCS Curriculum Framework*.

MAP STANDARDS

Map your district's current implementation of DLCS curriculum and learning opportunities. Use the appropriate *Standards Alignment* worksheet to record where DLCS standards are currently being addressed in school curriculum:

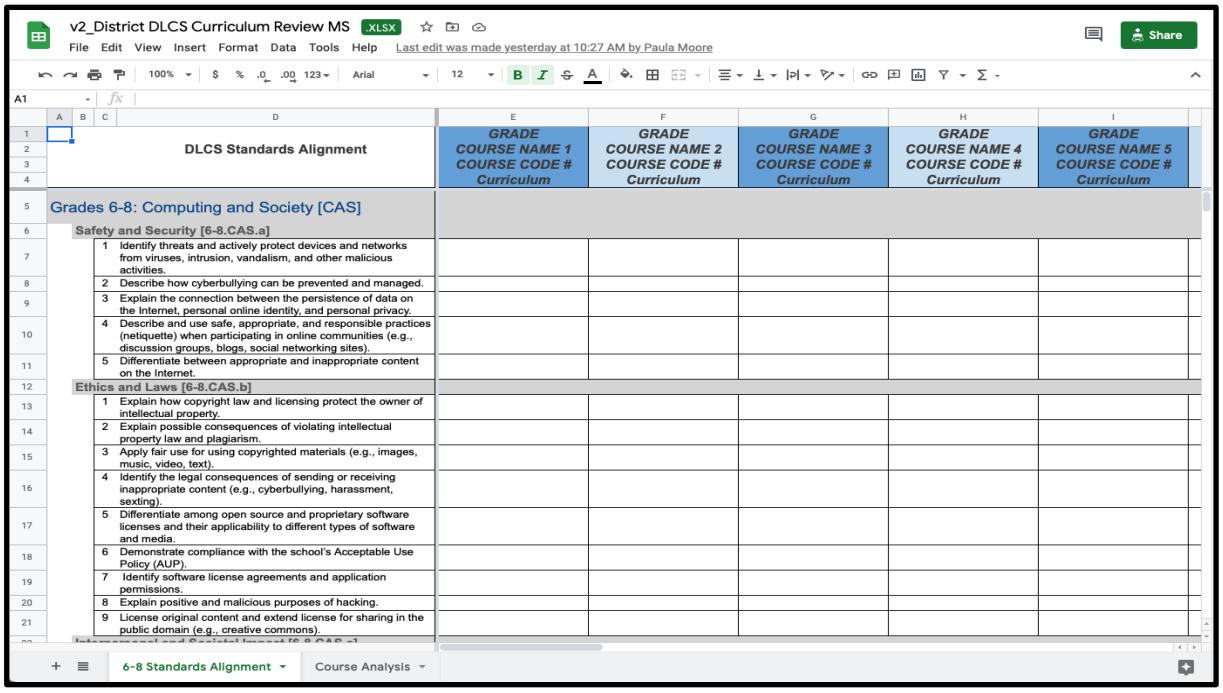
K-2 (www.doe.mass.edu/stem/dlcs/gk-2-standards-alignment.xlsx)

3-5 (www.doe.mass.edu/stem/dlcs/g3-5-standards-alignment.xlsx)

6-8 (www.doe.mass.edu/stem/dlcs/g6-8-standards-alignment.xlsx)

9-12 (www.doe.mass.edu/stem/dlcs/g9-12-standards-alignment.xlsx)

1. Identify in which classes or courses DLCS standards are being addressed. Look across the curriculum! For example, Digital Literacy standards are often addressed in core subjects such as ELA and History/Social Studies.
2. On the first tab, indicate in relevant cells where a particular standard is taught; leave cells blank if the standard is not addressed in the course. You may also add notes here to indicate if a standard is taught in depth, only touched upon, or if a particular course is a good place to teach this standard in the future.
3. On the second tab, include basic information for each class or course (such as grade level, whether it is required or elective, and student contact hours).



v2_District DLCS Curriculum Review MS XLSX

File Edit View Insert Format Data Tools Help Last edit was made yesterday at 10:27 AM by Paula Moore

Share

	DLCS Standards Alignment	GRADE COURSE NAME 1 COURSE CODE # Curriculum	GRADE COURSE NAME 2 COURSE CODE # Curriculum	GRADE COURSE NAME 3 COURSE CODE # Curriculum	GRADE COURSE NAME 4 COURSE CODE # Curriculum	GRADE COURSE NAME 5 COURSE CODE # Curriculum
Grades 6-8: Computing and Society [CAS]						
Safety and Security [6-8.CAS.a]						
1	Identify threats and actively protect devices and networks from viruses, intrusion, vandalism, and other malicious activities.					
2	Describe how cyberbullying can be prevented and managed.					
3	Explain the connection between the persistence of data on the Internet, personal online identity, and personal privacy.					
4	Describe and use safe, appropriate, and responsible practices (netiquette) when participating in online communities (e.g., discussion groups, blogs, social networking sites).					
5	Differentiate between appropriate and inappropriate content on the Internet.					
Ethics and Laws [6-8.CAS.b]						
1	Explain how copyright law and licensing protect the owner of intellectual property.					
2	Explain possible consequences of violating intellectual property law and plagiarism.					
3	Apply fair use for using copyrighted materials (e.g., images, music, video, text).					
4	Identify the legal consequences of sending or receiving inappropriate content (e.g., cyberbullying, harassment, sexting).					
5	Differentiate among open source and proprietary software licenses and their applicability to different types of software and media.					
6	Demonstrate compliance with the school's Acceptable Use Policy (AUP).					
7	Identify software license agreements and application permissions.					
8	Explain positive and malicious purposes of hacking.					
9	License original content and extend license for sharing in the public domain (e.g., creative commons).					

6-8 Standards Alignment | Course Analysis

COLLECT DLCS DATA

Obtain data for each course identified in the mapping exercise. Data can include:

1. Course code # (if not identified above)
2. Student enrollment; current and over time
3. Student demographics; current and over time
4. Other data or details as desired, particularly to examine access and performance of different student populations.
5. See the linked sample (https://tinyurl.com/dp2tucp6) for what data may be useful to collect, and how to analyze it to understand engagement of different student populations.
    - Raw data should be available from your district data office/person. The sample spreadsheet provides a template of what can be done with the data. Additional district data may be available that can support even further analysis.
    - The course types, "DL", "CS", "DLCS", are not found in typical district data. This sheet (https://tinyurl.com/yc3t4zdy) provides details to incorporate this in your analysis.

ANALYZE DLCS DATA

Using the STANDARDS MAP and DLCS DATA, engage your team in an examination of program strengths and needs. These sample facilitation questions may be useful:

1. Do all students have access to DLCS curriculum in all grades? All student populations, including SWD and ELL students?
2. Do our current offerings provide opportunities to learn all DLCS topics and standards (a comprehensive approach)? What are our strengths? Key needs?
3. How can we enhance access, opportunity, and interest to learn DLCS standards for all students, and all student populations?
4. What grade levels, DLCS topics, or standards need enhancement? Aim to comprehensively address DLCS topics and standards.
5. What are the particular curriculum design features and instructional strategies that our current DLCS programming emphasizes? How do those relate to other curriculum areas?
6. What are the particular curricular design features, format, or themes (e.g., integrated; stand-alone; project-based; web-based; physical computing/robotics; artificial intelligence) that will complement or advance existing programming?
7. What is offered to students in the grade(s) above and/or below, and how do those experiences contribute to student needs at the grade(s) being examined?
8. Are certain student groups reflected and centered in the curriculum while other groups (e.g., female, BIPOC) are not?
9. How are we ensuring that the DLCS content is reflecting varied student interests, including those from underrepresented groups?
10. Are there any infrastructure needs (e.g., number of teachers, number of course sections) that are impacting student participation?

SET GOALS FOR THIS YEAR

After mapping your current implementation and analyzing your district data, assess where your district is related to the DLCS vision for your district. Use this assessment to make goals for this year's implementation. Building a strong DLCS implementation in a district typically requires a phased approach with an implementation focus on one grade span per year. We recommend beginning with middle school the first year and expanding to high school or elementary (K-2 or 3-5) in subsequent years.

SELECT CURRICULA

Based on your team's analysis, choose core and/or supplementary curriculum to enhance your DLCS program and student learning opportunities.

1. Use the curriculum summaries in this *DLCS Curriculum Guide for Massachusetts Districts* (**Error! Hyperlink reference not valid.**, www.doe.mass.edu/stem/dlcs/curriculum-guide.pdf), shown below, to identify potential curricula that help meet needs identified in the analysis above.
2. Consider how your potential choices contribute to a coherent progression of learning. See the next section on Pathways Guidance for relevant considerations.
3. Identify a few units or lessons to review, try out, and/or pilot.
4. Identify necessary teacher professional development and materials needed to pilot a curricular choice.
5. Create a support team or PLC for teacher(s) piloting the course either within your district or with other districts.

ATTEND TRAINING

Successful implementation of any curriculum materials is enhanced when participating educators and school leaders are provided professional development on the goals, design, and practices of the curriculum. Training prior to initial implementation, and periodic support for participating educators during implementation, helps ensure that students engage with the curriculum as intended, and adjustments made to account for local conditions are consistent with the curriculum design. Note that a curriculum professional development workshop typically assumes that the teacher has DLCS content knowledge. If a teacher has limited or no CS background, they should take an introductory CS course prior to the curriculum PD.

IMPLEMENT CURRICULUM and MONITOR

Once a curriculum choice is settled on, and educators have engaged in relevant training, school and district efforts are on implementation and monitoring. Districts can refer to implementation guidance provided by DESE or others as helpful, such as the DESE IMplement MA Process (www.doe.mass.edu/instruction/impd/implement-ma-process/story.html).

K–12 DLCS PATHWAYS GUIDANCE

This section provides guidance for choosing a curriculum that contributes to a **coherent DLCS pathway** in a district.

To choose curricula that contribute to a thoughtful pathway across grades, consider how each curriculum in a progression:

1. Contributes to your program goals.
    a. All 'core curriculum' options should contribute to a **goal of a comprehensive DLCS program** (substantial standards alignment) at their respective grade spans.
    b. District and school DLCS programs should also include **goals that account for local context**, particularly prior student experience, cultural responsiveness, course structure (e.g., a dedicated, stand-alone DLCS course vs. integrating DLCS instruction into core academic subjects), and student contact time. In the early stages of building a district-wide DLCS progression, inequities will exist due to differing student experiences and prior opportunities to learn DLCS concepts and practices. Thoughtful implementation and monitoring as a program develops can reduce inequities and enhance DLCS learning and engagement for all students.

2. Purposefully builds from one curriculum to the next. Consider:
    a. **Alignment to grade level standards, including DLCS practices**, to provide an appropriate progression of learning goals across grade spans. Practices and skills such as problem-solving, analyzing, abstracting, creating artifacts, and collaborating are key components of a comprehensive and robust program of study.
    b. The progression of **how students are engaged in coding within and across curricula** (e.g., from block-based to text-based). Starting with block-based programming promotes conceptual understanding and a good foundation for text-based programming.
    c. Reasonable **progression of coding languages**. Consider the purpose of each coding language that is introduced across a progression; how different programming languages build upon each other or provide an opportunity for new applications or more advanced concepts. Try not to have progression with a large number of languages that appear randomly chosen, or do not relate to key goals or applications.
    d. Opportunities for **physical computing** that should, similar to coding languages, be thoughtfully chosen to ensure they have a purpose, contribute to progressive student skill development, and support thoughtful applications. Applications of different platforms (e.g., Dash/Dot, BeeBots, micro:bit, Raspberry Pi) should be thoughtfully planned across the curriculum progression.
    e. Whether **explicit connections** are made from one curriculum to the next, so both students and adults understand the trajectory of learning and application.

3. Contributes to pedagogical coherence.
    a. Ensure each curriculum in a progression r**eflects and builds on pedagogical approaches or strategies from curricula in prior grades, or other subjects** in the same grade span. Having an instructional vision brings coherence to student learning. Ensuring connections across curricula in regards to pedagogical strategies, content area practices, and curriculum design features all contribute to coherence.

4. Contributes to, or allows for, a focus on equity.
    a. Chosen curricula should provide for, or be flexible and adaptable enough to allow for, **scaffolding and differentiation for students** of various experience levels or backgrounds. This may be considered at the program, course, or unit level.
    b. Chosen curricula should be, or can be adapted to be, **culturally responsive** in a way that is appropriate to the grade span and student population.
    c. Each curriculum in a progression should support students and teachers to explicitly **address issues of equity and present strategies for cultural responsiveness in increasingly frequent and sophisticated ways**.

5. Provides for high school options.
    High school students are likely to have a wide range of goals relative to DLCS programming. Aim to provide options for students with different goals and prior experience. Two different approaches may include:
    a. Integration with other disciplines. This may involve, for example:
        i. Science teachers integrating computational thinking through modeling and simulation into core Biology, Chemistry, and Physics courses to provide real-world examples of CS in Science
        ii. History teachers using public data sets to provide student experience in collecting and analyzing data to enhance civic projects.
        iii. Potential for taking advanced courses such as an AP CSP.
    b. Advanced pathway options, with at least two (2) progressive courses, designed to address varied student goals. For example:
        i. Student goal: “I am interested in graphics design and app development.”
        ii. Student goal: “I want exposure to hardware and software development in robotics.”
        iii. Student goal: “I am interested in Business and want to know more about Data Analytics”

    Each of these goals implies different combinations of DLCS-related courses that may start with a general DLCS course (e.g., ECS, CTPS, CSP [non-AP]) then branch to different options for specialization and application contexts (e.g., digital media, animation, AI, cybersecurity, robotics, data analytics). Any pathway can include various advanced course options.

See the Appendix for an *Example Application of K–12 DLCS Pathways Guidance*.

(This page left intentionally blank.)

GRADES K THROUGH 5

CONSIDERATIONS FOR SELECTING ELEMENTARY SCHOOL CURRICULA

This guide presents a range of curriculum options for both the K–2 and 3–5 grade spans. A number of organizations offer curriculum that spans K–5; a few others are included that focus on one span or the other. A few of the curricula have a relatively broad scope that approaches comprehensive coverage of the topics in the MA DLCS standards, but no one curriculum addresses all. A number of organizations included in this section also have middle school offerings, providing for the possibility of pathways using curricula from the same organization.

If a district does not have a DLCS curriculum at the elementary school levels, or is looking to start anew, the following options are suggested as a starting point given their relatively broad scope to provide a strong foundation for a comprehensive DLCS program. This ensures coherence across topics, and can streamline professional development and materials.

- For grades K–2:
    - *Ellipsis Education Computer Science Foundations* (K–2)
    - *NYC CSforALL Computational Media Explorer – Computing Through Time* (K–2)
- For grades 3–5:
    - *Ellipsis Education Computer Science Fundamentals* (3–5)

Any DLCS curricula at the elementary grades will have gaps relative to the full MA DLCS standards. The suggestions below provide some examples of how different curricula pairings might result in a relatively comprehensive offering.

- For grades K–2:
    - Use *Computational Media Explorer – Computing Through Time* as a base and pair it with *Digital Citizenship K–2*.
    - Use *Growing with KIBO* as a base and pair it with *Digital Citizenship K–2*.
- For grades 3–5:
    - Use *Technokids* as a base and pair it with *Digital Citizenship 3–5*.
    - Pair *Digital Citizenship 3–5* with *Learn to Code* and *STEM+C*.
    - Pair *Digital Citizenship 3–5* (for Computing and Society [CAS] with selected *Technokids* projects that address DTC, and *Computer Science Fundamentals D-F* (for Computational Thinking [CT]).

If a district already has DLCS programming in place and is looking primarily to fill known gaps, then use the topical alignment chart (presented with each curriculum, or provided in the Appendix) to identify which options best address particular topics.

There are many other combinations of pairings possible, particularly if a district is already using resources not included in this guide.

ACTION FRACTIONS (3–5)

Canon Lab
canonlab.org/action-fractions-materials

Integrating mathematics and computational thinking, using Everyday Mathematics.

Nature of curriculum	Integrated
Format and time	Units (10–12 hours in both grades 3 and 4)
Key classroom routines	Warm up, Focus, Wrap Up; mix of whole-class, small-group, partner work, and independent work
Assessments, rubrics, examples	Yes
Differentiation supports	Yes
English Learner supports	Limited
Teacher content knowledge support	Yes
Cultural responsiveness	No
Programming platform or language	Scratch (online blocks-based environment)
Web-based	Yes

Gr. 3–5 DLCS Topics	**3–5 *Action Fractions* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	*Partial*
DTC.b. Collaboration & Communication	✓
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

ACTION FRACTIONS

Overview

This curriculum includes fraction units integrating computer science and computational thinking with *Everyday Mathematics*. The focus is on computational thinking concepts while embedding computer science skills. There are 11 lessons for grade 3 and 12 lessons for grade 4. Lessons clearly state when whole-class, small-group, partner, or independent structures are planned, and include a warm up focus and a wrap up. All lessons include "I can" statements, graphic organizers, and age-appropriate vocabulary. Teacher support is provided through Teacher Tips and Anticipated Barriers, highlighting issues that may rise when students are working on a concept. A student option is provided in each lesson, and the curriculum provides suggestions for how teachers could modify the curriculum if students are struggling with a barrier. Student work exemplars are provided for Scratch-related exercises and strategy tips are available to students on how to program on Scratch. Reflection questions are provided in wrap ups, and assessment opportunities are included through each unit, including early, middle, and late assessments. Every lesson includes teacher notes, slide decks, student printables, a summary on connections to other areas, and examples.

Technology requirements

- Requires computers or tablets with speaker or headphones for the computer- based design activities; Internet connectivity; up-to-date browser to connect to Scratch online. A downloadable version of Scratch is available.

Professional development offered

- None specific to this curriculum.

Costs

- The curriculum and Scratch software are free.

BE INTERNET AWESOME (3–5)

Google
beinternetawesome.withgoogle.com

Help students be safe, confident explorers of the online world.

Nature of curriculum	Stand-alone
Format and time	Course (5 lessons, ~5 weeks)
Key classroom routines	Discussions, unplugged activities, games
Assessments, rubrics, examples	Limited
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	No
Programming platform or language	Interland (online game)
Web-based	Yes

Gr. 3–5 DLCS Topics	**3–5 *Be Internet Awesome* Alignment**
CAS.a. Safety & Security	✓
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	X
DTC.b. Collaboration & Communication	X
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	X
CT.b. Algorithms	X
CT.c. Data	X
CT.d. Programming & Development	X
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

BE INTERNET AWESOME

Overview
The curriculum teaches digital safety and citizenship fundamentals through gamification techniques, using Interland, an interactive online game. Lessons provide collaborative activities with discussion talking points, examples, and activities. The curriculum is designed for any teacher to pick it up and teach without prior experience. The curriculum is also appropriate for use in middle school with students who have not had prior instruction in digital safety and citizenship. Assessment opportunities are not specifically provided, but are possible through occasional guidance given (e.g., 'cheat sheets') for selected discussions or activities. Limited teacher content knowledge support is provided through brief notes to the teacher at the beginning of some lessons. The curriculum is also available in Spanish.

Technology requirements

- Interland works on any device that has an Internet connection.

Professional development offered

- *Digital Citizenship and Safety* course offered online.

Costs

- All curriculum, presentations, handouts, and professional development are free.

ELLIPSIS EDUCATION COMPUTER SCIENCE COURSES (K–2; 3–5)*

Ellipsis Education (formerly Codelicious)
ellipsiseducation.com/courses/

Customizable courses to support learning of digital literacy and computer science principles.

Nature of curriculum	Stand-alone
Format and time	Courses (K–2 up to 28 modules; 3–5 up to 63 modules)
Key classroom routines	Group discussions, partner work, unplugged, plugged, centers
Assessments, rubrics, examples	Yes
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	No
Cultural responsiveness	Limited
Programming platform or language	ScratchJr (K–2), Scratch (3–5) (app or online blocks-based environments)
Web–based	Yes

Gr. K–5 DLCS Topics	***Computer Science Foundations (K–2)* Alignment**	***Computer Science Fundamentals (3–5)* Alignment**
CAS.a. Safety & Security	✓	✓
CAS.b. Ethics & Laws	✓	*Partial*
CAS.c. Interpersonal & Societal Impact	✓	*Partial*
DTC.a. Digital Tools	✓	*Partial*
DTC.b. Collaboration & Communication	✓	✓
DTC.c. Research	*Partial*	*Partial*
CS.a. Computing Devices	✓	*Partial*
CS.b. Human & Computer Partnerships	✓	✓
CS.c. Networks	✓	✓
CS.d. Services	NA	X
CT.a. Abstraction	✓	✓
CT.b. Algorithms	✓	✓
CT.c. Data	✓	*Partial*
CT.d. Programming & Development	✓	✓
CT.e. Modeling & Simulation	X	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

ELLIPSIS EDUCATION COMPUTER SCIENCE COURSES

Overview

Ellipsis Education computer science courses support learning of digital literacy and computer science principles. *Computer Science Foundations* (grades K–2) includes up to 28 modules and *Fundamentals* (developed for grades 3–8) includes up to 63 modules; schools can select modules based on their needs. Modules are customized to each school's timing requirements. Meaning the full year of curriculum can be delivered in 20min, 30min, 60min, or other as needed. Modules intertwine digital citizenship lessons, stem careers, unplugged and plugged lessons. Students learn how technology plays a role in our lives, in careers, and about its development through time. The plugged lessons use Scratch JR for grades K–2 and Scratch for grades 3–5. There are some mini-projects using Scratch Jr and Scratch to advance development of certain skills. All lessons include objectives, lesson procedures, examples, materials, and reflection prompts. Pre- and post-assessments include both informal and formal assessments. The lessons are accessed through Canvas, with teacher lesson plans also available via downloadable pdf files, which include detailed lessons to support implementation. Throughout the teacher and student materials specific attention is provided to representing a diversity of people in all images and highlighting diversity in related career descriptions.

Technology requirements

- ScratchJr can be used on either iOS, Android devices, or a Chromebook. It can be downloaded onto tablets and used offline.
- Up-to-date browser to connect to Scratch online. A downloadable version of Scratch is available.

Professional development offered

- Teacher training on Ellipsis Education curriculum and platform is available. PD is typically 3–4 hours, provided virtually with a live customer support team member.
- Customer and training support continues beyond the PD through a dedicated Customer Experience Team. Supports are provided for computer science integration, including adding computer science into an existing class period, a specials rotation, or introducing a stand-alone class.

Costs

- Curriculum pilots are available for $750. Full year courses are $75 per curriculum hour, with a per building cost cap of $2,000 for elementary schools (K–5), including multiple course purchases, with no educator or student seat limits and no kits to buy.
- Included with curriculum at no extra cost: product and platform video-based training, and ongoing customer support throughout the subscription period.
- Virtual live teacher training sessions: $500 per educator.

CODING AND INNOVATION USING MICRO:BITS (3–5)

Utah Coding Project
sites.google.com/view/utahcodingproject/microbits/coding-innovation

Introduction to computer science and innovation using micro:bits.

Nature of curriculum	Stand-alone
Format and time	Course (about 8 weeks)
Key classroom routines	Intro to Concept, Unplugged Activity, Guided Activity, Innovative Project, Reflection
Assessments, rubrics, examples	Yes
Differentiation supports	Yes
English Learner supports	Limited
Teacher content knowledge support	Yes
Cultural responsiveness	Limited
Programming platform or language	Blockly (online blocks-based environment); micro:bits
Web-based	Yes

Gr. 3–5 DLCS Topics	**3–5 *Coding & Innovation* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	✓
DTC.c. Research	X
CS.a. Computing Devices	✓
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

CODING AND INNOVATION USING MICRO:BITS

Overview

Coding and Innovation using micro:bits is designed for grades 5–7 but is very appropriate for grade 5. It is broken into 8 modules: Overview; Design and Making with micro:bit; Software and Hardware (Algorithms); Everything Counts (Variables); Making Decisions (Conditionals); Music, Designs and LEDs (Loops); Radio Communication; and, Innovative Project. The curriculum uses a "making" approach and embeds a mini-innovation project in each module to assess student understanding while also encouraging students to make personal connections. There is a heavy focus on collaboration and problem solving throughout the curriculum while weaving computing standards and computational thinking principles. The curriculum includes a student reflection book with reflection sheets, planning sheets, assessment rubrics, and sentence frames to support EL instruction. Each lesson has lesson objectives, detailed lesson plan, vocabulary, pair programming, unplugged and plugged activities, modifications, projects, reflection, and assessment. Each lesson has examples of final projects to support teacher understanding with details of how to write programming during the whole class lesson. This curriculum is designed to support teachers with minimal Computer Science background. Each of the first six modules are about 2–3 hours and the last module will take about 2 weeks for students to complete.

Technology requirements

- Access to the MakeCode platform via computer, laptop, or tablet with an internet connection, modern browser, and USB port.
- Recommended 1 micro:bit per student. Note that there is now a version 2 micro:bit available.

Professional development offered

- None noted.

Costs

- The curriculum is free.
- Individual micro:bits are approximately $15 each; consider mini-kits that include battery holder, USB cable, and such for approximately $20 each.
- Materials for activities, approximately $400. Many of the electronic components for use with the micro:bits are reusable.

CODING AS ANOTHER LANGUAGE (K–2)

DevTech Research Group at Tufts University
sites.tufts.edu/codingasanotherlanguage/

Relate stories, writing, and coding through literature with ScratchJr.

Nature of curriculum	Stand-alone or Integrated
Format and time	Course (24 lessons, 45 min per lesson)
Key classroom routines	Warm Up, Opening Tech Circle, Scratch Jr Time, Word time, Closing Tech Circle
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	Limited
Teacher content knowledge support	Limited
Cultural responsiveness	Yes
Programming platform or language	ScratchJr (blocks-based environment)
Web-based	No

Gr. K–2 DLCS Topics	**K–2 CAL *ScratchJr Curriculum* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	✓
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	NA
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	*Partial*
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

CODING AS ANOTHER LANGUAGE

Overview

The *Coding as Another Language* curriculum brings literature and coding together in a grades K, 1, and 2 curriculum. The design approach is to present coding as a playground, coding as another language, coding as a bridge, and coding as a palette of virtues. The curriculum consists of warm up activities, opening/closing technology circles, structured challenges, expressive exploration, unplugged activities, and word time. Students use a design journal throughout the lessons. Each grade includes two books, a biography and a fiction book featuring an animal. Kindergarten uses *A Computer Called Katharine: How Katharine Johnson Helped Put America on the Moon* and *Knuffle Bunny*; Grade 1 uses *Ada Byron Lovelace & the Thinking Machine* and *Where the Wild Things Are*; Grade 2 uses *Grace Hopper: Queen of Computer Code* and *Stellaluna*. The curriculum shows students of multiple ethnicities, highlights programmers that come from all walks of life, and brings in a variety of languages through the hello lesson. Opportunities for socio-emotional development are provided alongside the work with ScratchJr and literacy. The culmination of the unit is an open-ended project. Throughout the curriculum there are formative and summative assessments along with rubrics for the ScratchJr projects. The curriculum gives pictures to support teacher understanding of what to do on ScratchJr, and there are remote learning alternatives embedded into the curriculum. Limited differentiation support is provided through an introductory section that talks about ways to modify, structure, and group students based off of class needs. There are a few overlaps among lessons across grades but together they provide a depth of computer science concepts.

Technology requirements

- The curriculum states that one iPad per student is expected, however, activities can be adapted for groups of two or three students per device.
- ScratchJr can be used on either iOS, Android devices, or a Chromebook. It can be downloaded onto tablets and used offline.

Professional development offered

- DevTech Research Group professional development opportunities are typically 2 days and address hands-on introduction and curriculum integration with ScratchJr.
- There is also a webinar series for educators that addresses a variety of topics related to early childhood education and technology.

Costs

- The curriculum and software are free. The curriculum uses several books and some craft supplies that would need to be purchased at limited cost.
- Professional development opportunities are currently $100 per day per teacher.

COMPUTER SCIENCE ESSENTIALS (K–2, 3–5)

EiE, Museum of Science, Boston
https://www.eie.org/stem-curricula/computer-science

Go beyond coding and practice computational thinking with an engineering approach to problem solving.

Nature of curriculum	Stand-alone or Integrated
Format and time	5 Units (one unit per grade, 8-10 hrs each)
Key classroom routines	Storybook, collaborative coding, reflection
Assessments, rubrics, examples	Yes
Differentiation supports	Yes
English Learner supports	Yes
Teacher content knowledge support	Yes
Cultural responsiveness	Yes
Programming platform or language	K-2: ScratchJr (blocks-based environment), Code & Go Robot Mouse; 3-5: Scratch, MakeCode (online blocks-based environments), micro:bit
Web-based	No

Gr. K–5 DLCS Topics	**K–2 *CS Essentials* Alignment**	**3–5 *CS Essentials* Alignment**
CAS.a. Safety & Security	X	X
CAS.b. Ethics & Laws	X	X
CAS.c. Interpersonal & Societal Impact	X	X
DTC.a. Digital Tools	*Partial*	*Partial*
DTC.b. Collaboration & Communication	*Partial*	✓
DTC.c. Research	X	X
CS.a. Computing Devices	*Partial*	X
CS.b. Human & Computer Partnerships	✓	✓
CS.c. Networks	X	X
CS.d. Services	NA	X
CT.a. Abstraction	X	✓
CT.b. Algorithms	✓	✓
CT.c. Data	X	*Partial*
CT.d. Programming & Development	✓	✓
CT.e. Modeling & Simulation	✓	*Partial*

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

COMPUTER SCIENCE ESSENTIALS

Overview

CS Essentials includes 5 units, one for each grade from 1st through 5th, designed as a supplement to the *Engineering Essentials*™ curriculum from EiE. These units can, however, be used separate from the engineering curriculum. Each unit begins with a story of a child engaged in an engineering challenge, providing the original storybook from EiE and a related story epilogue to contextualize the CS unit. All five units include the same Preparation Lesson, "What is a Computer?," which would only be taught once if the entire curriculum is used. An optional math extension lesson is also provided in each unit. Each lesson is structured with a teacher preparation section, lesson introduction, student activity, and reflection. Pedagogical strategies sequenced across each unit include engaging with the story, an unplugged coding activity, then a technology-based coding and design activity, and an application with student choice. Assessments include a unit pre-post assessment, checks for understanding for each lesson, and suggested reflection questions on the lesson guiding question, student experience, and EiE Habits of Mind (Feel, Think, Do). Sample programming solutions are also provided. Each lesson provides some content background for the teacher, and teacher tips in the lesson materials on instructional strategies, differentiation, and EL learner strategies. All student materials are provided in English and Spanish. The key strategy to attend to cultural responsiveness are the anchoring storybooks, which highlight diverse characters engaged in inquiry and problem solving; associated character cards are also provided.

Technology requirements

- For Grade 1: Code & Go Robot Mouse; For Grade 3: micro:bit. Recommend 1 per student pair.
- Tablets or Chromebooks; recommend one per student or 1 per student pair.
- ScratchJr can be used on either iOS, Android devices, or a Chromebook. It can be downloaded onto tablets and used offline.
- Scratch and MakeCode can be accessed online. A version of Scratch can be downloaded.

Professional development offered

- Virtual online PD for up to 25 participants.
- Full day or multi-day workshops, on site or at Museum of Science, Boston, for 10-25 participants.

Costs

- 1-year digital subscription to the curriculum is $99
- Materials Kits are available for $149 per unit
- Individual Code & Go Robot Mice are approximately $30 each
- Individual micro:bits are approximately $15 each.
- ScratchJr, Scratch, and MakeCode are free to use.
- Full-day on-site professional workshop for 25 participants is $4000
- Three-part webinar series (90-120 minutes per part) for up to 40 participants is $2500
- Asynchronous professional development module (8-10 hours of on-demand learning) is $125/educator

COMPUTER SCIENCE FUNDAMENTALS (K–2; 3–5)*

Code.org
code.org/educate/curriculum/csf

An introductory series of courses designed to engage student problem solving through coding.

Nature of curriculum	Stand-alone
Format and time	6 courses (1 per grade, 12–21 hours each)
Key classroom routines	Unplugged activities, coding puzzles, projects, journals
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	Limited
Teacher content knowledge support	Yes
Cultural responsiveness	Limited
Programming platform or language	Blockly (online blocks-based environment)
Web–based	Yes

Gr. K–5 DLCS Topics	**K–2 *CS Fundamentals* *A–C* Alignment**	**3–5 *CS Fundamentals* *D–F* Alignment**
CAS.a. Safety & Security	*Partial*	*Partial*
CAS.b. Ethics & Laws	X	*Partial*
CAS.c. Interpersonal & Societal Impact	X	X
DTC.a. Digital Tools	*Partial*	*Partial*
DTC.b. Collaboration & Communication	X	✓
DTC.c. Research	X	X
CS.a. Computing Devices	X	X
CS.b. Human & Computer Partnerships	X	X
CS.c. Networks	X	X
CS.d. Services	NA	X
CT.a. Abstraction	✓	✓
CT.b. Algorithms	✓	✓
CT.c. Data	X	*Partial*
CT.d. Programming & Development	✓	✓
CT.e. Modeling & Simulation	X	✓

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

COMPUTER SCIENCE FUNDAMENTALS

Overview

CS Fundamentals includes six sequenced courses, from A to F, designed to be implemented across grades K–5. The grade and age designations are, however, flexible. Each course contains 18–20 lessons of 25–45 minutes each. The curriculum engages students in coding, fundamental computer science concepts, and basics of digital citizenship. Students engage in unplugged activities, coding tasks, and interactive games or stories in Blockly. All lessons have an unplugged and a bridging activity that helps the student connect the hands-on activity to coding. All unplugged activities have a teacher tutorial video to explain how to do the activity and the concept. Many have a video showing instruction of the lesson as an example. Some language supports are provided via pre-read and read-aloud strategies; A and B courses are designed for pre-readers and C–F are for readers. The platform can be changed to numerous different languages, but there is not explicit English language learners supports. An example of each solution is provided to the teacher to help the students with the puzzles. Programming options allow two or three students to work together on the same computer with one student as the "driver" and another as the "navigator." There are assessments for all unplugged activities and the completion of the puzzles can be used as an assessment. Lesson wrap-ups include discussions and journaling prompts. The curriculum touches upon multiple perspectives or bias in relation to social impacts of media use. A teacher interface allows for deciding which assessments and projects students will engage in, and a dashboard shows the progress of student work.

Technology requirements

- Laptop, Chromebook, or mobile device with at least a 1024x728 px screen size; at least one device per two students is recommended, with one-to-one being ideal.
- Internet connectivity (at least a 15 MBit/sec) and up-to-date browser.

Professional development offered

- Seven-hour *CS Fundamentals* and six-hour Deep Dive workshops are available, led by a certified Code.org K–5 facilitator.
- Schools and districts can engage a Code.org facilitator for dedicated workshops.
- Self-paced online course for teachers who wish to implement CSF curriculum.
- A large online community includes forums for CSF courses and technical support.

Costs

- The curriculum and software are free.
- All professional development opportunities are currently available at no cost to educators, based on grants and contributions secured by Code.org.

COMPUTING WITH MINECRAFT (3–5)

Minecraft Education
education.minecraft.net/class-resources/computer-science-subject-kit

Students learn computer science skills through designing a Minecraft city.

Nature of curriculum	Stand-alone
Format and time	Course (5 units, 20 hours of lessons)
Key classroom routines	Introduction, Coding Activity, Knowledge Check, Wrap Up
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	No
Cultural responsiveness	Limited
Programming platform or language	Blockly (online blocks-based environment)
Web-based	No

Gr. 3–5 DLCS Topics	3–5 *Computing with Minecraft* Alignment
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	*Partial*
DTC.b. Collaboration & Communication	✓
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	*Partial*
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	✓

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

COMPUTING WITH MINECRAFT

Overview

Computing with Minecraft teaches computational thinking and computer science skills through designing a Minecraft city. The course is designed for beginning coders to familiarize them with algorithms, event handlers, coordinates, absolute world positions, relative player positions, loops, creating and merging multiple lines of coding, and recognizing that there are many different ways to write code with the same results. Organized into 5 units designed for about 20 hours of instruction, each lesson includes an introduction, lead coding activity, knowledge check, wrap up, unit extensions, learning goals, and assessments. The five units are Agency, City Planner, A Zoo, and Wind Power. The students will learn to plan a city, create a small park and fountain for their city, create a zoo, and then create a wind farm including animated turbines and an electrical resource to power the city. This theme allows students to connect their learning to local community needs and concerns. The first unit has the students become comfortable with the platform, then students build specific components of the city. All units have an educator guide, unit presentation (with PowerPoint slides), MakeCode files, and Minecraft world files. Assessments consist of learning assessment questions that connect to each lesson, Warm Up prompts to reinforce concepts and link prior knowledge, and knowledge checks to assess students learning, with a particular focus on coding skills. Each unit includes an end project and has four extension activities which can be used to challenge students who need more opportunities. In the last unit students create a model of a wind farm and animate it to simulate a real wind farm. An updated version of this course will be available soon, with additional computer science content, digital fluency, and culturally responsive supports.

Technology requirements

- *Minecraft: Education Edition* can operate on Windows, MacOS, iOS, and ChromeOS. Check system requirements for minimum specifications (e.g., educommunity.minecraft.net/hc/en-us/articles/360047556591-Supported-platforms-for-Minecraft-Education-Edition).

Professional development offered

- There is an 11-hour self-paced introduction to Minecraft: Education Edition (education.microsoft.com/en-us/learningPath/3eede2ae).

Costs

- *Minecraft: Education Edition* is licensed via yearly subscriptions. Individual student license is $5, although various volume licensing options are available.
- The self-paced PD is free.

CREATIVE COMPUTING CURRICULUM (3–5)

Creative Computing Lab at Harvard University
creativecomputing.gse.harvard.edu/guide/

A computational thinking curriculum designed to promote student creativity and agency.

Nature of curriculum	Stand-alone or Integrated
Format and time	Course (7 units; ~4–8 hours per unit)
Key classroom routines	Creating, personalizing, sharing, reflecting
Assessments, rubrics, examples	Limited
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	No
Programming platform or language	Scratch (online blocks-based environment)
Web-based	Yes

Gr. 3–5 DLCS Topics	3–5 *Creative Computing* Alignment
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	X
DTC.b. Collaboration & Communication	✓
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	*Partial*
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

CREATIVE COMPUTING CURRICULUM

Overview

The *Creative Computing Curriculum* engages students in designing animations, stories, and games while learning computational thinking concepts. An emphasis is placed on reflection, design journals, personalization, and sharing projects to promote brainstorming, documenting, and sharing ideas and work. Reflection Prompts form the key assessment strategy. Brief teacher notes provide instructional tips and support for implementation. Differentiation opportunities are provided mainly through ability to personalize projects. The curriculum can be used in its entirety as a semester-long computing course, or can be selectively distributed across middle school programming (note that this curriculum is also a viable option for middle school grades, although it is not recommended to be used at both elementary and middle school in any one district). The curriculum has been translated by teachers into ten different languages.

Technology requirements

- Requires computers or tablets with speaker or headphones for the computer- based design activities; Internet connectivity; up-to-date browser to connect to Scratch online. A downloadable version of Scratch is available.
- Computers equipped with microphones and webcams are recommended.

Professional development offered

- ScratchEd Meetups are participatory opportunities for teachers to have hands-on experience with Scratch, offered in cities across the country.
- The ScratchEd Online Community Archive is a venue for educators to access discussions, stories, and resources for teaching with Scratch.
- The Teaching with Scratch Facebook group is an active forum to share ideas, questions, and resources related to teaching with Scratch.

Costs

- The curriculum and Scratch software are free.
- Available professional development options are also free.

CS FIRST (3–5)

Google
csfirst.withgoogle.com/s/en/home

An introductory computer science course designed for easy entry for teachers and students alike.

Nature of curriculum	Stand-alone
Format and time	Modules (8 sessions, 1–1.5 hours each)
Key classroom routines	Introduction, create projects, wrap-up, reflection
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	Limited
Teacher content knowledge support	Yes
Cultural responsiveness	Yes
Programming platform or language	Scratch (online blocks-based environment)
Web-based	Yes

Gr. 3–5 DLCS Topics	**3–5 *CS First* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	X
DTC.b. Collaboration & Communication	X
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	*Partial*

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

CS First

Overview

CS First engages students in storytelling, music and sound, friends, fashion, sports, gaming, and art while teaching computer science concepts. Facilitators need no prior experience teaching computer science to begin using the video content to teach coding basics with Scratch. Each module includes several one-hour activities and multi-day activities that align to student interests, such as Storytelling, Game Design, and Music and Sound. All lessons are self-paced, include tutorial videos for students, and support application of coding concepts learned as they create a Scratch project. There is a new CS unplugged lesson that focuses on encoding and sending secret messages (abstraction). The unplugged lesson highlights diversity in related career descriptions. After the initial instructional videos, students or educators can choose optional add-on video lessons to extend their learning and individualize each project. The lessons provide discussion questions at the end of each activity along with prompts for teachers to support the students. Assessments include student work examples, reflections, student check ins, and answer keys. Teacher content knowledge supports are provided through introductory videos. Occasional teacher tips for EL supports are also provided. Modules are offered at differing levels so a program can provide for groups of students at different levels based on their needs, however, within individual modules there is limited support for student differentiation. It is recommended that a school implement at least one beginner, intermediate, and advance module to address the depth of all aligned standards. These modules can be spread out over three years, or taught as a dedicated course. The curriculum is also sometimes used at middle school.

Technology requirements

- Requires computers or tablets with speaker or headphones for the computer-based designed activities; devices can be shared by students, if desired.
- Internet connectivity and an up-to-date browser to connect to Scratch online is recommended. *CS First* can be used with limited bandwidth by downloading videos from the *CS First* website and through the Scratch Offline Editor.

Professional development offered

- Getting Started with *CS First* and *CS First* Subject Integration workshops are offered by professional development providers in virtual and in-person formats.
- Self-paced resources to learn the curriculum.

Costs

- The *CS First* curriculum and Scratch software are free. This includes classroom kits that are shipped to educators.
- Professional development may have associated costs.

DATA HANDLING WITH MICRO:BIT (3–5)

micro:bit
microbit.org/lessons/data-handling-unit-summary/

A unit to build data handling skills through the use of a micro:bit.

Nature of curriculum	Stand-alone
Format and time	Unit (5 lessons, approximately 60 minutes each)
Key classroom routines	Pair work, small group, whole class, unplugged and plugged activities
Assessments, rubrics, examples	Yes
Differentiation supports	Yes
English Learner supports	No
Teacher content knowledge support	Yes
Cultural responsiveness	No
Programming platform or language	MakeCode (online blocks-based environment); micro:bits
Web-based	Yes

Gr. 3–5 DLCS Topics	**3–5 *Data Handling* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	X
DTC.b. Collaboration & Communication	X
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	*Partial*
CT.b. Algorithms	✓
CT.c. Data	✓
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

DATA HANDLING WITH MICRO:BIT

Overview

Data Handling, one of a variety of available micro:bit units, supports students' understanding of data and how to collect and manipulate data to answer a question. The unit is geared towards fourth and fifth grade, and uses the Microsoft MakeCode platform to program micro:bits. Each lesson explores the micro:bit sensors and how to write programs using them to collect or act on data. Lessons include an introduction, learning objectives, materials, plugged and unplugged activities, and extension ideas. Differentiation opportunities are provided through stretch and challenge exercises. All of the lessons provide informal and formal assessment, including through their programming, class discussions, and during the tinkering process. Instructional structures include pair work, small group, and whole class learning opportunities. Supporting teacher materials include sample Hex files and slide decks. Teacher tips about content concepts are provided, including in the overview section.

Technology requirements

- Access to the MakeCode platform via computer, laptop, or tablet with an internet connection, modern browser, and USB port.
- micro:bits, recommended 1 per pair of students. Note that there is now a version two micro:bit available.

Professional development offered

- None specific to this unit.

Costs

- The curriculum is free.
- Individual micro:bits are approximately $15 each; consider mini-kits that include battery holder, USB cable, and such for approximately $20 each.

DIGITAL CITIZENSHIP (K–2; 3–5)

Common Sense Education
commonsense.org/education/digital-citizenship

Help students take ownership of their digital lives.

Nature of curriculum	Stand-alone or Integrated
Format and time	3 modules (1 per grade; ~6 hours each)
Key classroom routines	Warm-up; Analyze, Apply or Create; Wrap-up
Assessments, rubrics, examples	Yes
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	Limited
Programming platform or language	NA
Web-based	Yes

Gr. K–5 DLCS Topics	K–2 *Digital Citizenship* Alignment	3–5 *Digital Citizenship* Alignment
CAS.a. Safety & Security	✓	✓
CAS.b. Ethics & Laws	✓	*Partial*
CAS.c. Interpersonal & Societal Impact	X	*Partial*
DTC.a. Digital Tools	X	X
DTC.b. Collaboration & Communication	X	X
DTC.c. Research	X	X
CS.a. Computing Devices	X	X
CS.b. Human & Computer Partnerships	X	X
CS.c. Networks	X	X
CS.d. Services	NA	X
CT.a. Abstraction	X	X
CT.b. Algorithms	X	X
CT.c. Data	X	X
CT.d. Programming & Development	X	X
CT.e. Modeling & Simulation	X	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

DIGITAL CITIZENSHIP

Overview

The goal of the *Digital Citizenship* curriculum is to develop digital literacy skills necessary for effective engagement and student safety in a connected world. There are three to seven modules for each grade, K–5, which can be taught on their own or integrated into another academic subject. The modules include many unplugged activities and options for online games. Each lesson provides instructional sequences supported by slides, videos, and handouts to support student activities. Assessments with keys are provided. The curriculum touches upon multiple perspectives or bias in relation to social impacts of media use. Some teacher content knowledge support is provided through brief Teacher Notes. The curriculum is also available in Spanish. There is one unit for 5th grade that focuses on gender biases and stereotypes.

Technology requirements

- Desktop or laptop with basic internet access to view lesson slides, videos and online games.
- Requires speakers or headphones.

Professional development offered

- Recorded webinar for overview of teaching the curriculum; monthly webinars and advice postings available.

Costs

- All curriculum and educator resources are free.

ELEMENTARY COMPUTING FOR ALL (3–5)

University of California Irving
elementarycomputingforall.org

A Scratch-based curriculum integrating language support and STEM identity.

Nature of curriculum	Stand-alone
Format and time	Course (8 units)
Key classroom routines	Intro, Unplugged activity, Memorable Role Model, Explore, TIPP and SEE, Plan, Build, Reflection
Assessments, rubrics, examples	Yes
Differentiation supports	No
English Learner supports	Yes
Teacher content knowledge support	Limited
Cultural responsiveness	Yes
Programming platform or language	Scratch (online blocks-based environment)
Web-based	Yes

Gr. 3–5 DLCS Topics	***Elementary Computing for ALL* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	*Partial*
DTC.c. Research	X
CS.a. Computing Devices	*Partial*
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	X
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

ELEMENTARY COMPUTING FOR ALL

Overview

The *Elementary Computing for ALL* curriculum uses Scratch to engage students in learning computational thinking and coding basics while developing language and STEM identity. There are 5 units geared towards students in 4th grade: CS and Scratch, Sequence, Events, Loops, and a Culminating Project. Units for 5th grade include: Animation, Loops with Conditions, and Parallelism and Synchronization. All of the lessons use a similar structure: an introduction to concepts, an unplugged activity, Memorable Role Models, Exploring a CS concept, TIPP & SEE strategies, Plan, Build, Reflection Journal, and Classmate Comments. There is a student workbook that aligns with the lessons and provides instructions, activities, resources, tutorials, assessments and reflections on activities throughout the curriculum. Assessment support includes self-assessments, rubrics, and share outs.

An emphasis is placed on understanding vocabulary along with different activities that support student understanding of vocabulary. Many new concepts or words are supported with images and throughout the student handbook graphic organizers and visuals support understanding of the coding. Along with visuals and activities there are many opportunities for students to engage in discourse. Units 2-5 have a final project. The Unit 2 project "All About Me" which helps to bring student's identity into their project. It asks the students to express who they are through their poem animation. Each unit talks about a role model that had to persevere in their careers. All of the images, role models, and examples represent different ethnicities and genders. They offer numerous suggestions on how students can share their work whether it is in the inside outside conversation circle, gallery walk, class comments etc. Throughout the curriculum there are notations and resources to support virtual learning, and support for teachers to restructure lessons in different ways.

Technology requirements

- Requires computers or tablets with speaker or headphones for the computer- based design activities; Internet connectivity; up-to-date browser to connect to Scratch online. A downloadable version of Scratch is available.

Professional development offered

- Professional development events are held periodically.

Costs

- The curriculum and Scratch are free.

GROWING WITH KIBO (K–2)*

KinderLab
kinderlabrobotics.com/teacher-materials/

Robotics programming for young children.

Nature of curriculum	Stand-alone
Format and time	Course (60 hours; 20 hours for each of 3 levels)
Key classroom routines	Inspire, connect, engage, and reflect
Assessments, rubrics, examples	Yes
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	Yes
Cultural responsiveness	Yes
Programming platform or language	KIBO robot
Web-based	No

Gr. K–2 DLCS Topics	**K–2 *Growing With KIBO* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	✓
DTC.c. Research	*Partial*
CS.a. Computing Devices	✓
CS.b. Human & Computer Partnerships	✓
CS.c. Networks	X
CS.d. Services	NA
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	✓
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	✓

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

GROWING WITH KIBO

Overview

Growing with KIBO is a robotics curriculum geared to PreK–2 and designed to be stand-alone, but with the potential for integration. It engages students in learning computational thinking principles and the engineering design process through hands-on and programming activities. The curriculum is centered on the KIBO platform and requires some materials; a list of possible recycled materials that may be needed is provided. Programming the KIBO does not require computers, laptops, or apps; the experience is screen-free. Activity cards are available to support the youngest students in station-based activities. The curriculum is organized into three 20-hour sections, including a novice, intermediate, and advanced set of lessons that can be used in any grade level based on students' prior knowledge. Each lesson is designed for a one-hour block with a common structure of inspire, connect, engage, and reflect. Numerous opportunities to assess student progress are provided and include student portfolios, workbook, and student interviews. All sets have extensions to continue learning. Each lesson includes tips for the teacher addressing equity issues, support, implementation (including ways to create "roles" for students in activities), classroom structure, and classroom management. A theme of Community building engages students through a variety of activities that allow them to make connections to the wider community, such as by to share family routines and cultural traditions, inviting family and community members in to participate in project showcases, and decorating the KIBO to represent themselves.

Technology requirements

- KIBO Robot, recommended to have one for each two to four students. No laptops or tablets are required.

Professional development offered

- Regular web-based training workshops.
- On-site workshops tailored to school needs and train-the-trainer workshops to build trainers in the district are available.

Costs

- *Growing with KIBO* curriculum $60.
- A KIBO 18 Kit, with one robot serving 2–4 students, is $470.
- There are various classroom packages available. A KIBO 18 Full Classroom Package combines 10 robots, curriculum materials and workbooks, and professional development, for $5,000.
- A variety of recommended books would need to be purchased. Student workbooks are an extra cost if not purchased as part of a classroom package.
- Web-based training workshops are $100; tailored workshop costs vary.

INTRO TO COMPUTER SCIENCE (K–2)

codeSpark
codespark.com/educators

Coding puzzles for young children.

Nature of curriculum	Stand-alone
Format and time	Modules (10 topics, 1.5 hours each)
Key classroom routines	Engage, explore, enrich, reflect (unplugged and coding activities)
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	Yes
Teacher content knowledge support	Limited
Cultural responsiveness	No
Programming platform or language	Codespark
Web-based	Optional

Gr. K–2 DLCS Topics	**K–2 *Intro to CS* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	*Partial*
DTC.a. Digital Tools	*Partial*
DTC.b. Collaboration & Communication	X
DTC.c. Research	X
CS.a. Computing Devices	*Partial*
CS.b. Human & Computer Partnerships	✓
CS.c. Networks	✓
CS.d. Services	NA
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

INTRO TO COMPUTER SCIENCE

Overview

The codeSpark *Intro to Computer Science* curriculum is broken into 10 computer science topics. Each topic includes three lessons focused on engage, explore and enrich, taking about 30 minutes each. Engage lessons focus on activating prior knowledge, use of different media or unplugged activities, and making connections. Explore lessons use the app and are mainly coding lessons with guided practice and independent practice opportunities. Enrich activities provide some limited differentiated activities or extensions. The final topic focuses on applying coding skills. There is a strong emphasis on turn and talk, pair work, vocabulary, and visuals, and some sentence starters that provide EL support. Each lesson provides a PowerPoint for teachers to guide them in what to say to students, discussion questions, and what to look for in student work. There is a teacher dashboard to help monitor the progress of the students. Assessments include checks for understanding, reflections, and a culminating skills assessment. Students can code on an app or web-based version of the codeSpark application.

Technology requirements

- A desktop, laptop, tablet, or iPad with Internet connectivity and up-to-date browser.

Professional development offered

- Activity-based online course, with five modules of 60 – 90 minutes each.

Costs

- The curriculum is free.
- The professional development workshop is $119 per teacher.

KEYBOARDING WITHOUT TEARS (K–2; 3–5)

Learning Without Tears
lwtears.com/kwt

Develop keyboarding skills and digital literacy a few minutes a day.

Nature of curriculum	Stand-alone or Integrated
Format and time	Course (36 weeks, 30–45 min per week)
Key classroom routines	Daily 5–10 minutes online
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	Limited
Teacher content knowledge support	Limited
Cultural responsiveness	Limited
Programming platform or language	NA
Web-based	Yes

Gr. K–5 DLCS Topics	**K–2 *Keyboarding Without Tears* Alignment**	**3–5 *Keyboarding Without Tears* Alignment**
CAS.a. Safety & Security	✓	✓
CAS.b. Ethics & Laws	✓	*Partial*
CAS.c. Interpersonal & Societal Impact	✓	*Partial*
DTC.a. Digital Tools	*Partial*	*Partial*
DTC.b. Collaboration & Communication	X	X
DTC.c. Research	X	*Partial*
CS.a. Computing Devices	✓	X
CS.b. Human & Computer Partnerships	*Partial*	X
CS.c. Networks	✓	X
CS.d. Services	NA	X
CT.a. Abstraction	*Partial*	X
CT.b. Algorithms	X	X
CT.c. Data	X	X
CT.d. Programming & Development	X	X
CT.e. Modeling & Simulation	X	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

KEYBOARDING WITHOUT TEARS

Overview

Keyboarding Without Tears is a 36-week curriculum with a primary focus on teaching students touch typing and keyboard navigation through topics such as phonics, sentence structure, math, vocabulary, and grammar skills. Each week includes a teacher-led digital citizenship and/or literacy lesson. The keyboarding program is self-paced with audio and visual instructions. Review and enrichment opportunities provide independent writing prompts, dictation activities, and the ability to add teacher-created activities. This curriculum is appropriate for center work and can be integrated into ELA. There is limited EL support through platform features. The application uses a color-coded, row-based methodology for the keyboard to support students of various levels. Assessments include speed and accuracy assessments and digital citizenship assessments, with progress reporting available via a teacher and administrator dashboard. There are no coding or programming components. The curriculum recently integrated Common Sense Media materials, which touches upon multiple perspectives or bias in relation to social impacts of media use. For K–2, the grade 2 curriculum is particularly important to address aligned K–2 standards; in 3–5 students must participate in all three years to address aligned 3–5 standards.

Technology requirements

- Windows, Chrome OS, or Mac OS computer, or iPad with iOS 11 or later.
- High-speed Internet connectivity and up-to-date browser.

Professional development offered

- Onsite professional development conducted by Learning Without Tears trainers can be customized to district implementation requirements.
- Additional virtual opportunities include, live virtual sessions, a virtual professional development hub, and virtual Q&A and coaching options.

Costs

- Curriculum subscription costs range between $3.50/student for a classroom to $2.85/student for a district-wide license.
- In-person professional development is $3,200/day ($6,000 for 2 days); virtual live sessions are $1,200/day or $600/half day, and $300/2-hour workshop; virtual PD Hub is between $250–$1,250 per year depending on options chosen.

KODABLE K–2

Kodable
kodable.com/schools-and-districts

Kodable for K–2 uses plugged and unplugged lessons to teach foundational logic skills.

Nature of curriculum	Stand-alone
Format and time	4 Units (60 minute lessons)
Key classroom routines	Unplugged, Plugged, Turn and Talk, Discussions, Independent, Partner work, Whole group
Assessments, rubrics, examples	Limited
Differentiation supports	Limited
English Learner supports	Limited
Teacher content knowledge support	Limited
Cultural responsiveness	No
Programming platform or language	Kodable
Web-based	Yes

Gr. K–2 DLCS Topics	K–2 *Kodable* Alignment
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	*Partial*
DTC.b. Collaboration & Communication	X
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	NA
CT.a. Abstraction	X
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

KODABLE K–2

Overview

Kodable K–2 includes four units: sequencing, conditions, loops, and functions. Each grade level continues the depth of CS content that they will learn. Kodable is a drag and drop platform in which students interact with characters called Foos; these characters at times can be designed by the students. Creative areas unlock for students when they complete a number of levels in the game. The creative areas are Maze Maker, Game Designer, and Fuzz Builder. When the students are doing puzzles there are step by step instructions with minimal amounts of text to support all learners. Any vocabulary uses grade level definitions along with bright images. For students who are colorblind there is a colorblind mode that can easily be turned on by the students.

Teachers can assign lessons that are appropriate for their students in the teacher dashboard. Students can sign in using a teacher code, password, or email. Lessons are scripted and include objectives, lesson materials, vocab, direction instructions, guided practice, independent practice, and informal assessment through completion of puzzles and some reflection. In the teacher dashboard it is possible to monitor where students are on the puzzles and to assign specific puzzles to students. There are unplugged activities embedded into almost all lessons. Kodable has a virtual learning component if there is a need to do some of the work remotely. There is a video library for teachers to use with the class, assign to students or to support their understanding of the concepts. Along with the videos, the teacher has access to try out all of the puzzles so that they can understand what the students are doing and how to best support them. If a student is stuck on a puzzle the program will give them clues to help them write the algorithm.

Technology requirements

- Ipad or Laptop

Professional development offered

- Professional development package consisting of: discovery phone call, personalized workshop course prep, one online beginner session focused on coding and *Kodable K–2* (1 hour), and one online intermediate session to discuss implementation plans (1 hour).

Costs

- Curriculum and game access is $1275 per year for a school, with extended curriculum materials available for an additional $750 per year.
- PD package is $1500 for up to 10 teachers.

LEARN TO CODE CURRICULUM (K–2; 3–5)

Wonder Workshop
education.makewonder.com/curriculum/learn-to-code

Computational thinking approach using Dash or Dot robots.

Nature of curriculum	Stand-alone or Integrated
Format and time	Course (6 modules, 7 to 14 hours each)
Key classroom routines	Warm-up, direct instruction, guided practice, independent practice, wrap-up
Assessments, rubrics, examples	Yes
Differentiation supports	Yes
English Learner supports	No
Teacher content knowledge support	Yes
Cultural responsiveness	No
Programming platform or language	Dash or Dot robots (and simulator), Blockly (online blocks-based environment)
Web-based	No

Gr. K–5 DLCS Topics	**K–2 *Learn to Code* Alignment**	**3–5 *Learn to Code* Alignment**
CAS.a. Safety & Security	X	X
CAS.b. Ethics & Laws	X	X
CAS.c. Interpersonal & Societal Impact	X	X
DTC.a. Digital Tools	✓	*Partial*
DTC.b. Collaboration & Communication	✓	✓
DTC.c. Research	X	X
CS.a. Computing Devices	X	X
CS.b. Human & Computer Partnerships	*Partial*	*Partial*
CS.c. Networks	X	X
CS.d. Services	NA	X
CT.a. Abstraction	✓	*Partial*
CT.b. Algorithms	✓	✓
CT.c. Data	X	X
CT.d. Programming & Development	✓	✓
CT.e. Modeling & Simulation	X	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

LEARN TO CODE CURRICULUM

Overview

The *Learn to Code Curriculum* is organized around six modules and six fundamental coding concepts, designed on Code.org's *CS Fundamentals* course structure for grades K–5. All the modules focus on 2 to 3 coding concepts. Within each module there are six lessons that are between 45 and 60 minutes each. Each lesson includes direct instruction, guided practice, independent practice, a wrap-up, and final assessment, framed around a design thinking problem and programming the Dash or Dot robot. Each assessment has a section with ideas on how to extend the lesson, suggestions for scaffolding and differentiating the assessment to supports a variety of learners. The K–2 and 3–5 alignments assume students complete the final digital presentation projects. There is an online teacher dashboard, Class Connect, to monitor student progress on puzzles, give the teacher answers to puzzles, and share what lesson connects with each puzzle, and identify what coding concept is being taught. This dashboard can also provide teachers with other ways to support student progress on the different computer science concepts and find integration suggestions. Additional handouts provide suggestions to differentiate reflections, checklists to support self-paced challenges, strategies for debugging, and a rubric to assess student progress. A virtual Dash simulator allows students to engage in robot programming when a device is not readily available. Any lessons listed with Dot can be used with Dash, but not the other way around; lessons or challenge cards that say Dash+Dot can be done with two Dash robots.

Technology requirements

- Mac iOS, Kindle, Chromebook, Android
- Dash robot, one for each group of three to four students.

Professional development offered

- *Introduction to Coding and Robotics* course with Dash robot.

Costs

- *Learn to Code* Challenge Cards and curriculum are $99.00.
- 6-pack of Dash Robots is $899.00.
- Various other classroom packages are on the website.
- Professional development course is $200 per teacher.

LITTLEBITS STEAM+ CODING (3–5)

Sphero
classroom.littlebits.com/curriculum/steam-kit-core

Quick hands-on STEAM learning embedded with coding concepts.

Nature of curriculum	Stand-alone
Format and time	Unit (10 lessons, 45–60 mi each)
Key classroom routines	Individual and small groups
Assessments, rubrics, examples	Limited
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	No
Cultural responsiveness	No
Programming platform or language	littleBits Fuse App (Blocks, JavaScript); littleBits
Web-based	Yes

Gr. 3–5 DLCS Topics	**3–5 *littleBits STEAM+ Coding* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	X
DTC.b. Collaboration & Communication	*Partial*
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	X
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

LITTLEBITS STEAM+ CODING

Overview

This review is based on the 10 lessons of the *STEAM+ Coding* unit and one lesson within *STEAM+ Core, Invent for Good*. The lessons can be done in small groups or individually and build on the littleBits invention cycle, create, play, remix, and share. The unit requires the littleBits STEAM+ Class Pack and the Fuse app. Each lesson consists of a description, essential question, lesson objectives, lesson guide and a concluding sharing activity. The lessons are written with explicit details on how to complete the projects. The very first activity is a discussion with the students about the relevant CS concept. The last few lessons of the unit, and the Invent for Good lesson, involve students using their knowledge to invent, such as creating an arcade to help save the world or creating something to help others. Assessment guidance is provided for projects. Additional units and lessons are available on how to integrate littleBits into other subjects, and how to use them with micro:bits.

Technology requirements

- STEAM+ Class Pack
- littleBits Fuse is a free web app. Fuse requires: Chrome browser version 81 or later; A computer with either Chromebook OS (81 or later), Android 9 or later, Windows 10 (version 1706) or later, or Mac OS 10.13 (High Sierra) or later.

Professional development offered

- On-Site Workshop: typically a one day (6-hour) interactive training for up to 25 instructors. Larger districts can have multiple days or multiple sessions.
- All-access Virtual PD: receive live, personalized one-on-one and/or group training tailored to the needs of up to 20 educators, scheduled as needed throughout a year.
- Self-Guided Course: Learn littleBits fundamentals and integration through five self-paced and hands-on sessions.

Costs

- The littleBits STEAM+ Class Pack Classroom set is $3500 (for up to 40 students).
- All-access virtual PD is $1500 per year for up to 20 educators from a school or district.
- On-site workshops are typically $3000, inclusive of travel expenses for the trainer.

MA STEM+C: INTEGRATED UNITS (K–2; 3–5)

EDC & Massachusetts Department of Elementary and Secondary Education
sites.google.com/site/stemcwithct/home

Integrate computational thinking with mathematics and science.

Nature of curriculum	Integrated
Format and time	K–2: 3 units (3–7 hr. each); 3–5: 2 units (5–6 hr. each)
Key classroom routines	Unplugged activities, simulations, performances
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	Yes
Cultural responsiveness	No
Programming platform or language	NA
Web-based	Yes

Gr. K–5 DLCS Topics	**K–2 *MA STEM+C: Integrated Units* Alignment**	**3–5 *MA STEM+C: Integrated Units* Alignment**
CAS.a. Safety & Security	X	X
CAS.b. Ethics & Laws	X	X
CAS.c. Interpersonal & Societal Impact	X	X
DTC.a. Digital Tools	X	X
DTC.b. Collaboration & Communication	X	X
DTC.c. Research	X	X
CS.a. Computing Devices	X	X
CS.b. Human & Computer Partnerships	X	X
CS.c. Networks	X	X
CS.d. Services	NA	X
CT.a. Abstraction	✓	✓
CT.b. Algorithms	✓	✓
CT.c. Data	✓	✓
CT.d. Programming & Development	✓	✓
CT.e. Modeling & Simulation	✓	✓

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

MA STEM+C: Integrated Units

Overview

The *MA STEM+C: Integrated Units* curriculum was designed by Massachusetts teachers with the intent to teach computational thinking with mathematics and science. For grades K–2, the grade 1 *Money Machines*, grade 1 *Light and Shadows*, and grade 2 *Measuring and Graphing* units address computation thinking topics quite well. For grades 3–5, the grade 3 *Populations and Habitat* unit, the grade 4 *Angles* unit, and the grade 5 *Plants Make Their Own Food* units work well together to address computational thinking. These units are only several of a set for each grade level, K–5. Given that they are designed around specific math or science topics, they can be used as a replacement unit in those subjects or as a supplement to existing units. Teacher content knowledge support is typically provided through articulation of common student conceptions and misconceptions, and overviews of instructional strategies and tips. Multiple formative assessments are typically provided, as well as a culminating performance assessment with scoring rubric. Some units provide occasional support for differentiation strategies. Additional units not in this review include grade 2: *Effects of Wind and Water*; grade 3: *Fractions, Build It Fix It*; grade 4: *Electrical Circuits, Weathering and Erosion*; and grade 5: *Number Fluency and Fractions, Water Cycle*.

Technology requirements

- Typically need computers of any sort with Internet connectivity and up-to-date browser.
- One K–2 unit uses a *Peep and the Big Wide World* interactive (online).
- The grade 3–5 units use online videos or activities (Concord Consortium) as well as the Annenberg Learner *Interactive Ecology Lab*.

Professional development offered

- A one-day professional development workshop on these units is held periodically.

Costs

- Units and online resources are free.

NYC CSforALL: COMPUTER SCIENCE UNITS (K–2; 3–5)*

New York City Department of Education
blueprint.cs4all.nyc/curriculum/

Explore important questions by creating visual media using Scratch.

Nature of curriculum	Stand-alone
Format and time	Units (K–2: 20 lessons, 45 min. each; 3–5: 20 lessons, 45 min. each)
Key classroom routines	Unplugged activities, coding, wrap up, journaling
Assessments, rubrics, examples	Yes
Differentiation supports	Yes
English Learner supports	Limited
Teacher content knowledge support	Yes
Cultural responsiveness	Yes
Programming platform or language	ScratchJr (K–2), Scratch (3–5) (app or online blocks-based environments)
Web-based	K–2: Optional; 3–5: Yes

Gr. K–5 DLCS Topics	**K–2 *Computing Through Time* or *Intro to CS* Alignment**	**3–5 *Build My City* or *Intro to Programming* Alignment**
CAS.a. Safety & Security	X	X
CAS.b. Ethics & Laws	X	X
CAS.c. Interpersonal & Societal Impact	✓	X
DTC.a. Digital Tools	✓	*Partial*
DTC.b. Collaboration & Communication	X	✓
DTC.c. Research	X	X
CS.a. Computing Devices	✓	*Partial*
CS.b. Human & Computer Partnerships	✓	X
CS.c. Networks	X	X
CS.d. Services	NA	X
CT.a. Abstraction	✓	✓
CT.b. Algorithms	✓	✓
CT.c. Data	✓	X
CT.d. Programming & Development	✓	✓
CT.e. Modeling & Simulation	X	*Partial*

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

NYC CSforALL: Computer Science Units

Overview
The original K–2 unit *Computing Through Time* and the 3–5 unit *Build My City* are reviewed below. A more recent versions of each, *Intro to CS* and *Intro to Programming* address similar standards just in different contexts, and are viable substitutions.

The K–2 *Computational Media Explorer: Computing Through Time* unit asks "How has technology helped make computers what they are today and what they may look like in the future?" This project-based unit is designed for grade 1 but can be used in K or 2 with adjustments. Lessons have slide decks and activity templates, and often photos or videos to support implementation. Many of the lessons are unplugged. Programming lessons are written for ScratchJr but teachers can choose the platform for programming. A final project can be adapted to different academic content. Assessments are included in each lesson, and exit tickets and a general rubric on collaboration, vocabulary, and connections provide for ongoing check-ins. Regular suggestions for differentiation and some suggestions for EL support are included.

The 3–5 *Computational Media Creator: Build My City* unit asks "How can we use computers to express ourselves?" The final project engages students in applying their learning to imagine, design, and program a new city. The curriculum is designed for grade 4 but can be modified for 3 or 5. The unit introduces computer science, algorithmic thinking, and creative programming with Scratch. The lessons are structured with a teacher demonstration, tutorials, student group work, student individual work, and checks for understanding. There are supports for differentiation, including mini lessons to support different learners. There is guidance on how to establish peer student groups based on needs, and how to implement the curriculum in different room structures such as rotating centers or a lab. Assessments are included, along with a general assessment rubric and formative assessments to check for understanding. The grade 3–5 alignment assumes that students do the final project path that includes modeling and simulation.

Both curricula are designed to engage the students in seeing themselves as computer scientists. Diverse images are provided and activities encourage students to consider who computer scientists are, and what computer science is used for.

Technology requirements

- For K–2, a tablet or iPad; for 3–5, a desktop or laptop with Internet connectivity.
- Software K–2: ScratchJr or other coding environment. Some examples: Playlab, ScratchJr and Foos for Pre-reader; ScratchJr can be used on either iOS, Android devices, or a Chromebook. It can be downloaded onto tablets and used offline.
- Software 3–5: Scratch (online). A downloadable version of Scratch is available.

Professional development offered

- None for Massachusetts educators at this time.

Costs

- All curriculum materials are free and accessible on a public Google drive.

PLTW LAUNCH: COMPUTER SCIENCE UNITS (K–2; 3–5)*

Project Lead the Way
pltw.org/our-programs/pltw-launch

Use a variety of programming apps in projects across elementary grades.

Nature of curriculum	Stand-alone
Format and time	Units (6 units, each about 10 hours)
Key classroom routines	Activity, Project, Problem-based learning
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	Yes
Cultural responsiveness	No
Programming platform or language	ScratchJr (K–2), Scratch (3), Tynker (4–5) (app or online blocks-based environments)
Web-based	K–2: Optional; 3–5: Yes

Gr. K–5 DLCS Topics	K–2 *PLTW Launch: CS units* Alignment	3–5 *PLTW Launch: CS units* Alignment
CAS.a. Safety & Security	X	X
CAS.b. Ethics & Laws	X	X
CAS.c. Interpersonal & Societal Impact	X	X
DTC.a. Digital Tools	✓	*Partial*
DTC.b. Collaboration & Communication	✓	*Partial*
DTC.c. Research	X	X
CS.a. Computing Devices	*Partial*	*Partial*
CS.b. Human & Computer Partnerships	*Partial*	X
CS.c. Networks	X	*Partial*
CS.d. Services	NA	X
CT.a. Abstraction	✓	✓
CT.b. Algorithms	✓	✓
CT.c. Data	*Partial*	*Partial*
CT.d. Programming & Development	✓	✓
CT.e. Modeling & Simulation	X	✓

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

PLTW LAUNCH: COMPUTER SCIENCE UNITS

Overview

PLTW Launch is a series of PreK–5 STEM-related units, with one DLCS-focused unit in each grade. The units use an activity-, project-, problem-based instructional approach, and each culminates in an application project. The computer science units for PreK–2 are: *Spatial Sense and Coding*, *Animals and Algorithms*, *Animated Storytelling*, and *Grids and Games*; and for 3–5: *Programming Patterns*, *Input/Output: Computer Systems*, and *Infection: Modeling and Simulation*. Lesson assessments as well as project rubrics and guides are provided. Teacher guides provide details for facilitation of activities, projects and problems and resources necessary for the learning experience. Additional resources help the teacher with concepts developed in the module, provide demonstrations for facilitating specific activities, projects, or problems, and call out guidance for addressing some aspects of Computing and Society, Digital Tools and Collaboration, and Computing Systems that are not explicit in student learning goals. Several accessibility features are provided for online materials. Each unit is available in Spanish.

Technology requirements

- Androids, Android-enabled Chromebooks, or iPad with internet connectivity and browser, at a recommended 4:1 student to device ratio.
- ScratchJr can be used on either iOS, Android devices, or a Chromebook. It can be downloaded onto tablets and used offline. Scratch can be accessed online; a downloadable version of Scratch is available.

Professional development offered

- Classroom Teacher Training (CTT) includes 16 hours of professional learning, over a two-day period or spread among multiple days, that may be completed in-person or online. All PLTW hosted trainings are led by PLTW Master Teachers. *PLTW Launch* Lead Teachers may lead CTT for in-school or district educators.
- Additional supports include email or phone support; an online community of teachers, *PLTW Launch* teachers and Master Teachers; a course specifically for *PLTW Launch* Lead teachers that build their capacity to support their *PLTW Launch* implementation; and opportunities to provide feedback to PLTW.

Costs

- PLTW materials kits are available for each unit, at a cost of $60–$220.
- Yearly fee for *PLTW Launch* is $950, which includes all *PLTW Launch* curriculum units and software. Funding and grants are available via the PLTW Grant program.
- CTT is $500 per teacher. *PLTW Launch* Lead Teacher Training costs $700 per teacher. District Transformation Training for up to 24 *PLTW Launch* classroom teachers is available at a cost of $9,500.

SFUSD CREATIVE COMPUTING K–2 CURRICULUM*

San Francisco Unified School District
sites.google.com/sfusd.edu/k-2cs/

Introduction to computer science as a creative, collaborative, and engaging discipline.

Nature of curriculum	Stand-alone
Format and time	3 Courses (20 lessons each grade)
Key classroom routines	Individual, Partner, small group, whole group
Assessments, rubrics, examples	Yes
Differentiation supports	Yes
English Learner supports	No
Teacher content knowledge support	No
Cultural responsiveness	Limited
Programming platform or language	ScratchJr (app or online blocks-based environments), BeeBots
Web-based	No

Gr. K–2 DLCS Topics	**K–2 *SFUSD Creative Computing* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	*Partial*
CAS.c. Interpersonal & Societal Impact	*Partial*
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	✓
DTC.c. Research	X
CS.a. Computing Devices	✓
CS.b. Human & Computer Partnerships	*Partial*
CS.c. Networks	X
CS.d. Services	NA
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

SFUSD Creative Computing K–2 Curriculum

Overview

The San Francisco Unified School District (SFUSD) *Creative Computing K–2 Curriculum* consists of 3 sequential grade level curricula for grades K, 1, and 2. Each grade level consists of 20 lessons (each lesson about 30–40 minutes) organized into 3 units: Unit 1 is unplugged, Unit 2 uses BeeBots, and Unit 3 uses Scratch Jr. Students learn about computer science principles and computing device skills through unplugged lessons, then apply their knowledge to robotics and visual block-based programming. Each lesson consists of warm up, main activity, debrief, and extensions. Through the lessons there are individual, small group, partner work and whole class activities. Extension activities, vocabulary, and lesson notes are provided to support the lesson. When students use Scratch Jr, the curriculum provides 'mild,' 'medium,' and 'spicy' levels to support all learners while also giving students choice; each level represents a set of criteria that get progressively harder and give students the opportunity to strive for what they feel they can accomplish. Templates for different writing activities are included to support all learners. The teacher notes occasionally encourage assessing student talk to see whether certain students dominate discussion, and whether there is a balance of female, male gender-fluid, mono and multilingual student contributions. The Grade 1 curriculum draws upon the *Hello Ruby: Adventures in Coding* book and grade 2 uses *Hello Ruby: Journey Inside a Computer*. The grades K and 1 curricula provide remote learning versions of in-person activities. When students are working on Scratch they are given project guidelines. All lessons have a reviewing student work section to support teacher assessment of the lesson.

Technology requirements

- BeeBots (or BlueBots; the curriculum does not require or use connectivity functions). The optimal group size is 4 students or fewer per Bee-Bot.
- ScratchJr can be used on either iOS, Android devices, or a Chromebook. It can be downloaded onto tablets and used offline.
- Tablet: 1 per pair of students recommended.

Professional development offered

- Not available at this time.

Costs

- Curriculum and Scratch Jr are free to access and use.
- A 6-pack of Beebots with docking station and command cards is approximately $600.
- Access to *Hello Ruby: Adventures in Coding* and *Hello Ruby: Journey Inside a Computer* for whole-class reading, and some ancillary materials for group work are required.

SPHERO COMPUTER SCIENCE FOUNDATIONS COURSE 1 (3–5)

Sphero
edu.sphero.com/cs-foundations

Computer science foundations through robotics.

Nature of curriculum	Stand-alone
Format and time	Course (24 lessons, 45–60 min each)
Key classroom routines	Exploration, Skill Building, Challenge, Sharing, Reflection, and Extended Challenge
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	No
Programming platform or language	Sphero Edu Draw Canvas, Sphero Edu Block Canvas; Sphero BOLT
Web-based	Yes

Gr. 3–5 DLCS Topics	**3–5 *CS Foundations 1* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	*Partial*
DTC.b. Collaboration & Communication	*Partial*
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	*Partial*
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	*Partial*

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

SPHERO COMPUTER SCIENCE FOUNDATIONS COURSE 1

Overview

Computer Science Foundations Course 1 teaches CS through the use of the Sphero BOLT robots. Course 1 contains 24 scaffolded lessons across three thematic areas: "A" in Steam, Shapes and Numbers, and Nature. The Nature unit is the only unit that addresses modeling and simulation. Students engage with CS through drawing, beginner blocks, and intermediate blocks. The unit themes allow for integration of the lessons into other disciplines, such as art, science, and math, although there is relatively limited support for this integration in the materials. Lessons can be sequenced so that all beginning lessons from each unit are taught first, or so that lessons in each unit are taught in sequence. Each lesson includes an Exploration (5–10 min), Skills Building and Challenge (25–30 minutes), Sharing and Reflection (5–15 minutes), and Extended Challenge (up to 60 min). Instructional formats include whole group, pair programming, and small group lessons. Each lesson includes assessment questions and reflections of the content to guide an informal or formal assessment. The frequency of assessments can vary based on complexity of lessons; more complex lessons ask an assessment question after each new skill while some simpler lessons provide fewer assessment questions. A bank of links to videos and documents are provided to supplement the curriculum guide. CSF 1 includes a printed Educator Guide for step-by-step instructional support, which includes occasional tips are provided on how to support a student who is struggling or how to refocus learning to the task at hand. Sphero Edu's teacher platform allows educators to sync Google Classroom or Clever, assign lessons and track student progress.

Technology requirements

- Sphero BOLT (recommend 1 for each pair of students)
- Desktop app (Windows, MAC, Chrome OS), Mobile App (iOS, Android, Fire OS), or browser interface
- Check Sphero's System Compatibility and Robot Compatibility chart for specifics (support.sphero.com/article/3tzmrkc6lx-edu)

Professional development offered

- On-site workshop includes a one day (6 hour) interactive training for up to 25 instructors. Multiple day or concurrent sessions also available.
- Virtual PD: Yearly unlimited access for up to 20 educators (from across the district). Educators access live, personalized one-on-one and/or group training. A list of suggested sessions is provided, but requests are also considered.

Costs

- Sphero *Computer Science Foundations Course 1* curriculum: $300.
- Sphero BOLT Power Pack: $2650, which includes 15 BOLTs (class set) in a portable, charging case on wheels.
- 1-day on-site workshop: $3,000, inclusive of travel expenses for the trainer. All-access Virtual PD: $1500 per year.

SPIKE ESSENTIAL UNITS (K–2, 3–5)

LEGO
education.lego.com/en-us/lessons?products=SPIKE™+Essential

Introduce physical computing with STEAM.

Nature of curriculum	Stand-alone
Format and time	Units (2 in K-2; 3 in 3-5; 5-6 hrs per unit)
Key classroom routines	Engage, Explore, Explain, Elaborate, Evaluate
Assessments, rubrics, examples	Yes
Differentiation supports	Yes
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	No
Programming platform or language	LEGO Education SPIKE App (block-based), LEGO SPIKE Essential Set
Web-based	No

Gr. K–5 DLCS Topics	***SPIKE Essential (K-2)* Alignment**	***SPIKE Essential (3-5)* Alignment**
CAS.a. Safety & Security	X	X
CAS.b. Ethics & Laws	X	X
CAS.c. Interpersonal & Societal Impact	X	X
DTC.a. Digital Tools	X	X
DTC.b. Collaboration & Communication	X	X
DTC.c. Research	X	X
CS.a. Computing Devices	X	X
CS.b. Human & Computer Partnerships	X	X
CS.c. Networks	X	X
CS.d. Services	NA	X
CT.a. Abstraction	X	Partial
CT.b. Algorithms	Substantial	Substantial
CT.c. Data	X	X
CT.d. Programming & Development	Partial	Substantial
CT.e. Modeling & Simulation	X	X

SPIKE Essential Units

Overview

SPIKE Essential is designed to be a supplemental physical computing curriculum that integrates Coding and sensors through the engineering design process. SPIKE Essential for grades 1-2 consists of two units, and three units for grades 3-5, with an extension unit to explore FIRST LEGO League. The units are designed to be modular so they can be adapted into existing curriculum, with a progression from Great Adventures (grade 1), to Amazing Amusement Park (grade 2), Happy Traveler (grade 3), Crazy Carnival Games (grade 4), and finally Quirky Creations (grade 5). Every lesson is built on the 5E model, with a common structure of engage, explore, explain, elaborate, and evaluate. A lesson page for each activity includes objectives, key questions to ask throughout the lesson, a video explaining the lesson, differentiation strategies for simplifying and increasing the difficulty. Activities engage students in small group and whole group instruction. There are numerous assessment options that are given in the evaluate section: rubric, peer feedback, self-assessment, and oral checklist are typical examples. Every lesson has an extension activity. The app gives the students a story with a task and then step by step instructions on building the structure and programming it. The language can be changed in the app and the story can be read to the students. Coding program blocks for grades 1-2 are all pictures or numbers while for grades 3-5 coding programs include relevant words. Many of the lessons additionally use inspiration blocks for students to modify their program and continue to iterate their design. The last lesson in each unit is an applied activity where students create a product using programming and the lesson concepts. All units use the LEGO SPIKE Essential kit, with the option of also combining that with the BricQ Motion Essential kit in two additional challenge-based lessons in the Carnival Games unit.

Technology requirements

- LEGO Education SPIKE Essential Set can be used with Windows, MacOS, iPad, or Chromebook. Devices should have Bluetooth.
- LEGO Education SPIKE App is based on Scratch; available through a downloadable app.

Professional development offered

- A variety of professional development formats are available online through the LEGO PD platform, including self-guided, facilitated, and micro credentials.
- PD and coaching is also offered with Master Trainers in person or virtually. These are typically 6 hours for up to 20 participants.

Costs

- The curriculum materials and App are free to access.
- One LEGO SPIKE Essential set for two students is approximately $275.
- Self-paced PD via the PD platform is free.
- The online self-paced PD options are free. Master Trainer PD and coaching costs are $3000 for 6 hours up to 20 people.

SPRINGFIELD CS4ALL (K–2; 3–5)*

Springfield Public Schools
www.springfieldpublicschools.com/departments/OITA/c_sfor_a_l_l

Integrate DLCS and physical computing into core academics.

Nature of curriculum	Integrated
Format and time	4 Units per year
Key classroom routines	Crafting, Composing Meaning, Processing
Assessments, rubrics, examples	Yes
Differentiation supports	Yes
English Learner supports	Limited
Teacher content knowledge support	No
Cultural responsiveness	Limited
Programming platform or language	Scratch, Codespark and MakeCode (online blocks-based environments), BeeBots, Makey Makey, micro:bits, Hummingbird kits
Web-based	No

Gr. K–5 DLCS Topics	***Springfield CS4ALL (K–2)* Alignment**	***Springfield CS4ALL (3–5)* Alignment**
CAS.a. Safety & Security	*Partial*	*Partial*
CAS.b. Ethics & Laws	✓	*Partial*
CAS.c. Interpersonal & Societal Impact	*Partial*	*Partial*
DTC.a. Digital Tools	✓	*Partial*
DTC.b. Collaboration & Communication	✓	✓
DTC.c. Research	✓	*Partial*
CS.a. Computing Devices	X	X
CS.b. Human & Computer Partnerships	*Partial*	*Partial*
CS.c. Networks	X	X
CS.d. Services	NA	X
CT.a. Abstraction	✓	✓
CT.b. Algorithms	✓	✓
CT.c. Data	✓	X
CT.d. Programming & Development	✓	✓
CT.e. Modeling & Simulation	✓	✓

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

SPRINGFIELD CS4ALL

Overview

Springfield CS4ALL is designed to integrate DLCS into ELA, mathematics, science, and social studies. There are typically 4 units provided for each year K-5, with between 3 and 9 lessons in each unit. Each unit has an application or project students work toward, such as developing a poem, illustrating an animal habitat, visualizing word problems, or developing an ecosystem model. Lessons are structured to start with Crafting, often involving teacher instructions and student turn and talk; followed by Composing Meaning in which students do research, coding, and partner work; and close with a Processing activity, typically a short class reflection on the lesson. Each lesson includes student handouts, suggestions for differentiation and remote learning, as well as guidance for assessment, often with rubrics for final projects. Several differentiation suggestions are specific to EL student populations. Several of the units include books representing diverse authors and characters, one second grade unit is about cultural celebrations, and at least one other lesson briefly engages students in a discussion of social impacts of natural disasters.

Technology requirements

- Internet connectivity and up-to-date browser.
- Scratch online; a downloadable version of Scratch is available.
- Codespark
- Access to the MakeCode platform via computer, laptop, or tablet with an internet connection, modern browser, and USB port.
- Recommended 6 BeeBots per class
- Recommended Makey Makey per pair of students
- Recommended Micro:Bit per pair of students
- Recommended Hummingbird Bit per groups of 3 to 4 students

Professional development offered

- Not Available

Costs

- The curriculum is free to access.
- Scratch is free to use.
- A 6-pack of Beebots with docking station and command cards is approximately $600.
- Makey Makey class set $699.95
- Individual micro:bits are approximately $15 each; consider mini-kits that include battery holder, USB cable, and such for approximately $20 each.
- Hummingbird Bit kits, set of 8: $1450.

TECHNOKIDS: SELECTED PROJECTS (K–2; 3–5)*

TechnoKids
technokids.com

Learn common productivity and coding applications.

Nature of curriculum	Stand-alone or Integrated
Format and time	K–2: 4 units (projects); 3–5: 5 units (projects)
Key classroom routines	Skill-building, project development
Assessments, rubrics, examples	Yes
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	No
Cultural responsiveness	No
Programming platform or language	Google or Microsoft applications; ScratchJr (app or online blocks-based environment); Python
Web-based	Optional

Gr. K–5 DLCS Topics	**K–2 *TechnoKids: Selected Projects* Alignment**	**3–5 *TechnoKids: Selected Projects* Alignment**
CAS.a. Safety & Security	X	*Partial*
CAS.b. Ethics & Laws	X	*Partial*
CAS.c. Interpersonal & Societal Impact	X	X
DTC.a. Digital Tools	✓	✓
DTC.b. Collaboration & Communication	✓	✓
DTC.c. Research	*Partial*	✓
CS.a. Computing Devices	✓	X
CS.b. Human & Computer Partnerships	X	X
CS.c. Networks	X	*Partial*
CS.d. Services	NA	X
CT.a. Abstraction	✓	✓
CT.b. Algorithms	✓	✓
CT.c. Data	*Partial*	✓
CT.d. Programming & Development	✓	✓
CT.e. Modeling & Simulation	X	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

TechnoKids: Selected Projects

Overview

For K–2, four TechnoKids projects are recommended as set: TechnoMe, TechnoStart, TechnoStories, and TechnoWhiz. For 3–5, the following five projects are recommended as a set: TechnoCandy, TechnoInternet, TechnoPresenter, TechnoResearch, and TechnoTurtle. At each grade level, there are additional projects that may be added or swapped for one of the sets, but check for any adjustments to the standards alignment. Additionally, a project or two may be removed from the recommended sets; for example, removal of TechnoWhiz and TechnoTurtle would remove the computer science coding topics and the remainder would focus on digital literacy applications. (Note: TechnoTurtle uses the Python programming language, which is beyond the expectation of a graphical environment noted in the standards.)

Each project uses a different technology tool/software title to complete a project—most of the projects can be integrated in core academics. Each project includes a mix of unplugged activities, skill building digital activities, and a project. Supporting materials include a project guide, workbook and resource files, as well as student workbook files and skill builder exercises. Almost all the projects can be web-based if Google or Microsoft online applications are chosen; however, use of the Python IDLE in the 3–5 TechnoTurtle unit requires downloaded software. Assessment guidance is provided for assessing skill development and for assessing the final project, along with regular review questions and skill reviews.

Technology requirements

- A desktop or laptop with Internet connectivity and up-to-date browser.
- If choosing Microsoft version of the projects, will need subscription to PC-based or online applications.
- ScratchJr; latest version of Python IDLE. ScratchJr can be used on either iOS, Android devices, or a Chromebook. It can be downloaded onto tablets and used offline.

Professional development offered

- TechnoKids provides free curriculum support by phone or email.

Costs

- Individual TechnoKids projects are $40 per school, or grade-level sets include all projects at those grades (Primary Set, $145; Junior Set, $275), or all project titles can be purchased via a complete package ($595 per school). If multiple schools are participating, one school pays the full price and additional schools receive a 50% reduction for the same product.
- Access to and use of ScratchJr and Python IDLE are free.

GRADES 6 THROUGH 8

CONSIDERATIONS FOR SELECTING MIDDLE SCHOOL CURRICULA

The middle school curricula included in this section provide a range of options spanning broad coverage to more topical approaches. If a district does not have a systematic DLCS offering, or is looking to start anew, consider a curriculum that addresses the broadest topical span as a foundation. This ensures coherence across topics, and can streamline professional development and materials. A number of organizations included in this section also have elementary or high school offerings, providing for the possibility of pathways using curricula from the same organization.

The following options are suggested as a starting point given their relatively broad scope to provide a strong foundation for a comprehensive middle school DLCS program:

- *Computer Science Discoveries* (code.org)
- *Computer Science Applications JavaScript* (Ellipsis Education)

Any DLCS curricula at the middle school grades will have gaps relative to the full MA DLCS standards. The suggestions below provide some examples of how different curricula pairings might result in a relatively comprehensive offering.

- Use *Computer Science Discoveries* as a base and pair it with *Digital Citizenship*.
- Use *Computer Skills – Digital Savvy* as a base and pair it with a Computational Thinking-focused option, such as: *Bootstrap:Algebra*, *Project GUTS*, *Creative Computing Curriculum*, or *Computational Thinking Curriculum*.

If a district already has DLCS programming in place and is looking primarily to fill known gaps, then use the topical alignment chart (presented with each curriculum and provided in the Appendix) to identify which options best address particular topics.

There are other combinations of pairings possible, particularly if a district is already using resources not included in this guide.

BOOTSTRAP:ALGEBRA

Brown University
bootstrapworld.org/

Integrate computational thinking into an Algebra course.

Nature of curriculum	Stand-alone or Integrated with mathematics
Format and time	9 units (about 25 hours)
Key classroom routines	Teacher-led, workbook activities, programming activities, closing discussions
Assessments, rubrics, examples	Yes
Differentiation supports	No
English Learner supports	Yes
Teacher content knowledge support	Limited
Cultural responsiveness	Limited
Programming platform or language	Scheme or Pyret (online programming environment)
Web-based	Yes

Gr. 6–8 DLCS Topics	***Bootstrap:Algebra*** **Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	X
DTC.b. Collaboration & Communication	Partial
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

BOOTSTRAP:ALGEBRA

Overview

Bootstrap:Algebra is a coding module meant to be integrated into an existing Algebra course or computer science course. This may also be used at high school, depending on when Algebra is taught. Can be taught as a separate computational thinking course as well, in which case the algebra content is reinforcing content. Students program a video game using linear functions, piecewise functions, the Pythagorean theorem, inequalities, nested functions, and more. Three additional modules are available (including *Data Science*), which could be used at middle school but are more aligned with high school and are described there. The *Algebra, Data Science*, and *Reactive* modules can be combined to form a full stand-alone computer science course.

Key instructional routines include workbook activities, programming activities, and closing discussions. The curriculum provides assessment guidance for teachers for providing feedback on student programs. Optional quizzes can be developed online and played live in the classroom, given as practice, or assigned for homework. Brief teacher notes provide suggested teaching tips and an occasional content point; suggested English Learner strategies are specifically provided in a variety of lessons. The curriculum is available in Spanish. Purposeful choice of names used in the curriculum reflect a mix of different cultural backgrounds and non-binary language and examples are used throughout. The *Bootstrap:Algebra* module is available in the Scheme/Racket programming language (a derivative of LISP), as well as Pyret (a Python-like language that behaves like LISP). Both options reinforce concepts found in a standard math class.

Technology requirements

- One computer (desktop or laptop) per pair of students with Internet access, an up-to-date browser, and a Google account. All programming materials run in the browser.

Professional development offered

- On-demand workshops are available, typically three days, for 25–40 teachers, and are highly recommended for beginning teachers.
- Active discussion group, moderated by Bootstrap staff and master teachers, and video office hours where anyone can ask questions.

Costs

- Professional development workshops are offered regionally, or district-specific workshops can be negotiated. Software licenses are built into the cost of PD.

COMPUTATIONAL THINKING CURRICULUM

MIT App Inventor
appinventor.mit.edu/explore/teach

Learn computational thinking through mobile app development.

Nature of curriculum	Stand-alone
Format and time	Course (10 units, full-year)
Key classroom routines	Coding applications, student sharing
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	Limited
Programming platform or language	MIT App Inventor (online blocks-based environment)
Web-based	Yes

Gr. 6–8 DLCS Topics	***Computational Thinking Curriculum* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	X
DTC.b. Collaboration & Communication	X
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	*Partial*
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	*Partial*
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

COMPUTATIONAL THINKING CURRICULUM

Overview

The *Computational Thinking Curriculum* is designed to teach students to think computationally by decomposing problems, abstracting and modularizing, reusing and remixing existing solutions, and testing to arrive at a working solution. Students program mobile apps using the MIT App Inventor, although schools do not need access to mobile phones or tablets to use the software and complete programming projects. Several of the units engage students in their community, including programming for local community landmarks and a final project designed to help someone in their community. Assessments are provided across the unit, with the final project emphasizing student progress through both self-assessments and peer assessments. Periodic teacher notes provide instructional tips, background information, and support for implementation. Differentiation of student skill levels can be addressed through three levels of the student guides.

Technology requirements

- Desktop or laptop with reliable Internet connectivity and up-to-date browser.
- Mobile devices (can use an emulator). One device for every two students is recommended. Android and iOS devices are supported.

Professional development offered

- Community Forum where educators can share and discuss ideas, lesson plans, and activities with a wider community.
- MIT App Inventor provides periodic summer professional development workshops and coordinates with other organizations to offer professional development.

Costs

- Access to and use of the curriculum, software, and Community Forum are free.
- Professional-development costs vary.

COMPUTER SCIENCE APPLICATIONS JAVASCRIPT*

Ellipsis Education (formerly Codelicious)
ellipsiseducation.com/course/computer-science-applications-javascript/

Introduction to text-based programming using JavaScript.

Nature of curriculum	Stand-alone
Format and time	Course (up to 70 modules/lessons)
Key classroom routines	Group discussions, partner work, unplugged, plugged, centers
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	Yes
Teacher content knowledge support	Limited
Cultural responsiveness	No
Programming platform or language	JavaScript, Repl.it
Web-based	Yes

Gr. 6–8 DLCS Topics	***CS Applications JavaScript* Alignment**
CAS.a. Safety & Security	✓
CAS.b. Ethics & Laws	Partial
CAS.c. Interpersonal & Societal Impact	Partial
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	✓
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	✓
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	Partial
CT.c. Data	✓
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	Partial

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

COMPUTER SCIENCE APPLICATIONS JAVASCRIPT

Overview

Computer Science Applications JavaScript (grades 6–8) introduces students to text-based programming with JavaScript. It is recommended that students have prior computer science experience, such as *Computer Science Fundamentals* (see grades 3–5 review) or similar. Schools can select from among 70 modules based on their needs. Modules are customized to each school's timing requirements. Meaning the full year of curriculum can be delivered in 20min, 30min, 60min, or other as needed. The course integrates coding, unplugged activities, digital citizenship, and STEM career lessons. All lessons include objectives, lesson procedures, examples, materials, and assessments. Both summative and formative assessments are included, including exit tickets and project assessments. The lessons are accessed through Canvas, which include detailed lessons to support implementation. Student slide decks and challenge activities are currently the only included options for differentiation. ELL Supplemental Teaching Guides offer options for various levels of ELL students. Teacher supports include materials to help teachers understand core concepts, typical student thinking of concepts, and pedagogical strategies to engage students in the concepts.

Technology requirements

- Supported platforms include Mac, Windows, and Chromebooks.

Professional Development

- Teacher training on Ellipsis Education curriculum and platform is available. PD is typically 3–4 hours, provided virtually with a live customer support team member.
- Customer and training support continues beyond the PD through a dedicated Customer Experience Team. Supports are provided for computer science integration, including adding computer science into an existing class period, a specials rotation, or introducing a stand-alone class.

Costs

- Curriculum pilots are available for $750. Full year courses are $75 per curriculum hour, with a per building cost cap of $3,500 for middle schools (6–8), including multiple course purchases, with no educator or student seat limits and no kits to buy.
- Included with curriculum at no extra cost: product and platform video-based training, and ongoing customer support throughout the subscription period.
- Virtual live teacher training sessions: $500 per educator.

COMPUTER SCIENCE DISCOVERIES*

Code.org
studio.code.org/courses/csd-2021/

An introductory survey course that touches upon many DLCS topics.

Nature of curriculum	Stand-alone
Format and time	2 semesters or 1 year-long course (6 units)
Key classroom routines	Student activities, small group discussions, reflections
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	Yes
Programming platform or language	GameLab and AppLab (online block- and/or text-based environments), HTML and CSS; Circuit Playground
Web-based	Yes

Gr. 6–8 DLCS Topics	***Computer Science Discoveries* Alignment**
CAS.a. Safety & Security	*Partial*
CAS.b. Ethics & Laws	✓
CAS.c. Interpersonal & Societal Impact	*Partial*
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	✓
DTC.c. Research	*Partial*
CS.a. Computing Devices	*Partial*
CS.b. Human & Computer Partnerships	✓
CS.c. Networks	*Partial*
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	*Partial*
CT.c. Data	*Partial*
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	✓

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

COMPUTER SCIENCE DISCOVERIES

Overview

Computer Science Discoveries is an introductory survey course designed to engage students in learning many digital literacy and computer science topics. Units contain a mix of unplugged and plugged activities through hands-on exploration and programming in different environments. The curriculum includes activity guides, videos, online development environments, and project-based summative assessments. Student resources, such as videos, include representations of diverse populations. Several of the projects encourage students to consider problems of the world, including how to help people in their own community. For example, *Artificial Intelligence and Machine Learning* engages students in a project using data and machine learning to solve a problem in their community, and evaluating their dataset and model to avoid unintended bias. Teachers are provided with a portal through which they can manage classes, review student work, and view assessment questions and responses. Selected teaching tips provide suggestions for specific instructional strategies and routines. Some general differentiation strategies are provided for units and lessons, although those tend to not be specific to particular student populations.

Technology requirements

- Requires a desktop or laptop computer. The smallest screen size is 1024x728 px. Chromebooks are compatible. Not compatible with tablets or mobile devices.
- At least a 15 MBit/sec Internet connection.
- Adafruit's Circuit Playground: a programmable board with sensors and LEDs.

Professional development offered

- Code.org-trained and approved facilitators run local summer institutes for one week, with ongoing quarterly workshops throughout the school year.
- An online community forum with active forum monitors to facilitate discussion.

Costs

- The curriculum and online student learning platform are free.
- All professional development opportunities are currently available at no cost to educators, based on grants and contributions secured by Code.org.
- Circuit Playground costs about $25 per board, with expectation of one per student pair.

COMPUTER SKILLS - DIGITAL SAVVY*

CompuScholar
compuscholar.com/schools/courses/computer-skills/

An introductory survey course that touches upon many DLCS topics.

Nature of curriculum	Stand-alone
Format and time	Year-long course
Key classroom routines	Online content, hands-on programming activities
Assessments, rubrics, examples	Yes
Differentiation supports	No
English Learner supports	Limited
Teacher content knowledge support	Limited
Cultural responsiveness	No
Programming platform or language	Scratch (online blocks-based environment), HTML
Web-based	Yes

Gr. 6–8 DLCS Topics	***Computer Skills - Digital Savvy* Alignment**
CAS.a. Safety & Security	✓
CAS.b. Ethics & Laws	*Partial*
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	✓
DTC.c. Research	✓
CS.a. Computing Devices	*Partial*
CS.b. Human & Computer Partnerships	X
CS.c. Networks	✓
CS.d. Services	*Partial*
CT.a. Abstraction	*Partial*
CT.b. Algorithms	*Partial*
CT.c. Data	*Partial*
CT.d. Programming & Development	*Partial*
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

COMPUTER SKILLS - DIGITAL SAVVY

Overview

The *Computer Skills - Digital Savvy* course covers a broad range of digital literacy and computer science topics, including operating systems and networking. The online portal, which can be integrated with various learning management systems, provides the main content delivery, activity portal, and resources repository. Assessments are provided via automated quizzes and tests, and rubrics are provided for activities. Teachers are provided limited content knowledge support in the curriculum, primarily through skill-building tips. Limited English Learner support is provided through access to Microsoft Language Portal for terminology searches, and the course layout can be viewed in Spanish, however, the text and videos are all in English.

Technology requirements

- Computer, laptop, or tablet with Windows or Mac operating system and any HTML5-compliant web browser with an Internet connection.
- Scratch can be accessed online. A downloadable version of Scratch is available.

Professional development offered

- Skill-building sessions and monthly enrichment webinars.
- CompuScholar PD and Orientation include live webinars at the beginning of each school year to prepare for implementing the curriculum, registration and logistics, using the online courses, and getting help from CompuScholar.
- Ongoing technical support is provided through the year.

Costs

- Costs are dependent on number of students, but approximately $500 per year for up to ten students, up to $2,500 per year for an unlimited campus license (all courses and students).
- Professional development costs are included in these costs.
- District-level licenses can be customized to accommodate distributions of students across multiple locations.

COMPUTING IDEAS*

CodeHS
codehs.com/info/curriculum/computing_ideas

A programming approach to learning computing systems.

Nature of curriculum	Stand-alone
Format and time	1-year course (~33 weeks)
Key classroom routines	Video tutorials, programming exercises, challenge problems
Assessments, rubrics, examples	Limited
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	No
Cultural responsiveness	No
Programming platform or language	Karel (online block- and/or text-based environment), HTML, CSS
Web-based	Yes

Gr. 6–8 DLCS Topics	***Computing Ideas* Alignment**
CAS.a. Safety & Security	✓
CAS.b. Ethics & Laws	*Partial*
CAS.c. Interpersonal & Societal Impact	*Partial*
DTC.a. Digital Tools	*Partial*
DTC.b. Collaboration & Communication	*Partial*
DTC.c. Research	X
CS.a. Computing Devices	*Partial*
CS.b. Human & Computer Partnerships	X
CS.c. Networks	✓
CS.d. Services	*Partial*
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

COMPUTING IDEAS

Overview

The *Computing Ideas* course introduces the basics of programming, designing a web page, and the functions of the Internet. Students learn to code using blocks to drag and drop, and they can switch between blocks and text as desired. Students create a personal portfolio website showing projects they build throughout the course. Each module provides short video tutorials, example programs, quizzes, programming exercises, challenge problems, and unit tests. Additional assessment opportunities are provided through class discussions, and selected open responses. Several supplemental materials and certain activities can be differentiated for individual students, although little guidance is provided for this. A problem bank written by teachers are available for supplemental activities or remediation. Students write and run programs in the browser using the CodeHS editor. Karel the Dog is available in Spanish.

Technology requirements

- One computer per student during class (any OS) with modern browser (up-to-date Chrome, Firefox, or Safari).
- Online connectivity at a minimum of 10MB/sec.
- Access to headphones for each student in class.

Professional development offered

- Online and in-person professional development options are available.

Costs

- Curriculum access for a single teacher (unlimited students) is currently $5,500 per year, then $2-5k for additional teachers (depending on the number). There is a free, limited, version of the curriculum available.
- Online professional development is $1,450 per course for around 30 hrs. Online self-paced and in-person options are available.

CREATIVE COMPUTING CURRICULUM

Creative Computing Lab at Harvard University
creativecomputing.gse.harvard.edu/guide/

A computational thinking curriculum designed to promote student creativity and agency.

Nature of curriculum	Stand-alone or Integrated
Format and time	Course (7 units; ~4–8 hours per unit)
Key classroom routines	Creating, personalizing, sharing, reflecting
Assessments, rubrics, examples	Limited
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	Limited
Programming platform or language	Scratch (online blocks-based environment)
Web-based	Yes

Gr. 6–8 DLCS Topics	***Creative Computing* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	X
DTC.b. Collaboration & Communication	✓
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

CREATIVE COMPUTING CURRICULUM

Overview

The *Creative Computing Curriculum* engages students in designing animations, stories, and games while learning computational thinking concepts. An emphasis is placed on reflection, design journals, personalization, and sharing projects to promote brainstorming, documenting, and sharing ideas and work. Students are supported to develop personal agency and reflect their own interests, context, and experiences through stories and creative visuals. Reflection Prompts form the key assessment strategy. Brief teacher notes provide instructional tips and support for implementation. Differentiation opportunities are provided mainly through ability to personalize projects. The curriculum can be used in its entirety as a semester-long computing course, or can be selectively distributed across middle school programming (note that this curriculum is also a viable option for upper elementary school grades, although it is not recommended to be used at both elementary and middle school in any one district). The curriculum has been translated by teachers into ten different languages.

Technology requirements

- Requires computers or tablets with speaker or headphones for the computer- based design activities; Internet connectivity and up-to-date browser to connect to Scratch online. A downloadable version of Scratch is available.
- Computers equipped with microphones and webcams are recommended.

Professional development offered

- ScratchEd Meetups are participatory opportunities for teachers to have hands-on experience with Scratch, offered in cities across the country.
- The ScratchEd Online Community Archive is a venue for educators to access discussions, stories, and resources for teaching with Scratch.
- The Teaching with Scratch Facebook group is an active forum to share ideas, questions, and resources related to teaching with Scratch.

Costs

- The *Creative Computing Curriculum* and Scratch software are free.
- Available professional development options are also free.

DEVELOPING AI LITERACY

MIT
aieducation.mit.edu/daily/index.html

Explore AI concepts, ethical issues in AI, creative expression using AI, and AI applications.

Nature of curriculum	Stand-alone or Integrated
Format and time	Semester course (4 units, about 20 hours)
Key classroom routines	Group work, discussions, journals
Assessments, rubrics, examples	Limited
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	Yes
Programming platform or language	Teachable Machine, Scratch (online blocks-based environment)
Web-based	Yes

Gr. 6–8 DLCS Topics	***Developing AI Literacy* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	✓
DTC.a. Digital Tools	Partial
DTC.b. Collaboration & Communication	Partial
DTC.c. Research	Partial
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	✓
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	Partial
CT.d. Programming & Development	Partial
CT.e. Modeling & Simulation	✓

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

DEVELOPING AI LITERACY

Overview

Developing AI Literacy (DAILy) is an exploration of artificial intelligence (AI) concepts and applications to help students understand the prevalence and impact of AI on themselves and society. In each unit, students are supported in considering issues of bias in datasets and algorithms, ethical design and community impacts. This can be taught as a short course or integrated into other curricular areas such as Science or Social Studies. Each unit includes slides for introducing lessons, teacher scripts to inform instructional sequencing, activities in Google Slides with links to the online activity or videos for students, regular opportunities for group work, discussions, and journal prompts. Activities include hands-on and some online and coding activities that can be adjusted for pacing. Exemplars are provided for some activities and are the extent of provided assessments. Slides and the lesson plans provided some support for the teacher in explaining what to look for and what is expected in terms of student concepts.

Technology requirements

- Most of the activities require Google Chrome and Internet access. A camera is also required.
- Phone or MP3 Player for an activity
- Scratch can be accessed online. A downloadable version of Scratch is available.

Professional development offered

- Summer PD is available.

Costs

- The curriculum is free.

DIGITAL CITIZENSHIP

Common Sense Education
commonsense.org/education/digital-citizenship/curriculum?grades=6%2C7%2C8

Help students take ownership of their digital lives.

Nature of curriculum	Stand-alone or Integrated
Format and time	3 modules (1 per grade; ~6 hours each)
Key classroom routines	Warm up; Analyze, Apply or Create; Wrap up
Assessments, rubrics, examples	Yes
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	Limited, via teacher prompts
Cultural responsiveness	No
Programming platform or language	NA
Web-based	Yes

Gr. 6–8 DLCS Topics	***Digital Citizenship* Alignment**
CAS.a. Safety & Security	✓
CAS.b. Ethics & Laws	*Partial*
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	X
DTC.b. Collaboration & Communication	X
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	X
CT.b. Algorithms	X
CT.c. Data	X
CT.d. Programming & Development	X
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

DIGITAL CITIZENSHIP

Overview

The goal of the *Digital Citizenship* curriculum is to develop digital literacy skills necessary for effective engagement and student safety in a connected world. The three middle school modules can be taught as its own curriculum distributed across grades 6–8 or in one year. The lessons could also be integrated into another academic subject. Each lesson provides instructional sequences supported by slides, videos, and handouts to support student activities. Assessments with keys are provided. The curriculum is also available in Spanish.

Technology requirements

- Desktop or laptop with basic internet access to view lesson slides, videos and online games.
- Requires speakers or headphones.

Professional development offered

- Recorded webinar for overview of teaching the curriculum; monthly webinars and advice postings are available.

Costs

- All educator resources are free.

FOUNDATIONS OF PHYSICAL COMPUTING: LEGO SPIKE PRIME*

LEGO Education
education.lego.com/en-us/product-resources/spike-prime/teacher-resources/computer-science-courses

Use LEGO SPIKE Prime to engage students in physical computing.

Nature of curriculum	Stand-alone
Format and time	Course (full year)
Key classroom routines	Unplugged activities, group work, discussion, coding, building, journaling
Assessments, rubrics, examples	Yes
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	Limited
Programming platform or language	SPIKE app (uses Scratch, online blocks-based environment); LEGO SPIKE Prime
Web-based	No

Gr. 6–8 DLCS Topics	***Foundations of Physical Computing* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	Partial
DTC.a. Digital Tools	X
DTC.b. Collaboration & Communication	Partial
DTC.c. Research	X
CS.a. Computing Devices	✓
CS.b. Human & Computer Partnerships	✓
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	Partial

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

FOUNDATIONS OF PHYSICAL COMPUTING: LEGO SPIKE PRIME

Overview

The *Foundations of Physical Computing* curriculum uses LEGO Education's SPIKE Prime robotics set to teach computer science principles through physical computing. The curriculum includes 10 units, with 5–8 lessons each of 45–90 minutes. The overall lesson format is based on the 5E model and includes a range of unplugged activities, building and coding activities, group work, journals, and challenges and projects. Computer science lessons emphasize computational thinking, decision making, decomposition of problems, problem solving, and communication. There are several mini-challenges, and a four-phase culminating project that guides students through research and presentation of their work. Each unit includes a section where students research the connections between information technology and 16 career paths such as agriculture, medicine, manufacturing, law, hospitality and the arts. This provides an opportunity for students of a variety of backgrounds to see the potential of related careers. Student assessment comes in the form of presentations, projects which include rubrics and steps for teacher and peer review, and opportunities for self-assessment (both individual and team based). Lessons provide a brief explanation of the subject matter for teachers.

Technology requirements

- Can be used with Windows, MacOS, iPad, or Chromebook. Devices should have Bluetooth. Internet connection needed for software updates and student research.
- SPIKE Prime app is based on Scratch; available through a downloadable app.

Professional development offered

- Face to face training, virtual training, and asynchronous webinars are available options. These are typically 6 hours for up to 20 participants.

Costs

- The curriculum is free to use.
- Spike Prime costs $340; Expansion kit is $105.
- In-person and virtual PD is $3000 for 6 hours up to 20 people. There are also some free online self-paced PD options.

INTRO TO CS

Microsoft MakeCode for micro:bit
makecode.microbit.org/courses/csintro-educator

A physical computing approach to introductory computer science.

Nature of curriculum	Stand-alone
Format and time	Semester or 1-year course (~14 weeks)
Key classroom routines	Unplugged activity, micro:bit activity, project
Assessments, rubrics, examples	Yes
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	Yes
Cultural responsiveness	No
Programming platform or language	MakeCode (online blocks-based environment); micro:bits
Web-based	Yes

Gr. 6–8 DLCS Topics	***Intro to CS* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	X
DTC.b. Collaboration & Communication	X
DTC.c. Research	X
CS.a. Computing Devices	*Partial*
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	X
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

INTRO TO CS

Overview

Intro to CS uses the micro:bit (a micro-controller device) to engage students in physical computing. Students program using Microsoft MakeCode to control physical elements on or attached to the micro:bit. Typical lessons are structured to include an unplugged activity, such as an offline game or activity that demonstrates the concept/topic, a micro:bit activity where students program their micro:bit, and a project in which each student creates to demonstrate their understanding of the skills and concepts covered in this lesson. Assessments and rubrics are provided for each lesson and key projects. Example projects are provided in the lessons.

Technology requirements

- Computer, laptop, or tablet with an internet connection, modern browser, and USB port. Speakers or headphones are recommended.
- micro:bits (curriculum assumes one per student). Note that there is now a version 2 micro:bit available.

Professional development offered

- A variety of Microsoft Education Training Partners and Learning Consultants can provide professional development on Intro to CS.

Costs

- The *Intro to CS* curriculum is free.
- Individual micro:bits are approximately $15 each; consider mini-kits that include battery holder, USB cable, and such for approximately $20 each.
- Some electronic accessories are needed, such as alligator clips, jumper cables, servo motors, copper tape, speakers, or other micro:bit accessories.
- Some craft materials are also needed, such as cardboard, colored paper, markers, pipe cleaners, and other assorted craft and/or household materials
- Professional development costs vary by vendor.

INTRODUCTION TO CYBERSECURITY

CodeHS
codehs.com/course/cyber_caesar/overview

An introduction to digital citizenship, cyber hygiene, and information literacy.

Nature of curriculum	Stand-alone
Format and time	Course (about 90 hours)
Key classroom routines	Video lessons, Connections, class discussion
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	Yes
Cultural responsiveness	No
Programming platform or language	NA
Web-based	Yes

Gr. 6–8 DLCS Topics	***Intro to Cybersecurity* Alignment**
CAS.a. Safety & Security	✓
CAS.b. Ethics & Laws	Partial
CAS.c. Interpersonal & Societal Impact	Partial
DTC.a. Digital Tools	Partial
DTC.b. Collaboration & Communication	Partial
DTC.c. Research	Partial
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	X
CT.b. Algorithms	X
CT.c. Data	X
CT.d. Programming & Development	X
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

INTRODUCTION TO CYBERSECURITY

Overview

Introduction to Cybersecurity helps students to understand the basics of cybersecurity, Internet etiquette, staying safe online, digital footprints, and information literacy. The Caesar version of this course is designed for students with no background in computer science. The course includes a final project in which students create a public service announcement about an aspect of cybersecurity. Lessons are centered on video lessons, class discussions, and includes connection examples and free response prompts for students to show understanding of the concepts. Assessments are provided, including pre- and post-tests, quizzes, and indicators to aid grading of assignments. Teacher support is provided in the Pro version which includes Problem Guides to support teachers in the lesson, handouts, planning notes, teaching strategies, prior knowledge, and video slides. Differentiation supports include a few additional activity options and general tips for adjusting lessons for specific learners.

Technology requirements

- One computer per student during class (any OS) with modern browser (up-to-date Chrome, Firefox, or Safari).
- Online connectivity at a minimum of 10MB/sec.
- Access to headphones for each student in class.

Professional development offered

- Online and in-person professional development options are available.

Costs

- Curriculum access for 100+ students is currently (approximately) $6,100 per year. $2,100 for 30 or fewer students. There is a free, limited, version available.
- Online professional development is $1,750 per course for around 30 hrs. Online self-paced and in-person options are available.

MICRO:BIT UNITS

micro:bit
microbit.org/lessons/energy-awareness/; microbit.org/lessons/helping-plants-grow/

Supplementary units to engage students with data, modeling, and networks using micro:bits.

Nature of curriculum	Stand-alone
Format and time	2 Units (about 8 hours)
Key classroom routines	Pair work, small group, whole class, unplugged and plugged activities
Assessments, rubrics, examples	Limited
Differentiation supports	Yes
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	No
Programming platform or language	MakeCode (online blocks-based environment), Python; micro:bits
Web-based	No

Gr. 6–8 DLCS Topics	***Energy Awareness*** **Alignment**	***Helping Plants Grow*** **Alignment**
CAS.a. Safety & Security	X	X
CAS.b. Ethics & Laws	X	X
CAS.c. Interpersonal & Societal Impact	X	X
DTC.a. Digital Tools	X	X
DTC.b. Collaboration & Communication	Partial	X
DTC.c. Research	X	X
CS.a. Computing Devices	Partial	Partial
CS.b. Human & Computer Partnerships	Partial	✓
CS.c. Networks	X	✓
CS.d. Services	X	Partial
CT.a. Abstraction	X	X
CT.b. Algorithms	X	X
CT.c. Data	✓	X
CT.d. Programming & Development	X	X
CT.e. Modeling & Simulation	Partial	✓

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

MICRO:BIT UNITS

Overview

Two micro:bit units, *Energy Awareness* and *Helping Plants Grow*, provide opportunities for students to engage with data, modeling, and networks using micro:bits. The *Energy Awareness* unit engages students in collecting real data, visualizing it, interpreting it and analyzing it to make decisions. The *Helping Plants Grow* helps students understand networks and modeling. While designed as stand-alone units, each has the potential to be integrated into other disciplines, such as science or engineering. Lessons include an introduction, learning objectives, materials, plugged and unplugged activities, and extension ideas. Differentiation opportunities are provided through stretch and challenge exercises. Assessments are limited, through informal and observational tips and general success criteria for activities. Teacher support materials include lesson plans, lesson slides, and coding files to support the lesson as it progresses.

Technology requirements

- Access to the MakeCode platform or Python code editor via computer, laptop, or tablet with an internet connection, modern browser, and USB port.
- Micro:bits, recommended 1 per pair of students. Note that there is now a version 2 micro:bit available.

Professional development offered

- None specific to these particular units.

Costs

- The curriculum is free.
- Individual micro:bits are approximately $15 each; consider mini-kits that include battery holder, USB cable, and such for approximately $20 each.
- Some materials for *Helping Plants Grow* Unit, such as a breakout board, 3v dual relays, and header wires.

NFTE STARTUP TECH*

Network for Teaching Entrepreneurship
nfte.com/programs/nfte-startup-tech-coding-meets-entrepreneurial-mindset/

A blended learning course combining entrepreneurship and technology skills.

Nature of curriculum	Stand-alone
Format and time	1-year course
Key classroom routines	Programming activities, entrepreneurship activities, sharing and presenting
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	No
Cultural responsiveness	Yes
Programming platform or language	MIT App Inventor (online blocks-based environment)
Web-based	Yes

Gr. 6–8 DLCS Topics	***NFTE Startup Tech* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	*Partial*
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	X
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	*Partial*
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	*Partial*
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	*Partial*

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

NFTE Startup Tech

Overview

NFTE Startup Tech is a project-based learning curriculum that engages students in developing both entrepreneurial skills and technical skills. Students use MIT App Inventor to develop an app that addresses a community need. They develop a business plan and present in a showcase. Opportunities for competitions are a unique aspect of this program. It is a thoughtfully organized course that uses Canvas and a teacher support community for student engagement and teacher collaboration. Assessments are included to provide feedback on key concepts and project development. Differentiation opportunities are provided primarily through personalization of the projects. The curriculum supports students to identify and address solutions relevant to their own communities in ways that can enhance equity.

Technology requirements

- Desktop or laptop with a reliable Internet access and an up-to-date browser.
- Mobile devices (can use an emulator). One device for every two students is recommended. Android and iOS devices are supported.
- Uses Canvas LMS to deliver content.

Professional development offered

- Virtual curriculum training specific to the course.
- Webinar-based MIT App Inventor training.
- Quarterly Professional Learning Communities meetings.
- Teacher Hub that includes program guide, Canvas guide, scope and sequence, and resources shared by teachers.
- Four-day National Summit.

Costs

- $2,500 for *NFTE Startup Tech* course per year, plus various activation fees and an annual partnership fee of $5000 for the first year and $1500 for subsequent years.
- Training is $750 per day, with a typical total of $2,250.

OZARIA

CodeCombat
ozaria.com

Adventure game format to keep students engaged while learning computer science concepts.

Nature of curriculum	Stand-alone
Format and time	Course (4 chapters, about 20 weeks)
Key classroom routines	Unplugged activities, online games, reflections
Assessments, rubrics, examples	Yes
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	Yes
Cultural responsiveness	No
Programming platform or language	JavaScript or Python
Web-based	Yes

Gr. 6–8 DLCS Topics	***Ozaria* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	X
DTC.b. Collaboration & Communication	Partial
DTC.c. Research	X
CS.a. Computing Devices	Partial
CS.b. Human & Computer Partnerships	X
CS.c. Networks	✓
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	Partial
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	Partial

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

OZARIA

Overview

In *Ozaria*, students progress through an adventure game as they learn concepts and coding along the way. Students can learn JavaScript or Python when playing. Lessons include unplugged activities, reflections, games, and the online *Ozaria* platform to support the students' adventure. Interactives include formative checks for student knowledge, presented in a 'multiple-choice' or 'fill in' format. Interactives accompany Cinematics (expositional dialog that introduces concepts) and reinforce the concept(s) introduced. *Ozaria* incorporates shareable Capstone Projects at the end of each Chapter, where students create their own games. Capstone projects feature more structural support in the form of the in-game helper and Intros, which guide students through high-level steps to complete each project. Rubrics and exemplars are provided for Capstone levels. A scaffolded student tutorial is provided along with teacher-led lesson presentations including slides and teacher notes. Lessons can be adapted for varied schedules.

Technology requirements

- *Ozaria* runs best on computers with at least 4GB of RAM, on a modern browser. Chromebooks with 2GB of RAM may have minor graphics issues in later units. iPads and Android Tablets are not supported at this time.
- A minimum of 200 Kbps bandwidth per student is required, although 1+ Mbps is recommended.

Professional development offered

- 40-hour PD focused on computer science topics, practical classroom applications, real-world connections, and differentiated instruction.

Costs

- Yearly subscription costs range from $50 per student (for up to 100 students) to $15 per student (for more than 1000 students).
- PD is $2000 per teacher, with options for multi-teacher discounts.

PROJECT GUTS

Project Growing Up Thinking Scientifically (GUTS)
projectguts.org/; teacherswithguts.org/resources

Computational thinking in science with a strong emphasis on modeling and simulation.

Nature of curriculum	Stand-alone or Integrated with science
Format and time	4 modules (~10–25 hours per module)
Key classroom routines	"Use-Modify-Create" progression, physical and programmed simulation activities
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	Limited
Teacher content knowledge support	Limited
Cultural responsiveness	No
Programming platform or language	StarLogo Nova (online blocks-based agent-based modeling environment)
Web-based	Yes

Gr. 6–8 DLCS Topics	***Project GUTS* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	X
DTC.b. Collaboration & Communication	X
DTC.c. Research	X
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	*Partial*
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	✓

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

PROJECT GUTS

Overview

Project GUTS engages students with modeling and simulation coding using StarLogo Nova to explore science concepts related to disease transmission, water use, ecosystems, and chemical reactions. The units can be coordinated with a district's science scope and sequence. Each module assumes prior knowledge of science concepts; these are not explicitly taught. Each of the five lessons in each module includes suggestions regarding various student learning needs. These provide brief comments about how principles of Universal Design are or can be achieved, some English Language Learner comments, and addressing diverse populations, but these are not typically framed as specific instructional actions. Instructional strategies such as small group work, student partners, and sharing of models are consistently employed through the curriculum. Occasional callouts in the curriculum provide tips or hints to teachers, but not substantive content knowledge support. Some links are provided to background information for things like using StarLogo Nova or water cycle content. Assessment questions are provided in each lesson, and samples of assessments and rubrics are available online. Spanish language translations of many activities are available online.

Technology requirements

- Desktop or laptop computer running any major OS or iPad.
- Browser to run StarLogo Nova 2.0 (does not require Flash).
- Internet bandwidth supporting downloads of 5 megabits per student.

Professional development offered

- Workshops available for 25–40 teachers, including: pre-workshop preparation (~1 day, online); intensive PD workshop (three days, face-to-face); mini-workshops (two one-day workshops, face-to-face or virtual); and online support.
- An online PD course and a moderated online PD network.

Costs

- All curriculum resources—lesson plans, slide decks, videos, and supplemental extension resources for teachers—are provided at no cost to schools.
- The sponsoring district is required to provide the location, materials, meals or meal vouchers, and stipends, and to recruit and register participants. Sponsoring districts contract with veteran *Project GUTS* facilitators and cover their transportation and stipends for offering workshops.

PLTW GATEWAY: COMPUTER SCIENCE UNITS*

Project Lead the Way
pltw.org/our-programs/pltw-gateway

Programming for apps and the physical world.

Nature of curriculum	Stand-alone
Format and time	2 units (45 days each)
Key classroom routines	Activity, project, and problem structure
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	Yes
Cultural responsiveness	Limited
Programming platform or language	MakeCode (online blocks-based environment, with micro:bits); MIT App Inventor (online blocks-based environment)
Web-based	Yes

Gr. 6–8 DLCS Topics	***PLTW Gateway: CS Units* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	*Partial*
DTC.b. Collaboration & Communication	X
DTC.c. Research	X
CS.a. Computing Devices	✓
CS.b. Human & Computer Partnerships	*Partial*
CS.c. Networks	*Partial*
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	*Partial*
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

PLTW GATEWAY: COMPUTER SCIENCE UNITS

Overview

PLTW Gateway is a series of STEM-related units, of which *Computer Science for Innovators and Makers* (*CSIM*, for grade 6) and *App Creators* (for grades 7 or 8) are the two with a computer science focus. The units use an activity-, project-, problem-based instructional approach, and each culminates in an application project. *CSIM* teaches students to program for the physical world by blending hardware design and software development. *App Creators* teaches students how to computationally analyze and develop solutions through mobile app development. In several projects students engage with or share aspects of their community, including designs to help someone in their community. Lesson assessments are provided, along with project rubrics and guides. Several accessibility features are provided for online materials. Each unit is available in Spanish. Teacher content knowledge support is found in teacher guides and professional development.

Technology requirements

- Desktop or laptop with Internet connectivity and an up-to-date browser.
- Android or iOS tablets and micro:bits: one for each per pair of students.

Professional development offered

- A 40-hour professional development course is offered in person or online, led by PLTW Master Teachers. Training addresses unit content, pedagogy, and facilitation support. After a course, teachers can access additional resources and communities to reinforce skills learned throughout the school year, including post-training webinars, online asynchronous training, and unit-specific resources.
- Additional supports include email or phone support; an online community of teachers, PLTW Launch teachers and Master Teachers; a course specifically for PLTW Launch Lead teachers that build their capacity to support their PLTW Launch implementation; and opportunities to provide feedback to PLTW.

Costs

- CSIM materials kits include micro:bits and a variety of materials such as conductive thread, alligator clips, conductive paint, copper tape, potentiometer, pressure sensor, light sensor, LEDs, SG90 servo, and battery holders. Initial purchase cost is approximately $1,000 per class. Subsequent replacement of the few consumables is needed on a periodic basis.
- All software used in the PLTW Gateway CS units is free.
- Yearly fee for schools is $950, which includes all 10 Gateway units. Schools may apply for funding and grants opportunities through the PLTW Grant program.
- Core training opportunities for beginning teachers are $1,200 per teacher per unit.

SPHERO COMPUTER SCIENCE FOUNDATIONS COURSES 2 & 3

Sphero
edu.sphero.com/cs-foundations

Computer science foundations through robotics.

Nature of curriculum	Stand-alone
Format and time	Courses (24 lessons each course, 45–60 min lessons)
Key classroom routines	Exploration, Skill Building, Challenge, Sharing, Reflection, and Extended Challenge
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	No
Cultural responsiveness	No
Programming platform or language	Sphero Edu Blocks Canvas, Sphero Edu JavaScript Text Canvas; Sphero BOLT
Web-based	Yes

Gr. 6–8 DLCS Topics	***Sphero CS Foundations 2 & 3* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	X
DTC.b. Collaboration & Communication	X
DTC.c. Research	X
CS.a. Computing Devices	Partial
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	X
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

SPHERO COMPUTER SCIENCE FOUNDATIONS COURSES 2 & 3

Overview

The *Computer Science Foundations Courses 2 and 3* teach CS through the use of Sphero BOLT robots. Course themes include: Empathy (an SEL-based unit), Game Design, Storytelling (in Course 2) and Brain Breakers, Missions, and Navigation (Course 3). Both courses focus on computational thinking, with Course 2 teaching block-based coding and Course 3 progressing to text-based coding. A school can choose to implement one or both courses. Each course can be used as a full year course, semester, or quarterly. Units within courses could also be integrated into other curricular areas by skipping around content themes and programming levels. Sphero Edu's student platform provides students with pre-loaded lessons and a programming canvas for their BOLT robots. There are unplugged activities, coding activities, and a mix of individual and group activities as well as reflections. Some lessons have videos for additional support. Typically, the finished programs are the assessments for the lessons, supplemented with some questions or reflection activities to check for understanding. No rubrics are provided although there are tips for teachers on what to look for and suggested solutions. There are options for Extended Challenges if time allows and this could be used for students who are ready to advance further. Some of the activities can be adjusted for various learners but no specific differentiation suggestions are provided. Both courses include a printed Educator Guide which provided a clear instructional sequence and tips to help support lesson implementation. Additionally, Sphero Edu's teacher platform allows educators to sync Google Classroom or Clever, assign lessons and track student progress.

Technology requirements

- Sphero BOLT (recommend 1 for each pair of students)
- Desktop app (Windows, MAC, Chrome OS), Mobile App (iOS, Android, Fire OS), or browser interface
- Check Sphero's System Compatibility and Robot Compatibility chart for specifics (support.sphero.com/article/3tzmrkc6lx-edu)

Professional development offered

- On-site workshop includes a one day (6 hour) interactive training for up to 25 instructors. Multiple day or concurrent sessions also available.
- Virtual PD: Yearly unlimited access for up to 20 educators (from across the district). Educators access live, personalized one-on-one and/or group training. A list of suggested sessions is provided, but requests are also considered.

Costs

- Each *Sphero Computer Science Foundations Course* curriculum: $300 per course.
- Sphero BOLT Power Pack: $2650, which includes 15 BOLTs (class set) in a portable, charging case on wheels.
- 1-day on-site workshop: $3,000, inclusive of travel expenses for the trainer. All-access Virtual PD: $1500 per year.

GRADES 9 THROUGH 12

CONSIDERATIONS FOR SELECTING HIGH SCHOOL CURRICULA

Curricula included in this section are primarily semester to full-year high school courses with a broad scope across DLCS strands and topics. There are, however, a number of options provided that have more of a topical focus or could be used to fill gaps in existing DLCS programming. A number of organizations included in this section also have middle school offerings, providing for the possibility of pathways using curricula from the same organization.

Districts should consider several perspectives for high school DLCS programming:

1. What DLCS programming has been provided at earlier grades (elementary and middle school). There may be more need for introductory courses if there has been little DLCS programming at earlier grades in the district.
2. Pathways within DLCS topics. This may include, for example:
   - An introductory course, intermediate course, and an advanced course option(s).
   - Overview courses and specialized courses (e.g., cybersecurity).
3. Relationships of DLCS courses to other STEM course offerings.

If a school does not currently have substantive DLCS offerings, the following options are suggested as a starting point given their relatively broad scope to provide a strong foundation for a comprehensive high school DLCS course:

- *Beauty and Joy of Computing* (UC Berkely & EDC)
- *code.org Computer Science Principles* (code.org)
- *Computational Thinking and Problem Solving* (CSforMA)
- *Exploring Computer Science* (University of Oregon)
- *Mobile Computer Science Principles* (Mobile CSP Project)
- *PLTW Computer Science Principles* (PLTW)

This guide only focuses on courses substantially aligned to the state DLCS standards. This guide does not include AP-level course options, many specialized topical courses, or courses focused on programming syntax.

The Board of Elementary and Secondary Education (BESE) has established guidance on using computer science courses for MassCore. Adopted in 2007, MassCore is a state-recommended program of study intended to align high school coursework with college and workforce expectations. The guidance provides for certain computer science courses to substitute for a mathematics or science course.[1] From DESE:

> The ability to effectively use and create technology to solve complex problems is an essential literacy skill. Recognizing the importance of these competencies, BESE and the Board of Higher Education (BHE) amended MassCore, in June 2018, to allow approved computer science courses to substitute for either a mathematics course or a laboratory science course. In January 2019, BESE approved the following initial set of courses:

[1] See DESE MassCore guidance at **www.doe.mass.edu/ccte/ccr/masscore/**, particularly questions #19–21 of the FAQ.

1. #10019 - AP Computer Science Principles, any College Board approved curriculum
2. #10011 - Computer Science Principles, any College Board approved curriculum, non-AP
3. #10012 - Exploring Computer Science, an NSF developed curriculum

The curriculum included in this guide include five College Board-approved Computer Science Principles (CSP) courses as well as Exploring Computer Science. There are other College Board-approved CSP courses, which can be found at: https://apcentral.collegeboard.org/courses/ap-computer-science-principles/classroom-resources/curricula-pedagogical-support.

(This page left intentionally blank.)

AI FOUNDATIONS*

UBTECH Education
ubtecheducation.com/ai-foundations-curriculum/

Explore Artificial Intelligence through the 5 big ideas of AI.

Nature of curriculum	Stand-alone
Format and time	2 Units (each 18–24 sessions of ~60 minutes each)
Key classroom routines	Discussion, research, presentations, using webtools
Assessments, rubrics, examples	Yes
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	Yes
Programming platform or language	NA
Web-based	Yes

Gr. 9–12 DLCS Topics	***AI Foundations* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	*Partial*
DTC.a. Digital Tools	*Partial*
DTC.b. Collaboration & Communication	X
DTC.c. Research	✓
CS.a. Computing Devices	*Partial*
CS.b. Human & Computer Partnerships	✓
CS.c. Networks	X
CS.d. Services	✓
CT.a. Abstraction	*Partial*
CT.b. Algorithms	*Partial*
CT.c. Data	✓
CT.d. Programming & Development	X
CT.e. Modeling & Simulation	*Partial*

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

AI FOUNDATIONS

Overview

Each lesson of the *AI Foundations* curriculum addresses one of the 5 Big Ideas of Artificial Intelligence: perception, representation and reasoning, learning, natural interaction, and social impact. This curriculum can serve to develop the foundation concepts as a precursor to coding AI in robots. The lessons are organized around the 5E model and use a variety of unplugged activities with internet research. There is an emphasis on students discussing and debating elements of different articles and videos accessed from the internet. Students analyze sensors in their cell phones, search engines, and map models. Students examine the connections between data and a mind map. Teaching techniques include discussion, jigsaw activities, debates, presentations, and activities using cell phones, google maps and search engines. Students explore how their cell phones can recognize fingerprints, images, and voices. Students explore machine learning algorithms in map apps on devices. Students train an AI on images of real objects using Teachable Machines by Google Labs. The second unit addresses topics such as autonomous vehicles, artificial neural networks, inclusive AI systems, and sorting algorithms. Although there are no EL or differentiation supports, the teacher can modify the variety of unplugged lessons to fit student needs. Each lesson has an 'Evaluate' step, with activities such as reflections, comparing results to peers, explaining results to peers, check for understanding questions, and design improvements to eliminate bias in an AI algorithm. At various moments in the curriculum students reflect, respond to a driving question, and present ideas. Problems of bias and prejudice in training AI are addressed, and students explore how to redesign a biased AI system to benefit society. Students look at real life examples of AI algorithms demonstrating bias and prejudice such as racism embedded in healthcare, corporate hiring algorithms, and algorithms used by the courts in correctional systems. In unit 2, students explore AI issues related to inclusion, exclusion and bias. This curriculum does not have any traditional 'coding' and would pair well with a coding curriculum, although it does discuss models and algorithms as represented in artificial intelligence.

Technology requirements

- Internet-enabled devices with webcams and Chrome browser.

Professional development offered

- One full day or half day in-person PD is available.

Costs

- *AI Foundations* is $1499 per grade band which includes a 1-year building license with up to 5 secure logins to access the curriculum and supporting materials.
- A half-day in-person PD session is $2000.
- Purchase of the curriculum plus one full day of PD is $3000.

BEAUTY AND JOY OF COMPUTING*

University of California, Berkeley, and Education Development Center (EDC)
bjc.berkeley.edu

Develop a sense of programming aesthetics using recursion and higher-order functions.

Nature of curriculum	Stand-alone
Format and time	Full-year course
Key classroom routines	Virtual programming labs, think out-louds, projects
Assessments, rubrics, examples	Yes
Differentiation supports	Yes
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	No
Programming platform or language	Snap*!* (online visual programming environment)
Web-based	Yes

Gr. 9–12 DLCS Topics	***Beauty and Joy of Computing* Alignment**
CAS.a. Safety & Security	*Partial*
CAS.b. Ethics & Laws	*Partial*
CAS.c. Interpersonal & Societal Impact	✓
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	✓
DTC.c. Research	*Partial*
CS.a. Computing Devices	*Partial*
CS.b. Human & Computer Partnerships	*Partial*
CS.c. Networks	✓
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	*Partial*
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	*Partial*

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

BEAUTY AND JOY OF COMPUTING

Overview

The *Beauty and Joy of Computing* (BJC) course is designed to attract nontraditional computing students and encourage the joy of programming. Based on the Computer Science Principles (CSP) framework, it takes a project-based learning approach with applications in games, art, and mathematics using visual blocks-based programming. *BJC* supports students to develop computational habits of mind and aesthetics of programming through a focus on abstraction and modularity, algorithms, modeling, and the use of conditionals, recursive functions, higher-order functions, and other control structures. A strong emphasis is placed on the social issues of computing, with labs on, for example, privacy, copyright, artificial intelligence, and cybersecurity. Assessments are included through self-check quizzes, project-based end-of-unit assessments, and example solutions. Key pedagogical strategies include virtual programming labs, thinking out loud sessions, collaboration, debugging, and projects. A variety of differentiation opportunities are provided throughout the lessons. A Spanish version of the curriculum will be available during the 2020–2021 school year. Limited teacher content knowledge is provided in the curriculum, with additional opportunities provided through the online educator forum. This is a College Board-approved CSP course so it can be a substitute for a mathematics or science course in MassCore.

Technology requirements

- Desktops or laptops, using any OS, with Internet connectivity and an up-to-date browser. Recommended one device for every two students.

Professional development offered

- *BJC* offers a one-week summer workshop in locations across the United States for teachers intending to teach *BJC* the following year.
- Teachers can join periodic online gatherings with project leaders and mentor teachers. There is also an active online community of *BJC* teachers, providers, and curriculum developers.

Costs

- The curriculum and software are free to use.
- The cost of the summer professional development workshop is $75 per teacher.
- *BJC* uses the book *Blown to Bits* (Abelson, Ledeen, and Lewis, 2008) as a reference text. This is available online for no cost.

BOOTSTRAP:DATA SCIENCE AND BOOTSTRAP:ALGEBRA*

Brown University
bootstrapworld.org/

Modules to explore real-world questions through programming, data analysis, and Algebra.

Nature of curriculum	Stand-alone or Integrated
Format and time	Modules (approximately 25 hours)
Key classroom routines	Launch, investigate, synthesize, close
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	Limited
Teacher content knowledge support	Limited
Cultural responsiveness	Yes
Programming platform or language	Pyret (online programming environment)
Web-based	Yes

Gr. 9–12 DLCS Topics	***Bootstrap:Data Science* Alignment**	***Bootstrap:Algebra* Alignment**
CAS.a. Safety & Security	X	X
CAS.b. Ethics & Laws	*Partial*	X
CAS.c. Interpersonal & Societal Impact	X	X
DTC.a. Digital Tools	*Partial*	*Partial*
DTC.b. Collaboration & Communication	✓	*Partial*
DTC.c. Research	✓	X
CS.a. Computing Devices	X	X
CS.b. Human & Computer Partnerships	*Partial*	X
CS.c. Networks	X	X
CS.d. Services	X	X
CT.a. Abstraction	✓	*Partial*
CT.b. Algorithms	*Partial*	*Partial*
CT.c. Data	✓	✓
CT.d. Programming & Development	✓	✓
CT.e. Modeling & Simulation	*Partial*	*Partial*

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

BOOTSTRAP:DATA SCIENCE AND BOOTSTRAP:ALGEBRA

Overview

The *Bootstrap:Data Science* module emphasizes analysis of data, including large datasets, to address real world questions such as: What factors make some people live longer than others? Are more expensive restaurants really better? Is voter fraud a problem? Students are supported to form their own questions, analyze data using multiple methods, and write a research paper about their findings. Datasets allow for considerations of race, poverty, and incarceration, and sense-making and drawing connections are often about social structures, resources, or power dynamics. Sense-making in the course supports student identification of varied narratives and considerations of how the data supports those. The module focuses on programming and data analysis, including topics such as functions, looping and iteration, data visualization, and linear regression. A wide range of project activities are available for each unit. The module can be taught as its own curriculum or integrated into another applicable course. Each lesson is organized around a launch, investigate, synthesize, and closing format. A student workbook allows for pencil-and-paper activities not requiring a computer. Occasional opportunities for differentiation are provided in extension and optional activities. Assessments are provided through activities in the student workbook (with answer keys for the teacher). Limited teacher content knowledge supports are provided in the curriculum in the form of common student misconceptions related to lesson concepts. The *Bootstrap:Data Science* module uses Pyret (a Python-like functional programming language).

Bootstrap:Algebra is a coding module meant to be integrated into an existing Algebra course or computer science course. This can be offered at middle or high school, depending on when Algebra is offered (see the entry in middle school for a full write up). An additional module, *Reactive*, is also available. These courses can be combined to form a full stand-alone computer science course.

Technology requirements

- One computer per pair of students (desktop or laptop) with Internet access, an up-to-date browser, and a Google account. All programming materials run in the browser.

Professional development offered

- On-demand workshops are available, typically three days, for 25–40 teachers, and are highly recommended for beginning teachers. An optional 4th day and school-year support is offered for first time educators to the curriculum. Sessions are held in various locations in the US or district-based sessions can be arranged.
- Active discussion group, moderated by Bootstrap staff and master teachers, and video office hours where anyone can ask questions.

Costs

- All curriculum materials and the Pyret IDE are free.
- Professional development workshops are currently $500 per teacher.

CODE.ORG COMPUTER SCIENCE PRINCIPLES*

Code.org
curriculum.code.org/csp-20/

Encourage students to be creative, express themselves, and share their creations.

Nature of curriculum	Stand-alone
Format and time	Full-year course
Key classroom routines	Unplugged activities, programming activities, journaling, discussions, projects
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	Yes
Cultural responsiveness	Limited
Programming platform or language	Internet Simulator (online), AppLab (JavaScript blocks-to-text environment), Widgets (online)
Web-based	Yes

Gr. 9–12 DLCS Topics	***Code.org CSP* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	*Partial*
CAS.c. Interpersonal & Societal Impact	✓
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	✓
DTC.c. Research	✓
CS.a. Computing Devices	*Partial*
CS.b. Human & Computer Partnerships	X
CS.c. Networks	*Partial*
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	✓
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	*Partial*

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

CODE.ORG COMPUTER SCIENCE PRINCIPLES

Overview

Computer Science Principles (CSP) is a full-year course designed to support students and teachers who are new to computer science. No computer science prerequisites are expected. Using inquiry-based activities, videos, computing tools, and projects, teachers can guide and learn alongside students as they develop core computing concepts and come to understand foundations of modern computing. Some issues of social, cultural, or political impacts of the internet are considered, as well as a bit about bias in analytical processes. Key instructional strategies include unplugged and plugged activities, programming, journaling, small-group discussions, peer feedback and collaborative activities, debugging, and projects. Assessments include end-of-unit assessments, unit project guides and (through CodeStudio) multiple choice or matching questions and free-response text fields. Limited differentiation opportunities are provided in a number of lessons. Teacher content knowledge support is provided through notes and videos for the teacher. *CSP* uses AppLab, a JavaScript programming environment, a variety of Widgets, and an Internet Simulator to explore networks. This is a College Board-approved CSP course so it can be a substitute for a mathematics or science course in MassCore.

Technology requirements

- Desktop or laptop with Internet connectivity and up-to-date browser (one device per student).
- Internet connection of at least 15 MB/sec is recommended.

Professional development offered

- Code.org-trained and approved facilitators run local summer institutes for one week, with ongoing quarterly workshops throughout the school year.
- An online community forum with active forum monitors to facilitate discussion.

Costs

- The curriculum and software are free.
- Some materials are required, many of which may be considered regular classroom items that are readily available or easy to purchase at low cost. Additional costs may include printing and journals, if using paper versions.
- All professional development opportunities are currently available at no cost to educators, based on grants and contributions secured by Code.org.

CODEHS: SELECTED COMPUTER SCIENCE COURSES*

CodeHS
codehs.com/info/curriculum/pathways/hs

Support development of foundational computing principles and practices in various contexts.

Nature of curriculum	Stand-alone
Format and time	Full-year course (3 course options)
Key classroom routines	Video tutorials, programming exercises, projects
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	No
Programming platform or language	Karel (online block- and/or text-based environment), HTML, CSS, CodeHS editor (online), JavaScript, Processing.js
Web-based	Yes

Gr. 9–12 DLCS Topics	***Intro to CS with JavaScript* Alignment**	***CS Principles** Alignment*	***Intro to Cybersecurity* Alignment**
CAS.a. Safety & Security	X	X	✓
CAS.b. Ethics & Laws	X	X	*Partial*
CAS.c. Interpersonal & Societal Impact	X	*Partial*	X
DTC.a. Digital Tools	*Partial*	✓	*Partial*
DTC.b. Collaboration & Communication	X	✓	*Partial*
DTC.c. Research	X	*Partial*	*Partial*
CS.a. Computing Devices	X	X	X
CS.b. Human & Computer Partnerships	X	X	X
CS.c. Networks	X	*Partial*	✓
CS.d. Services	X	X	X
CT.a. Abstraction	✓	✓	X
CT.b. Algorithms	*Partial*	✓	X
CT.c. Data	*Partial*	*Partial*	X
CT.d. Programming & Development	✓	✓	X
CT.e. Modeling & Simulation	*Partial*	*Partial*	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

CodeHS Computer Science Courses

Overview
Three CodeHS courses provide different options and opportunities for students, and contribute to several computer science pathways. Each course is primarily web-based, and all units provide short video tutorials, example programs, quizzes, programming exercises, a culminating project, and unit tests. Additional assessment opportunities are provided through class discussions, open responses, and projects. Several supplemental materials and certain activities provide limited differentiation opportunities for students. Students write and run programs in the browser using the CodeHS editor.

Introduction to Computer Science with JavaScript (Golden) focuses on graphics, animations, and game design. The course engages students in programming using Karel (also available in Spanish), and basic JavaScript with an emphasis on graphics.

Computer Science Principles (CSP) uses a blended classroom approach with web-based and physical activities. Students write and run both blocks- and text-based JavaScript code in the browser using the CodeHS editor, Karel the Dog, and Processing.js, create websites and digital artifacts, and complete research projects. This is a College Board-approved CSP course so it can be a substitute for a mathematics or science course in MassCore.

Introduction to Cybersecurity (Vigenère) is an online blended course for students with some exposure to computer science. It engages students in foundational cybersecurity topics such as cyber hygiene, cryptography, software security, and networking, all through the CodeHS platform. Students complete projects, participate in simulated cyber attacks on safe sites in order to learn how to mitigate cyber attacks, learn basic SQL, and utilize basic HTML and JavaScript.

Technology requirements

- One computer per student (any OS) Internet connectivity and up-to-date browser.
- Online connectivity at a minimum of 10MB/sec.

Professional development offered

- Online and in-person professional development options are available

Costs

- Curriculum access for 100+ students is currently about $6,100 per year and $2,100 for 30 or fewer students.
- Online professional development is $1,750 per course for around 30 hrs. Online self-paced and in-person options are available.

COMPUTATIONAL THINKING AND PROBLEM SOLVING*

CSforMA, Inc
csforma.org/?page_id=7751

Students collaborate on projects using computational thinking and problem solving.

Nature of curriculum	Stand-alone
Format and time	Full-year course
Key classroom routines	Projects, videos, groupwork, unplugged activities, coding activities
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	Limited
Teacher content knowledge support	No
Cultural responsiveness	Limited
Programming platform or language	MIT App Inventor (online blocks-based environment), CyberCiege (cybersecurity gaming app)
Web-based	No

Gr. 9–12 DLCS Topics	***Computational Thinking and Problem Solving* Alignment**
CAS.a. Safety & Security	*Partial*
CAS.b. Ethics & Laws	✓
CAS.c. Interpersonal & Societal Impact	✓
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	✓
DTC.c. Research	✓
CS.a. Computing Devices	*Partial*
CS.b. Human & Computer Partnerships	✓
CS.c. Networks	✓
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	✓
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

COMPUTATIONAL THINKING AND PROBLEM SOLVING

Overview

The *Computational Thinking and Problem Solving* curriculum engages students in collaborative projects that employ computational thinking and problem-solving strategies. The course is designed to be a yearlong class for high school students who have digital literacy skills. Six projects are contextualized to be relevant to students and workplace needs. The projects include system design and proposal development, energy consumption and data analysis, website design and development, mobile application development, and privacy and security (cybersecurity). Instructional strategies are focused on teacher acting as a facilitator, providing scaffolding and guidance rather than direct instruction. The curriculum uses groupwork, discussions, unplugged activities, and projects. Assessments are provided through exit tickets and project rubrics. This course engages students in problem-based learning scenarios based on workplace needs, where students own the problem and create solutions that are unique to them and their work group. Such problems encourage students to consider multiple perspectives. There are limited tips for differentiation (primarily focused on encouraging group work) and pedagogical tips for the teacher.

Technology requirements

- Laptop or desktop with Internet connectivity and up-to-date browser.
- The curriculum expects the use of Microsoft application products, but Google suite or similar applications can be used.

Professional development offered

- CSforMA Professional Learning Institutes are offered over 5 days during the summer. Ongoing technical assistance for implementation is included.

Costs

- The curriculum and software are free (not including Microsoft applications).
- Some materials are required, many of which may be considered regular classroom or household items that are readily available or easy to purchase at low cost.
- Professional development workshops are $1,500 for Massachusetts teachers. Scholarships are available to educators at schools with diverse populations.

COMPUTING WITH ROBOTICS

UC Davis C-STEM Center; RoboBlockly
c-stem.ucdavis.edu/curriculum/

Help students move from block based coding to C++, with options for Arduino-based kits.

Nature of curriculum	Stand-alone or integrate with mathematics
Format and time	Course
Key classroom routines	Projects, coding, problem solving
Assessments, rubrics, examples	Yes
Differentiation supports	Yes
English Learner supports	Limited
Teacher content knowledge support	No
Cultural responsiveness	Limited
Programming platform or language	Blockly and C++; optional LinkBots (Arduino-based robots)
Web-based	Yes

Gr. 9–12 DLCS Topics	***Computing with Robotics* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	✓
DTC.c. Research	X
CS.a. Computing Devices	*Partial*
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	*Partial*
CT.b. Algorithms	*Partial*
CT.c. Data	*Partial*
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	✓

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

COMPUTING WITH ROBOTICS

Overview

This is a full year course for grades 10–12 that introduces students to robotics. Topics include introduction to programming, drawing, shapes, music, controlling a robot, math concepts and functions, graphics and image processing, and animations. The curriculum has 5 units with 62 chapters. Students can program virtual sprites, virtual robots, physical robots, and Arduino electronics kits. Students can control single and multiple robots which use sensors. The course could be taught without hardware but their inclusion does enhance the course. The course emphasizes hands-on robotics activities with a concentration on mathematical modeling and computer programming for solving problems in math. Math activities relate moving a robot to a graph. Some lessons integrate art and music and video production. There are competitions within the curriculum including robot competition and a video competition which emphasizes art and music. There are differentiated levels of activities and homework assignments. Students can start out in Blockly and work up to C++. There is instant feedback to students on coding and teachers can monitor student progress in a heat map. Audio and translation is available to students. Characters used for sprites provide some images of diversity.

Technology requirements

- RoboBlockly software
- Windows, Mac, or Chromebook.
- LinkBot robotics kits (using Arduinos) and Arduino starter kits are optional; students may use virtual robotics kits instead. If used, a maximum of 1 kit per 4 students is recommended.

Professional development offered

- 2.5 hour trainings are available.

Costs

- $1000 for high school site license with unlimited seats.
- RoboBlockly software is free.
- $300 for Linkbot Advanced Kit (Arduino based robot; barobo.com/product-page/linkbot-advanced-kit)
- $40 for Arduino compatible starter kit (barobo.com/product-page/linkbot-advanced-kit)
- Professional development is $150/person.

CYBER.ORG: SELECTED CYBER COURSES

Cyber.org
cyber.org/curricula-search

Prepare students for the cyber workforce and society with real-world applications and contexts.

Nature of curriculum	Stand-alone
Format and time	Full-year course (3 course options)
Key classroom routines	Slideshows, concept activities, challenge tasks
Assessments, rubrics, examples	Yes
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	No
Cultural responsiveness	Yes (*Cyber Society*)
Programming platform or language	BASIC IDE (text-based programming environment, specific app can vary), Parallax Boe-Bot or Shield-Bot or cyber:bot robots
Web-based	No

Gr. 9–12 DLCS Topics	***Cyber Literacy 2* Alignment**	***Cyber Society* Alignment**	***Cyber Science* Alignment**
CAS.a. Safety & Security	*Partial*	*Partial*	X
CAS.b. Ethics & Laws	*Partial*	✓	*Partial*
CAS.c. Interpersonal & Societal Impact	X	✓	*Partial*
DTC.a. Digital Tools	*Partial*	✓	X
DTC.b. Collaboration & Communication	*Partial*	*Partial*	X
DTC.c. Research	*Partial*	✓	X
CS.a. Computing Devices	✓	X	*Partial*
CS.b. Human & Computer Partnerships	X	X	X
CS.c. Networks	X	*Partial*	✓
CS.d. Services	X	X	X
CT.a. Abstraction	X	X	✓
CT.b. Algorithms	*Partial*	X	*Partial*
CT.c. Data	X	X	*Partial*
CT.d. Programming & Development	*Partial*	X	*Partial*
CT.e. Modeling & Simulation	X	X	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

CYBER.ORG: SELECTED CYBER COURSES

Overview

Cyber.org offers a wide range of cyber-focused courses; this summary includes three courses that can be implemented independently, or for a pathway using three courses in series. *Cyber Society* engages students in considering how technology is used and affects our everyday lives, in the workforce and society. Topics range from cyber law, ethics, terrorism, communications, and business. *Cyber Literacy 2* considers the cyber implications on liberal arts and blends in exploration of robotics and engineering. *Cyber Science* focuses on computer science fundamentals, again through the use of robotics and a humanities context. In each course, slideshows provide instructional background information with supporting visuals, students engage in activities to build an understanding of the concepts and are given challenges and more complex tasks to work through. Some additional challenges are provided for extension. Course assessments include tests, solution guides, and rubrics.

Technology requirements

- Desktop or laptop (Mac or PC) with Internet connectivity and up-to-date browser. 1 Mbps is sufficient for networking activities.
- *Cyber Literacy 2* and *Cyber Science* courses use the Parallax Boe-Bot (or Parallax Shield-Bot with Arduino or Parallax cyber:bot with micro:bit). One robot (and supply kit) per pair of students recommended.

Professional development offered

- Professional development is customizable from a few hours of content introduction to a four- or five-day workshop.
- Educators can engage with each other and Cyber.org staff through the learning management system.
- Cyber.org staff and subject-matter experts are available via email and telephone to assist with implementation and help customize content.

Costs

- Boe-Bots can be purchased for approximately $200 each.
- Supply kits for *Cyber Literacy 2* and *Cyber Science* are between $60–$120 each.
- Professional development is free for all first-time users of cyber.org curriculum.

EXPLORING COMPUTER SCIENCE*

University of Oregon
exploringcs.org

Weave inquiry and equity throughout exploration of computer science concepts.

Nature of curriculum	Stand-alone
Format and time	Full-year course
Key classroom routines	Small group work, reflections, inquiry, unplugged activities, coding activities
Assessments, rubrics, examples	Yes
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	Yes
Programming platform or language	Scratch (online blocks-based environment), HTML/CSS editor, Edison robots (optional) and EdScratch (online blocks-based environment)
Web-based	Yes

Gr. 9–12 DLCS Topics	***Exploring Computer Science* Alignment**
CAS.a. Safety & Security	✓
CAS.b. Ethics & Laws	*Partial*
CAS.c. Interpersonal & Societal Impact	✓
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	*Partial*
DTC.c. Research	✓
CS.a. Computing Devices	*Partial*
CS.b. Human & Computer Partnerships	✓
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	✓
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	✓

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

EXPLORING COMPUTER SCIENCE

Overview

Exploring Computer Science (ECS) is an introductory computer science curriculum designed around inquiry-based instruction and equity, which highlights societal impacts of computer science. There are no prerequisites. *ECS* is designed as a full-year course; if needed, units one through four can be taught as a semester course. Instructional strategies include think-pair-share activities, small-group work, journaling, and reflection. The curriculum uses a combination of unplugged and coding activities. ECS includes a variety of culturally-responsive lessons, including several lessons using *Culturally Situated Design Tools* such as Bead Loom and Cornrow Curves. There are 6 units in which the first four (required) cover a wide variety of computer science concepts. Schools can make a choice for the additional (optional) two units from four options: data analysis, physical computing (via Edison robotics), e-textiles, or artificial intelligence. (Note: the alignment provided in this guide assumes the use of the data analysis unit.) Rubrics are provided for most lessons (where applicable) and summative assessments are provided for each unit. Differentiation opportunities in the curriculum are limited to collaboration and potential personalization through reflections and some activities. Additional differentiation strategies and teacher content knowledge support is provided through professional development. Version 9.0 of the curriculum is available in Spanish. *ECS* can be a substitute for a mathematics or science course in MassCore.

Technology requirements

- Desktop or laptop (any OS) with Internet connectivity and up-to-date browser.
- Edison robot (optional) for Unit 6.
- Scratch can be accessed online. A downloadable version of Scratch is available.

Professional development offered

- Professional development for educators beginning with *ECS* is highly recommended.
- A two-year sequenced series of courses is available: A one-week summer institute, four unit-specific workshops, and a second week-long summer institute.
- An online option is available for selected workshops.
- A variety of online communities are available.

Costs

- The curriculum and software are free.
- Professional development is free for beginning educators. District-based PD would hire facilitators (two for groups greater than 15).
- Edison robot (optional Unit 6) can be purchased online for about $50 apiece, or in quantities up to 30 for about $1,000. Legos are also recommended for this unit.
- Student journals are recommended.

FIRST TECH CHALLENGE: ROBOTICS ENGINEERING*

FIRST Robotics
info.firstinspires.org/curriculum

A PBL curriculum using robotics to address real world problem solving and community impact.

Nature of curriculum	Stand-alone
Format and time	Full year course (180 hours)
Key classroom routines	Projects, collaboration, reflection, peer assessment, presentations
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	No
Cultural responsiveness	Yes
Programming platform or language	Blockly and Java; CAD software; REV robots
Web-based	Yes

Gr. 9–12 DLCS Topics	***FIRST Tech Challenge* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	*Partial*
DTC.a. Digital Tools	*Partial*
DTC.b. Collaboration & Communication	✓
DTC.c. Research	✓
CS.a. Computing Devices	✓
CS.b. Human & Computer Partnerships	*Partial*
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	*Partial*
CT.c. Data	*Partial*
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	*Partial*

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

FIRST TECH CHALLENGE: ROBOTICS ENGINEERING

Overview

In this curriculum students design, build, and code a robot to solve a challenge or game. A strong emphasis is placed on developing teamwork and collaboration skills. Students model a robot with an online simulation in addition to working with the physical robot. They keep engineering notebooks to document their work, and engage in a final capstone project. The students must do research for this final project and it is encouraged that the project contributes to a positive impact for the community. Students develop a portfolio to present the work from the capstone project. Students may use a variety of software in ways similar to professional team settings, including CAD software and project management software. Students use the CAD software to design a robot arm and then run a simulation to test the actions of the robot. There are two portfolio-based summative assessments, and the curriculum contains rubrics for assessing the engineering notebook, career readiness, and robot design. There is also a career practice rubric and an emphasis on 'core values.' There are 11 units with teacher planning support & lesson guides, each with a slide deck for student-guided learning. As a project-based learning curriculum, students are supported to pace themselves and work through productive struggle. The curriculum emphasizes the value in allowing students to make mistakes and redesign their work. Students work to develop technical skills, document the design process in an engineering notebook, and present their designs. Students also self-assess to set improvement goals for themselves based on their skill level. Social Emotional Learning practices and Social Awareness are built into the Career Ready Practices rubric and the Community Project, with an emphasis on Gracious Professionalism, Coopertition, and Relationships. There are three explicit modules in the course on equity, diversity, and inclusion, one of which is on discrimination and bias.

Technology requirements

- REV Robotics Education Kit (revrobotics.com/rev-45-1517/)
- Computers able to run CAD and programming software (such as with a graphics card with 1GB of VRAM)
- CAD Design Software
- Some kind of collaboration software or apps

Professional development offered

- A 14-hour PD is available remotely or in person in Manchester, NH.
- PD is highly recommended if new to Project Based Learning.

Costs

- Digital access to the curriculum and a mini-game field for field elements is $900.
- REV Robotic Edu Kits are $775 for a robot kit for 3 students; Robot Upgrade kit is $300.
- PD is $500 per teacher.

LOCOXTREME

LocoRobo
locorobo.co/locoxtreme-for-teachers.html

A physical computing curriculum using ground robots.

Nature of curriculum	Stand-alone
Format and time	Full year course
Key classroom routines	Videos, Coding Activities, Quizzes, Challenges
Assessments, rubrics, examples	Yes
Differentiation supports	No
English Learner supports	No
Teacher content knowledge support	No
Cultural responsiveness	No
Programming platform or language	Python; LocoXtreme Robots
Web-based	Yes

Gr. 9–12 DLCS Topics	*LocoXtreme* Alignment
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	*Partial*
DTC.b. Collaboration & Communication	X
DTC.c. Research	X
CS.a. Computing Devices	✓
CS.b. Human & Computer Partnerships	*Partial*
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	*Partial*
CT.b. Algorithms	*Partial*
CT.c. Data	*Partial*
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

LocoXtreme

Overview

The *LocoXtreme* (ground robot) curriculum is a step-by-step introduction to physical computing with Python, including lessons for all LocoXtreme sensors and actuators and all basic Python programming features, including data types. The curriculum includes 64 step-by-step lessons that include video instruction, quiz worksheets, coding activities to use the robot, and 'artifact' demonstrations; there are no unplugged activities. The curriculum is web-based (with Bluetooth connections to some physical devices) and includes a learning management system and gradebook. The LocoXtreme ground robot has dual motors, LED ring, and piezoelectric actuators and ultrasonic, accelerometer, gyroscope, shaft encoder, and temperature sensors. Assessments include per-lesson vocabulary, concept, and code quiz multiple-choice worksheets. Occasional challenge lessons synthesize previous concepts, with the bulk of them coming towards the end of the course. LocoRobo offers additional physical computing options as well.

Technology requirements

- One computer and LocoXtreme robot per two students. Robots connect to computers through USB tethers (or custom Bluetooth dongles).
- Accounts that access the curriculum and learning management system are linked to login credentials for specific users.

Professional development offered

- LocoRobo offers seminar options ranging from hourly on-line to full-day onsite.
- LocoRobo provides live support technicians available by e-mail and voice.

Costs

- Curriculum materials are paired with devices @ $515 for 1 LocoXtreme ground robot (with custom Bluetooth dongle) + 2 lifetime LocoRobo Academy Voyager licenses. Lifetime licenses with access to the learning management system are permanent, no future renewals are required, and all licenses are transferable between students for new quarters, trimesters, semesters, and years.
- Additional LocoXtreme Voyager logins allowing students to share robots with access to individual curricula are $100.
- Hourly online PD is $200 per hour; full day onsite PD is $1800.

MOBILE COMPUTER SCIENCE PRINCIPLES*

The Mobile CSP Project
course.mobilecsp.org

Engage in computer science by building socially useful mobile apps.

Nature of curriculum	Stand-alone
Format and time	Full-year course
Key classroom routines	Collaborative programming, projects, group inquiry
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	Limited
Teacher content knowledge support	Yes
Cultural responsiveness	Limited
Programming platform or language	MIT App Inventor (online blocks-based environment)
Web-based	Yes

Gr. 9–12 DLCS Topics	***Mobile CSP* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	*Partial*
CAS.c. Interpersonal & Societal Impact	✓
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	✓
DTC.c. Research	*Partial*
CS.a. Computing Devices	*Partial*
CS.b. Human & Computer Partnerships	*Partial*
CS.c. Networks	✓
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	✓
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	*Partial*

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

MOBILE COMPUTER SCIENCE PRINCIPLES

Overview

Mobile Computer Science Principles is designed to engage students by building socially useful mobile apps. Students learning programming and computer science principles through a project-based and process-oriented group-inquiry learning (POGIL) approach. The course uses MIT App Inventor as the main creation and coding platform, resulting in apps that can be used on mobile devices. While the course can be run entirely online, Android devices are recommended for testing student apps. Students develop a portfolio of their work through the course. Assessments include performance tasks, rubrics and example solutions, quizzes, and midterm and final exams. Approximately half of the course lessons are focused on app-building. Programming activities are structured in a three-step, pair-programming model, including a hands-on tutorial, a set of problem-solving tasks and programming challenges, and student app development or enhancements to an existing app. Some English Language needs have been addressed through purposeful attention to vocabulary, color coded tables, and reading levels. Teacher lesson plans include suggestions for differentiation (primarily limited to practice and enrichment) and teacher content knowledge. This is a College Board-approved CSP course so it can be a substitute for a mathematics or science course in MassCore.

Technology requirements

- Desktop or laptop with reliable Internet connectivity and up-to-date browser.
- Mobile devices (can use an emulator). One device for every two students is recommended. Android and iOS devices are supported.

Professional development offered

- Immersion PD is a 50-hour course (including out-of-class assignments) recommended for teachers who have significant background in computer science.
- Extended PD is 90–100 hours, equivalent to taking a college-level CS Principles course, recommended for teachers with no prior education or teaching experience in computer science.
- *Mobile Computer Science Principles* teachers are supported by a large, active online community. Support during the school year also includes monthly webinars.

Costs

- The curriculum, teacher materials, and software are free.
- Professional development fees are $1,750 per teacher for the Immersion PD and $3,000 per teacher for the Extended PD.

NEUROMAKER MODULES (9-12)

BrainCo, Inc.
neuromakerstem.com/courses/

Biomedical-based project kits to engage students in helping others through design and coding.

Nature of curriculum	Stand-alone
Format and time	2 Units (12 or 25 lessons per module)
Key classroom routines	PBL, research, writing, coding, design
Assessments, rubrics, examples	Yes
Differentiation supports	Yes
English Learner supports	No
Teacher content knowledge support	Yes
Cultural responsiveness	Limited
Programming platform or language	Python, Jupyter
Web-based	No

Gr. 9–12 DLCS Topics	***BCI Classroom Module* Alignment**	***Applied AI Module* Alignment**
CAS.a. Safety & Security	X	X
CAS.b. Ethics & Laws	✓	X
CAS.c. Interpersonal & Societal Impact	*Partial*	*Partial*
DTC.a. Digital Tools	*Partial*	X
DTC.b. Collaboration & Communication	X	X
DTC.c. Research	*Partial*	*Partial*
CS.a. Computing Devices	*Partial*	✓
CS.b. Human & Computer Partnerships	✓	✓
CS.c. Networks	X	X
CS.d. Services	X	X
CT.a. Abstraction	*Partial*	*Partial*
CT.b. Algorithms	*Partial*	*Partial*
CT.c. Data	✓	*Partial*
CT.d. Programming & Development	*Partial*	*Partial*
CT.e. Modeling & Simulation	✓	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

NEUROMAKER MODULES

Overview
The two modules included in this review provide opportunities for physical computing with a hands-on approach to learning. Each module has an online teacher's guide with teacher background knowledge, classroom routines, and tech tips. Each module uses a physical kit that students program and modify.

In *BCI Classroom,* students code AI in Python to analyze brain waves collected with a Brain Computer Interface, hardware that measures electrical signals from the brain, and use this to control tasks such as video games. This module includes 25 lessons of 45–60 minutes each. Students learn to collect and analyze a dataset using Python and Jupyter notebooks. Students use signal processing to analyze data sets such as sound files. They then combine signal processing with AI to filter and interpret brain waves. All students do some coding, research into ethical issues, and consider the design of the hardware. Three different pathways are then available for differentiation or to allow students to specialize. One path specializes in exploring ethics, in which students write a science fiction story exploring the societal issues behind a Brain Computer Interface. The second pathway focuses on the design of hardware and software to solve a societal problem using BCI technology. Students in the third pathway code and train an AI to analyze BCI data to solve a problem. All students have a final presentation on their work for assessment. A teacher guide with sample solutions is provided.

Applied Artificial Intelligence contains 12 lessons (50 minutes each) where students learn about AI and use it to control a robotic prosthetic hand to explore the 5 Big Ideas of AI. Students need some programming skills before doing this module and need access to an assembled hand (see kit below). This module includes an emphasis on how AI works and related ethical issues, including a discussion of bias, identity, and responsibility in Artificial Intelligence. Students also learn to code the hand to use sign language. Students explore machine learning and train the AI on data sets, and discuss training an AI for differences in values, cultures, race, and ethnic groups. There are three real world AI challenges included.

Technology requirements

- Either the Neuromaker Hand kit or the Brain Computer Interface kit.
- PC for interface with BCI.
- Prosthetic Hand kit includes an Arduino, so either a PC or Mac to interface with the kit.

Professional development offered

- Virtual PD is available by the hour.

Costs

- Curriculum itself is free.
- The kit for Neuromaker Hand is $500; the kit for Brain Computer Interface is $400.
- PD is $500/hour.

PLTW COMPUTER SCIENCE*

Project Lead the Way
pltw.org/our-programs/pltw-computer-science

Three courses to build computational thinking skills using a wide variety of computing tools.

Nature of curriculum	Stand-alone
Format and time	Full-year course (3 course options)
Key classroom routines	Activity, project, and problem structure
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	Yes
Cultural responsiveness	Limited
Programming platform or language	MIT App Inventor (online blocks-based environment), Visual Studio Code (text-based Python environment), AWS Cloud9, VEXcode V5 Blocks (online block-based environment), Trinket (online Python editor), NetLogo (online modeling environment); VEX vehicle
Web-based	CSE: Yes; CSP: Yes; Cybersecurity: Yes

Gr. 9–12 DLCS Topics	***PLTW CS Essentials* Alignment**	***PLTW CS Principles** Alignment**	***PLTW Cybersecurity* Alignment**
CAS.a. Safety & Security	X	X	X
CAS.b. Ethics & Laws	*Partial*	*Partial*	*Partial*
CAS.c. Interpersonal & Societal Impact	X	*Partial*	*Partial*
DTC.a. Digital Tools	*Partial*	✓	✓
DTC.b. Collaboration & Communication	✓	✓	✓
DTC.c. Research	X	✓	*Partial*
CS.a. Computing Devices	X	✓	✓
CS.b. Human & Computer Partnerships	*Partial*	*Partial*	✓
CS.c. Networks	X	✓	✓
CS.d. Services	X	X	*Partial*
CT.a. Abstraction	✓	✓	*Partial*
CT.b. Algorithms	*Partial*	✓	X
CT.c. Data	X	✓	X
CT.d. Programming & Development	✓	✓	X
CT.e. Modeling & Simulation	*Partial*	✓	X

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

PLTW COMPUTER SCIENCE

Overview

PLTW Computer Science include three full-year courses: *Computer Science Essentials (CSE)*, *Computer Science Principles (CSP)*, and *Cybersecurity*. Course units use an activity-, project-, and problem-based instructional approach, and each unit culminates in an application problem. Lesson assessments are provided, along with project rubrics and guides. Students maintain a computer science notebook through each course. Limited differentiation strategies are outlined in teacher materials and opportunities for student choice in some projects. Teacher content knowledge support is found in teacher guides and professional development. *CSE* helps students to develop fundamental computer science concepts and skills using block- and text-based programming. This course does not require any prerequisite computing experience. Student projects include creating apps and developing websites. *CSE* uses a variety of platforms, including MIT App Inventor, VEXcode V5 Blocks, and AWS Cloud9, and involves some programming in Python (using Trinket). *CSP* develops computational thinking principles related to career paths that use computing. Students develop Python coding skills using Trinket, before advancing to Vex Coding Studio. They model with NetLogo and explore the workings of the Internet with AWS Cloud9. This is a College Board-approved CSP course so it can be a substitute for a mathematics or science course in MassCore. *Cybersecurity* engages students with a broad range of digital and information security, including social media, networks, hacking prevention, and e-commerce. The curriculum emphasizes socially responsible choices and ethical behavior. Most work is done in PLTW Security Labs via virtual labs using a browser.

Technology requirements

- Desktop or laptop (Mac or Windows), with Internet and up-to-date browser.
- For CSE: Android tablet and VEX Self-Driving Vehicle (one per two students).

Professional development offered

- A 40-hour professional development course is offered in person or online, led by PLTW Master Teachers. Trainings addresses unit content, pedagogy, and facilitation support. Subsequent resources and communities include post-training webinars, online asynchronous training, and unit-specific resources.
- Additional supports include email or phone support; an online community of teachers, PLTW Launch teachers and Master Teachers; and opportunities to provide feedback to PLTW.

Costs

- All software used in PLTW Computer Science courses are free.
- Materials are needed when starting a CSP course (approximately $2,390; includes Vernier Sensors) or a CSE course (about $4,000; includes VEX vehicles). Some consumable materials are also needed, about $75 per class per year.
- The fee for schools to participate in PLTW Computer Science is $2,000 per year, which covers participation in all three courses for a school. Schools may also apply for funding and grants opportunities through the PLTW Grant program.
- Core training opportunities for beginning teachers are $2,400 per teacher, per unit.

SCIENCE+C

EDC

scienceplusc.org

Computational modeling and simulation for Biology, Chemistry, and Physics courses.

Nature of curriculum	Integrated
Format and time	6 to 8 supplemental units per course
Key classroom routines	Simulation, Modeling, Claims-Evidence-Reasoning, Using, Decoding, Modifying
Assessments, rubrics, examples	Yes
Differentiation supports	Yes
English Learner supports	No
Teacher content knowledge support	Limited
Cultural responsiveness	No
Programming platform or language	NetLogo Web (online text-based agent-based modeling environment)
Web-based	Yes

Gr. 9–12 DLCS Topics	***Science+C* Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	*Partial*
DTC.b. Collaboration & Communication	*Partial*
DTC.c. Research	*Partial*
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	X
CS.d. Services	X
CT.a. Abstraction	✓
CT.b. Algorithms	✓
CT.c. Data	*Partial*
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	✓

Key: ✓ substantial alignment; *Partial* alignment; X no alignment

SCIENCE+C

Overview

Science+C provides focused units to engage students in computational modeling and simulation of scientific systems related to a variety of topics in Biology, Chemistry, and Physics. There are 7-10 units for each discipline that can be integrated into an existing science curriculum. Using NetLogo Web, students explore science phenomena related to topics such as ecosystem dynamics, enzyme reactions, kinetics, REDOX reactions, kinematics, and simple harmonic motion. Each unit assumes prior knowledge of science concepts that can be coordinated with instruction in typical science courses. Each unit includes a pre-built model in NetLogo and explore the model through three lessons organized in a Use, Decode, and Modify sequence which is structured to reflect a 5E instructional model. All units engage students in coding simulations to model and experiment with a specific system, and to develop a claims-evidence-reasoning (CER) explanation about the phenomenon. Each unit starts with a career connection, includes formative assessment tips, exit tickets, and coding examples, and provides prerequisite concepts and common student misconceptions among the student handouts.

Technology requirements

- Desktop or laptop computer running any major OS or iPad.
- Browser to run NetLogo Web (does not require Flash).
- Internet bandwidth supporting downloads of 5 megabits per student.

Professional development offered

- Summer PD is available.

Costs

- The curriculum and software are provided at no cost to schools.
- Grant funding for PD may be available; contact EDC for details.

APPENDIX

REVIEW PROCESS

The review process was undertaken in three phases:

1. Identified a wide range of digital literacy and computer science educational products.
2. For each product, identified whether there is teacher-focused curriculum (vs. tutorials or aggregated lessons) and conducted initial standards alignment to determine scope.
    a. This filtered out a large number of programming courses, robotics products, CAD software and such that offered tutorials on how to set up or program the robot or software, or that focused on coding language and syntax without an emphasis on computational thinking principles.
    b. For those that had a curriculum, ensured substantial coverage of at least one topic of the MA DLCS standards.
    c. Resources eliminated during this phase could be considered as supplemental resources for DLCS instruction; that analysis is beyond the scope of this review.
3. For those curricula that showed promise in the first phase, a more thorough review of each curriculum was conducted using the criteria below. Curricula were chosen to ensure a variety of educational approaches are available at each grade span, to ensure options for pairing curricula to achieve substantive DLCS standards coverage, and to ensure no-cost options.
4. A description of each curriculum was then written. The draft description was sent to the curriculum provider to ensure accuracy and obtain additional information about context such as technology requirements, costs, and professional development availability.

REVIEW CRITERIA

Criteria for evaluation of each DLCS curriculum:

- Nature of curriculum
    - Integrated: designed to be, or strong potential to be, integrated with another core academic discipline such as mathematics, science, or literacy
    - Stand-alone: designed to be offered as a curriculum on its own
- Scope of curriculum
    - Module: a set of related materials on a theme or topic; not designed as a unified curriculum per se but coherent when used together or structured as a unit (e.g., thematic lesson set)
    - Unit: intended to be a coherent curriculum focused on a single theme or topic
    - Course: intended to be a coherent quarter, semester, year-long, or multi-year course or sequence covering multiple topics or including multiple units
- Standards alignment
    - Alignment of the learning objectives to the MA DLCS standards (as relevant to the scope of the curriculum)
        - To be considered aligned to a particular standard, the curriculum must be explicit about learning goals as well as instruction or student learning opportunities relevant to that standard.
        - Reviewers exercised professional judgement to determine if a curriculum exhibited “partial” or “substantial” alignment to standards.

    - Report coverage and gaps
    - Based on the individual standards alignment, summarize topical coverage
        - "Partial" coverage when the curriculum addressed between 33% and 65% of the standards in the topic (to determine percentage, partially aligned standards counted as 0.5, substantially aligned counted as 1.0)
        - "Substantial" coverage when the curriculum addressed more than 66% of the standards in the topic
- Teacher usability (drawn from MA CURATE Project review rubric; www.doe.mass.edu/instruction/curate/resources.html). Any determinations of "None," "Limited," or "Substantial" were professional judgements of the reviewers, with discussion among all reviewers to calibrate for consistency.
    - Classroom routines or Approach to instruction. Briefly describe the instructional design (e.g., lecture, online videos, scaffolded student tutorial, individual vs. group work, unplugged activities, reflections).
    - Assessments, rubrics, and examples (None, Limited, Yes). Includes both informal and formal assessments, with rubrics, exemplars, or other resources to define expectations of student performance.
    - Differentiation (None, Limited, Yes). Includes explicit or clearly identifiable strategies in the curriculum such as designed student choice, multiple opportunities for accessing the same content or demonstrating learning, helping teachers meet the diverse needs of students with disabilities or those not at grade level.
    - English Language supports (None, Limited, Yes). Includes explicit articulation in the curriculum of strategies for the development of academic language in English.
    - Support for teachers' subject matter knowledge (None, Limited, Yes). Includes explicit support within the curriculum or teacher materials to help teachers understand core concepts, typical student thinking of concepts, or pedagogical strategies to engage students in the concepts.
    - Cultural responsiveness. Presence of one or more of the following sections, from the *Culturally Responsive Scorecard* (research.steinhardt.nyu.edu/scmsAdmin/media/users/atn293/ejroc/CRE-Rubric-2018-190211.pdf)
        - Diversity of characters
        - Accurate portrayals
        - Decolonization/Power and privilege
        - Centering multiple perspectives
        - Connecting learning to real life and action
    - Web-based (Yes, No, Optional). Whether the curriculum can be done without applications on local drives (can the curriculum be done, for example, on a Chromebook). This does not include considerations for robots or development boards which are used in person but distinct from curriculum access.
- Programming language or platform. Specify what programming language or platform is used, if any, along with robot or development board, if any.
- Technology requirements
    - Minimum computer specs, if any
    - Accessories such as robots

    - Special software
- Professional development offered. Describe the range, if any, of PD offered.
- Cost (ensure at least one free option at each grade span)
    - Describe the basic costs of the program, including whether the curriculum is free, low cost, subscription cost, or purchase cost. Include curriculum and professional development costs.

LIMITATIONS OF THIS REVIEW

It will be helpful to recognize a number of limitations that may have influenced aspects of the curriculum review process.

- The first review of curricula took place between March–June, 2020; additional review was conducted between March and May, 2021; the most recent additional review occurred between October, 2021 and February, 2022. DLCS-related curricula constantly change. Always check for the most recent versions and consider any significant changes that may have been made since this review.
- The review team knew some curricula better than others, having participated in professional development on a number of them or used a number of them in their own practice. The team worked to comprehensively and thoughtfully investigate all curricula as equally as possible, including applying as fresh a perspective as possible to ones previously experienced, and treating new options with equal consideration. This included asking for input of other team members who may not have had direct experience with a program. In selected cases the team reached out to curriculum providers directly to better understand the design and use of particular curricula.
- The review team completed only a brief review of a wide variety of resources that did not have coherent curriculum materials and are not included in this Guide. These resources could, however, be excellent supplements to a DLCS curriculum. These resources include many robotics kits, programmable boards, design software applications (e.g., CAD-related apps), and digital equipment (e.g., 3D printers, vinyl cutters) that could be effective supplements to enrich student engagement in DLCS. Some curricula included in this report make use of such options, but only when they are integrated into a coherent curriculum.
- The review team also did not include a wide variety of curricula that were focused primarily on learning a programming language and syntax. These options typically addressed a few Computational Thinking (CT) standards, such as conditionals and looping, but they did not provide a broader CT principles framework called for in the standards.
- The review did not include analysis of treatment of student data or information by the curriculum organization (per the Family Educational Rights and Privacy Act Policy [“FERPA”] or U.S. Children’s Online Privacy Protection Act [“COPPA”]).

MA DLCS STRANDS, TOPICS, AND PRACTICES

Digital Literacy and Computer Science (DLCS) Overview

The standards for Kindergarten to grade12 are organized by **grade span**: Kindergarten to grade 2, grade 3 to grade 5, grade 6 to grade 8, and grade 9 to grade 12. Within each grade span, standards are grouped in four **strands**: Computing and Society, Digital Tools and Collaboration, Computing Systems, and Computational Thinking. Each strand is further subdivided into **topics** comprised of related **standards**. Standards define performance expectations, as well as what students should know and be able to do. Standards from different strands or topics may sometimes be closely related. Standards in every grade span and strand demonstrate a range of cognitive complexity such as reflected in Bloom's Revised Taxonomy: remembering, understanding, applying, analyzing, evaluating, and creating.[1]

Vision
Digital Literacy and Computer Science (DLCS) knowledge, reasoning, and skills are essential both to prepare students for personal and civic efficacy in the twenty-first century and to prepare and inspire a much larger and more diverse number of students to pursue the innovative and creative careers of the future. The abilities to effectively use and create technology to solve complex problems are the new and essential literacy skills of the twenty-first century.

Learning Progression

Grade Spans	Strands			
K-2	**CAS: Computing and Society** a. Safety and Security b. Ethics and Laws c. Interpersonal and Societal Impact	**DTC: Digital Tools and Collaboration** a. Digital Tools b. Collaboration and Communication c. Research	**CS: Computing Systems** a. Computing Devices b. Human and Computer Partnerships c. Networks d. Services	**CT: Computational Thinking** a. Abstraction b. Algorithms c. Data d. Programming and Development e. Modeling and Simulation
3-5				
6-8				
9-12				

Practices: Connecting, Creating, Abstracting, Analyzing, Communicating, Collaborating, Research

[1] Anderson, L.W. (Ed.), Krathwohl, D.R. (Ed.), Airasian, P.W., Cruikshank, K.A., Mayer, R.E., Pintrich, P.R., Raths, J., & Wittrock, M.C. (2001). *A taxonomy for learning, teaching, and assessing: A revision of Bloom's Taxonomy of Educational Objectives* (Complete edition). New York: Longman.

Page 7 of the *2016 Massachusetts Digital Literacy and Computer Science Framework.*

DLCS to CSTA Crosswalk

The Computer Science Teachers Association (CSTA) has developed national computer science education standards. To see alignment between the Massachusetts Digital Literacy and Computer Science standards and the CSTA standards, refer to the downloadable Excel file DLCS to CSTA Standards Crosswalk.

EXAMPLE APPLICATION OF K–12 DLCS PATHWAYS GUIDANCE

The following sections use two curricula per grade span to illustrate an analysis of the progression from one curriculum to another across grade spans. Using the pathways guidance elements from above, comments are made about how each curriculum relates to (progresses from) the two curricula in the prior grade span (starting with grades 3–5, since there is no progression prior to K–2). This exercise can be done with any DLCS curricula; we have chosen 2 examples from each grade span for illustration purposes only.

- Grades K–2
  - Code.org K-2 CS Fundamentals
  - Springfield K-2 CSforALL
- Grades 3–5
  - Code.org 3-5 CS Fundamentals
  - Springfield 3-5 CSforALL
- Grades 6–8
  - Code.org CS Discoveries
  - CodeHS Computing Ideas
- Grades 9–12
  - Computational Thinking and Problem Solving (CTPS)
  - Exploring Computer Science (ECS)

Code.org CS Fundamentals (grades 3–5)

Program Goal	• Designed to achieve coverage of a variety of grade 3–5 DLCS topics and standards. • Designed to be a stand-alone curriculum, with some flexibility for different contexts.	
Description	• Aligns to a variety of grade 3–5 DLCS standards, with a concentration on computational thinking topics. Some additional digital literacy and computing systems lessons may be needed to supplement this curriculum • Students exclusively use Blockly, an online, block-based programming language.	
Transitioning from:	**Code.org K–2 CS Fundamentals**	**Springfield K–2 CSforALL**
Purposeful Progression	• This curriculum follows the same structure as K–2 CS Fundamentals. Students in both grade spans are engaged in interactive games, stories, and coding using the same platform. There is not a physical computing element to either grade span curriculum.	• The Springfield K–2 curriculum employs block-based programming, consistent with the block-based approach of the 3–5 CS Fundamentals curriculum. The Springfield K-2 curriculum does include physical computing, but that is not a detriment to transitioning to 3–5 CS Fundamentals.

Pedagogical Coherence	• The 3–5 CS Fundamentals curriculum is a direct progression from the K–2 curriculum. As with K–2, this curriculum is designed to be a stand-alone curriculum with some flexibility for other contexts, and uses the same coding platform and pedagogical strategies, including a context-setting activity, skill building, online coding, interactive games or stories, and collaborative work.	• The Springfield K–2 CSforALL curriculum includes some similar pedagogical strategies as the 3–5 CS Fundamentals curriculum. The Springfield curriculum includes group projects, independent student work, partner work, coding activities, turn and talk activities, unplugged algorithm development, and student reflections.
Equity	• The K–2 CS Fundamentals curriculum is similar to grade 3–5, with limited attention to equity through some pre-read strategies and occasional attention to perspective or bias in relation to social impacts of media use. • Differentiation supports in both K–2 and 3–5 CS Fundamentals are limited to selected activities in which teachers can personalize for individual students.	• The treatment of equity in the Springfield K–2 curriculum includes diverse authors in text choices and character and professional representation. An occasional project is contextualized in students' home life or culture. This represents a different approach to equity than in 3–5 CS Fundamentals. • Springfield K–2 CSforALL includes significantly more support for differentiation, with specific suggestions for many lessons, than does 3–5 CS Fundamentals, which provides only articulated occasional differentiation opportunities.

Springfield CSforALL (grades 3–5)

Program Goal	• Designed to achieve substantial coverage of grade 3–5 DLCS topics and standards. • Designed to be integrated into core subjects (ELA, mathematics, science, and social studies). • Aims to reflect and engage diverse student populations in DLCS.	
Description	• Aligns to a substantial portion of grade 3–5 DLCS standards. Every module has at least one digital citizenship standard embedded into the lessons, however, some additional digital literacy lessons may be needed to supplement this curriculum. • Students engage in block-based programming through a variety of coding platforms. • Students work with Scratch in the context of using several physical computing devices, including Makey Makey, micro:bit, and Hummingbird Robotics kits.	
Transitioning from:	**Code.org K–2 CS Fundamentals**	**Springfield K–2 CSforALL**

Purposeful Progression	• The K–2 CS Fundamentals curriculum focuses on block-based programming using Blockly, supporting a progression to Springfield 3–5 CSforALL. There is not, however, a physical computing element in K–2 CS Fundamentals. The Springfield 3–5 CSforALL curriculum assumes students will have some prior experience with Makey Makey, so they may need additional instruction and support prior to lessons using a Makey Makey device.	• This curriculum follows the same structure as K–2. Coding and physical computing platforms are coordinated for a purposeful progression from K–2 to 3–5. In K–2, Kindergarten curriculum uses Beebots, 1st grade uses Codespark and Scratch, and 2nd grade uses Scratch and Makey Makey, all employing block-based programming.
Pedagogical Coherence	• The K–2 CS Fundamentals curriculum can be, but is not specifically designed for, integration into other academic disciplines as the Springfield 3–5 curriculum is. • The K–2 CS Fundamentals curriculum uses some similar pedagogical strategies, including a context-setting activity (such as unplugged, hands-on activities), skill building, online coding, interactive games or stories, and collaborative work.	• The 3–5 curriculum is a natural progression from the K–2 curriculum. As with K–2, this curriculum is designed to be integrated with other academic disciplines. Pedagogical strategies are also consistent with K–2, including independent work, partner work, and group projects, coding activities, turn and talk activities, unplugged algorithm development, and student reflections.
Equity	• The K–2 CS Fundamentals curriculum is limited in attention to equity, relying on some pre-read strategies and occasional attention to perspective or bias in relation to social impacts of media use. This is a different approach to addressing equity than is taken in the 3–5 CSforALL curriculum which includes diverse authors in text choices and character or professional representation. • Springfield 3–5 CSforALL includes significantly more support for differentiation, with specific suggestions for many lessons, than does K–2 CS Fundamentals, which provides only occasional differentiation opportunities to personalize certain student activities.	• The treatment of equity in the Springfield K–2 curriculum is of a similar design as 3–5, relying on inclusion of diverse authors in text choices and character or professional representation. As with K–2, an occasional project is contextualized in students' home life or culture. • Differentiation supports in Springfield K–2 and 3–5 CSforALL are similar, with specific differentiation tips included in many lessons across the curricula.

Code.org CS Discoveries (grades 6–8)

Program Goal	• Designed to achieve coverage of a variety of grade 6–8 DLCS topics and standards. • Designed to be a stand-alone curriculum.	
Description	• Aligns to a variety of grade 6–8 DLCS standards, with a concentration on computational thinking topics. Some additional digital literacy and computing systems lessons may be needed to supplement this curriculum • Students use both block- and text-based programming environments, and use a Circuit Playground development board.	
Transitioning from:	**Code.org 3–5 CS Fundamentals**	**Springfield 3–5 CSforALL**
Purposeful Progression	• CS Discoveries is designed to progress from CS Fundamentals, including moving students from block-based programming to text-based programming, using GameLab and AppLab, as well as a physical computing option with Circuit Playground.	• The Springfield 3–5 CSforALL curriculum engages students in block-based programming using Scratch, with application using Makey Makey, micro:bit, and Hummingbird Kits. The CS Discoveries curriculum builds on those experiences to support student movement from block-based programming to text-based programming, using GameLab and AppLab, as well as a physical computing option with Circuit Playground.
Pedagogical Coherence	• CS Discoveries provides a consistent progression from CS Fundamentals. Similar strategies for lesson design and student engagement are included, such as student activities, small group discussions, and reflections. CS Discoveries is designed to be a stand-alone curriculum with some flexibility for other contexts, similar to CS Fundamentals. • Ideally units are done in order, however several could be skipped or rearranged if needed. The Artificial Intelligence and Machine Learning unit is optional; this could be used with older middle school students who have finished some of the more in-depth CS Discovery units.	• Students in Springfield 3–5 CSforALL engaged with DLCS instruction in the context of core subjects (ELA, mathematics, science, and social studies). CS Discoveries is a stand-alone course, although it does provide some flexibility to be integrated into other contexts. • Overall pedagogical strategies of CS Discoveries are consistent with Springfield 3–5 CSforALL, including group projects, partner work, unplugged and coding activities, and student reflections.
Equity	• CS Discoveries expands upon its attention to equity from CS Fundamentals, with some projects encouraging students to consider problems of the world, including how to help people in their own community. One project engages students in how to evaluate a dataset and model to avoid unintended bias. CS Discoveries includes several videos showing diverse students and adults in CS.	• The treatment of equity in the Springfield 3–5 curriculum relies on inclusion of diverse authors in text choices and character or professional representation, with an occasional project contextualized in students' home life or culture. CS Discoveries also contextualizes several projects by encouraging students to consider problems of the world, including

	• Both CS Fundamentals and CS Discoveries provide only occasional differentiation strategies, primarily through potential personalization of selected student activities.	how to help people in their own community. • Springfield 3–5 CSforALL includes significantly more support for differentiation, with specific suggestions for many lessons, than does CS Discoveries, which provides only occasional differentiation strategies, primarily through potential personalization of student activities.

CodeHS Computing Ideas (grades 6–8)

Program Goal	• Designed to achieve coverage of a variety of grade 6–8 DLCS topics and standards. • Designed to be a stand-alone curriculum.	
Description	• Aligns to a variety of grade 6–8 DLCS standards, with a concentration on computational thinking topics. Some additional digital literacy and computing systems lessons may be needed to supplement this curriculum. • Students engage in programming via Karel, an online block- and/or text-based environment.	
Transitioning from:	**Code.org 3–5 CS Fundamentals**	**Springfield 3–5 CSforALL**
Purposeful Progression	• While CodeHS is more focused on a programming approach, it too transitions students quickly from block-based programming to text-based programming using Karel, and introduces additional programming languages (HTML). • There is no physical computing component to the CodeHS curriculum, consistent with CS Fundamentals.	• The Springfield 3–5 CSforALL curriculum engages students in block-based programming through a variety of coding platforms, including Scratch, Makey Makey, micro:bit, and Hummingbird Kits. While CodeHS is more focused on a programming approach, it too transitions students quickly from block-based programming to text-based programming using Karel, and introduces additional programming languages (HTML). There is no physical computing component to the CodeHS curriculum.
Pedagogical Coherence	• The CodeHS Computing Ideas course is focused on a programming approach to learning, which is different from CS Fundamentals. Where CS Fundamentals emphasizes more of a collaborative approach to lessons, Computing Ideas emphasizes more individual work with some small group work. Both, however, engage students in development of projects in some of the units. Use of Karel the Dog advances students quickly to text-based programming, as well as additional programming languages (HTML).	• Students in Springfield 3–5 CSforALL engaged with DLCS instruction in the context of core subjects (ELA, mathematics, science, and social studies). Computing Ideas is a stand-alone course, although it does provide some flexibility to be integrated into other contexts. • The CodeHS Computing Ideas course is focused on a programming approach to learning, which is different from Springfield CSforALL. Springfield 3–5 CSforALL emphasizes group projects,

	• While both CS Fundamentals and Computing Ideas are designed as stand-alone curricula, both have elements that can be integrated into other contexts or subject areas.	partner work, unplugged and coding activities, and student reflections. Computing Ideas emphasizes video tutorials, programming exercises, and challenge problems, some of which include student small group work. Both, however, engage students in development of projects in some of the units.
Equity	• CodeHS Computing Ideas does not attend to cultural responsiveness. The experiences of students in CS Fundamentals to consider problems of the world or how to help people in their own community would not be extended in Computing Ideas. • Both CS Fundamentals and Computing Ideas include limited differentiation supports, primarily found in certain activities that can be differentiated for individual students.	• CodeHS Computing Ideas does not attend to cultural responsiveness. The experiences of students in Springfield 3–5 CSforALL with diverse authors in text choices and character or professional representation, with an occasional project contextualized in students' home life or culture, would not be extended in Computing Ideas. • Springfield 3–5 CSforALL includes significantly more support for differentiation, with specific suggestions for many lessons, than does Computing Ideas, which is limited to certain activities that can be differentiated for individual students.

Computational Thinking & Problem Solving (High School)

Program Goal	• Designed to achieve comprehensive coverage of grade 9–12 DLCS topics and standards. • Designed to be a stand-alone semester or full-year course.	
Description	• A curriculum designed to engage students in collaborative projects that employ computational thinking and problem-solving strategies.	
Transitioning from:	**Code.org 6–8 CS Discoveries**	**CodeHS 6–8 Computing Ideas**
Purposeful Progression	• CS Discoveries moves students from block-based programming to text-based programming, using HTML/CSS, GameLab and AppLab, as well as a physical computing option with Circuit Playground. CTPS also starts with block-based programming, using MIT App Inventor. Unlike CS Discoveries, there is no physical computing element to CTPS.	• CodeHS transitions students quickly from block-based programming to text-based programming using Karel, and introduces additional programming languages (HTML). CTPS also starts with block-based programming, using MIT App Inventor. Similar to CodeHS, there is no physical computing component to CTPS.
Pedagogical Coherence	• CTPS provides a consistent progression from CS Discoveries. Both employ similar strategies for lesson design and student	• Computing Ideas emphasizes video tutorials, programming exercises, and challenge problems, some of which include

	engagement, such as student activities, group work, unplugged and coding activities. CTPS puts a larger emphasis on projects and application, particularly to workplace contexts. • CS Discoveries is designed to be a stand-alone curriculum, with some flexibility for other contexts; CTPS is designed as a stand-alone course.	student small group work. CTPS is more focused on student activities and projects based on workplace contexts. • Both CodeHS 6–8 Computing Ideas and CTPS are designed as stand-alone courses.
Equity	• CS Discoveries attends to equity through encouraging students to consider problems of the world, including how to help people in their own community, in a few projects. CTPS takes a somewhat similar approach by encouraging students to consider multiple perspectives as they work through problems and solutions grounded in workplace needs. Overall, both curricula are relatively limited in attention to cultural responsiveness. • CTPS includes limited tips for differentiation, primarily focused on encouraging group work and several pedagogical tips for the teacher. This is similar to CS Discoveries.	• The inclusion of multiple perspectives in CTPS projects, while limited, is more than is provided in Computing Ideas, which does not explicitly attend to equity or cultural responsiveness. • Both CTPS and Computing Ideas include limited differentiation supports, primarily found in certain activities that can be differentiated for individual students.

Exploring Computer Science (High School)

Program Goal	• Designed to achieve substantial coverage of grade 9–12 DLCS topics and standards. • Designed to be a stand-alone semester or full-year course.	
Description	• An introductory computer science curriculum designed around inquiry-based instruction, highlighting societal impacts of computer science.	
Transitioning from:	**Code.org 6–8 CS Discoveries**	**CodeHS 6–8 Computing Ideas**
Purposeful Progression	• CS Discoveries moves students from block-based programming to text-based programming, using HTML/CSS, GameLab and AppLab, as well as a physical computing option with Circuit Playground. ECS also starts with block-based programming, using Scratch, and moves students to text-based programming, including HTML/CSS. ECS does include an option for using Edison	• CodeHS transitions students quickly from block-based programming to text-based programming using Karel, and introduces HTML. ECS also starts with block-based programming, using Scratch, and moves students to text-based programming, including HTML/CSS. Similar to CodeHS, ESC does not include a physical computing component in the base curriculum, but does provide an optional unit using Edison robots.

	robots, which would build in the physical computing option in CS Discoveries.	
Pedagogical Coherence	• ECS provides a consistent progression from CS Discoveries. Both employ similar strategies for lesson design and student engagement, such as student activities, group work, unplugged and coding activities. • CS Discoveries is designed to be a stand-alone curriculum, with some flexibility for other contexts; ECS is designed as a stand-alone course.	• Computing Ideas emphasizes video tutorials, programming exercises, and challenge problems, some of which include student small group work. ECS is more focused on inquiry-based instruction rather than programming. • Both CodeHS 6–8 Computing Ideas and ECS are designed as stand-alone courses.
Equity	• CS Discoveries attends to equity through encouraging students to consider problems of the world, including how to help people in their own community, in a few projects. ECS builds significantly on this through highlighting the social impacts of computer science throughout the curriculum and explicit culturally-responsive lessons. • Differentiation opportunities in ECS are limited to collaboration and potential personalization through reflection and some activities. Similarly, CS Discoveries provides only occasional differentiation strategies.	• ECS is a significant change in the focus on equity and cultural responsiveness from Computing Ideas, which does not address this. ECS units highlight the social impacts of computer science and include explicit culturally-responsive lessons. • Differentiation opportunities in ECS are limited to collaboration and potential personalization through reflection and some activities. Similarly, Computing Ideas includes limited differentiation supports, primarily found in several supplemental activities and certain activities that can be differentiated for individual students.

SPECIAL TOPICS IN DLCS CURRICULA

	Grades K–2	Grades 3–5	Grades 6–8	Grades 9–12
Integration with math & science	• MA STEM+C: Integrated Units • Springfield CSforALL	• Action Fractions • MA STEM+C: Integrated Units • Springfield CSforALL	• Bootstrap: Algebra • Project GUTS	• Bootstrap: Data Science • Science+C
Physical computing	• Computer Science Essentials (Code & Go Robot Mouse) • Growing with KIBO • Learn to Code (Dash/Dot) • LEGO SPIKE Essential • SFUSD Creative Computing K–2 (BeeBots) • Springfield CSforALL (BeeBots, Makey Makey)	• Coding and Innovation using micro:bits • Computer Science Essentials (micro:bits) • Data Handling with micro:bit • Learn to Code (Dash/Dot) • LEGO SPIKE Essential • littleBits STEAM+ Coding • Sphero Computer Science Foundations 1 (BOLT) • Springfield CSforALL (Makey Makey, Hummingbird kits, micro:bits)	• Computer Science Discoveries (Circuit Playground) • Foundations of Physical Computing (LEGO SPIKE) • Intro to CS (micro:bits) • micro:bit Units • PLTW Gateway (micro:bits) • Sphero Computer Science Foundations 2 & 3 (BOLT)	• Computing with Robotics (LinkBot) • Cyber.org (Boe-Bots) • Exploring Computer Science (option, Edison Robots) • FIRST Tech Challenge (REV Robots) • LocoXtreme • NeuroMaker BCI Classroom • PLTW Computer Science (VEX vehicles)
Cyber-security	NA	NA	• CodeHS Introduction to Cybersecurity	• CodeHS Introduction to Cybersecurity • Computational Thinking and

	Grades K–2	Grades 3–5	Grades 6–8	Grades 9–12
				Problem Solving (unit) • Cyber.org: Selected Cyber Courses • PLTW Cybersecurity
Artificial intelligence	NA	NA	• Computer Science Discoveries (new optional unit) • Developing AI Literacy • Exploring Computer Science (optional unit)	• AI Foundations • NeuroMaker Applied AI

PROGRAMMING LANGUAGES USED IN CURRICULA

Language	Grades K–2	Grades 3–5	Grades 6–8	Grades 9–12
ScratchJr	• Ellipsis Education Computer Science • Coding as Another Language • Computer Science Essentials • NYC CSforALL • PLTW Launch • SFUSD Creative Computing K-2 Curriculum • Technokids: Selected Projects			
Scratch	• Springfield CSforALL	• Action Fractions • Ellipsis Education Computer Science • Computer Science Essentials • Creative Computing Curriculum • CS First • Elementary Computing for All • NYC CSforALL • PLTW Launch • Springfield CSforALL	• Computer Skills – Digital Savvy • Developing AI Literacy • Foundations of Physical Computing	• Exploring Computer Science

Language	Grades K–2	Grades 3–5	Grades 6–8	Grades 9–12
Blockly	• code.org Computer Science Fundamentals	• Coding & Innovation with micro:bits • code.org Computer Science Fundamentals • Computer Science Essentials • Computing with Minecraft • Data Handling with micro:bits • Learn to Code Curriculum • Springfield CSforALL	• Intro to CS • micro:bit Units • PLTW Gateway	• Computational Thinking and Problem Solving • Computing with Robotics • FIRST Tech Challenge
Other block-based	• Intro to Computer Science • Kodable K-2 • LEGO SPIKE Units	• LEGO SPIKE Units • littleBits STEAM+ Coding • Sphero Computer Science Foundations 1	• Computer Science Discoveries • Computing Ideas • Project GUTS • Sphero Computer Science Foundations 2 & 3	• Beauty and Joy of Computing • Code.org Computer Science Principles • CodeHS Computer Science
JavaScript		• littleBits STEAM+ Coding	• Computer Science Applications JavaScript • Ozaria (option) • Sphero Computer Science Foundations 2 & 3	• code.org Computer Science Principles • CodeHS Computer Science • FIRST Tech Challenge

Language	Grades K–2	Grades 3–5	Grades 6–8	Grades 9–12
Python		• Technokids: Selected Projects	• micro:bit Units • Ozaria (option)	• LocoXtreme • Neuromaker Modules • PLTW Computer Science
Pyret			• Bootstrap:Algebra	• Bootstrap:Data Science
HTML, CSS			• Computer Science Discoveries • Computer Skills – Digital Savvy • Computing Ideas	• CodeHS Computer Science • Exploring Computer Science
C++				• Computing with Robotics
BASIC				• Cyber.org Cyber Courses
NetLogo				• PLTW Computer Science • Science+C

DETAILED STANDARDS ALIGNMENT BY CURRICULUM

Grades K–2	Ellipsis Education Computer Science Foundations*	Coding as Another Language (Tufts DevTech Research Group)	Computer Science Essentials (EiE, MoS, Boston)	Computer Science Fundamentals A-C* (code.org)	Digital Citizenship K-2 (Common Sense Education)	Growing with KIBO* (KinderLab)	Intro to Computer Science (codeSpark)	Keyboarding Without Tears (K-2)	Kodable K-2
Kindergarten – Grade 2: Computing and Society (CAS)									
K-2.CAS.a Safety and Security									
K-2.CAS.a.1	Substantial							Substantial	
K-2.CAS.a.2	Substantial				Partial				
K-2.CAS.a.3	Substantial	Substantial	Substantial					Substantial	
K-2.CAS.a.4	Substantial			Substantial	Substantial			Substantial	
K-2.CAS.a.5	Substantial			Substantial	Substantial			Substantial	
K-2.CAS.a.6	Substantial			Substantial	Substantial			Substantial	
K-2.CAS.a.7	Substantial			Substantial	Substantial			Substantial	
K-2.CAS.a.8	Substantial			Substantial	Substantial			Substantial	
K-2.CAS.b Ethics and Laws									
K-2.CAS.b.1	Substantial			Substantial	Substantial	Substantial		Substantial	
K-2.CAS.b.2	Partially				Substantial			Substantial	
K-2.CAS.b.3	Substantial				Substantial				
K-2.CAS.b.4	Substantial				Substantial			Substantial	
K-2.CAS.c Interpersonal and Societal Impact									
K-2.CAS.c.1	Substantial					Partial	Substantial	Partial	
K-2.CAS.c.2	Substantial			Partial	Partial			Substantial	
Kindergarten – Grade 2: Digital Tools and Collaboration (DTC)									
K-2.DTC.a Digital Tools									
K-2.DTC.a.1	Substantial	Substantial				Substantial		Partial	Partial
K-2.DTC.a.2	Substantial	Substantial	Substantial					Substantial	Substantial
K-2.DTC.a.3	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial	Substantial	Substantial
K-2.DTC.a.4	Substantial	Substantial		Partial		Substantial	Substantial		
K-2.DTC.b Collaboration and Communication									
K-2.DTC.b.1	Substantial	Substantial	Substantial	Partial		Substantial	Partial		
K-2.DTC.b.2									
K-2.DTC.b.3	Substantial	Substantial				Substantial			
K-2.DTC.c Research									
K-2.DTC.c.1									
K-2.DTC.c.2	Substantial					Substantial		Partial	
K-2.DTC.c.3	Partially								
Kindergarten – Grade 2: Computing Systems (CS)									
K-2.CS.a Computing Devices									
K-2.CS.a.1	Substantial		Partial			Partial	Substantial	Substantial	Substantial
K-2.CS.a.2	Substantial		Partial			Partial		Substantial	
K-2.CS.a.3	Substantial		Partial			Substantial	Substantial	Substantial	
K-2.CS.a.4	Substantial	Substantial	Partial			Substantial		Substantial	
K-2.CS.b Human and Computer Partnerships									
K-2.CS.b.1	Substantial		Substantial	Partial		Substantial	Substantial	Partial	Partial
K-2.CS.b.2	Substantial		Substantial			Substantial			
K-2.CS.b.3	Substantial					Substantial	Substantial	Substantial	
K-2.CS.c Networks									
K-2.CS.c.1	Substantial						Substantial	Substantial	
K-2.CS.d Services									
NA									

An asterisk (*) designates ‘comprehensive’ curricular materials.

Kindergarten – Grade 2: Computational Thinking (CT)									
K-2.CT.a Abstraction									
K-2.CT.a.1	Substantial	Substantial		Substantial		Substantial	Substantial	Partial	
K-2.CT.b Algorithms									
K-2.CT.b.1	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial		Substantial
K-2.CT.b.2	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial		Substantial
K-2.CT.b.3	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial		Substantial
K-2.CT.c Data									
K-2.CT.c.1	Substantial			Partial		Partial			
K-2.CT.c.2	Substantial	Substantial		Partial		Substantial			
K-2.CT.c.3	Substantial	Substantial		Partial		Substantial	Substantial		
K-2.CT.c.4	Substantial					Substantial			
K-2.CT.c.5	Substantial		Partial			Substantial			Substantial
K-2.CT.d Programming and Development									
K-2.CT.d.1	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial		Substantial
K-2.CT.d.2	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial		Substantial
K-2.CT.d.3	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial		Substantial
K-2.CT.e Modeling and Simulation									
K-2.CT.e.1			Partial			Substantial			
K-2.CT.e.2			Substantial			Partial			

DLCS to CSTA Crosswalk

The Computer Science Teachers Association (CSTA) has developed national computer science education standards. To see alignment between the Massachusetts Digital Literacy and Computer Science standards and the CSTA standards, refer to the downloadable Excel file DLCS to CSTA Standards Crosswalk.

Grades K–2	Learn to Code Curriculum (K-2, Wonder Workshop)	MA Stem+C: Integrated K-2 Units (EDC & DESE)	NYC CSforALL*	PLTW Launch: K-2 Computer Science Units*	SFUSD Creative Computing K-2 Curriculum*	SPIKE Essential Units (LEGO)	Springfield CSforALL* (Springfield Public Schools)	Technokids: Selected K-2 Projects*
Kindergarten – Grade 2: Computing and Society (CAS)								
K-2.CAS.a Safety and Security								
K-2.CAS.a.1							Substantial	
K-2.CAS.a.2					Substantial		Partial	
K-2.CAS.a.3				Substantial	Substantial		Substantial	Substantial
K-2.CAS.a.4							Substantial	Substantial
K-2.CAS.a.5								
K-2.CAS.a.6								Partial
K-2.CAS.a.7								
K-2.CAS.a.8							Partial	
K-2.CAS.b Ethics and Laws								
K-2.CAS.b.1				Partial			Substantial	
K-2.CAS.b.2					Substantial		Substantial	Partial
K-2.CAS.b.3							Substantial	
K-2.CAS.b.4					Partial		Substantial	
K-2.CAS.c Interpersonal and Societal Impact								
K-2.CAS.c.1			Substantial	Partial	Substantial			
K-2.CAS.c.2			Substantial				Substantial	
Kindergarten – Grade 2: Digital Tools and Collaboration (DTC)								
K-2.DTC.a Digital Tools								
K-2.DTC.a.1	Substantial		Substantial	Partial	Substantial		Substantial	Substantial
K-2.DTC.a.2			Substantial	Partial	Substantial		Substantial	Substantial
K-2.DTC.a.3	Substantial		Substantial	Substantial	Substantial		Substantial	Substantial
K-2.DTC.a.4	Substantial		Substantial	Substantial	Substantial		Substantial	Substantial
K-2.DTC.b Collaboration and Communication								
K-2.DTC.b.1	Substantial			Substantial	Substantial		Substantial	Partial
K-2.DTC.b.2				Partial				Partial
K-2.DTC.b.3	Substantial			Partial	Substantial		Substantial	Substantial
K-2.DTC.c Research								
K-2.DTC.c.1							Substantial	
K-2.DTC.c.2				Partial				Partial
K-2.DTC.c.3							Substantial	Partial
Kindergarten – Grade 2: Computing Systems (CS)								
K-2.CS.a Computing Devices								
K-2.CS.a.1			Substantial	Partial	Substantial			Substantial
K-2.CS.a.2			Substantial	Partial	Substantial			Substantial
K-2.CS.a.3			Substantial	Partial	Substantial		Substantial	Substantial
K-2.CS.a.4			Substantial	Substantial	Substantial			Substantial
K-2.CS.b Human and Computer Partnerships								
K-2.CS.b.1	Partial		Substantial	Substantial	Substantial		Substantial	
K-2.CS.b.2	Partial		Substantial					
K-2.CS.b.3			Substantial	Partial	Partial			
K-2.CS.c Networks								
K-2.CS.c.1								
K-2.CS.d Services								
NA								

An asterisk (*) designates ‘comprehensive’ curricular materials.

Kindergarten – Grade 2: Computational Thinking (CT)								
K-2.CT.a Abstraction								
K-2.CT.a.1	Substantial	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial
K-2.CT.b Algorithms								
K-2.CT.b.1	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial
K-2.CT.b.2	Substantial	Substantial	Substantial	Substantial	Substantial	Partial	Substantial	Substantial
K-2.CT.b.3	Substantial	Substantial	Substantial	Substantial	Substantial	Partial	Substantial	Substantial
K-2.CT.c Data								
K-2.CT.c.1		Substantial	Substantial	Substantial		Substantial	Substantial	Partial
K-2.CT.c.2	Partial	Substantial	Partial				Substantial	Partial
K-2.CT.c.3	Partial	Substantial	Partial	Substantial			Substantial	
K-2.CT.c.4		Substantial	Partial	Substantial			Substantial	
K-2.CT.c.5			Substantial				Substantial	Substantial
K-2.CT.d Programming and Development								
K-2.CT.d.1	Substantial	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial
K-2.CT.d.2	Substantial	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial
K-2.CT.d.3	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial
K-2.CT.e Modeling and Simulation								
K-2.CT.e.1		Substantial		Partial	Partial		Substantial	
K-2.CT.e.2		Substantial					Substantial	

DLCS to CSTA Crosswalk

The Computer Science Teachers Association (CSTA) has developed national computer science education standards. To see alignment between the Massachusetts Digital Literacy and Computer Science standards and the CSTA standards, refer to the downloadable Excel file DLCS to CSTA Standards Crosswalk.

Grades 3–5	Action Fractions (Canon Lab)	Be Internet Awesome (Google)	Ellipsis Education Computer Science Courses*	Coding & Innovation using micro:bits (Utah Coding Project)	Computer Science Essentials (EiE, MoS, Boston)	Computer Science Fundamentals D-F* (Code.org)	Computing with Minecraft (Minecraft Education)	Creative Computing Curriculum (Creative Computing Lab, Harvard)	CS First (Google)	Data Handling with micro:bit (micro:bit)	Digital Citizenship 3-5 (Common Sense Education)
Grades 3 – 5: Computing and Society (CAS)											
3-5 CAS.a Safety and Security											
3-5 CAS.a.1											
3-5 CAS.a.2		Substantial	Substantial								Substantial
3-5 CAS.a.3		Substantial	Substantial			Substantial					Substantial
3-5 CAS.a.4		Substantial	Substantial								Substantial
3-5 CAS.a.5		Substantial	Substantial			Partial					Substantial
3-5 CAS.a.6		Substantial	Substantial			Substantial					Substantial
3-5 CAS.a.7		Substantial	Substantial			Substantial					Substantial
3-5.CAS.b Ethics and Laws											
3-5.CAS.b.1					Partial						
3-5.CAS.b.2			Substantial			Substantial					Substantial
3-5.CAS.b.3			Substantial			Partial					Substantial
3-5.CAS.b.4			Substantial			Partial					Substantial
3-5.CAS.b.5											
3-5.CAS.c Interpersonal and Societal Impact											
3-5.CAS.c.1										Partial	Substantial
3-5.CAS.c.2										Partial	Substantial
3-5.CAS.c.3			Partial								
3-5.CAS.c.4			Substantial			Partial				Partial	
3-5.CAS.c.5											
3-5.CAS.c.6			Substantial			Partial					
3-5.CAS.c.7		Substantial	Substantial			Substantial					Substantial
Grades 3 – 5: Digital Tools and Collaboration (DTC)											
3-5.DTC.a Digital Tools											
3-5.DTC.a.1											
3-5.DTC.a.2				Substantial			Substantial				
3-5.DTC.a.3	Substantial		Substantial	Substantial	Substantial	Substantial	Partial				
3-5.DTC.b Collaboration and Communication											
3-5.DTC.b.1	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Partial		
3-5.DTC.b.2	Substantial		Substantial	Substantial	Partial	Substantial	Partial	Substantial			
3-5.DTC.c Research											
3-5.DTC.c.1											
3-5.DTC.c.2											
3-5.DTC.c.3			Substantial								
3-5.DTC.c.4											
3-5.DTC.c.5											
3-5.DTC.c.6			Substantial								
3-5.DTC.c.7			Substantial								
Grades 3 – 5: Computing Systems (CS)											
3-5.CS.a Computing Devices											
3-5.CS.a.1			Substantial	Substantial			Substantial				
3-5.CS.a.2				Substantial	Partial					Substantial	
3-5.CS.a.3				Substantial	Partial						
3-5.CS.a.4					Partial		Partial				
3-5.CS.a.5			Substantial	Substantial							
3-5.CS.a.6											
3-5.CS.b Human and Computer Partnerships											
3-5.CS.b.1			Substantial		Substantial						
3-5.CS.b.2			Substantial			Partial					
3-5.CS.b.3					Substantial						

An asterisk (*) designates ‘comprehensive’ curricular materials.

3-5.CS.c Networks											
3-5.CS.c.1			Substantial			Partial					
3-5.CS.c.2			Substantial								
3-5.CS.c.3			Substantial								
3-5.CS.c.4			Substantial								
3-5.CS.d Services											
3-5.CS.d.1											
Grades 3 – 5: Computational Thinking (CT)											
3-5.CT.a Abstraction											
3-5.CT.a.1	Substantial		Substantial	Substantial	Substantial	Substantial			Substantial		
3-5.CT.a.2	Substantial		Substantial	Substantial		Substantial			Substantial		
3-5.CT.a.3	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Partial	Substantial	
3-5.CT.b Algorithms											
3-5.CT.b.1	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial	
3-5.CT.b.2	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	
3-5.CT.b.3	Substantial		Substantial	Substantial	Partial	Substantial	Substantial	Substantial	Substantial	Substantial	
3-5.CT.b.4	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	
3-5.CT.b.5	Substantial		Substantial	Substantial	Partial	Substantial	Substantial	Substantial	Substantial	Substantial	
3-5.CT.c Data											
3-5.CT.c.1					Partial					Substantial	
3-5.CT.c.2			Substantial		Partial	Substantial				Substantial	
3-5.CT.d Programming and Development											
3-5.CT.d.1	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	
3-5.CT.d.2	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	
3-5.CT.d.3	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	
3-5.CT.d.4	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	
3-5.CT.e Modeling and Simulation											
3-5.CT.e.1					Partial		Substantial		Partial		
3-5.CT.e.2					Partial	Substantial	Substantial		Partial		
3-5.CT.e.3						Substantial					

DLCS to CSTA Crosswalk

The Computer Science Teachers Association (CSTA) has developed national computer science education standards. To see alignment between the Massachusetts Digital Literacy and Computer Science standards and the CSTA standards, refer to the downloadable Excel file DLCS to CSTA Standards Crosswalk.

Grades 3–5	Elementary Computing for All (UC Irving)	Keyboarding Without Tears (3-5)	Learn to Code Curriculum (3-5; Wonder Workshop)	littleBits STEAM+ Coding (Sphero)	MA Stem+C: Integrated 3-5 Units (EDC & DESE)	NYC CSforALL*	PLTW Launch: 3-5 Computer Science Units*	Sphero Computer Science Foundations Course 1	SPIKE Essential Units (LEGO)	Springfield CSforALL* (Springfield Public Schools)	Technokids: Selected 3-5 Projects*
Grades 3 – 5: Computing and Society (CAS)											
3-5 CAS.a Safety and Security											
3-5 CAS.a.1		Substantial								Substantial	
3-5 CAS.a.2		Partial								Substantial	Partial
3-5 CAS.a.3		Substantial					Partial			Partial	Substantial
3-5 CAS.a.4		Partial								Partial	Partial
3-5 CAS.a.5		Substantial								Partial	Partial
3-5 CAS.a.6		Substantial								Partial	Partial
3-5 CAS.a.7		Substantial								Substantial	Partial
3-5.CAS.b Ethics and Laws											
3-5.CAS.b.1		Substantial								Substantial	Partial
3-5.CAS.b.2										Substantial	Partial
3-5.CAS.b.3											Partial
3-5.CAS.b.4		Substantial								Partial	Partial
3-5.CAS.b.5											
3-5.CAS.c Interpersonal and Societal Impact											
3-5.CAS.c.1										Substantial	
3-5.CAS.c.2										Partial	
3-5.CAS.c.3											
3-5.CAS.c.4		Partial		Partial			Substantial			Substantial	
3-5.CAS.c.5		Substantial									
3-5.CAS.c.6											
3-5.CAS.c.7		Substantial								Partial	Substantial
Grades 3 – 5: Digital Tools and Collaboration (DTC)											
3-5.DTC.a Digital Tools											
3-5.DTC.a.1		Substantial									
3-5.DTC.a.2	Substantial						Partial				Substantial
3-5.DTC.a.3	Substantial	Partial	Substantial			Substantial	Substantial	Substantial		Substantial	Substantial
3-5.DTC.b Collaboration and Communication											
3-5.DTC.b.1	Substantial	Partial	Substantial	Substantial		Substantial	Substantial	Substantial		Substantial	Substantial
3-5.DTC.b.2			Substantial			Partial				Partial	Substantial
3-5.DTC.c Research											
3-5.DTC.c.1		Partial								Partial	Substantial
3-5.DTC.c.2										Substantial	Substantial
3-5.DTC.c.3		Partial									Substantial
3-5.DTC.c.4		Partial								Substantial	Substantial
3-5.DTC.c.5		Substantial	Substantial	Substantial			Partial			Substantial	Substantial
3-5.DTC.c.6		Substantial									Substantial
3-5.DTC.c.7							Partial				Substantial
Grades 3 – 5: Computing Systems (CS)											
3-5.CS.a Computing Devices											
3-5.CS.a.1	Substantial	Partial				Substantial	Partial			Substantial	
3-5.CS.a.2	Substantial	Partial		Partial		Substantial	Partial	Partial			
3-5.CS.a.3				Partial				Partial			Partial
3-5.CS.a.4						Partial	Partial				Partial
3-5.CS.a.5						Substantial					
3-5.CS.a.6											
3-5.CS.b Human and Computer Partnerships											
3-5.CS.b.1			Partial			Partial	Partial			Substantial	
3-5.CS.b.2			Substantial								
3-5.CS.b.3											

An asterisk (*) designates ‘comprehensive’ curricular materials.

3-5.CS.c Networks											
3-5.CS.c.1							Partial				
3-5.CS.c.2											Partial
3-5.CS.c.3							Substantial				Partial
3-5.CS.c.4											Substantial
3-5.CS.d Services											
3-5.CS.d.1											
Grades 3 – 5: Computational Thinking (CT)											
3-5.CT.a Abstraction											
3-5.CT.a.1			Partial		Substantial	Substantial	Substantial			Substantial	Partial
3-5.CT.a.2						Substantial	Partial		Partial	Substantial	Substantial
3-5.CT.a.3			Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Partial
3-5.CT.b Algorithms											
3-5.CT.b.1	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Partial	Substantial	Substantial
3-5.CT.b.2	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial
3-5.CT.b.3	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial		Substantial
3-5.CT.b.4	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial
3-5.CT.b.5	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial
3-5.CT.c Data											
3-5.CT.c.1					Partial						Partial
3-5.CT.c.2					Substantial		Substantial		Partial	Partial	Substantial
3-5.CT.d Programming and Development											
3-5.CT.d.1	Substantial		Substantial	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial
3-5.CT.d.2	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial
3-5.CT.d.3	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial
3-5.CT.d.4			Partial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Partial	Substantial
3-5.CT.e Modeling and Simulation											
3-5.CT.e.1			Partial		Substantial	Substantial	Partial	Substantial		Substantial	
3-5.CT.e.2					Substantial		Partial	Partial		Substantial	
3-5.CT.e.3					Substantial		Substantial		Partial	Substantial	

DLCS to CSTA Crosswalk

The Computer Science Teachers Association (CSTA) has developed national computer science education standards. To see alignment between the Massachusetts Digital Literacy and Computer Science standards and the CSTA standards, refer to the downloadable Excel file DLCS to CSTA Standards Crosswalk.

Grades 6–8	Bootstrap: Algebra (Brown University)	Computational Thinking Curriculum (MIT App Inventor)	Computer Science Applications JavaScript* (Ellipsis)	Computer Science Discoveries* (code.org)	Computer Skills - Digital Savvy* (CompuScholar)	Computing Ideas* (CodeHS)	Creative Computing Curriculum (Creative Computing Lab, Harvard)	Developing AI Literacy (MIT)	Digital Citizenship 6-8 (Common Sense Education)
Grades 6 to 8: Computing and Society [CAS]									
6-8.CAS.a Safety and Security									
6-8.CAS.a.1			Substantial	Substantial	Substantial	Substantial			Partial
6-8.CAS.a.2					Partial	Substantial			Substantial
6-8.CAS.a.3			Substantial	Substantial	Substantial	Substantial	Partial	Partial	Substantial
6-8.CAS.a.4			Substantial		Substantial	Substantial	Partial	Partial	Substantial
6-8.CAS.a.5			Substantial		Substantial	Partial	Partial	Partial	Substantial
6-8.CAS.b Ethics and Laws									
6-8.CAS.b.1	Partial		Substantial	Substantial	Substantial	Partial		Partial	Substantial
6-8.CAS.b.2			Substantial	Substantial	Substantial	Substantial		Partial	
6-8.CAS.b.3			Substantial	Substantial	Substantial	Substantial		Partial	Substantial
6-8.CAS.b.4			Substantial	Substantial		Substantial		Partial	Substantial
6-8.CAS.b.5				Substantial	Substantial	Substantial			
6-8.CAS.b.6					Partial				
6-8.CAS.b.7				Substantial	Partial				
6-8.CAS.b.8					Partial				
6-8.CAS.b.9			Partial	Substantial					
6-8.CAS.c Interpersonal and Societal Impact									
6-8.CAS.c.1			Substantial	Substantial	Partial	Substantial		Substantial	
6-8.CAS.c.2			Substantial		Substantial				
6-8.CAS.c.3			Partial	Substantial		Substantial		Substantial	
6-8.CAS.c.4				Partial				Substantial	Partial
6-8.CAS.c.5				Partial				Substantial	Partial
Grades 6 to 8: Digital Tools and Collaboration [DTC]									
6-8.DTC.a Digital Tools									
6-8.DTC.a.1			Substantial	Substantial	Substantial			Substantial	
6-8.DTC.a.2	Partial		Partial		Substantial	Partial			
6-8.DTC.a.3			Substantial	Substantial	Substantial	Partial		Partial	
6-8.DTC.a.4	Partial			Substantial	Substantial	Substantial		Substantial	
6-8.DTC.a.5			Substantial	Substantial					
6-8.DTC.b Collaboration and Communication									
6-8.DTC.b.1		Partial	Substantial	Substantial	Substantial	Substantial	Partial	Substantial	
6-8.DTC.b.2	Substantial		Substantial	Substantial	Substantial		Substantial		
6-8.DTC.b.3			Substantial		Substantial		Partial		
6-8.DTC.c Research									
6-8.DTC.c.1				Substantial	Substantial				
6-8.DTC.c.2					Partial	Substantial		Substantial	
6-8.DTC.c.3			Substantial	Substantial	Substantial				
6-8.DTC.c.4			Partial	Substantial	Partial			Substantial	
6-8.DTC.c.5					Substantial			Partial	

An asterisk (*) designates ‘comprehensive’ curricular materials.

Grades 6 to 8: Computing Systems [CS]									
6-8.CS.a Computing Devices									
6-8.CS.a.1			Partial		Substantial	Substantial			
6-8.CS.a.2			Substantial		Substantial	Substantial			
6-8.CS.a.3				Substantial	Substantial	Substantial			
6-8.CS.a.4		Partial					Substantial		
6-8.CS.a.5		Partial		Substantial			Substantial		
6-8.CS.a.6		Partial	Partial	Substantial	Partial				
6-8.CS.a.7				Substantial	Substantial				
6-8.CS.b Human and Computer Partnerships									
6-8.CS.b.1				Substantial				Substantial	
6-8.CS.b.2				Substantial		Partial		Substantial	
6-8.CS.c Networks									
6-8.CS.c.1					Substantial	Partial			
6-8.CS.c.2			Substantial		Substantial	Substantial			
6-8.CS.c.3			Substantial	Substantial	Partial	Substantial		Partial	
6-8.CS.d Services									
6-8.CS.d.1		Partial			Partial	Partial			
Grades 6 to 8: Computational Thinking [CT]									
6-8.CT.a Abstraction									
6-8.CT.a.1	Substantial	Partial			Partial			Substantial	
6-8.CT.a.2	Substantial	Substantial	Substantial	Substantial	Partial	Substantial	Substantial	Partial	
6-8.CT.a.3		Partial	Substantial	Substantial		Substantial	Substantial	Substantial	
6-8.CT.b Algorithms									
6-8.CT.b.1	Substantial	Substantial		Substantial	Partial	Substantial	Substantial	Substantial	
6-8.CT.b.2	Substantial	Substantial	Partial		Partial	Substantial	Substantial	Substantial	
6-8.CT.b.3	Substantial	Substantial	Substantial	Substantial	Partial	Substantial	Substantial	Substantial	
6-8.CT.b.4	Partial	Substantial	Partial	Substantial	Partial	Substantial	Substantial	Substantial	
6-8.CT.b.5	Substantial	Partial			Partial	Substantial	Substantial	Substantial	
6-8.CT.c Data									
6-8.CT.c.1			Substantial			Partial	Substantial		
6-8.CT.c.2		Partial	Substantial				Partial		
6-8.CT.c.3		Substantial	Substantial	Substantial	Substantial			Substantial	
6-8.CT.c.4		Partial	Substantial	Substantial	Substantial			Substantial	
6-8.CT.c.5		Substantial	Substantial	Substantial	Partial			Substantial	
6-8.CT.d Programming and Development									
6-8.CT.d.1	Partial	Substantial	Partial	Substantial	Partial	Substantial	Substantial	Partial	
6-8.CT.d.2	Substantial	Substantial	Substantial			Substantial	Substantial		
6-8.CT.d.3	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	
6-8.CT.d.4	Partial	Substantial		Substantial	Partial	Substantial	Substantial	Partial	
6-8.CT.d.5		Substantial	Substantial	Substantial	Partial	Substantial	Substantial	Substantial	
6-8.CT.d.6	Substantial	Substantial	Substantial	Substantial	Partial	Substantial	Substantial		
6-8.CT.e Modeling and Simulation									
6-8.CT.e.1			Substantial	Substantial				Substantial	
6-8.CT.e.2				Substantial				Substantial	

DLCS to CSTA Crosswalk

The Computer Science Teachers Association (CSTA) has developed national computer science education standards. To see alignment between the Massachusetts Digital Literacy and Computer Science standards and the CSTA standards, refer to the downloadable Excel file DLCS to CSTA Standards Crosswalk.

Grades 6–8	Foundations of Physical Computing* (LEGO Education)	Intro to CS (Microsoft MakeCode)	Introduction to Cybersecurity (CodeHS)	micro:bit Units (micro:bit)	NFTE Startup Tech*	Ozaria (CodeCombat)	Project GUTS	PLTW Gateway: Computer Science Units*	Sphero Computer Science Foundations Courses 2 & 3
Grades 6 to 8: Computing and Society [CAS]									
6-8.CAS.a Safety and Security									
6-8.CAS.a.1	Substantial		Substantial			Substantial			
6-8.CAS.a.2			Substantial						Substantial
6-8.CAS.a.3			Substantial		Partial			Partial	
6-8.CAS.a.4			Substantial					Partial	
6-8.CAS.a.5			Partial						
6-8.CAS.b Ethics and Laws									
6-8.CAS.b.1			Substantial			Substantial			
6-8.CAS.b.2			Partial			Partial			
6-8.CAS.b.3			Substantial			Substantial			
6-8.CAS.b.4			Substantial						
6-8.CAS.b.5									
6-8.CAS.b.6								Substantial	
6-8.CAS.b.7									
6-8.CAS.b.8			Substantial					Partial	
6-8.CAS.b.9									
6-8.CAS.c Inte	Partial								
6-8.CAS.c.1	Substantial		Substantial	Substantial	Substantial	Substantial		Partial	Partial
6-8.CAS.c.2	Substantial				Substantial			Partial	
6-8.CAS.c.3			Partial	Partial					
6-8.CAS.c.4									
6-8.CAS.c.5			Partial						
Grades 6 to 8: Digital Tools and Collaboration [DTC]									
6-8.DTC.a Digital Tools									
6-8.DTC.a.1		Partial	Partial		Substantial				Partial
6-8.DTC.a.2					Partial			Substantial	Partial
6-8.DTC.a.3	Partial		Substantial		Substantial			Substantial	
6-8.DTC.a.4	Substantial		Substantial		Substantial			Substantial	
6-8.DTC.a.5					Substantial				
6-8.DTC.b Collaboration and Communication									
6-8.DTC.b.1	Substantial		Substantial	Substantial	Partial	Substantial		Partial	
6-8.DTC.b.2									
6-8.DTC.b.3			Partial						
6-8.DTC.c Research									
6-8.DTC.c.1	Partial								
6-8.DTC.c.2			Substantial		Partial				
6-8.DTC.c.3					Partial				
6-8.DTC.c.4	Substantial				Partial	Substantial		Partial	
6-8.DTC.c.5			Substantial						

An asterisk (*) designates ‘comprehensive’ curricular materials.

Grades 6 to 8: Computing Systems [CS]									
6-8.CS.a Computing Devices									
6-8.CS.a.1	Partial					Substantial			
6-8.CS.a.2	Substantial				Partial				
6-8.CS.a.3	Partial	Partial				Substantial		Substantial	
6-8.CS.a.4	Substantial	Substantial		Substantial				Substantial	Substantial
6-8.CS.a.5	Substantial	Substantial		Substantial	Partial			Substantial	Substantial
6-8.CS.a.6	Substantial			Substantial	Partial			Substantial	Substantial
6-8.CS.a.7	Substantial	Partial		Partial		Partial		Substantial	
6-8.CS.b Human and Computer Partnerships									
6-8.CS.b.1	Substantial			Substantial	Partial			Partial	
6-8.CS.b.2	Substantial			Substantial	Partial			Partial	
6-8.CS.c Networks									
6-8.CS.c.1		Partial		Partial		Partial		Substantial	
6-8.CS.c.2				Substantial		Substantial			
6-8.CS.c.3	Partial			Substantial	Partial	Substantial		Partial	
6-8.CS.d Services									
6-8.CS.d.1				Partial					
Grades 6 to 8: Computational Thinking [CT]									
6-8.CT.a Abstraction									
6-8.CT.a.1	Substantial				Substantial	Substantial	Partial	Partial	Substantial
6-8.CT.a.2	Substantial				Substantial	Substantial	Substantial	Substantial	Substantial
6-8.CT.a.3	Substantial				Partial	Substantial		Substantial	Partial
6-8.CT.b Algorithms									
6-8.CT.b.1	Substantial	Substantial			Substantial	Substantial	Substantial	Substantial	Substantial
6-8.CT.b.2	Substantial	Substantial			Partial	Substantial	Substantial	Partial	Substantial
6-8.CT.b.3	Substantial	Substantial			Substantial	Substantial	Substantial	Substantial	Substantial
6-8.CT.b.4	Substantial	Substantial		Partial	Substantial	Substantial	Substantial	Partial	Substantial
6-8.CT.b.5	Substantial	Partial		Partial	Partial	Substantial	Substantial	Partial	Substantial
6-8.CT.c Data									
6-8.CT.c.1		Partial				Partial			Substantial
6-8.CT.c.2				Partial	Partial	Substantial			
6-8.CT.c.3		Partial		Substantial	Substantial	Substantial		Substantial	
6-8.CT.c.4				Substantial	Partial			Partial	
6-8.CT.c.5	Substantial			Substantial	Partial			Substantial	Partial
6-8.CT.d Programming and Development									
6-8.CT.d.1	Substantial	Substantial		Partial	Partial	Substantial	Substantial	Substantial	Substantial
6-8.CT.d.2	Substantial	Substantial			Partial	Substantial	Substantial	Substantial	Substantial
6-8.CT.d.3	Substantial	Substantial			Substantial	Substantial	Substantial	Substantial	Substantial
6-8.CT.d.4	Substantial	Substantial			Partial	Substantial	Substantial	Substantial	Substantial
6-8.CT.d.5	Substantial	Substantial			Substantial	Substantial	Substantial	Substantial	Substantial
6-8.CT.d.6	Substantial	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial
6-8.CT.e Modeling and Simulation									
6-8.CT.e.1	Substantial			Substantial	Partial		Substantial		
6-8.CT.e.2				Substantial		Substantial	Substantial		
6-8.CT.e.3				Substantial	Partial		Substantial		

DLCS to CSTA Crosswalk

The Computer Science Teachers Association (CSTA) has developed national computer science education standards. To see alignment between the Massachusetts Digital Literacy and Computer Science standards and the CSTA standards, refer to the downloadable Excel file DLCS to CSTA Standards Crosswalk.

Grades 9–12	AI Foundations * (UBTECH Education)	Beauty and Joy of Computing* (UC Berkeley & EDC)	Bootstrap: Algebra (Brown Univ)	Bootstrap: Data Science* (Brown Univ)	Code.org Computer Science Principles *	CodeHS Computer Science Principles *	CodeHS Intro to CS with JavaScript	CodeHS Intro to Cybersecurity	Computational Thinking & Problem Solving* (CSforMA)	Computing with Robotics (UC Davis, RoboBlockly)	Cyber Literacy II (Cyber.org)
Grades 9–12: Computing and Society (CAS)											
9-12.CAS.a Safety and Security											
9-12.CAS.a.1											
9-12.CAS.a.2		Substantial						Substantial	Partial		Substantial
9-12.CAS.a.3		Substantial						Substantial			Substantial
9-12.CAS.a.4		Substantial						Substantial	Substantial		
9-12.CAS.a.5									Partial		
9-12.CAS.a.6								Substantial			
9-12.CAS.b Ethics and Laws											
9-12.CAS.b.1									Substantial		
9-12.CAS.b.2	Partial	Substantial		Substantial	Substantial			Substantial	Substantial		Substantial
9-12.CAS.b.3	Partial	Substantial		Substantial	Substantial			Substantial	Partial		Substantial
9-12.CAS.b.4								Partial	Partial		
9-12.CAS.c Interpersonal and Societal Impact											
9-12.CAS.c.1		Substantial			Substantial	Substantial					
9-12.CAS.c.2	Substantial	Substantial			Substantial	Substantial		Partial	Substantial		
9-12.CAS.c.3	Partial										
9-12.CAS.c.4		Partial			Partial				Substantial	Substantial	
9-12.CAS.c.5	Substantial	Substantial			Substantial	Substantial			Substantial		Partial
9-12.CAS.c.6								Substantial	Substantial		
9-12.CAS.c.7	Partial	Partial			Partial	Partial			Substantial	Partial	
9-12.CAS.c.8	Substantial	Substantial			Substantial				Partial		
9-12.CAS.c.9		Substantial			Substantial			Substantial	Partial		Partial
Grades 9–12: Digital Tools and Collaboration (DTC)											
9-12.DTC.a Digital Tools											
9-12.DTC.a.1	Partial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial
9-12.DTC.a.2	Partial	Substantial			Substantial	Substantial			Substantial	Partial	
9-12.DTC.b Collaboration and Communication											
9-12.DTC.b.1		Substantial	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial	Substantial	Substantial
9-12.DTC.b.2	Partial	Partial		Partial	Partial	Partial			Partial	Substantial	
9-12.DTC.c Research											
9-12.DTC.c.1	Substantial	Substantial		Substantial	Substantial	Substantial		Partial	Substantial	Partial	
9-12.DTC.c.2	Substantial			Partial					Partial	Partial	Substantial
9-12.DTC.c.3	Substantial			Partial	Substantial				Substantial		Substantial
9-12.DTC.c.4	Partial	Substantial		Substantial	Substantial	Substantial		Substantial	Substantial	Partial	Substantial
9-12.DTC.c.5	Partial	Substantial		Substantial	Substantial	Substantial		Substantial	Substantial		
Grades 9–12: Computing Systems (CS)											
9-12.CS.a Computing Devices											
9-12.CS.a.1	Substantial								Substantial	Partial	Substantial
9-12.CS.a.2	Substantial	Substantial			Substantial				Partial	Substantial	Substantial
9-12.CS.a.3	Substantial									Substantial	Substantial
9-12.CS.a.4	Partial	Substantial								Substantial	Substantial
9-12.CS.a.5					Substantial				Substantial		
9-12.CS.a.6		Substantial			Substantial	Substantial			Partial		
9-12.CS.b Human and Computer Partnerships											
9-12.CS.b.1	Substantial	Partial		Partial					Substantial		

An asterisk (*) designates ‘comprehensive’ curricular materials.

9-12.CS.c Networks											
9-12.CS.c.1		Substantial			Substantial	Partial		Substantial	Substantial		
9-12.CS.c.2		Substantial			Partial	Partial		Substantial	Substantial		
9-12.CS.c.3		Partial						Substantial			
9-12.CS.d Services											
9-12.CS.d.1	Substantial	Partial			Partial	Partial					
9-12.CS.d.2	Partial										
Grades 9–12: Computational Thinking (CT)											
9-12.CT.a Abstraction											
9-12.CT.a.1	Partial	Substantial	Partial	Substantial	Substantial	Substantial	Substantial		Substantial	Partial	
9-12.CT.b Algorithms											
9-12.CT.b.1		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial		Substantial	Partial	Substantial
9-12.CT.b.2	Partial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial		Substantial		Substantial
9-12.CT.b.3	Partial	Substantial			Substantial	Substantial			Substantial	Substantial	
9-12.CT.b.4	Partial	Partial	Partial	Substantial	Partial	Partial	Substantial			Substantial	
9-12.CT.b.5	Partial		Partial						Partial		
9-12.CT.c Data											
9-12.CT.c.1	Substantial	Substantial	Substantial		Substantial	Substantial					
9-12.CT.c.2		Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Partial	
9-12.CT.c.3	Partial	Partial	Substantial	Substantial	Substantial	Partial			Substantial	Substantial	
9-12.CT.c.4	Substantial	Partial	Partial	Substantial	Partial				Substantial	Partial	
9-12.CT.c.5	Substantial		Substantial	Substantial			Substantial		Substantial	Partial	
9-12.CT.d Programming and Development											
9-12.CT.d.1		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial	
9-12.CT.d.2		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial	Partial
9-12.CT.d.3		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial		Partial	Substantial	
9-12.CT.d.4	Partial	Substantial	Partial	Partial	Substantial	Substantial	Partial		Substantial	Substantial	
9-12.CT.d.5		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial	Substantial
9-12.CT.d.6		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial	Substantial
9-12.CT.d.7			Substantial	Substantial		Substantial	Substantial			Substantial	
9-12.CT.d.8		Substantial	Partial		Substantial	Substantial			Partial	Substantial	
9-12.CT.d.9	Partial		Partial	Substantial			Substantial			Substantial	Substantial
9-12.CT.d.10		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial	Substantial
9-12.CT.d.11	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial	Substantial
9-12.CT.d.12			Substantial						Substantial	Substantial	
9-12.CT.e Modeling and Simulation											
9-12.CT.e.1	Partial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial		Partial	Substantial	
9-12.CT.e.2	Partial									Partial	

DLCS to CSTA Crosswalk

The Computer Science Teachers Association (CSTA) has developed national computer science education standards. To see alignment between the Massachusetts Digital Literacy and Computer Science standards and the CSTA standards, refer to the downloadable Excel file DLCS to CSTA Standards Crosswalk.

Grades 9–12	Cyber Science (Cyber.org)	Cyber Society (Cyber.org)	Exploring Computer Science* (UCLA)	FIRST Tech Challenge* (FIRST Robotics)	LocoXtreme (LocoRobo)	Mobile Computer Science Principles*	NeuroMaker BCI Classroom Module (BrainCo)	NeuroMaker Applied AI Module (BrainCo)	PLTW Computer Science Essentials	PLTW Computer Science Principles*	PLTW Cybersecurity	Science+C (EDC)
Grades 9–12: Computing and Society (CAS)												
9-12.CAS.a Safety and Security												
9-12.CAS.a.1			Partial	Partial								
9-12.CAS.a.2	Partial	Substantial	Substantial			Substantial				Partial		
9-12.CAS.a.3			Substantial							Partial	Partial	
9-12.CAS.a.4			Substantial									
9-12.CAS.a.5		Substantial	Substantial									
9-12.CAS.a.6		Substantial	Substantial	Partial								
9-12.CAS.b Ethics and Laws												
9-12.CAS.b.1											Partial	
9-12.CAS.b.2	Substantial	Substantial	Substantial			Substantial	Substantial	Partial	Substantial	Substantial	Substantial	
9-12.CAS.b.3	Partial	Substantial	Substantial			Substantial	Substantial	Partial	Substantial	Substantial	Substantial	
9-12.CAS.b.4		Substantial					Substantial					
9-12.CAS.c Interpersonal and Societal Impact												
9-12.CAS.c.1	Substantial	Substantial	Substantial			Substantial		Partial				
9-12.CAS.c.2	Partial	Substantial	Substantial	Substantial		Substantial		Partial	Substantial	Substantial	Substantial	
9-12.CAS.c.3		Substantial					Substantial	Substantial		Partial		
9-12.CAS.c.4								Substantial		Partial		
9-12.CAS.c.5	Substantial	Substantial	Substantial	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Partial	
9-12.CAS.c.6	Partial			Partial						Substantial	Substantial	
9-12.CAS.c.7			Substantial	Partial		Partial	Partial	Partial	Partial	Partial	Partial	
9-12.CAS.c.8	Partial	Substantial	Substantial			Substantial	Partial	Partial				
9-12.CAS.c.9	Substantial	Substantial	Substantial			Substantial	Partial	Partial		Substantial	Substantial	
Grades 9–12: Digital Tools and Collaboration (DTC)												
9-12.DTC.a Digital Tools												
9-12.DTC.a.1		Substantial	Substantial	Partial	Substantial	Substantial	Substantial		Substantial	Substantial	Substantial	Substantial
9-12.DTC.a.2		Substantial	Substantial	Partial		Substantial		Partial		Substantial	Substantial	
9-12.DTC.b Collaboration and Communication												
9-12.DTC.b.1		Substantial	Substantial	Substantial	Partial	Substantial	Partial		Substantial	Substantial	Partial	Substantial
9-12.DTC.b.2				Substantial		Partial			Partial	Partial	Substantial	
9-12.DTC.c Research												
9-12.DTC.c.1		Substantial	Substantial	Substantial		Substantial	Partial	Partial		Substantial	Partial	Substantial
9-12.DTC.c.2		Substantial		Substantial				Partial				
9-12.DTC.c.3		Substantial	Substantial	Partial				Partial		Partial	Substantial	
9-12.DTC.c.4		Substantial	Partial	Partial		Substantial	Partial			Substantial	Substantial	Partial
9-12.DTC.c.5			Substantial	Substantial		Substantial	Substantial	Substantial		Substantial	Partial	Substantial
Grades 9–12: Computing Systems (CS)												
9-12.CS.a Computing Devices												
9-12.CS.a.1	Substantial			Substantial	Substantial		Partial	Substantial		Partial	Substantial	
9-12.CS.a.2	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Partial	Substantial	Substantial	
9-12.CS.a.3				Substantial	Substantial		Partial	Partial			Substantial	
9-12.CS.a.4			Partial	Substantial	Substantial	Substantial	Partial	Substantial	Substantial	Substantial	Partial	
9-12.CS.a.5			Substantial							Substantial	Substantial	
9-12.CS.a.6						Substantial	Partial	Partial		Substantial		
9-12.CS.b Human and Computer Partnerships												
9-12.CS.b.1			Substantial	Partial	Partial	Partial	Substantial	Substantial	Partial	Partial	Substantial	

An asterisk (*) designates ‘comprehensive’ curricular materials.

9-12.CS.c Networks												
9-12.CS.c.1	Substantial	Partial		Partial		Substantial				Substantial	Substantial	
9-12.CS.c.2	Substantial	Substantial				Substantial				Substantial	Substantial	
9-12.CS.c.3										Substantial	Substantial	
9-12.CS.d Services												
9-12.CS.d.1						Partial		Partial				
9-12.CS.d.2	Partial										Substantial	
Grades 9–12: Computational Thinking (CT)												
9-12.CT.a Abstraction												
9-12.CT.a.1	Substantial		Substantial	Substantial	Partial	Substantial	Partial	Partial	Substantial	Substantial	Partial	Substantial
9-12.CT.b Algorithms												
9-12.CT.b.1	Substantial		Substantial	Partial	Partial	Substantial		Partial	Substantial	Substantial		Partial
9-12.CT.b.2	Substantial		Substantial	Substantial	Substantial	Substantial			Substantial	Substantial		Substantial
9-12.CT.b.3						Substantial	Substantial	Partial		Substantial		Substantial
9-12.CT.b.4	Partial		Substantial	Substantial	Substantial	Partial	Substantial	Partial		Substantial		Partial
9-12.CT.b.5			Substantial	Partial			Partial	Substantial				Partial
9-12.CT.c Data												
9-12.CT.c.1	Partial		Partial	Partial		Substantial	Partial	Partial	Partial	Substantial		
9-12.CT.c.2	Partial				Substantial	Substantial	Substantial			Substantial		
9-12.CT.c.3	Substantial		Substantial	Partial		Partial	Substantial	Substantial		Substantial		Substantial
9-12.CT.c.4			Substantial	Partial	Substantial	Substantial	Substantial	Partial		Partial	Partial	Partial
9-12.CT.c.5			Substantial	Substantial			Substantial	Substantial			Partial	Partial
9-12.CT.d Programming and Development												
9-12.CT.d.1			Substantial	Substantial	Substantial	Substantial	Substantial	Partial	Substantial	Substantial		Substantial
9-12.CT.d.2	Substantial		Partial	Substantial	Substantial	Substantial	Partial		Substantial	Substantial		Substantial
9-12.CT.d.3	Substantial		Substantial	Partial	Substantial	Substantial	Partial					
9-12.CT.d.4			Substantial	Substantial		Substantial	Substantial	Partial		Substantial		Partial
9-12.CT.d.5	Substantial		Substantial	Substantial	Substantial	Substantial	Partial		Substantial	Substantial		Substantial
9-12.CT.d.6	Substantial		Substantial	Substantial	Substantial	Substantial	Partial		Substantial	Substantial		Substantial
9-12.CT.d.7				Partial	Substantial		Partial	Partial	Partial	Partial		Substantial
9-12.CT.d.8	Substantial		Substantial	Substantial	Substantial	Substantial	Partial		Substantial	Substantial		Substantial
9-12.CT.d.9					Substantial	Substantial	Partial	Substantial	Partial	Partial		
9-12.CT.d.10	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial		Substantial
9-12.CT.d.11	Substantial		Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial	Substantial		Substantial
9-12.CT.d.12			Substantial	Substantial				Partial				Substantial
9-12.CT.e Modeling and Simulation												
9-12.CT.e.1		Partial	Substantial	Substantial	Partial	Substantial	Substantial		Substantial	Substantial		Substantial
9-12.CT.e.2			Substantial				Substantial			Partial		Substantial

DLCS to CSTA Crosswalk

The Computer Science Teachers Association (CSTA) has developed national computer science education standards. To see alignment between the Massachusetts Digital Literacy and Computer Science standards and the CSTA standards, refer to the downloadable Excel file DLCS to CSTA Standards Crosswalk.

CONTRIBUTORS

Jake Foster, Ph.D., founder of STEM Learning Design, LLC, supports organizations to develop STEM programming and learning spaces. He has contributed to a variety of K–12 STEM initiatives across Massachusetts, including facilitating the development of the state's DLCS standards. Jake led science and technology/engineering initiatives at ESE for more than a decade, as well as mathematics and computer science initiatives his last several years. He also led the revision of the most recent state standards and curriculum frameworks for Science and Technology/ Engineering, and Mathematics. Jake also has extensive experience reviewing and implementing curriculum, instruction and assessment at the classroom, school and district levels, with attention to alignment to standards and best practices for STEM education.

Lisa Manzi, an Instructional Technology/Computer Science Teacher at Michael E. Smith Middle School in South Hadley, MA, has worked with the DLCS Implementation Panel, contributing to recommendations related to DLCS licensure and Subject Matter Knowledge (SMK) requirements, PD needs, and support of DLCS Framework implementation and credentialing. She developed the middle school DLCS curriculum for grades 5–8 for South Hadley Public Schools through research of various curricula and tools to teach and meet the 2016 DLCS Curriculum Framework. She continually reviews and refines the curriculum.

David Petty is a Computing and Robotics Teacher at Brookline High School in Brookline, MA. He was a member of the DLCS standards review committee to develop the 2016 DLCS Framework and has taught several levels of computer science for 15 years. He has also taught an autonomous robotics curriculum. David was a contributor to the 2017 MA K–12 Computer Science Curriculum Guide (edc.org/massachusetts-k-12-computer-science-curriculum-guide), a survey of 32 curricula in computing. As past Co-President of the Computer Science Teachers Association Greater Boston chapter, he has promoted computer science programming and standards-aligned curricula for teachers and students across the state.

Melissa Zeitz, K–5 Digital Literacy and Computer Science Teacher at Liberty Elementary School in Springfield, MA, has worked to support DLCS curricula in her school and across her district. Melissa is the technology and curriculum resource coordinator for the CSforALL Springfield NSF-funded grant which aims to write computer science curriculum to be integrated into the district's academic curriculum. In these roles she has guided numerous teachers to different DLCS resources and curricula, and advised the district in deciding what computer science resources may be beneficial for the elementary level. Melissa is also the CSTA of Western Mass President, a Code.org Fundamental trainer, and a DLCS Ambassador with DESE. She was recently awarded the Presidential Award of Excellence in Science for her work with computer science education at the elementary level.

UPDATES

2022-11-23:

- Add link to SPRINGFIELD CS4ALL (K–2; 3–5) curriculum site.
- Add link to "DLCS to CSTA Crosswalk" spreadsheet download for users outside of Massachusetts.

2023-04-06:

- Codelicious has been renamed to Ellipsis Education.