CODE KIT CURRICULAR CROSSWALK

This chart provides an overview of the CSTA and NGSS Standards that can be met by, or extended to meet, the Code Kit lessons. Use the Code Master Workbook (after going through tutorials) and Invention Log (after creating inventions) as a way to assess whether your students have fulfilled these standards. See Curriculum Guide for additional information on assessment strategies.

CSTA ALIGNMENT

IDENTIFIER	CSTA STANDARD	FRAME CONCEPT	FRAMEWORK PRACTICE	LESSONS THAT MEET THIS STANDARDS	WAYS TO ADD THIS STANDARD TO ANY LESSON
ELEMENTARY (3–5)					
1B-A-2-1	Apply collaboration strategies to support problem solving within the design cycle of a program.	Algorithms and Programs	Collaborating	Invention Lessons: Ultimate Shootout; Hot Potato...of Doom!; Rockstar Guitar	
1B-A-5-3	Create a plan as part of the iterative design process, both independently and with diverse collaborative teams (e.g., storyboard, flowchart, pseudo-code, story map).	Algorithms and Programs	Creating Computational Artifacts	Invention Lessons: Ultimate Shootout; Hot Potato...of Doom!; Rockstar Guitar	
1B-A-5-4	Construct programs, in order to solve a problem or for creative expression, that include sequencing, events, loops, conditionals, parallelism, and variables, using a block-based visual programming language or text-based language, both independently and collaboratively (e.g., pair programming).	Algorithms and Programs	Creating Computational Artifacts	- All Tutorial Lessons: Inputs and Outputs; Loops; Logic; Variables; Functions - Invention Lessons: Ultimate Shootout; Hot Potato...of Doom!; Rockstar Guitar"	
1B-A-5-5	Use mathematical operations to change a value stored in a variable.	Algorithms and Programs	Creating Computational Artifacts		Meet this standard during the Remix phase in any lesson by asking students to track score or add a second player's score in their games.
1B-A-3-7	Construct and execute an algorithm (set of step-by-step instructions) which includes sequencing, loops, and conditionals to accomplish a task, both independently and collaboratively, with or without a computing device.	Algorithms and Programs	Recognizing and Defining Computational Problems	- Hello World - All Tutorial Lessons: Inputs and Outputs; Loops; Logic; Variables; Functions - Invention Lessons: Ultimate Shootout; Hot Potato...of Doom!; Rockstar Guitar	
1B-A-6-8	Analyze and debug (fix) an algorithm that includes sequencing, events, loops, conditionals, parallelism, and variables.	Algorithms and Programs	Testing and Refining	Invention Lessons: Ultimate Shootout; Hot Potato...of Doom!; Rockstar Guitar	Meet this standard in any lesson by presenting students with an altered version of the invention code that you know doesn't work. Challenge students to fix it. The debugging checklist can be used as a reference.
1B-C-7-9	Model how a computer system works. [Clarification: Only includes basic elements of a computer system, such as input, output, processor, sensors, and storage.]	Computing Systems	Communicating about Computing		Meet this standard in any lesson by asking students to describe how inputs, outputs, and the codeBit function in their circuit.
1B-C-6-11	Identify, using accurate terminology, simple hardware and software problems that may occur during use, and apply strategies for solving problems (e.g., reboot device, check for power, check network availability, close and reopen app).	Computing Systems	Testing and Refining	Hello World	Meet this standard in any lesson by asking students, before they playtest their inventions, to predict what problems they may encounter with the hardware or software.
1B-D-4-14	Use numeric values to represent non-numeric ideas in the computer (binary, ASCII, pixel attributes such as RGB).	Data and Analysis	Developing and Using Abstractions	Tutorial Lessons: Loops; Logic; Variables; Functions	Meet this standard during the Remix phase in any lesson by challenging students to add the [COLOR RGB] block to their code. This block allows students to customize the color of a pixel by specifying values for Red, Green, and Blue.

CSTA ALIGNMENT

IDENTIFIER	CSTA STANDARD	FRAME CONCEPT	FRAMEWORK PRACTICE	LESSONS THAT MEET THIS STANDARDS	WAYS TO ADD THIS STANDARD TO ANY LESSON
MIDDLE (6–8)					
2-A-2-1	Solicit and integrate peer feedback as appropriate to develop or refine a program.	Algorithms and Programming	Collaborating	All Invention Lessons: Ultimate Shootout; Hot Potato...of Doom!; Rockstar Guitar; Tug of War	Meet this standard in any lesson by inviting students to share their prototypes with peers during the Remix phase of the lesson, and then integrate the feedback into further prototypes.
2-A-7-4	Interpret the flow of execution of algorithms and predict their outcomes.	Algorithms and Programming	Communicating about Computing		Meet this standard in any lesson by asking students to read the code before they run it for the first time, and predict what their circuit will do when they run it.
2-A-5-6	Develop programs, both independently and collaboratively, that include sequences with nested loops and multiple branches.	Algorithms and Programming	Creating Computational Artifacts	Tutorial Lessons: Loops; Logic; Variables; Functions	
2-A-5-7	Create variables that represent different types of data and manipulate their values.	Algorithms and Programming	Creating Computational Artifacts	Tutorial Lessons: Loops and Variables	
2-A-4-8	Define and use procedures that hide the complexity of a task and can be reused to solve similar tasks.	Algorithms and Programming	Developing and Using Abstractions	Invention Lesson: Tug of War	
2-A-6-10	Use an iterative design process (e.g., define the problem, generate ideas, build, test, and improve solutions) to solve problems, both independently and collaboratively.	Algorithms and Programming	Testing and Refining	- All Invention Lessons: Ultimate Shootout; Hot Potato...of Doom!; Rockstar Guitar; Tug of War - Coding Challenge	Meet this standard in any lesson by having students use the Invention Log to guide the design process and record their explorations.
HIGH (9–10)					
3A-A-2-1	Design and develop a software artifact working in a team.	Algorithms and Programming	Collaborating	- Invention Lesson: Tug of War - Coding Challenge	
3A-A-5-4	Design, develop, and implement a computing artifact that responds to an event (e.g., robot that responds to a sensor, mobile app that responds to a text message, LED monster that responds to a broadcast).	Algorithms and Programming	Creating Computational Artifacts	- Invention Lesson: Tug of War - Coding Challenge	
3A-C-5-14	Create, extend, or modify existing programs to add new features and behaviors using different forms of inputs and outputs (e.g., inputs such as sensors, mouse clicks, data sets; outputs such as text, graphics, sounds).	Computing Systems	Communicating About Computing	- Invention Lesson: Tug of War - Coding Challenge	
HIGH (11–12)					
3B-A-7-3	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality).	Algorithms and Programmin	Communicating About Computing	- Invention Lesson: Tug of War - Coding Challenge	

NGSS ALIGNMENT

IDENTIFIER	PERFORMANCE EXPECTATION	LESSONS THAT MEET THIS STANDARDS
ELEMENTARY (3–5)		
3-5-ETS1-2	Generate and compare multiple possible solutions to a problem based on how well each is likely to meet the criteria and constraints of the problem.	Invention Lessons: Ultimate Shootout; Hot Potato...of Doom!; Rockstar Guitar
3-5-ETS1-3	Plan and carry out fair tests in which variables are controlled and failure points are considered to identify aspects of a model or prototype that can be improved.	Invention Lessons: Ultimate Shootout; Hot Potato...of Doom!; Rockstar Guitar
MIDDLE (6–8)		
MS-ETS1-1	Define the criteria and constraints of a design problem with sufficient precision to ensure a successful solution, taking into account relevant scientific principles and potential impacts on people and the natural environment that may limit possible solutions.	Coding Challenge
MS-ETS1-2	Evaluate competing design solutions using a systematic process to determine how well they meet the criteria and constraints of the problem.	- All Invention Lessons: Ultimate Shootout; Hot Potato...of Doom!; Rockstar Guitar; Tug of War - Coding Challenge
MS-ETS1-3	Analyze data from tests to determine similarities and differences among several design solutions to identify the best characteristics of each that can be combined into new solutions to better meet the criteria for success.	- All Invention Lessons: Ultimate Shootout; Hot Potato...of Doom!; Rockstar Guitar; Tug of War - Coding Challenge
HIGH (9–12)		
HS-ETS1-1	Analyze a major global challenge to specify qualitative and quantitative criteria and constraints for solutions that account for societal needs and wants.	Coding Challenge
HS-ETS1-2	Design a solution to a complex real-world problem by breaking it down into smaller, more manageable problems that can be solved through engineering.	Coding Challenge