Computational Thinking in LEGO® Education SPIKE™ Essential Lessons

Computational thinking is a set of problem-solving skills named for the processes a computer uses to solve problems. We use these skills – decomposition, generalization, algorithmic thinking, debugging, and abstract thinking – every day to solve problems. Practicing and learning computational thinking through LEGO® Education SPIKE™ Essential lessons supports students' success in and beyond the classroom.

Key Computational Thinking Skills

This skill	Helps you...	Everyday examples
Decomposition	Break down problems into smaller parts to explain or solve the problem more easily.	Preparing to travel includes subtasks, like booking the flight, reserving a hotel, packing a suitcase, etc. Completing these subtasks makes the larger task more manageable. We can quickly and easily complete familiar tasks, such as packing a suitcase, without having to develop a "new" solution. This saves time for completing new and unfamiliar tasks like finding things to do and booking activities at the destination.
Generalization	Recognize patterns you can use to create a solution.	Have you ever assembled a piece of furniture? Then you'll understand the importance of recognizing patterns. For example, when assembling multiple drawers for a small cabinet it'll likely take you much longer to assemble the first drawer than the fourth or fifth. When we repeat steps, we learn from our mistakes and can complete the steps more quickly.
Algorithmic thinking	Visualize and organize the steps needed to achieve the desired outcome.	When you cook from a recipe, you follow a series of steps to prepare a meal. You find the ingredients, rinse the vegetables, chop the ingredients, then cook and finally serve the meal. This step-by-step process also enables us to create and implement a solution to a problem.
Evaluating and debugging	Evaluate a solution to identify areas for improvement and then debug to correct errors in the solution.	If your bread always fails to rise as it bakes, you can test the recipe to try to find the problem. By changing one element at a time, like the amount of yeast or the baking time, you'll eventually find out where it's going wrong. You test and evaluate the results to find and correct the problem. That's debugging!
Abstract thinking	Explain the chosen solution and conceptualize the idea with minimal detail.	When describing a bicycle, we use details like it has two wheels, one in front of the other. These are defining characteristics of a bicycle. Details such as its type and color add interest but aren't required. Abstract thinking is the process of filtering out specific details to create a representation of an idea we want to explain.

The table below shows how your students will learn and develop their understanding of these computational thinking skills throughout the SPIKE Essential lessons. Use this guidance to help you prepare to teach these lessons with a focus on driving the development of your students' computational thinking skills. As you prepare to teach each lesson, you can:

- Use the information in the last column to identify places in the lesson where your students will practice and learn the skills.
- Use the information in the last column to assess your students' progress. Give formative feedback about their progress with using computational thinking skills. Ask questions like these: *How did you create the program? Why did you use those programming blocks? What would you do to improve the program?*
- Expand your knowledge and start talking with your students and colleagues about computational thinking skills in an everyday language. What do they look like, how do we use them to solve problems, and what's the connection between the skills you learn from coding and the skills you use to solve everyday problems?

	Lesson	**Key Objective(s)**	**Students will use computational thinking skills when they...**
Unit: Great Adventures	1: Boat Trip	Follow instructions to create a program.	Create different program sequences with actions that control the motors to push the boat in the water **(Algorithmic thinking)**.
	2: Arctic Ride	Break a problem down into smaller parts.	Break down the problem of creating a program into smaller parts to understand which Movement Blocks to use **(Decomposition)**. Create the program sequence for Leo's round trip journey using different Movement Blocks to control the snowmobile **(Algorithmic thinking)**.
	3: Cave Car	Describe a program's sequence of events, goals, and expected outcome.	Create different program sequences with actions that control the cave car's light **(Algorithmic thinking)**.
	4: Animal Alarm	Demonstrate an understanding of cause and effect to develop a program to solve a problem.	Create a program sequence for an animal alarm using an Event Block to control when the alarm goes off **(Algorithmic thinking)**. Identify patterns and actions in the existing program that can be reused to improve the program by using another Event Block to set up actions for a different color **(Generalization)**.
	5: Underwater Quest	Develop programs that use simple loops to address a problem.	Create a program sequence to move the submarine using Motor Blocks and a loop **(Algorithmic thinking)**. Identify patterns and actions in the existing program that can be reused to change the program sequence based on modifications to the model **(Generalization)**.
	6: Treehouse Camp	Identify and fix errors in a program (test and debug).	Create a program sequence to open the treehouse roof using Motor Blocks **(Algorithmic thinking)**. Come up with a solution to fix the program. Change the program based on modifications to the model **(Evaluating and debugging)**.
	7: Great Desert Adventure	Apply computational thinking skills to solve the given problem.	Apply all of the **computational thinking skills** in an open-ended challenge to create a way for the team to get to the pyramids.

	Lesson	Key Objective(s)	Students will use computational thinking skills when they...
Unit: Amazing Amusement Park	1: The Fast Lane	Practice brainstorming to generate ideas.	Create a program sequence that uses Event Blocks to turn on the Fast Lane's light when a yellow "ticket" is shown **(Algorithmic thinking)**. Identify patterns and actions in the existing program that can be reused to create more actions using a Loop Block to automate the Fast Lane, or a Bar Graph Block to count how many times the Fast Lane has been activated **(Generalization)**.
	2: Classic Carousel	Improve and refine a prototype as part of the design process.	Create a program sequence that uses Motor Blocks to make the carousel spin **(Algorithmic thinking)**. Use patterns and actions from the existing program to refine and improve the program **(Generalization)**.
	3: The Perfect Swing	Change a solution to meet the needs or wants of others.	Create a program sequence that uses Motor Blocks to move the swing **(Algorithmic thinking)**. Identify patterns and actions in the existing program that can be reused in a loop to automate the swing and improve the program **(Generalization)**.
	4: Snack Stand	Practice testing prototypes to ensure that they meet a need by modifying and remixing a solution.	Create a program sequence that uses an Event Block to control when the snack stand will deliver a snack **(Algorithmic thinking)**. Identify patterns and actions in the existing program that can be reused to improve the program **(Generalization)**.
	5: Twirling Teacups	Modify a solution while considering a specific goal or outcome by refining and improving the prototype.	Create a program sequence for the teacups using a Motor Block for movement and Event Blocks to run parallel actions **(Algorithmic thinking)**. Identify patterns and actions in the existing program that can be reused to improve the program **(Generalization)**.
	6: The Spinning Ferris Wheel	Modify an existing solution to make it work properly.	Create a program sequence for the Ferris Wheel using Motor Blocks for movement and a loop to control the rotations **(Algorithmic thinking)**. Identify patterns and actions in the existing program that can be reused to improve the program so the Ferris Wheel also stops so that passengers can get on and off **(Generalization)**.
	7: The Most Amazing Amusement Park	Apply engineering design skills to solve a problem.	Apply all of the **computational thinking skills** in an open-ended challenge to create a new amusement park ride.

Unit: Happy Traveler	Lesson	Key Objective(s)	Students will use computational thinking skills when they…
	1: River Ferry	Develop a sequence to solve a problem. Decompose problems into smaller parts.	Break down Daniel's problem into smaller parts by discussing how to get from one place to another **(Decomposition)**. Create a program sequence for the ferry using Motor Blocks **(Algorithmic thinking)**.
	2: Taxi! Taxi!	Identify and fix errors in a program (test and debug).	Break down the problem of creating a program into smaller parts to understand which Movement Blocks to use for the taxi's trip to the art museum **(Decomposition)**. Create and program a sequence that moves the taxi forward and uses Movement Blocks to make a turn **(Algorithmic thinking)**. Identify patterns and actions in the existing program that can be reused for next trip **(Generalization)**.
	3: Hovering Helicopter	Describe the choices they've made when creating a program. Create and test automated solutions.	Create a program sequence that uses Motor Blocks to move the helicopter's rotor blades **(Algorithmic thinking)**. Look for patterns and actions that can be applied to the Tilt Sensor in the second challenge when the helicopter is tilted **(Generalization)**. Describe what they've learned about the choices they made when creating their program **(Abstract thinking)**.
	4: Swamp Boat	Identify the parts of an existing program that should be modified. Carry out tests to identify where a program can be modified.	Create a program sequence to identify crocodiles by turning on the swamp boat's light when "green" is detected **(Algorithmic thinking)**. Identify patterns and actions that can be used to modify the program by adding sounds, more lights, and bar graphs to count the crocodiles **(Generalization)**.
	5: Cable Car	Identify and fix errors in a program to ensure it works as intended (test and debug).	Create a program sequence that uses Motor Blocks to move the cable car and a loop to control it **(Algorithmic thinking)**. Come up with a solution to fix errors in the cable car's movement **(Evaluating and debugging)**.
	6: Big Bus	Test and evaluate solutions to determine whether they meet a specific need.	Create a program sequence that uses Motor Blocks and the Color Sensor to make the bus stop at the green bus stop **(Algorithmic thinking)**.
	7: Get Around Town	Apply computational thinking skills to solve a problem.	Apply all of the **computational thinking skills** in an open-ended challenge to create a way for the team to get to the SPIKE Castle.

	Lesson	Key Objective(s)	Students will use computational thinking skills when they...
Unit: Crazy Carnival Games	1: Mini Mini-Golf	Explore the basic principles of energy and their connection to an object's speed. Identify and describe the relationship between speed and energy.	Create a program sequence to get a hole-in-one using Motor Blocks to control hitting the golf ball **(Algorithmic thinking)**. Describe what they've learned about speed and energy **(Abstract thinking)**.
	2: Bowling Fun!	Predict outcomes of the changes in energy that occur when objects collide. Observe and describe the relationship between energy and force.	Create a program sequence to get a strike using Motor Blocks to control the ball's movement **(Algorithmic thinking)**. Describe what they've learned about the relationship between energy and force **(Abstract thinking)**.
	3: High Stick Hockey	Observe and describe how energy can be transferred. Predict how energy moves from place to place.	Create a program sequence using Motor Blocks to run the hockey game. Identify patterns and actions to use in a program loop **(Generalization)**. Describe what they've learned about how energy moves from place to place **(Abstract thinking)**.
	4: A-Maze-Ing	Observe and explain how interactions between two objects can impact the energy of an object.	Create a program sequence for the maze game using Motor Blocks and a sensor to control the sound **(Algorithmic thinking)**. Describe what they've learned about how interactions between two objects can impact an object's energy **(Abstract thinking)**.
	5: Avoid the Edge	Explore and describe energy conversion.	Create a program sequence for the game using the Tilt Sensor to count the number of tilts **(Algorithmic thinking)**. Describe what they've learned about energy conversion **(Abstract thinking)**.
	6: Junior Pinball	Apply ideas to refine a solution that converts energy. Test the solution to improve and refine its function.	Create a program sequence for the pinball machine using Motor Blocks to start the game **(Algorithmic thinking)**. Modify the program by identifying patterns and actions to use in a loop **(Generalization)**.
	7: Creative Carnival Games	Apply existing scientific knowledge of energy transfer and collision to solve a problem.	Apply all of the computational thinking skills in an open-ended challenge to create a new carnival game.

	Lesson	Key Objective(s)	Students will use computational thinking skills when they…
Unit: Quirky Creations	1: Good Morning Machine	Define and understand a problem. Brainstorm and iterate to create a solution.	Break down Leo's problem into smaller parts in order to define and understand the problem **(Decomposition)**. Create a program sequence for the waving machine using Motor Blocks to move the hand. Modify the program by identifying patterns and actions to use in a loop **(Generalization)**.
	2: Big Little Helper	Create a possible solution to a problem that has constraints. Improve on others' ideas to develop a new program.	Create a program sequence that controls the robot helper using Movement Blocks **(Algorithmic thinking)**. Identify patterns and actions in the existing program that can be reused to enable the robot helper to follow Daniel home **(Generalization)**.
	3: High-Tech Playground	Use the design process to improve an existing object. Develop, test, and refine prototypes.	Create a program sequence for the seesaw using Motor Blocks and the Tilt Sensor **(Algorithmic thinking)**. Identify patterns and actions in the existing program that can be reused to improve the seesaw **(Generalization)**.
	4: Trash Monster Machine	Explore the benefits of automated solutions. Refine a prototype.	Create a program sequence for the Trash Monster using Motor Blocks and the Color Sensor **(Algorithmic thinking)**. Identify patterns and actions in the existing program that can be reused to make the Trash Monster react to different-colored trash **(Generalization)**.
	5: Winning Goal	Identify the failure points of a model or program. Consider failure points in order to make improvements.	Break down the problem by identifying the failures causing the program to not be able to save a goal **(Decomposition)**. Create a program sequence that uses Motor Blocks to move the goal, making it harder to score **(Algorithmic thinking)**. Identify patterns and actions in the existing program that can be reused to improve the program using a loop **(Generalization)**.
	6: Literary Randomizer	Define success criteria to help evaluate a solution. Compare and contrast different solutions.	Create a program sequence that picks a book genre using Light Blocks and a loop to control the random selection **(Algorithmic thinking)**.
	7: Your School Creation	Apply engineering design skills in order to solve a problem. Practice brainstorming as part of the design process.	Apply all of the **computational thinking skills** in an open-ended challenge to design a new creation for the team's classroom.