Digital Literacy and Computer Science Curriculum Guide for Massachusetts Districts

June 30, 2020

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

Table of Contents

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.*

www.stemlearningdesign.com

CONTEXT FOR USING THIS GUIDE

This guide provides overviews of curricula that schools can use to engage students in learning of digital literacy and computer science (DLCS) concepts and skills that reflect the standards of 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. (www.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 University of Massachusetts Boston (regional partner for Code.org) to implement a comprehensive district change model that embeds computer science coursework for middle school students in the 15 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.
- *Broadening Advanced Technological Education Connections* (BATEC), 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 recent state competitive grant designed in partnership with M.A.S.S. to promote K–12 DLCS education in participating districts through developing implementation plans, choosing middle school 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, alignment to MA DLCS topics, and suggestions for potential pairings of curricula, depending on district need (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.

- 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.
  - Coding for Assessment, Differentiation, English Learner, and Teacher content knowledge supports: uses "Yes," "Limited," or "No" labels, with explanations included in the summary paragraph on the second page.
  - Web-based: 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.

CONSIDERATIONS FOR CHOOSING DLCS CURRICULA

To achieve a goal of comprehensive coverage of DLCS standards, a district may be considering building a program anew or adding to programming already in place. This guide provides suggestions for curriculum that could provide a strong comprehensive foundation as well as curricula that are more suited to filling particular gaps in existing programming.

To use this guide effectively, districts may find it useful to first:

- Complete a map of current DLCS topic coverage in existing programs. This will help to identify what topic or grade span gaps may exist, and whether current programming is coherent or consistent.
- Articulate whether the overall approach is to integrate DLCS into other areas of study or develop a DLCS program as its own subject. This may vary by grade span.

Curricula included in this guide emphasize computational thinking principles over programming language development (a primary focus on syntax and structure). There are a variety of options not included in this guide that provide instruction in coding for a variety of programming languages. There are also additional products that could be excellent supplements which provide for physical computing (such as via robotics kits or development boards) and design (such as CAD-based software packages). Most of these options were not included in this guide as they did not provide a coherent curriculum sufficiently aligned to the MA DLCS standards.

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.

Besides alignment to standards, important considerations for choosing a curriculum include how it matches the overall design or approach of the district's program, technology requirements, professional development support, and costs. Brief information about each of these is provided on the second page of each summary. Additional considerations for each grade span, including potential curriculum pairings to comprehensively address the DLCS standards, are included at the start of each grade span section.

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. There is not a single curriculum at either elementary school grade span that comprehensively addresses the DLCS standards. In particular, at grades 3–5, there is not an identified curriculum that effectively addresses the Computing Systems (CS) strand.

If a district does not have a DLCS curriculum at the elementary school levels, or is looking to start anew, the following suggested options could be a useful starting point.

- For grades K–2:
    1. Use *Computational Media Explorer – Computing Through Time* as a base and pair it with *Digital Citizenship K–2*.
    2. Use *Growing with KIBO* as a base and pair it with *Digital Citizenship K–2*.
    3. Pair *Keyboarding Without Tears* with one of the following: *Learning to Code*, *Technokids*, or *Animated Genres*.
- For grades 3–5:
    1. Use *Technokids* as a base and pair it with *Digital Citizenship 3–5*.
    2. Pair *Digital Citizenship 3–5* with *Learn to Code* and *STEM+C*.
    3. Pair *Digital Citizenship 3–5* (for Computing and Society [CAS] with selected *Technokids* projects that address DTC, and *CS Fundamentals* (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 also considering supplemental resources not included in this guide.

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
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 safey 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.

Curriculum pairing notes

- For comprehensive coverage: Since *Be Internet Awesome* is particularly focused on Safety and Security (CAS.a), it should be paired with another curriculum, such as *TechnoKids*, to meet the range of topics.
- For strand focus: The curriculum forms a good basis for a focus on Computing and Society (CAS), but would need to be supplemented to more fully address Ethics and Laws (CAS.b) and Interpersonal and Societal Impacts (CAS.c).

Professional development offered

- *Digital Citizenship and Safety* course offered online.

Costs

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

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

Code.org
code.org/educate/curriculum/elementary-school

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	Yes
Teacher content knowledge support	Yes
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*	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		*Partial*
CT.d. Programming & Development	✓	✓
CT.e. Modeling & Simulation		✓

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

CS 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 to support EL learners. 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. A teacher interface allows for deciding which assessments and projects students will engage in, and a dashboard shows the progress of the students as they 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.

Curriculum pairing notes

- For comprehensive coverage: This could be paired with a curriculum such as *Keyboarding Without Tears* at K–2, or selected projects from *TechnoKids* at 3–5.
- For strand focus: This is primarily a Computational Thinking (CT) curriculum, needing supplements only at K–2 for Data (CT.c) and Modeling and Simulation (CT.e).

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.

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
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.

Curriculum pairing notes

- For comprehensive coverage: Given the limited scope of this curriculum, it should be paired with other courses to achieve comprehensive coverage, such as *TechnoKids* and *Digital Citizenship*.
- For strand focus: This provides a good basis for a focus on Computational Thinking (CT) but would need supplements to address Data (CT.c) and Modeling and Simulation (CT.e).

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
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	*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

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. 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.

Curriculum pairing notes

- For comprehensive coverage: Given the limited scope of this curriculum, it should be paired with a survey course to achieve comprehensive topic coverage, such as selected projects from *TechnoKids*.
- For strand focus: A good basis for a focus on Computational Thinking (CT) but would need supplements to address Data (CT.c).

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.

DIGITAL CITIZENSHIP (K–2; 3–5)

Common Sense Education
www.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
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. Some teacher content knowledge support is provided through brief Teacher Notes. 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.

Curriculum pairing notes

- For comprehensive coverage: This curriculum should be paired with a survey course to achieve comprehensive coverage, such as *Growing with KIBO* or *Computational Media Explorer* at K–2, or *Technokids* or *Computational Media Creator* at grades 3–5.
- For strand focus: A good base for a focus on Computing and Society (CAS), particularly for grades 3–5, but for K–2 would need supplements to address Interpersonal and Societal Impact (CAS.c).

Professional development offered

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

Costs

- All curriculm and educator resources 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
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 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. The curriculum is organized into three 20-hour sections, including a novice, intermediate, and advance 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.

Technology requirements

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

Curriculum pairing notes

- For comprehensive coverage: This curriculum provides a strong foundation for a K–2 comprehensive course, but needs to be paired with a curriculum addressing Computing and Society (CAS) such as *Digital Citizenship*.
- For strand focus: given the breadth of this curriculum, it is not a good choice for focus on a particular strand.

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 $60.00
- A KIBO 18 Kit, with one robot serving 2–4 students, is $470.00.
- 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
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.

Curriculum pairing notes

- For comprehensive coverage: This curriculum would need to be paired with another curriculum such as *Keyboarding Without Tears*.
- For strand focus: This is a good foundation for a focus on Computational Thinking (CT), needed some supplements for Data (CT.c) and Modeling and Simulation (CT.e).

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
www.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	Yes
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*	X
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 different 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 very 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. For K–2, the grade 2 curriculum is particularly important to address the aligned K–2 standards; in 3–5 students must participate in all three years to address to all the 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.

Curriculum pairing notes

- For comprehensive coverage: At K–2, this curriculum provides a relatively broad foundation, but should be paired with a curriculum that addresses Digital Tools and Collaboration (DTC) and Computational Thinking (CT), such as *Learning to Code* or *TechnoKids.* At grades 3–5, this curriculum should be paired with a more comprehensive curriculum such as *TechnoKids* or *Computational Media Creator*.
- For strand focus: This curriculum is a good foundation for a focus on Computing and Society (CAS) at K–2 or 3–5, and on Computing Systems (CS) at K–2.

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 $1,600/half day or $2,400/day; virtual PD Hub is between $250–$1,250 per year depending on options chosen.

LEARN TO CODE (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
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

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.

Curriculum pairing notes

- For comprehensive coverage: This curriculum can provide a base for a comprehensive option, but needs to be paired with a digital literacy-focused course such as *Digital Citizenship*. Additional supplements would still be needed to address Data (CT.c) and Modeling and Simulation (CT.e).
- For strand focus: this curriculum is a bit too broad in scope to provide a particular focus on any one strand.

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.

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
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 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.

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*.

Curriculum pairing notes

- For comprehensive coverage: Given the focus on Computational Thinking (CT), this curriculum should be paired with another such as *Keyboarding Without Tears* at K–2 or *TechnoKids* at K–2 or 3–5 to address the range of standards.
- For strand focus: This curriculum provides a strong basis for a focus on Computational Thinking (CT) at either grade span.

Professional development offered

- A one-day professional development workshop on these units is held periodically.
- DESE supports DLCS Ambassadors, which will continue in 2020–2021. Ambassadors can support districts and schools in planning for DLCS curriculum integration.

Costs

- Units and online resources are free.

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

New York City Department of Education

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
Assessments, rubrics, examples	Yes
Differentiation supports	Yes
English Learner supports	Yes
Teacher content knowledge support	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 *Computational Media Explorer: Computing Through Time* Alignment**	**3–5 *Computational Media Creator: Build My City* Alignment**
CAS.a. Safety & Security	X	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	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 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?" The 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 English Learner support are included in lesson supports.

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 final project path that includes modeling and simulation.

Technology requirements

- For K–2, a tablet or iPad; for 3–5, a desktop or laptop with Internet connectivity and up-to-date browser.
- Software for K–2: ScratchJr or other coding environment. Some examples: Playlab, ScratchJr and Foos for Pre-reader
- Software for 3–5: Scratch and Scratch accounts under the Teacher Dashboard

Curriculum pairing notes

- For comprehensive coverage: The curriculum at both grade spans provides a relatively broad foundation but should be paired with a curriculum addressing Computing and Society (CAS), such as *Digital Citizenship*.
- For strand focus: This curriculum provides a strong focus on Computational Thinking (CT) in either grade span, with supplements needed only at K–2 to address Modeling and Simulation (CT.e) and at 3–5 to address Data (CT.c).

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
www.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
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.

Curriculum pairing notes

- For comprehensive coverage: The curriculum provides a relatively broad foundation but should be paired with a curriculum addressing Computing and Society (CAS), such as *Digital Citizenship*.
- For strand focus: The curriculm provides a strong base for Computational Thinking (CT), with supplement needed for K–2 Modeling & Simulation (CT.e).

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.

SCRATCHJR CODING CURRICULUM (K–2)

DevTech Research Group at Tufts University
sites.tufts.edu/devtech/research/coding-as-literacy/

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

Nature of curriculum	Integrated or stand-alone
Format and time	Units (2 units of 12 hours each)
Key classroom routines	Warm-ups, design challenges, writing activities, design projects, reflections
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	Limited
Teacher content knowledge support	Yes
Programming Platform or Language	ScratchJr (app or online blocks-based environment)
Web-based	Optional

Gr. K–2 DLCS Topics	**K–2 *ScratchJr Coding Curriculum* Alignment**
CAS.a. Safety & Security	*Partial*
CAS.b. Ethics & Laws	*Partial*
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	*Partial*
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

SCRATCHJR CODING CURRICULUM

Overview

ScratchJr Coding Curriculum for Emergent Readers and *Readers* are two related units that engage students in computational thinking through literature and programming. Both units involve the design and coding of a final computational and textual project. The *Emergent Readers* unit is designed for grade K. It introduces how to care for a device and be safe and responsible with it along with beginning coding with ScratchJr. It relates the writing and design processes using the story of *Knuffle Bunny* (by Mo Williems) as a context. The *Readers* curriculum is designed from grades 1 and 2. It goes deeper into computer science skills and a writing workshop approach using *Giraffes Can't Dance* (by Giles Andreae and Guy Parker-Rees) as a core story text. There are a few overlaps among lessons of both curricula but together they provide a depth of computer science concepts. Lessons include a warm up, design challenge, free exploration, a creative writing task, project, and reflection. In addition, opportunities for socio-emotional development are provided alongside the work with ScratchJr and literacy. A lot of detail and educator support make this possible for educators of any level to implement. Lesson resources include sample anchor charts, sample student work, assessments, picture commands to print, and a design journal. Limited differentiation support is provided through an introductory section that talks about ways to modify, structure, and group students based off of class needs. Limited English Learner supports are provided through sample sentence starters and anchor charts.

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.

Curriculum pairing notes

- For comprehensive coverage: This curriculum addresses only some topics so it would need to be paired with another, such as *Keyboarding Without Tears.*
- For strand focus: This curriculum provides a good base for a focus on Computational Thinking (CT), but needs supplements to address Data (CT.c) and Modeling and Simulation (CT.e).

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.

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

TechnoKids
www.technokids.com

Learn common productivity and coding applications.

Nature of curriculum	Integrated or stand-alone
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
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	✓	*Partial*
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.

Curriculum pairing notes

- For comprehensive coverage: At K–2, the curriculum provides a relatively broad foundation but should be paired with a curriculum addressing Computing and Society (CAS), such as *Digital Citizenship*. At grades 3–5, this curriculum provides a comprehensive scope.
- For strand focus: individual projects can be selected to address particular strands.

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 comprehensive 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. Possible options for a comprehensive approach include:

- Single survey middle school courses that offer coverage of all strands, but not necessarily all topics, include:
    1. *Computer Science Discoveries*
    2. *Computer Skills – Digital Savvy*
    3. *Computing Ideas*
- Several of those courses could be paired with another to achieve a more comprehensive coverage of all four DLCS strands:
    1. Use *Computer Science Discoveries* as a base and pair it *Digital Citizenship*.
    2. 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*, *Computational Thinking Curriculum*, or *NFTE Startup Tech*.

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 considering supplemental resources not included in this guide.

BOOTSTRAP:ALGEBRA

Brown University
www.bootstrapworld.org/

Integrate computational thinking into an Algebra course.

Nature of curriculum	Integrated with mathematics or stand-alone
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
Programming Platform or Language	WeScheme 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; suggestions English Learner strategies are specifically provided in a variety of lessons. The curriculum is available in Spanish. 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.

Curriculum pairing notes

- For comprehensive coverage: Given the limited scope of this curriculum, it should be paired with a survey course to achieve comprehensive topic coverage, such as *Computer Science Discoveries* or *Computer Skills - Digital Savvy*.
- For strand focus: A good base for a focus on Computational Thinking (CT) but would need supplements to address Data (CT.c) and Modeling and Simulation (CT.e). Could pair with *Project GUTS* to address Modeling and Simulation.

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
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 (Android) 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. 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 devices are supported; a beta version for iOS is available.

Curriculum pairing notes

- For comprehensive coverage: Given the limited scope of this curriculum, it should be paired with a survey course to achieve comprehensive topic coverage, such as *Computer Science Discoveries* or *Computer Skills - Digital Savvy*.
- For strand focus: A good base for a focus on Computational Thinking (CT) but would need supplements to address Modeling and Simulation (CT.e).

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.
- Porfessional-development costs vary.

COMPUTER SCIENCE DISCOVERIES

Code.org
https://curriculum.code.org/csd-19/

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	Yes
Programming Platform or Language	GameLab and AppLab (online block- and/or text-based environments), HTML and CSS
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. 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 moments. Occasional differentiation strategies are provided at both the student levels and class levels.

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.

Curriculum pairing notes

- For comprehensive coverage: Since this curriculum does not fully address all topics, it would best be supplemented with a curriculum that addresses Networks (CS.c) and Services (CS.d), such as *Computing Ideas*. Could also be paired with a curriculum that is stronger in Safety and Security (CAS.a), such as *Digital Citizenship* or *Be Internet Awesome*.
- For strand focus: As a survey course, this is not ideal to focus on any one strand.

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
www.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
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. 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.

Curriculum pairing notes

- For comprehensive coverage: While this course does touch upon Computational Thinking (CT), it would benefit from being paired with any other curriculum that focuses on Computational Thinking (CT). Purchase of the *Computer Skills - Digital Savvy* curriculum also provides access to an (unreviewed) Python course. This curriculum could also be supplemented with materials that addresses Interpersonal and Societal Impacts (CAS.c) and Human and Computer Partnerships (CS.b).
- For strand focus: As a survey course, this is not ideal to focus on any one strand.

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	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	Limited
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, open responses, and projects. Several supplemental materials and certain activities can be differentiated for individual students. 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.

Curriculum pairing notes

- For comprehensive coverage: *Computing Ideas* is relatively comprehensive, but needs supplements in several areas.
- For strand focus: The curriculum forms a reasonable basis for a focus on Computing Systems (CS), but would need to be supplemented to more fully address Human and Computer Partnerships (CS.b) in particular.

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.

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
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. 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.

Curriculum pairing notes

- For comprehensive coverage: Given the limited scope of this curriculum, it should be paired with a survey course to achieve comprehensive topic coverage, such as *Computer Science Discoveries* or *Computer Skills - Digital Savvy*.
- For strand focus: The curriculum provides a good basis for a focus on Computational Thinking (CT) but needs supplements to address Data (CT.c) and Modeling and Simulation (CT.e).

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.

DIGITAL CITIZENSHIP

Common Sense Education
www.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
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

DIGITIAL 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.

Curriculum pairing notes

- For comprehensive coverage: This curriculum should be paired with a survey course to achieve comprehensive coverage, such as *Computer Science Discoveries*.
- For strand focus: The curriculum provides a good base for a focus on Computing and Society (CAS) but would need supplements to fully address Ethics and Laws (CAS.b) and Interpersonal and Societal Impact (CAS.c).

Professional development offered

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

Costs

- All educator resources are free.

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
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).

Curriculum pairing notes

- For comprehensive coverage: This curriculum should be paired with a survey course to achieve comprehensive coverage, such as *Computer Science Discoveries* or *Computer Skills - Digital Savvy*.
- For strand focus: This curriculum is a good basis for a focus on Computational Thinking (CT) but needs supplements to address Abstraction (CT.a), Data (CT.c), and Modeling and Simulation (CT.e).

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.

NFTE STARTUP TECH

Network for Teaching Entrepreneurship
www.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
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 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.

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 devices are supported; a beta version for iOS is available.
- Uses Canvas LMS to deliver content.

Curriculum pairing notes

- For comprehensive coverage: Given the focus on Computational Thinking (CT) and entrepreneurship in this curriculum, it should be paired with a survey course to achieve comprehensive topic coverage, such as *Computer Science Discoveries* or *Computer Skills - Digital Savvy*.
- For strand focus: A strong base for a focus on Computational Thinking (CT) but would need supplements to address Modeling and Simulation (CT.e).

Professional development offered

- Face-to-face 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 one course for a year, which includes professional development.

PROJECT GUTS

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

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

Nature of curriculum	Integrated with science or stand-alone
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
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.

Curriculum pairing notes

- For comprehensive coverage: Given the limited scope of this curriculum, it should be paired with a survey course to achieve comprehensive topic coverage, such as *Computer Science Discoveries* or *Computer Skills - Digital Savvy*.
- For strand focus: This is one of few Computational Thinking-related curricula that emphasize Modeling and Simulation (CT.e). To fully address all CT topics, this curriculum needs to be supplemented with material to address Data (CT.c).

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
www.pltw.org/our-programs/pltw-gateway

Programming for apps and the physical world.

Nature of curriculum	Stand-alone
Format and time	3 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
Programming Platform or Language	MakeCode (online blocks-based environment, with micro:bits); MIT App Inventor (online blocks-based environment)
Web-based	No

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. 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 tablets and micro:bits: one for each per pair of students.

Curriculum pairing notes

- For comprehensive coverage: Given the primarily Computational Thinking (CT) focus of these units, this should be paired with a survey course, such as *Computer Science Discoveries* or *Computer Skills - Digital Savvy*.
- For strand focus: To fully address Computational Thinking (CT), this curriculum would need supplements to address Modeling and Simulation (CT.e) in particular.

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 are 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 11 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.

GRADES 9 THROUGH 12

CONSIDERATIONS FOR SELECTING HIGH SCHOOL CURRICULA

Curricula included in this section are typically 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.

This guide only focuses on those 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. #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: apcentral.collegeboard.org/courses/ap-computer-science-principles/classroom-resources/curricula-pedagogical-support.

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

BEAUTY AND JOY OF COMPUTING

University of California, Berkeley, and Education Development Center (EDC)
bjc.edc.org

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
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

Brown University
www.bootstrapworld.org/

A module to explore real-world questions through programming and data analysis.

Nature of curriculum	Stand-alone or integrated
Format and time	Module (approximately 25 hours)
Key classroom routines	Launch, investigate, synthesize, close
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	Limited
Programming Platform or Language	Pyret (online programming environment)
Web-based	Yes

Gr. 9–12 DLCS Topics	***Bootstrap:Data Science*** **Alignment**
CAS.a. Safety & Security	X
CAS.b. Ethics & Laws	*Partial*
CAS.c. Interpersonal & Societal Impact	X
DTC.a. Digital Tools	*Partial*
DTC.b. Collaboration & Communication	✓
DTC.c. Research	✓
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	*Partial*
CT.c. Data	✓
CT.d. Programming & Development	✓
CT.e. Modeling & Simulation	*Partial*

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

BOOTSTRAP:DATA SCIENCE

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. The module focuses on programming and data analysis, including topics such as functions, looping and iteration, data visualization, and linear regression. 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.

Three additional modules are available (including *Algebra*, described in the middle school section); the *Algebra, Data Science*, and *Reactive* modules can be combined to form a full stand-alone computer science course. The *Bootstrap:Data Science* module uses Pyret (a Python-like functional programming language).

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
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

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. 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
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
batec.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	Groupwork, unplugged activities, projects
Assessments, rubrics, examples	Yes
Differentiation supports	Limited
English Learner supports	No
Teacher content knowledge support	No
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	X
CAS.b. Ethics & Laws	X
CAS.c. Interpersonal & Societal Impact	*Partial*
DTC.a. Digital Tools	✓
DTC.b. Collaboration & Communication	✓
DTC.c. Research	✓
CS.a. Computing Devices	X
CS.b. Human & Computer Partnerships	X
CS.c. Networks	*Partial*
CS.d. Services	X
CT.a. Abstraction	✓
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

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. Each project is 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. 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. 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 during the summer. Workshops are five days and includes ongoing technical assistance for implementation.

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.

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
Programming Platform ovr Language	PBASIC 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 in quantity 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 California, Los Angeles
www.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	Limited
English Learner supports	No
Teacher content knowledge support	Limited
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. 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. An older version (v6.2; current version is v9.0) 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.

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.

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
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 (CSP) 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 (using an Android or a beta iOS companion app). 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 devices are supported; a beta version for iOS is available.

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

PLTW COMPUTER SCIENCE

Project Lead the Way
www.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
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)
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.

APPENDIX

REVIEW PROCESS

The review process was undertaken in three phases:

1. Identified a wide reange 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. Of these curricula, approximately 10–12 were chosen for each grade span to ensure a variety of educational approaches are available, to ensure options for pairing curricula to achieve substantive DLCS standards coverage, and to ensure at least one no-cost option.
4. A description of each curriculum was then written, then checked by the curriculum provider to ensure information about context such as technology requirements, costs, and professional development availability were up-to-date.

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.

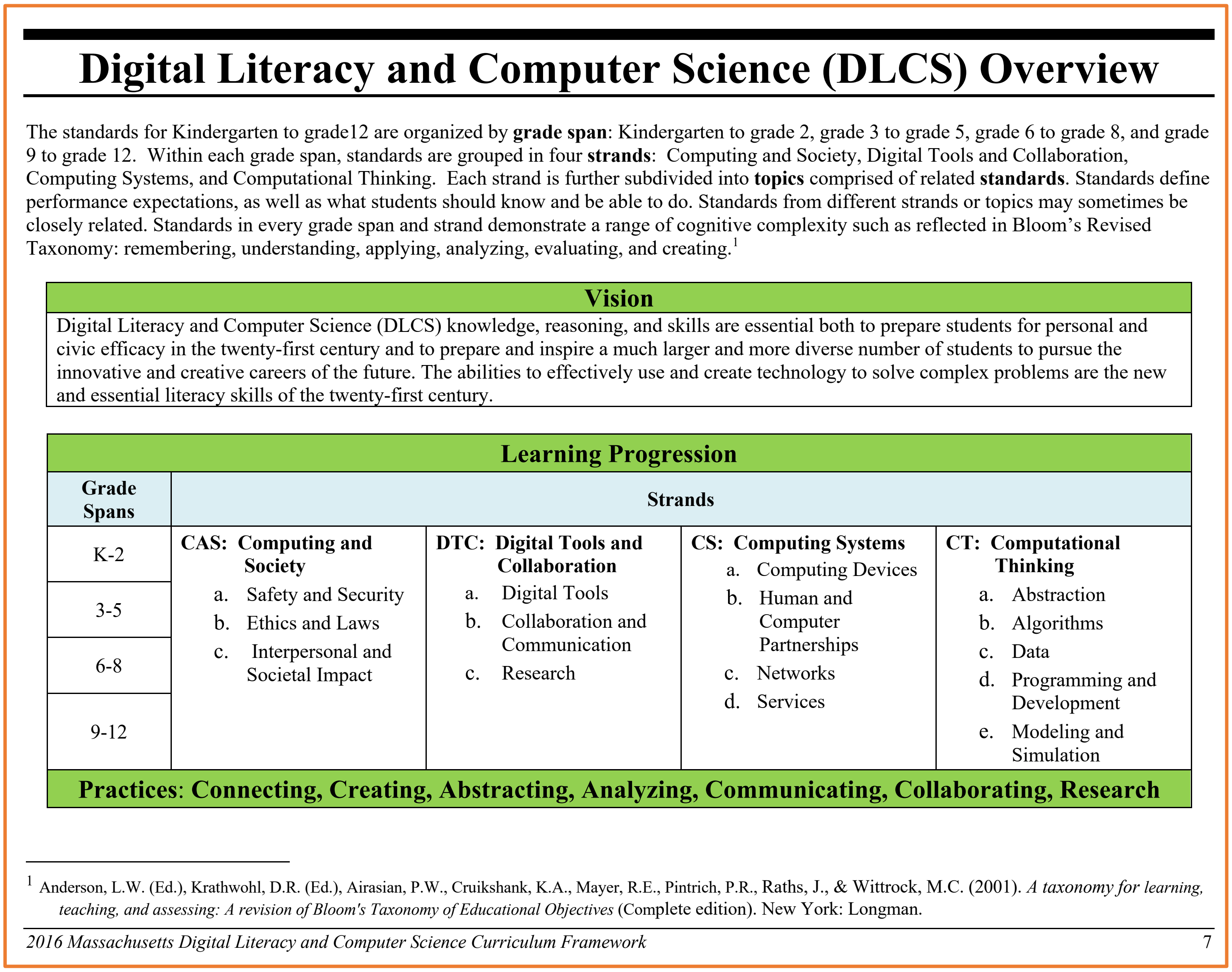
        - Report coverage and gaps
        - Based on the individual standards alignment, summarize topical coverage
            - "Partial" coverage when the curriculum addressed between 33% and 65% of the standards in the topic (to determine percentage, partially aligned standards counted as 0.5, substantially 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.
    - 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).
- Programming language or platform. Specify what programming language or platform is used, 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 review of curricula took place between March–June, 2020. 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.
- Much of this review took place during the spring 2020 COVID-19 shutdown. In a few cases, curricular options were not reviewed because the review team could not access curriculum overviews or materials and the curriculum provider did not respond after multiple attempts. The shutdown may have impacted the ability of some curriculum providers to respond.
- The review team completed only a brief review of the variety of resources that did not have coherent curriculum materials. These could, however, be excellent supplements to a DLCS curriculum. This includes many robotics kits, programmable boards, design software (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.

2016 Massachusetts Digital Literacy and Computer Science Curriculum Framework 7

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

DETAILED STANDARDS ALIGNMENT BY CURRICULUM

Grades K–2	Computer Science Fundamentals A-C (code.org)	Digital Citizenship K-2 (Common Sense Education)	Growing with Kibo (KinderLab Robotics)	Intro to Computer Science (codeSpark)	Keyboarding Without Tears (K-2)	Learn to Code Curriculum (K-2, Wonder Workshop)	MA Stem+C: Selected K-2 Units (EDC & DESE)	NYCSforALL: Computational Media Explorer - Computing Through Time	PLTW Launch: Selected K-2 Units	ScratchJr Coding Curriculum (Tufts DevTech Research Group)	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*					*Substantial*	
K-2.CAS.a.2		*Partial*								*Substantial*	
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*						
K-2.CAS.a.6	Substantial	*Substantial*			*Substantial*						*Partial*
K-2.CAS.a.7	Substantial	*Substantial*			*Substantial*						
K-2.CAS.a.8	Substantial	*Substantial*			*Substantial*						
K-2.CAS.b Ethics and Laws											
K-2.CAS.b.1	Substantial	*Substantial*	*Substantial*		*Substantial*				*Partial*	*Partial*	
K-2.CAS.b.2		*Substantial*			*Substantial*					*Substantial*	*Partial*
K-2.CAS.b.3		*Substantial*									
K-2.CAS.b.4		*Substantial*									
K-2.CAS.c Interpersonal and Societal Impact											
K-2.CAS.c.1			*Partial*	*Substantial*	*Partial*			*Substantial*	*Partial*		
K-2.CAS.c.2	*Partial*	*Partial*			*Substantial*			*Substantial*			
Kindergarten – Grade 2: Digital Tools and Collaboration (DTC)											
K-2.DTC.a Digital Tools											
K-2.DTC.a.1			*Substantial*		*Partial*	*Substantial*		*Substantial*	*Partial*	*Partial*	*Substantial*
K-2.DTC.a.2					*Substantial*			*Substantial*	*Partial*		*Substantial*
K-2.DTC.a.3	*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*
K-2.DTC.a.4	*Partial*		*Substantial*	*Substantial*		*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*
K-2.DTC.b Collaboration and Communication											
K-2.DTC.b.1	*Partial*		*Substantial*	*Partial*		*Substantial*			*Substantial*		*Partial*
K-2.DTC.b.2									*Partial*		*Partial*
K-2.DTC.b.3			*Substantial*			*Substantial*			*Partial*	*Substantial*	*Substantial*
K-2.DTC.c Research											
K-2.DTC.c.1											
K-2.DTC.c.2			*Substantial*		*Partial*				*Partial*	*Substantial*	*Partial*
K-2.DTC.c.3											*Partial*
Kindergarten – Grade 2: Computing Systems (CS)											
K-2.CS.a Computing Devices											
K-2.CS.a.1			*Partial*	*Substantial*	*Substantial*			*Substantial*	*Partial*		*Substantial*
K-2.CS.a.2			*Partial*		*Substantial*			*Substantial*	*Partial*		*Substantial*
K-2.CS.a.3			*Substantial*	*Substantial*	*Substantial*			*Substantial*	*Partial*		*Substantial*
K-2.CS.a.4			*Substantial*		*Substantial*			*Substantial*	*Substantial*		*Substantial*
K-2.CS.b Human and Computer Partnerships											
K-2.CS.b.1	*Partial*		*Substantial*	*Substantial*	*Partial*	*Partial*		*Substantial*	*Substantial*	*Partial*	
K-2.CS.b.2			*Substantial*			*Partial*		*Substantial*		*Partial*	
K-2.CS.b.3			*Substantial*	*Substantial*	*Substantial*			*Substantial*	*Partial*		
K-2.CS.c Networks											
K-2.CS.c.1				*Substantial*	*Substantial*						
K-2.CS.d Services											
NA											
Kindergarten – Grade 2: Computational Thinking (CT)											
K-2.CT.a Abstraction											
K-2.CT.a.1	Substantial		*Substantial*	*Substantial*	*Partial*	*Substantial*	Substantial	*Substantial*	*Substantial*		*Substantial*
K-2.CT.b Algorithms											
K-2.CT.b.1	Substantial		*Substantial*	*Substantial*		*Substantial*	Substantial	*Substantial*	*Substantial*	*Substantial*	*Substantial*
K-2.CT.b.2	Substantial		*Substantial*	*Substantial*		*Substantial*	Substantial	*Substantial*	*Substantial*	*Substantial*	*Substantial*
K-2.CT.b.3	Substantial		*Substantial*	*Substantial*		*Substantial*	Substantial	*Substantial*	*Substantial*	*Substantial*	*Substantial*
K-2.CT.c Data											
K-2.CT.c.1	*Partial*		*Partial*				Substantial	*Substantial*	*Substantial*		*Partial*
K-2.CT.c.2	*Partial*		*Substantial*			*Partial*	Substantial	*Partial*			*Partial*
K-2.CT.c.3	*Partial*		*Substantial*	*Substantial*		*Partial*	Substantial	*Partial*	*Substantial*		
K-2.CT.c.4			*Substantial*				Substantial	*Partial*	*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*	*Substantial*	*Substantial*
K-2.CT.d.2	Substantial		*Substantial*	*Substantial*		*Substantial*	Substantial	*Substantial*	*Substantial*	*Substantial*	*Substantial*
K-2.CT.d.3	Substantial		*Substantial*	*Substantial*		*Substantial*	Substantial	*Substantial*	*Substantial*	*Substantial*	*Substantial*
K-2.CT.e Modeling and Simulation											
K-2.CT.e.1			*Substantial*				Substantial		*Partial*		
K-2.CT.e.2			*Partial*				Substantial				

Grades 3–5	Be Internet Awesome (Google)	Computer Science Fundamentals D-F (Code.org)	Creative Computing Curriculum (ScratchEd)	CS-First (Google)	Digital Citizenship 3-5 (Common Sense Education)	Keyboarding Without Tears 3-5	Learn to Code Curriculum (3-5; Wonder Workshop)	MA Stem+C: Selected 3-5 Units (EDC & DESE)	NYC CSforALL: Computational Media Creator - Build My City	PLTW Launch: Selected 3-5 Units	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*					
3-5 CAS.a.2	*Substantial*				*Substantial*	*Partial*					Partial
3-5 CAS.a.3	*Substantial*	*Substantial*			*Substantial*	*Substantial*				Partial	Substantial
3-5 CAS.a.4	*Substantial*				*Substantial*	*Partial*					Partial
3-5 CAS.a.5	*Substantial*	*Partial*			*Substantial*	*Substantial*					Partial
3-5 CAS.a.6	*Substantial*	*Substantial*			*Substantial*	*Substantial*					Partial
3-5 CAS.a.7	*Substantial*	*Substantial*			*Substantial*	*Substantial*					Partial
3-5.CAS.b Ethics and Laws											
3-5.CAS.b.1						*Substantial*					Partial
3-5.CAS.b.2		*Substantial*			*Substantial*						Partial
3-5.CAS.b.3		*Partial*			*Substantial*						Partial
3-5.CAS.b.4		*Partial*			*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					*Substantial*						
3-5.CAS.c.3											
3-5.CAS.c.4		*Partial*				*Partial*				Substantial	
3-5.CAS.c.5						*Substantial*					
3-5.CAS.c.6		*Partial*									
3-5.CAS.c.7	*Substantial*	*Substantial*			*Substantial*	*Substantial*					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										Partial	Substantial
3-5.DTC.a.3						*Partial*	*Substantial*		Substantial	Substantial	Substantial
3-5.DTC.b Collaboration and Communication											
3-5.DTC.b.1		*Partial*	*Substantial*	*Partial*		*Partial*	*Substantial*		Substantial	Substantial	Substantial
3-5.DTC.b.2			*Substantial*				*Substantial*		Partial		Substantial
3-5.DTC.c Research											
3-5.DTC.c.1						*Partial*					Substantial
3-5.DTC.c.2											Substantial
3-5.DTC.c.3						*Partial*					Substantial
3-5.DTC.c.4						*Partial*					Substantial
3-5.DTC.c.5						*Substantial*	*Substantial*			Partial	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						*Partial*			Substantial	Partial	
3-5.CS.a.2						*Partial*			Substantial	Partial	
3-5.CS.a.3											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	
3-5.CS.b.2		*Partial*					*Substantial*				
3-5.CS.b.3											
3-5.CS.c Networks											
3-5.CS.c.1		*Partial*								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		*Substantial*					*Partial*	Substantial	Substantial	Substantial	Partial
3-5.CT.a.2		*Substantial*		*Partial*					Substantial	Partial	Substantial
3-5.CT.a.3		*Substantial*	*Substantial*	*Partial*			*Substantial*	Substantial	Substantial	Substantial	Partial

Grades 3–5	Be Internet Awesome (Google)	Computer Science Fundamentals D-F (Code.org)	Creative Computing Curriculum (ScratchEd)	CS-First (Google)	Digital Citizenship 3-5 (Common Sense Education)	Keyboarding Without Tears 3-5	Learn to Code Curriculum (3-5; Wonder Workshop)	MA Stem+C: Selected 3-5 Units (EDC & DESE)	NYC CSforALL: Computational Media Creator - Build My City	PLTW Launch: Selected 3-5 Units	Technokids: Selected 3-5 Projects
3-5.CT.b Algorithms											
3-5.CT.b.1		*Substantial*		*Substantial*			*Substantial*	Substantial	Substantial	Substantial	Substantial
3-5.CT.b.2		*Substantial*	*Substantial*	*Substantial*			*Substantial*	Substantial	Substantial	Substantial	Substantial
3-5.CT.b.3		*Substantial*	*Substantial*	*Substantial*			*Substantial*	Substantial	Substantial	Substantial	Substantial
3-5.CT.b.4		*Substantial*	*Substantial*	*Substantial*			*Substantial*	Substantial	Substantial	Substantial	Substantial
3-5.CT.b.5		*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		Substantial	Substantial
3-5.CT.d Programming and Development											
3-5.CT.d.1		*Substantial*	*Substantial*	*Substantial*			*Substantial*		Substantial	Substantial	Substantial
3-5.CT.d.2		*Substantial*	*Substantial*	*Substantial*			*Substantial*	Substantial	Substantial	Substantial	Substantial
3-5.CT.d.3		*Substantial*	*Substantial*	*Substantial*			*Substantial*	Substantial	Substantial	Substantial	Substantial
3-5.CT.d.4		*Substantial*	*Substantial*	*Substantial*			*Partial*	Substantial	Substantial	Substantial	Substantial
3-5.CT.e Modeling and Simulation											
3-5.CT.e.1				*Partial*			*Partial*	Substantial	Substantial	Partial	
3-5.CT.e.2		*Substantial*		*Partial*				Substantial		Partial	
3-5.CT.e.3		*Substantial*						Substantial		Substantial	

Grades 6–8	Bootstrap Algebra (Brown University)	Computational Thinking Curriculum (MIT App Inventor)	Computer Science Discoveries (code.org)	Computer Skills - Digital Savvy (CompuScholar)	Computing Ideas (CodeHS)	Creative Computing Curriculum (ScratchEd)	Digital Citizenship 6-8 (Common Sense Education)	Intro to CS (Microsoft MakeCode)	NFTE Startup Tech	Project GUTS	PLTW Gateway: Computer Science Units
Grades 6 to 8: Computing and Society [CAS]											
6-8.CAS.a Safety and Security											
6-8.CAS.a.1			*Substantial*	*Substantial*	*Substantial*		*Partial*				
6-8.CAS.a.2				*Partial*	*Substantial*		*Substantial*				
6-8.CAS.a.3			*Substantial*	*Substantial*	*Substantial*	*Partial*	*Substantial*		*Partial*		*Partial*
6-8.CAS.a.4				*Substantial*	*Substantial*	*Partial*	*Substantial*				*Partial*
6-8.CAS.a.5				*Substantial*	*Partial*	*Partial*	*Substantial*				
6-8.CAS.b Ethics and Laws											
6-8.CAS.b.1	*Partial*		*Substantial*	*Substantial*	*Partial*		*Substantial*				
6-8.CAS.b.2			*Substantial*	*Substantial*	*Substantial*						
6-8.CAS.b.3			*Substantial*	*Substantial*	*Substantial*		*Substantial*				
6-8.CAS.b.4			*Substantial*		*Substantial*		*Substantial*				
6-8.CAS.b.5			*Substantial*	*Substantial*	*Substantial*						
6-8.CAS.b.6				*Partial*							*Substantial*
6-8.CAS.b.7			*Substantial*	*Partial*							
6-8.CAS.b.8				*Partial*							*Partial*
6-8.CAS.b.9			*Substantial*								
6-8.CAS.c Interpersonal and Societal Impact											
6-8.CAS.c.1			*Substantial*	*Partial*	*Substantial*				*Substantial*		*Partial*
6-8.CAS.c.2				*Substantial*					*Substantial*		*Partial*
6-8.CAS.c.3			*Substantial*		*Substantial*						
6-8.CAS.c.4			*Partial*				*Partial*				
6-8.CAS.c.5			*Partial*				*Partial*				
Grades 6 to 8: Digital Tools and Collaboration [DTC]											
6-8.DTC.a Digital Tools											
6-8.DTC.a.1			*Substantial*	*Substantial*				*Partial*	*Substantial*		
6-8.DTC.a.2	*Partial*			*Substantial*	*Partial*				*Partial*		*Substantial*
6-8.DTC.a.3			*Substantial*	*Substantial*	*Partial*				*Substantial*		*Substantial*
6-8.DTC.a.4	*Partial*		*Substantial*	*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*	*Partial*			*Partial*		*Partial*
6-8.DTC.b.2	*Substantial*		*Substantial*	*Substantial*		*Substantial*					
6-8.DTC.b.3				*Substantial*		*Partial*					
6-8.DTC.c Research											
6-8.DTC.c.1			*Substantial*	*Substantial*							
6-8.DTC.c.2				*Partial*	*Substantial*				*Partial*		
6-8.DTC.c.3			*Substantial*	*Substantial*					*Partial*		
6-8.DTC.c.4			*Substantial*	*Partial*					*Partial*		*Partial*
6-8.DTC.c.5				*Substantial*							
Grades 6 to 8: Computing Systems [CS]											
6-8.CS.a Computing Devices											
6-8.CS.a.1				*Substantial*	*Substantial*						
6-8.CS.a.2				*Substantial*	*Substantial*				*Partial*		
6-8.CS.a.3			*Substantial*	*Substantial*	*Substantial*			*Partial*			*Substantial*
6-8.CS.a.4		*Partial*				*Substantial*		*Substantial*			*Substantial*
6-8.CS.a.5		*Partial*	*Substantial*			*Substantial*		*Substantial*	*Partial*		*Substantial*
6-8.CS.a.6		*Partial*	*Substantial*	*Partial*					*Partial*		*Substantial*
6-8.CS.a.7			*Substantial*	*Substantial*				*Partial*			*Substantial*
6-8.CS.b Human and Computer Partnerships											
6-8.CS.b.1			*Substantial*						*Partial*		*Partial*
6-8.CS.b.2			*Substantial*		*Partial*				*Partial*		*Partial*
6-8.CS.c Networks											
6-8.CS.c.1				*Substantial*	*Partial*			*Partial*			*Substantial*
6-8.CS.c.2				*Substantial*	*Substantial*						
6-8.CS.c.3			*Substantial*	*Partial*	*Substantial*				*Partial*		*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*	*Partial*	*Partial*
6-8.CT.a.2	*Substantial*	*Substantial*	*Substantial*	*Partial*	*Substantial*	*Substantial*			*Substantial*	*Substantial*	*Substantial*
6-8.CT.a.3		*Partial*	*Substantial*		*Substantial*	*Substantial*			*Partial*		*Substantial*

Grades 6–8	Bootstrap Algebra (Brown University)	Computational Thinking Curriculum (MIT App Inventor)	Computer Science Discoveries (code.org)	Computer Skills - Digital Savvy (CompuScholar)	Computing Ideas (CodeHS)	Creative Computing Curriculum (ScratchEd)	Digital Citizenship 6-8 (Common Sense Education)	Intro to CS (Microsoft MakeCode)	NFTE Startup Tech	Project GUTS	PLTW Gateway: Computer Science Units
6-8.CT.b Algorithms											
6-8.CT.b.1	*Substantial*	*Substantial*	*Substantial*	*Partial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*
6-8.CT.b.2	*Substantial*	*Substantial*		*Partial*	*Substantial*	*Substantial*		*Substantial*	*Partial*	*Substantial*	*Partial*
6-8.CT.b.3	*Substantial*	*Substantial*	*Substantial*	*Partial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*
6-8.CT.b.4	*Partial*	*Substantial*	*Substantial*	*Partial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Partial*
6-8.CT.b.5	*Substantial*	*Partial*		*Partial*	*Substantial*	*Substantial*		*Partial*	*Partial*	*Substantial*	*Partial*
6-8.CT.c Data											
6-8.CT.c.1					*Partial*	*Substantial*		*Partial*			
6-8.CT.c.2		*Partial*				*Partial*			*Partial*		
6-8.CT.c.3		*Substantial*	*Substantial*	*Substantial*				*Partial*	*Substantial*		*Substantial*
6-8.CT.c.4		*Partial*	*Substantial*	*Substantial*					*Partial*		*Partial*
6-8.CT.c.5		*Substantial*	*Substantial*	*Partial*					*Partial*		*Substantial*
6-8.CT.d Programming and Development											
6-8.CT.d.1	*Partial*	*Substantial*	*Substantial*	*Partial*	*Substantial*	*Substantial*		*Substantial*	*Partial*	*Substantial*	*Substantial*
6-8.CT.d.2	*Substantial*	*Substantial*			*Substantial*	*Substantial*		*Substantial*	*Partial*	*Substantial*	*Substantial*
6-8.CT.d.3	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*
6-8.CT.d.4	*Partial*	*Substantial*	*Substantial*	*Partial*	*Substantial*	*Substantial*		*Substantial*	*Partial*	*Substantial*	*Substantial*
6-8.CT.d.5		*Substantial*	*Substantial*	*Partial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*
6-8.CT.d.6	*Substantial*	*Substantial*	*Substantial*	*Partial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*
6-8.CT.e Modeling and Simulation											
6-8.CT.e.1			*Substantial*						*Partial*	*Substantial*	
6-8.CT.e.2			*Substantial*							*Substantial*	
6-8.CT.e.3			*Substantial*						*Partial*	*Substantial*	

Grades 9–12	Beauty and Joy of Computing (UC Berkely)	Bootstrap:Data Science (Brown University)	Code.org Computer Science Principles	CodeHS Computer Science Principles	CodeHS Intro to Computer Science with JavaScript	CodeHS Intro to Cybersecurity	Computational Thinking and Problem Solving (CSforMA)	Cyber Literacy II	Cyber Science	Cyber Society	Exploring Computer Science (UCLA)	Mobile Computer Science Principles	PLTW Computer Science Essentials	PLTW Computer Science Principles	PLTW Cybersecurity
Grades 9–12: Computing and Society (CAS)															
9-12.CAS.a Safety and Security															
9-12.CAS.a.1											*Partial*				
9-12.CAS.a.2	*Substantial*					*Substantial*		*Substantial*	*Partial*	*Substantial*	*Substantial*	*Substantial*		*Partial*	
9-12.CAS.a.3	*Substantial*					*Substantial*		*Substantial*			*Substantial*			*Partial*	*Partial*
9-12.CAS.a.4	*Substantial*					*Substantial*					*Substantial*				
9-12.CAS.a.5										*Substantial*	*Substantial*				
9-12.CAS.a.6						*Substantial*				*Substantial*	*Substantial*				
9-12.CAS.b Ethics and Laws															
9-12.CAS.b.1															*Partial*
9-12.CAS.b.2	*Substantial*	*Substantial*	*Substantial*			*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*
9-12.CAS.b.3	*Substantial*	*Substantial*	*Substantial*			*Substantial*		*Substantial*	*Partial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*
9-12.CAS.b.4						*Partial*	*Substantial*			*Substantial*					
9-12.CAS.c Interpersonal and Societal Impact															
9-12.CAS.c.1	*Substantial*		*Substantial*	*Substantial*					*Substantial*	*Substantial*	*Substantial*	*Substantial*			
9-12.CAS.c.2	*Substantial*		*Substantial*	*Substantial*		*Partial*	*Partial*		*Partial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*
9-12.CAS.c.3							*Partial*			*Substantial*				*Partial*	
9-12.CAS.c.4	*Partial*		*Partial*				*Substantial*							*Partial*	
9-12.CAS.c.5	*Substantial*		*Substantial*	*Substantial*			*Partial*	*Partial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Partial*
9-12.CAS.c.6						*Substantial*	*Substantial*		*Partial*					*Substantial*	*Substantial*
9-12.CAS.c.7	*Partial*		*Partial*	*Partial*			*Partial*				*Substantial*	*Partial*	*Partial*	*Partial*	*Partial*
9-12.CAS.c.8	*Substantial*		*Substantial*						*Partial*	*Substantial*	*Substantial*	*Substantial*			
9-12.CAS.c.9	*Substantial*		*Substantial*			*Substantial*		*Partial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*
Grades 9–12: Digital Tools and Collaboration (DTC)															
9-12.DTC.a Digital Tools															
9-12.DTC.a.1	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*
9-12.DTC.a.2	*Substantial*		*Substantial*	*Substantial*			*Substantial*			*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*
9-12.DTC.b Collaboration and Communication															
9-12.DTC.b.1	*Substantial*	*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Partial*
9-12.DTC.b.2	*Partial*	*Partial*	*Partial*	*Partial*			*Partial*					*Partial*	*Partial*	*Partial*	*Substantial*
9-12.DTC.c Research															
9-12.DTC.c.1	*Substantial*	*Substantial*	*Substantial*	*Substantial*		*Partial*	*Substantial*			*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Partial*
9-12.DTC.c.2		*Partial*					*Partial*	*Substantial*		*Substantial*					
9-12.DTC.c.3		*Partial*	*Substantial*				*Substantial*	*Substantial*		*Substantial*	*Substantial*			*Partial*	*Substantial*
9-12.DTC.c.4	*Substantial*	*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Partial*	*Substantial*		*Substantial*	*Substantial*
9-12.DTC.c.5	*Substantial*	*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*				*Substantial*	*Substantial*		*Substantial*	*Partial*
Grades 9–12: Computing Systems (CS)															
9-12.CS.a Computing Devices															
9-12.CS.a.1							*Partial*	*Substantial*	*Substantial*					*Partial*	*Substantial*
9-12.CS.a.2	*Substantial*		*Substantial*					*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Partial*	*Substantial*	*Substantial*
9-12.CS.a.3								*Substantial*							*Substantial*
9-12.CS.a.4	*Substantial*						*Partial*	*Substantial*			*Partial*	*Substantial*	*Substantial*	*Substantial*	*Partial*
9-12.CS.a.5			*Substantial*								*Substantial*			*Substantial*	*Substantial*
9-12.CS.a.6	*Substantial*		*Substantial*	*Substantial*								*Substantial*		*Substantial*	
9-12.CS.b Human and Computer Partnerships															
9-12.CS.b.1	*Partial*	*Partial*									*Substantial*	*Partial*	*Partial*	*Partial*	*Substantial*
9-12.CS.c Networks															
9-12.CS.c.1	*Substantial*		*Substantial*	*Partial*		*Substantial*	*Substantial*		*Substantial*	*Partial*		*Substantial*		*Substantial*	*Substantial*
9-12.CS.c.2	*Substantial*		*Partial*	*Partial*		*Substantial*			*Substantial*	*Substantial*		*Substantial*		*Substantial*	*Substantial*
9-12.CS.c.3	*Partial*					*Substantial*								*Substantial*	*Substantial*
9-12.CS.d Services															
9-12.CS.d.1	*Partial*		*Partial*	*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*	*Substantial*	*Substantial*		*Substantial*		*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Partial*
9-12.CT.b Algorithms															
9-12.CT.b.1	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*	
9-12.CT.b.2	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*	
9-12.CT.b.3	*Substantial*		*Substantial*	*Substantial*								*Substantial*		*Substantial*	
9-12.CT.b.4	*Partial*	*Substantial*	*Partial*	*Partial*	*Substantial*		*Substantial*		*Partial*		*Substantial*	*Partial*		*Substantial*	
9-12.CT.b.5											*Substantial*				
9-12.CT.c Data															
9-12.CT.c.1	*Substantial*		*Substantial*	*Substantial*					*Partial*		*Partial*	*Substantial*	*Partial*	*Substantial*	
9-12.CT.c.2	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Partial*		*Partial*			*Substantial*		*Substantial*	
9-12.CT.c.3	*Partial*	*Substantial*	*Substantial*	*Partial*			*Substantial*		*Substantial*		*Substantial*	*Partial*		*Substantial*	
9-12.CT.c.4	*Partial*	*Substantial*	*Partial*				*Substantial*				*Substantial*	*Substantial*		*Partial*	*Partial*
9-12.CT.c.5		*Substantial*			*Substantial*		*Partial*				*Substantial*				*Partial*
9-12.CT.d Programming and Development															
9-12.CT.d.1	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*		*Substantial*				*Substantial*	*Substantial*	*Substantial*	*Substantial*	
9-12.CT.d.2	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Partial*	*Substantial*		*Partial*	*Substantial*	*Substantial*	*Substantial*	
9-12.CT.d.3	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*				*Substantial*		*Substantial*	*Substantial*			
9-12.CT.d.4	*Substantial*	*Partial*	*Substantial*	*Substantial*	*Partial*		*Substantial*				*Substantial*	*Substantial*		*Substantial*	
9-12.CT.d.5	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*	
9-12.CT.d.6	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*	
9-12.CT.d.7		*Substantial*		*Substantial*	*Substantial*								*Partial*	*Partial*	
9-12.CT.d.8	*Substantial*		*Substantial*	*Substantial*					*Substantial*		*Substantial*	*Substantial*	*Substantial*	*Substantial*	
9-12.CT.d.9		*Substantial*			*Substantial*			*Substantial*				*Substantial*	*Partial*	*Partial*	
9-12.CT.d.10	*Substantial*	*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*	*Substantial*	*Substantial*	
9-12.CT.d.12											*Substantial*				
9-12.CT.e Modeling and Simulation															
9-12.CT.e.1	*Substantial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*					*Partial*	*Substantial*	*Substantial*	*Substantial*	*Substantial*	
9-12.CT.e.2											*Substantial*			*Partial*	

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 (www.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.