TCBOE K-12 CT Progression

	K-2	3-5	6-8	9-12
Key Words	if/then, start/end, forward/backward, left/right, loop/repeat, sequence		variable, run series, algorithm, debugging, sensors, functions, iterations, operators, logic, parameters	
Computational Thinking	Create algorithms, or series of ordered steps, to solve problems.			
	Decompose a problem, into smaller, more manageable parts.			
	Collect, analyze, and represent data effectively.			
	Demonstrate an understanding of how information is represented, stored, and processed by a computer.			
	Optimize an algorithm for execution by a computer.			
	Create simulations / models to understand natural phenomena and test hypotheses.			
	Engineer software and/or hardware solutions for real-world problems.			
	Evaluate algorithms by their efficiency, correctness, and clarity.			
Computing Practice and Programming	Use hands-on learning and the physical environment to explore computing concepts.			
	Write programs using visual (block-based) programming languages.			
	Write programs using text-based programming languages.			
	Locate and debug errors in a program.			
	Read a program and translate it into English. Explain how a particular program functions.			
	Design, code, test, and execute a program that corresponds to a set of specifications.			
	Modify and create animations, and present work to teammates.		Design, develop, publish, and present products (e.g., web pages, mobile apps, animations) to demonstrate and communicate curriculum concepts.	
	Create web pages with a practical, personal, and/or societal purpose.			
	Identify strengths and limitations of different programming languages.			
Examples of Suggested Languages & Platforms	Scratch Jr.	Scratch/ScratchED		Snap!
	Dash and Dot			JavaScript
	Ozobot/Ozoblocky			HTML/CSS
	Daisy the Dinosaur	Sphero/Ollie/Sphero Edu		
	Bee-Bots/Blue-Bots		VEX/BEST Robotics	
	Move the Turtle	Hummingbird/Create Lab		
	Lego WeDo		Lego Mindstorm	
	Cubelets		Drones	
Coding Curricula	Code.org Course 1	Code.org Course 2	Code.org Course 3	Code.org Course 4	Code.org CS Principles
	Code.org CS Discoveries			
	Google CS First			
	CodeHS			
	Project Lead the Way			
	Kodable		Code Combat	

Exemplary Learning Activities														
	Determine and input a series of six sequential directions into a Bee-Bot to follow the pictures of a story (e.g., Goldilocks and the three bears) from beginning to end.	Determine and input a series of 10+ sequential directions into a Bee-Bot to navigate a maze or accomplish a basic task (e.g., find the sight word, avoid the opposite, find the sum of 2+3).	Use basic loops to repeat a sequence of commands, in order to guide fuzz balls through a maze in Kodable.	Create and present a Scratch, Jr. interactive collage involving multiple, animated characters.	Create and share an animated, interactive story using sequence, loops, and event-handlers in Code.org's PlayLab.	Remix a Scratch project to add and customize features. Debug a project to correct errors and achieve a given objective.	Draw complex shapes and patterns by decomposing and combining smaller shapes, using nested loops and randomization.	Create web pages with a practical, personal, and/or societal purpose. Read the code behind a Flappy Bird-like game and translate it into English.	Develop a model of a local ecosystem using StarLogo Nova, to simulate predator-prey relationships and population dynamics.	Use algebraic concepts to design a game that detects collisions, handles keystrokes, and determines how characters move and interact.	Build, code, and test a robot that solves a stated problem. Create a chase, escape, or platform game, and use variables to keep score.	Create an Ants vs. SomeBees action game (inspired by Plants vs. Zombies) with complicated interaction using object-oriented programming.	Evaluate U.S. and world trends by develop a geographic visualization of Twitter data using lists, and data abstraction techniques to create a modular program.	Design and develop an an app, game, website, or program to solve a real-world, community- based problem.