LEGO® Education WeDo 2.0 Computational Thinking

Teacher's Guide

WeDo 2.0

LEGO education

Table of Contents

The LEGO® Education community is an online community for teachers, administrators, and other professionals in education.
It is a place to connect and share ideas, engage in discussions, and share lesson plans and projects.
The LEGO Education community is only in English.

Developing Computational Thinking with WeDo 2.0 Projects

In this chapter, you will discover how you can use WeDo 2.0 to develop computational thinking skills in a science context.

Develop Computational Thinking with LEGO® Education WeDo 2.0 Projects

LEGO® Education is pleased to present these projects, which have been specifically designed for use in elementary school classrooms to develop students' computational thinking skills.

Computational thinking is a set of skills that everybody can use to solve everyday life problems. In WeDo 2.0, these skills are developed throughout each phase of every project. Development opportunities have been identified for you in each of the projects, it is up to you to focus on the ones that are most relevant to you and your students.

Every project in WeDo 2.0 combines the use of the LEGO bricks with an iconic programming language, enabling your students to find solutions to problems while being introduced to programming principles.

WeDo 2.0 develops computational thinking through coding activities, which bring students' creations to life, generating smiles and the desire to discover more.

Computer Science, Computational Thinking, Coding

While the science and engineering fields originated in the early ages of humankind, computer science has a much younger history. Nevertheless, this young discipline has influenced not only the way we approach science and engineering, but also the way we live our lives.

Computer Science is a STEM discipline, sharing attributes with science, technology, engineering, and mathematics.

All STEM disciplines present opportunities to develop a mindset and a lifelong set of practices. Among these practices, we find the ability to ask questions, to design solutions, and to communicate results.

Computational thinking is another one of these practices. It is a way in which we think and it is a way in which everybody can solve problems.

Computational thinking can be described as a group of skills, one of these skills being algorithmic thinking. “Code” or “coding” can be used to describe the action of creating an algorithm.

Coding is therefore one vehicle by which to develop computational thinking within a STEM context.

STEM Disciplines

Science, Technology, Engineering, Mathematics, Computer Science

Develop a mindset and life long set of practices

1. Ask questions and solve problems.
2. Use models.
3. Design prototypes.
4. Investigate.
5. Analyze and interpret data.
6. Use computational thinking.
    a. Decompose
    b. Abstract
    c. Think algorythmically (code)
    d. Evaluate
    e. Generalise
7. Engage in argument from evidence.
8. Obtain, evaluate, and communicate information.

What is Computational thinking ?

The expression “computational thinking” was first used by Seymour Papert, but Professor Jeannette Wing is known to have popularized the idea. She defined computational thinking as:

”the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent.” (Wing, 2011)

Computational thinking is used in various fields and situations, and we use it in our daily lives. Computational thinking skills are present in science, engineering, and mathematics. These skills can be defined as the following:

Decomposition

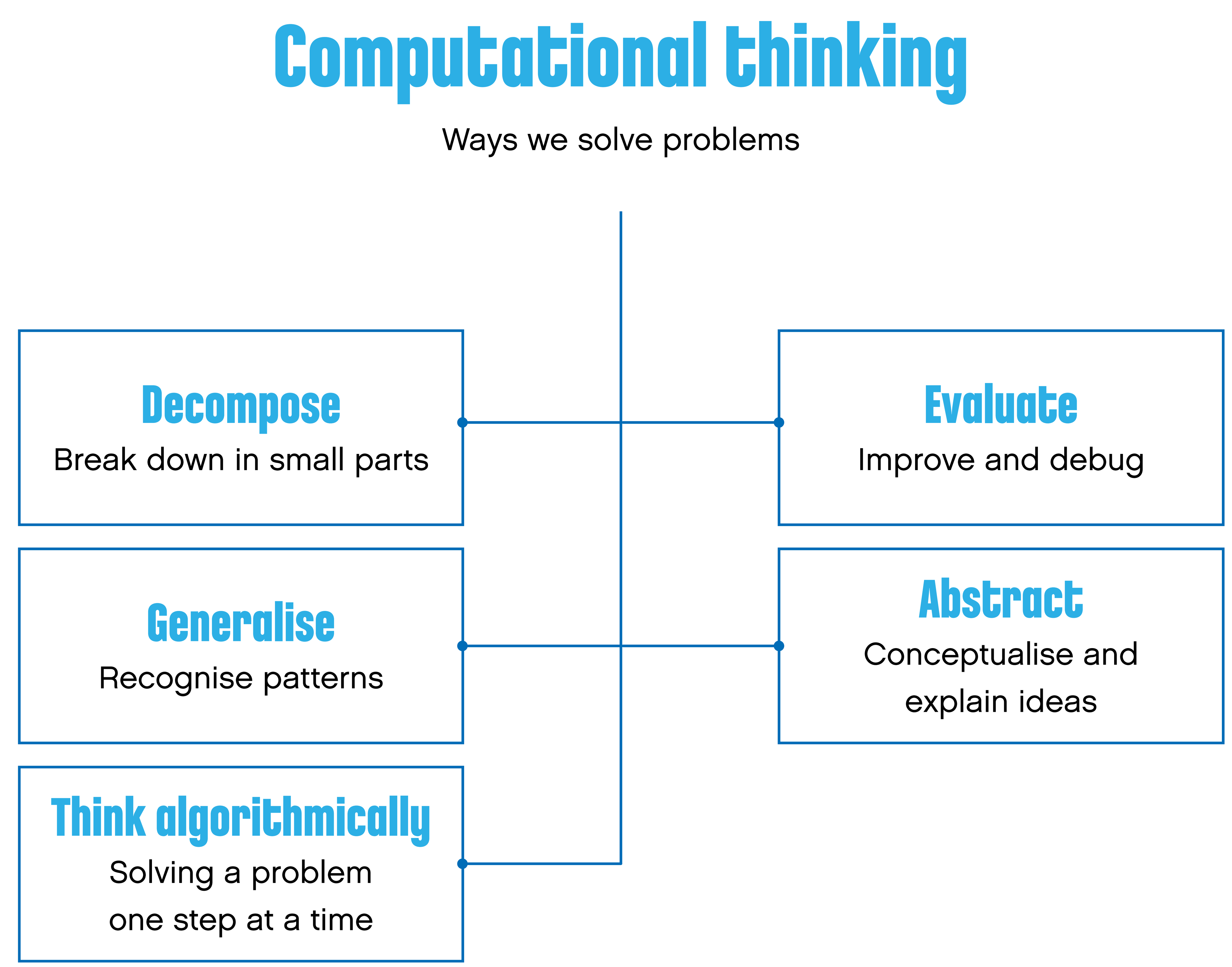
Decomposition is the ability to simplify a problem into smaller parts in order to ease the process of finding a solution. By doing so, the problem becomes easier to explain to another person, or to separate into tasks. Decomposition frequently leads to Generalization.

Example: When going on vacation, the preparation (or project) can be separated into subtasks: booking the airfare, reserving a hotel, packing a suitcase, etc.

Generalization (Pattern Recognition)

Generalization is the ability to recognize the parts of a task that are known, or have been seen somewhere else. This frequently leads to easier ways of designing algorithms.

Example: Traffic lights work by repeating the same series of actions forever.



What is computational thinking ?

Algorithmic Thinking

Algorithmic Thinking is the ability to create an ordered series of steps with the purpose of solving a problem.

Example one: when we cook from a recipe, we are following a series of steps in order to prepare a meal.
Example two: when playing with computers, we can code a sequence of actions that tell the computer what to do.

Evaluating or Debugging

This is the ability to verify whether or not a prototype works as intended, and if not, the ability to identify what needs to be improved. It is also the process a computer programmer goes through in order to find and correct mistakes within a program.

Example one: when we're cooking, we will periodically taste the dish to check whether or not it is seasoned correctly.
Example two: when we look for spelling mistakes and missing punctuation in our written work, we are debugging it so that it can be read correctly.

Abstraction

Abstraction is the ability to explain a problem or a solution by removing unimportant details. In other words, being able to conceptualize an idea.

Example: When describing a bicycle, we use only some details to describe it. We might mention its type and color, and add more details for someone who has a real interest in bikes.

A Process for Developing Computational Thinking Skills

Using an Engineering Design Process

When looking for solutions to a problem, engineers use a design process. They go through a series of phases that guide them toward a solution. During each of these phases, some of their skills are used or developed. It is those skills that we we refer to as "computational thinking skills."

In WeDo 2.0, students follow a similar process:

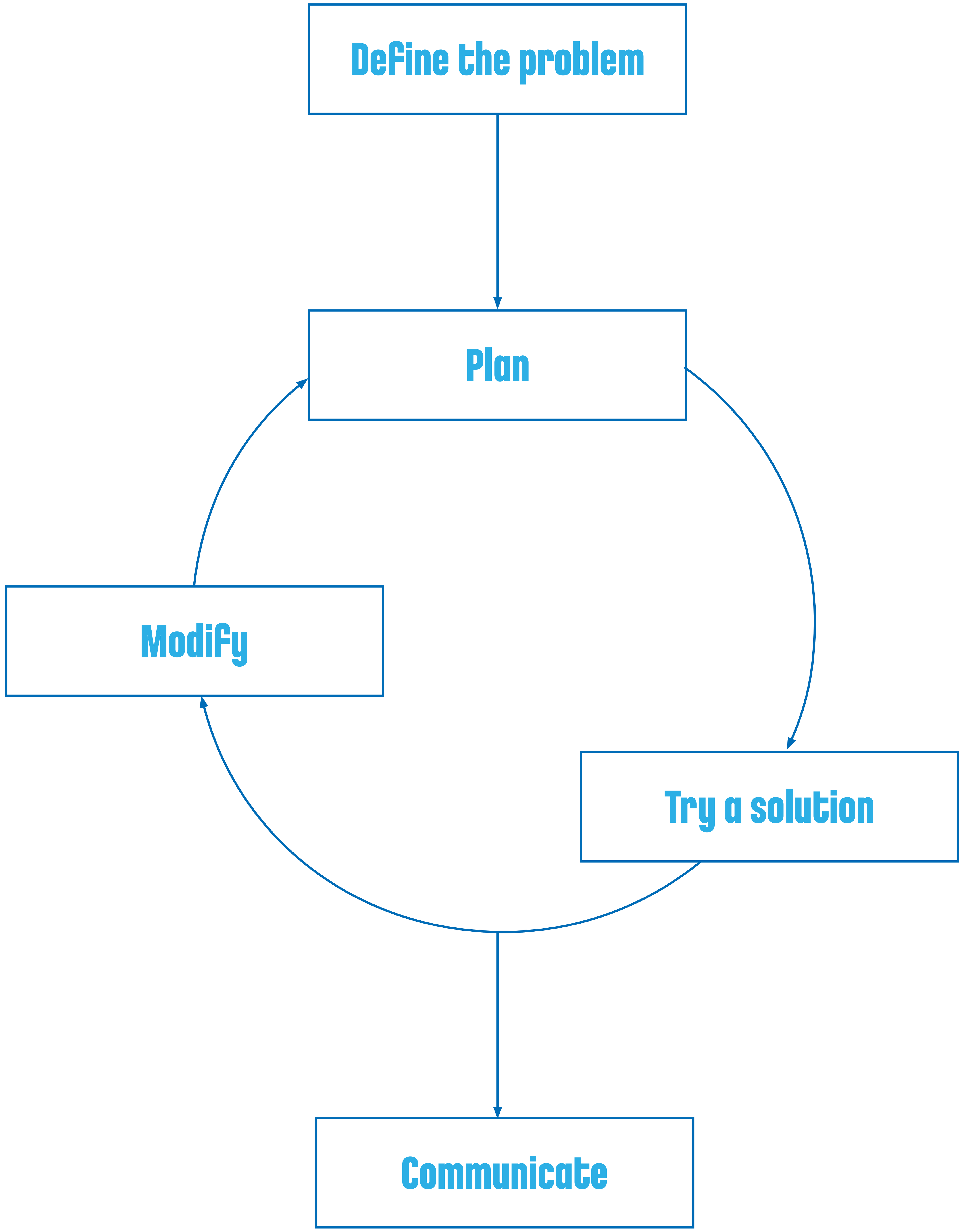
Defining the Problem

Students are presented with a topic that guides them to a problem or to a situation they wish to improve. Sometimes, a problem can have a lot of details. To make it easier to solve, the problem can be broken down into smaller parts.

By defining the problem in a simple way and by identifying some success criteria, students will develop a skill called "Decomposition."

In other words:

- *Is the student able to explain the problem by themselves?*
- *is the student able to describe how they will evaluate whether or not they were successful in solving the problem?*
- *Is the student able to break down the problem into smaller and more manageable parts?*



A Process for Developing Computational Thinking Skills

Planning
Students should spend some time imagining different solutions to the problem, and then make a detailed plan for executing one of their ideas. They will define the steps they will need to go through in order to reach the solution. By identifying the parts of the task they might have see before, they will develop a skill called "Generalization."

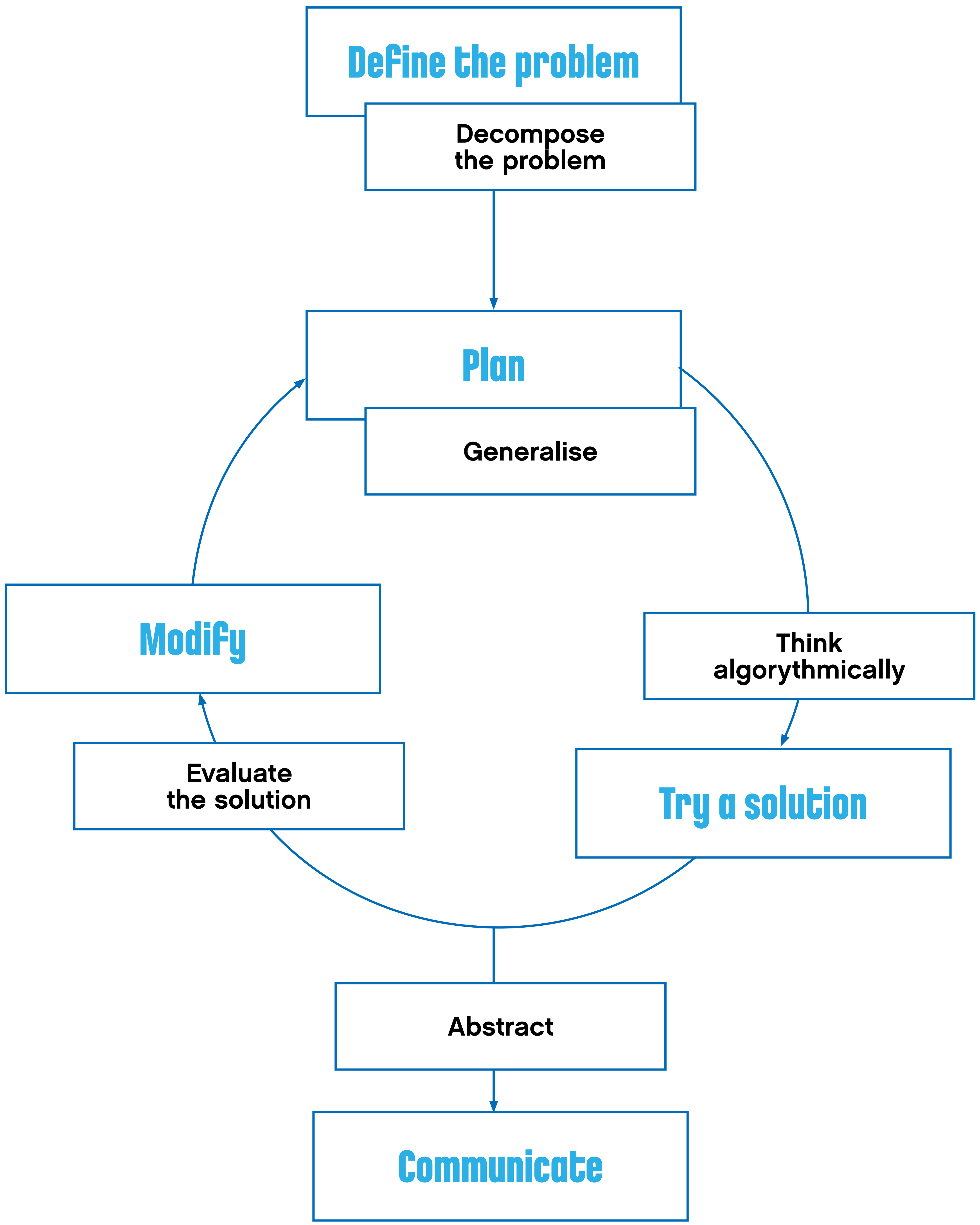
In other words:
- *Is the student able to make a list of actions to program?*
- *Is the student able to identify parts of programs that he or she could use?*
- *Is the student able to reuse parts of programs?*

Trying
Each student is then tasked with creating the final version of their solution. In this phase of the process, they use iconic programing language to activate their LEGO® models. As the students code their ideas, they develop their Algorithmic Thinking skills.

In other words:
- *Is the student able to program a solution to a program?*
- *Is the student able to use sequence, loops, conditional statements, etc.?*

Modifying
Students will evaluate their solution according to whether or not their program and model meet the success criteria. Using their Evaluation skills, they will determine whether they need to change, fix, debug, or improve some part of their program.

In other words:
- *Is the student making iterations of their program?*
- *Is the student fixing problems in their program ?*
- *Is the student able to judge if the solution is linked to the problem ?*



A Process for Developing Computational Thinking Skills

Communicating

Students will present the final version of their solution to the class, explaining how their solution meets the success criteria. By explaining their solution with the right level of detail, they will develop their Abstraction and communication skills.

In other words:

- Is the *student explaining the most important part of their solution?*
- *Is the student giving enough detail to enhance comprehension?*
- *Is the student making sure to explain how their solution meets the success criteria*?

Developing Computational Thinking through Coding

In order to develop their Algorithmic Thinking, students will be introduced to some programming principles. As they develop their solutions, they will organize a series of actions and structures that will bring their models to life.

The most common WeDo 2.0 programming principles students will use are:

1. Output
Output is something that is controlled by the program students are writing. Examples of outputs for WeDo 2.0 are sounds, lights, display, and turning motors on and off.

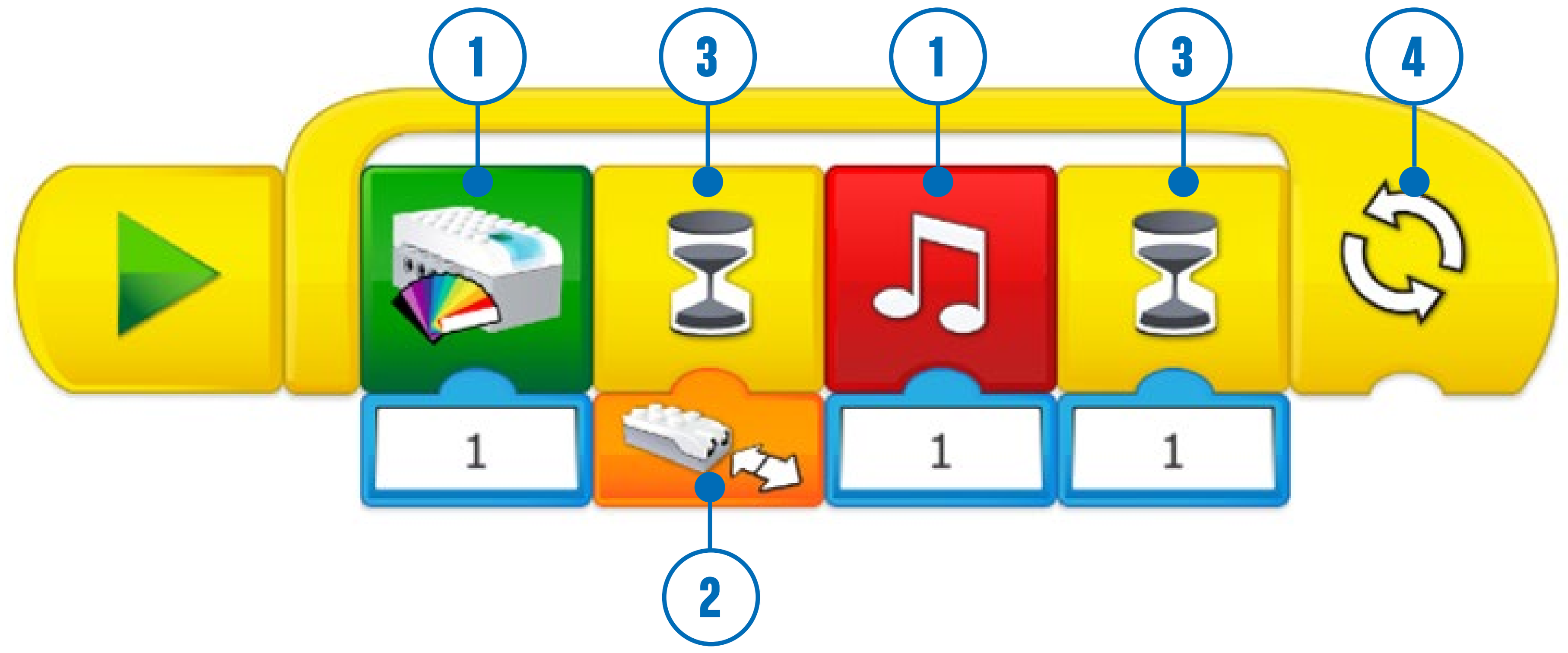
2. Input
Input is information that a computer or device receives. It can be entered through the use of sensors in the form of a numeric or text value. For example, a sensor that detects or measures something (such as distance) converts that value into a digital input signal so it can be used in a program.

3. Events (Wait for)
Students can tell their program to wait for something to happen before continuing to execute the sequence of actions. Programs can wait for a specific amount of time, or wait for something to be detected by a sensor.

4. Loop
Students can program actions to be repeated either forever, or for a specific length of time.

5. Functions
Functions are a group of actions that are to be used together in specific situations. For example, the group of blocks that could be used to make a light blink would together be called, “the blink function”.

6. Conditions
Conditions are used by students in order to program actions that are to be executed only under certain circumstances. Creating conditions within a program means that some part of the program will never be executed if the condition is never met. For example, if the Tilt Sensor is tilted left, the motor will start, and if the sensor is tilted right, the motor will stop; if the Tilt Sensor never tilts left, the motor will never start and if it never tilts right, then the motor will never stop.



WeDo 2.0 in Curriculum

The LEGO® Education WeDo 2.0 projects combine LEGO® bricks with Next Generation Science Standards (NGSS). All of the WeDo 2.0 projects are designed to develop students' computational thinking skills.

Computational Thinking in Curriculum

The world is changing, and whether we realize it or not, technology and computer science shape nearly every aspect of our lives. Students are rapidly becoming active citizens, and equipping them with the right set of skills has become one of the nation's first priorities.

Computational thinking is a set of skills that is spreading worldwide, becoming a key practice to develop in relation to technology. Already identified by the NGSS as a practice essential to the Science and Engineering field, computational thinking has found roots in many other national curriculums both domestically and abroad.

Computational thinking has become the foundation of standards issued by the Computer Science Teacher Association (CSTA) and other associations such as ISTE, Code.org, and Computing at School (the British association responsible for a globally recognized computing curriculum). All of these organizations have aligned their curriculums with an emphasis on the development of computational thinking skills.

These important skills can be developed through engaging activities or projects that are rooted in real life problem-based situations. To support this development, LEGO® Education is adding a dedicated series of computational thinking projects to the science projects that are already available in WeDo 2.0.

Visual Overview of Guided Projects

1. Moon Base
This project is about designing a solution in which a robot would be able to assemble a base on the moon.

2. Grabbing Objects
This project is about designing a solution for a prosthetic arm that is able to move small objects around.

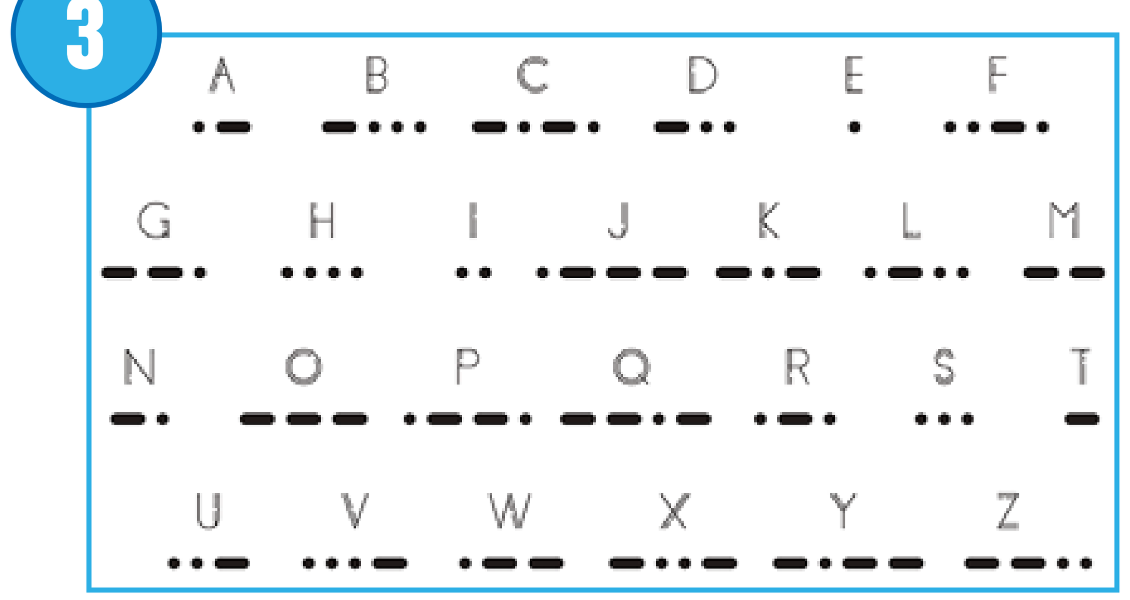
3. Send Messages
This project is about designing a solution for exchanging information using a system of signals organized in patterns.

4. Volcano Alert
This project is about designing a device for improving the monitoring of volcanic activity in order to guide scientific exploration.

1

2

3



4

Visual Overview of Open Projects

5. Inspection
This project is about designing a solution in which a robot is able to inspect narrow spaces, guiding its motion with sensors.

6. Emotional Design
This project is about designing a solution in which a robot can display positive emotions when interacting with people.

7. City Safety
This project is about designing a solution to improve safety in a city.

8. Animal Senses
This project is about modeling how animals use their senses to interact with their environment.

5

6

7

8

Potential Flow to develop Computational thinking skills

You can organize the projects as you wish. Each project highlights opportunities for developing computational thinking skills, and it is it up to you to focus on the ones that are most relevant to you and your students. Here is one suggested sequence, which is based on an increasing level of complexity in the programming concepts covered:

Getting Started

Use two lessons of 45 minutes each to introduce your students to WeDo 2.0.
Lesson 1, Milo, the Science Rover
Lesson 2, combine Milo's Motion Sensor, Milo's Tilt Sensor, and Collaborating

Guided Projects

Use two lessons of 45 minutes each, during which students will program a sequence of actions.
Lesson 3, Moon Base (Explore and Create phase)
Lesson 4, Moon Base (Test and Share phase)

Use two lessons of 45 minutes each, during which students will use sensors (inputs).
Lesson 5, Grabbing Objects (Explore and Create phase)
Lesson 6, Grabbing Objects (Test and Share phase)

Use two lessons of 45 minutes each, during which students will use sensors (inputs), loops, and parallel programming.
Lesson 7, Send Messages (Explore and Create phase)
Lesson 8, Send Messages (Test and Share phase)

Use two lessons of 45 minutes each to introduce your students to conditions, and how to integrate all of the other programming principles.
Lesson 9, Volcano Alert (Explore and Create phase)
Lesson 10, Volcano Alert (Test and Share phase)

Open Projects

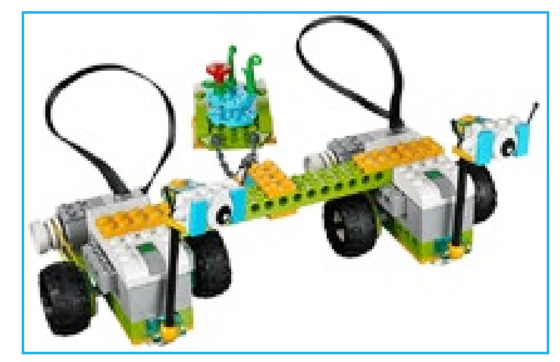
Use two or three lessons of 45 minutes each to make you own project based on one of the suggested Open Projects. This project should integrate all of the programming principles, as well as the computational thinking skills developed during the Guided Projects.

Potential Flow to develop Computational thinking skills

Getting Started

Introduce your students to WeDo 2.0

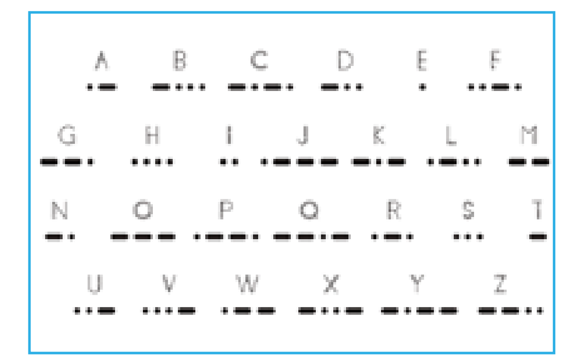


45 minutes | 45 minutes

Guided Project - Send Messages

Students will use sensors (inputs), loops, and parallel programming.



Using a condensed lesson flow
2 x 45 minutes

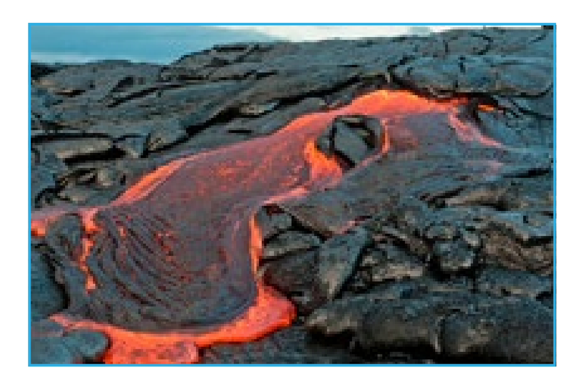
Guided Project - Moon Base

Students will program sequences of actions.

Using a condensed lesson flow
2 x 45 minutes

Guided Project - Volcano Alert

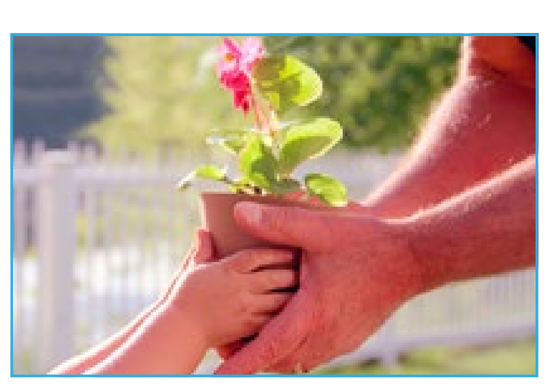
Students will be introduced to conditions, and to other programming principles.



Using a condensed lesson flow
2 x 45 minutes

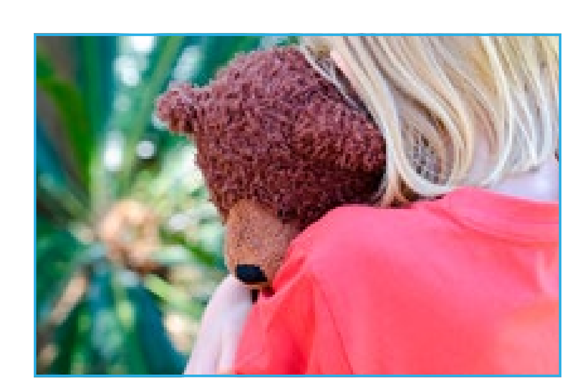
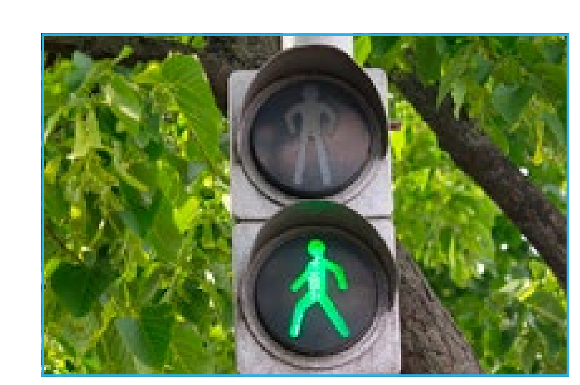
Guided Project - Grabbing Objects

Students will use sensors (inputs).



Using a condensed lesson flow
2 x 45 minutes

Open Projects


Curriculum Overview of Guided Projects Organized by NGSS Disciplinary Core Ideas

	1 Moon Base	2 Grabbing Objects	3 Send Messages	4 Volcano Alert
Life Sciences				
Earth and Space Sciences				4-ESS3-2.
Physical Sciences			4-PS4-3.	
Engineering, Technology, and Applications of Science	K-2-ETS1-3. 3-5-ETS1-1. 3-5-ETS1-2. 3-5-ETS1-3.	3-5-ETS1-1. 3-5-ETS1-2. 3-5-ETS1-3.	3-5-ETS1-1. 3-5-ETS1-2. 3-5-ETS1-3.	3-5-ETS1-1. 3-5-ETS1-2. 3-5-ETS1-3.

Curriculum Overview of Open Projects Organized by NGSS Disciplinary Core Ideas

	5 Inspection	6 Emotional Design	7 City Safety	8 Animal Senses
Life Sciences				4-LS1-2.
Earth and Space Sciences				
Physical Sciences				
Engineering, Technology, and Applications of Science	K-2-ETS1-3. 3-5-ETS1-1. 3-5-ETS1-2. 3-5-ETS1-3.	K-2-ETS1-3. 3-5-ETS1-1. 3-5-ETS1-2. 3-5-ETS1-3.	K-2-ETS1-3. 3-5-ETS1-1. 3-5-ETS1-2. 3-5-ETS1-3.	K-2-ETS1-3. 3-5-ETS1-1. 3-5-ETS1-2. 3-5-ETS1-3.

NGSS Performance Expectations: Grade Two

Life Science

2-LS2-1. Plan and conduct an investigation to determine if plants need sunlight and water to grow.
2-LS2-2. Develop a simple model that mimics the function of an animal in dispersing seeds or pollinating plants.
2-LS4-1. Make observations of plants and animals to compare the diversity of life in different habitats.

Physical Science

2-PS1-1. Plan and conduct an investigation to describe and classify different kinds of materials by their observable properties.
2-PS1-2. Analyze data obtained from testing different materials to determine which materials have the properties that are best suited for an intended purpose.
2-PS1-3. Make observations to construct an evidence-based account of how an object made of a small set of pieces can be disassembled and made into a wholly new object.
2-PS1-4. Construct an argument with evidence that some changes caused by heating or cooling can be reversed and some cannot.

Earth and Space Science

2-ESS1-1. Use information from several sources to provide evidence that earth events can occur quickly or slowly.
2-ESS2-1. Compare multiple solutions designed to slow or prevent wind or water from changing the physical shape of the land.
2-ESS2-2. Develop a model to represent the shapes and kinds of land and bodies of water in an area.
2-ESS2-3. Obtain information to identify where water is found on earth and understand that it can be solid or liquid.

Engineering

K-2-ETS1-1. Ask questions, make observations, and gather information about a situation people want to change in order to define a simple problem that can be solved through the development of a new or improved object or tool.
K-2-ETS1-2. Develop a simple sketch, drawing, or physical model to illustrate how the shape of an object helps it function as needed to solve a problem.
K-2-ETS1-3. Analyze data from tests of two objects designed to solve the same problem to compare the strengths and weaknesses of how each performs.

NGSS Performance Expectations: Grade Three

Physical Science

3-PS2-1. Plan and conduct an investigation to provide evidence of the effects of balanced and unbalanced forces on the motion of an object.
3-PS2-2. Make observations and/or measurements of an object's motion to provide evidence that a pattern can be used to predict future motion.
3-PS2-3. Ask questions to determine cause and effect relationships of electric or magnetic interactions between two objects not in contact with each other.
3-PS2-4. Define a simple design problem that can be solved by applying scientific ideas about magnets.

Earth and Space Science

3-ESS2-1. Represent data in tables and graphic displays to describe typical weather conditions expected during a particular season.
3-ESS2-2. Obtain and combine information to describe climates in different regions of the world.
3-ESS3-1. Make a claim about the merit of a design solution that reduces the impact of a weather-related hazard.

Engineering

3-5-ETS1-1. Define a simple design problem reflecting a need that includes specified criteria for success, and constraints on materials, time, or cost.
3-5-ETS1-2. Generate and compare multiple, possible solutions to a problem based on how well each meets the criteria and constraints of the problem.
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.

Life Science

3-LS2-1. Construct an argument that some animals form groups that help members survive.
3-LS4-1. Analyze and interpret data from fossils to provide evidence of organisms and the environments in which they lived long ago.
3-LS4-3. Construct an argument with evidence that in a particular habitat, some organisms can survive well, some survive less well, and some cannot survive at all.
3-LS4-4. Make a claim about the merit of a solution to a problem that is caused when the environment changes and the types of plants and animals that live there may also change.
3-LS1-1. Develop models to describe that organisms have unique and diverse life cycles, but all have in common birth, growth, reproduction, and death.
3-LS3-1. Analyze and interpret data to provide evidence that plants and animals have traits inherited from parents and that variations of these traits exist within a group of similar organisms.
3-LS3-2. Use evidence to support the explanation that traits can be influenced by the environment.
3-LS4-2. Use evidence to construct an explanation for how the variations in characteristics among individuals of the same species may provide advantages in surviving, finding mates, and reproducing.

NGSS Performance Expectations: Grade Four

Energy

4-PS3-1. Use evidence to construct an explanation relating the speed of an object to the energy of that object.
4-PS3-2. Make observations to provide evidence that energy can be transferred from place to place by sound, light, heat, and electric currents.
4-PS3-3. Ask questions and predict outcomes about the changes in energy that occur when objects collide.
4-PS3-4. Apply scientific ideas to design, test, and refine a device that converts energy from one form to another.
4-ESS3-1. Obtain and combine information to describe the fact that energy and fuels are derived from natural resources and that their use will affect the environment.

Structure, Function, and Information Processing

4-PS4-2. Develop a model to describe how light reflecting from objects and entering the eye of a sighted person allows objects to be seen.
4-LS1-1. Construct an argument that plants and animals have internal and external structures that function to support their survival, growth, behavior, and reproduction.
4-LS1-2. Use a model to describe how animals receive different types of information through their senses, then process the information in their brain, and respond to the information in a range of different ways.

Waves: Waves and Information

4-PS4-1. Develop a model of waves to describe patterns in terms of amplitude and wavelength, and that waves can cause objects to move.
4-PS4-3. Generate and compare multiple solutions that use patterns for the transfer of information.

Earth's Systems: Processes That Shape the Earth

4-ESS1-1. Identify evidence from patterns in rock formations and fossils in rock layers to support an explanation for changes in a landscape over time.
4-ESS2-1. Make observations and/or measurements to provide evidence of the effects of weathering or the rate of erosion by water, ice, wind, or vegetation.
4-ESS2-2. Analyze and interpret data from maps to describe patterns of earth's features.
4-ESS3-2. Generate and compare multiple solutions to reduce the impacts of natural earth processes on humans.

Engineering

3-5-ETS1-1. Define a simple design problem reflecting a need or a want that includes criteria for success, and constraints on materials, time, or cost.
3-5-ETS1-2. Generate and compare possible solutions to a problem based on how well each meets the criteria and constraints of the problem.
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.

Curriculum Overview of Guided Projects Organized by NGSS Practices

	1 Moon Base	2 Grabbing Objects	3 Send Messages	4 Volcano Alert
Practice One: Ask questions and define problems	●	●	●	●
Practice Two: Develop and use models				
Practice Three: Plan and carry out investigations				
Practice Four: Analyze and interpret data				
Practice Five: Use mathematics and computational thinking	●	●	●	●
Practice Six: Construct explanations and design solutions	●	●	●	●
Practice Seven: Engage in argument from evidence	●	●	●	●
Practice Eight: Obtain, evaluate, and communicate information	●	●	●	●

Curriculum Overview of Open Projects Organized by NGSS Practices

	5 Inspection	6 Emotional Design	7 City Safety	8 Animal Senses
Practice One: Ask questions and define problems	●	●	●	●
Practice Two: Develop and use models				●
Practice Three: Plan and carry out investigations				
Practice Four: Analyze and interpret data				
Practice Five: Use mathematics and computational thinking	●	●	●	●
Practice Six: Construct explanations and design solutions	●	●	●	
Practice Seven: Engage in argument from evidence	●	●	●	●
Practice Eight: Obtain, evaluate, and communicate information	●	●	●	●

Curriculum Overview of Guided and Open Projects Organized by CSTA Standards

Identifier		Interim CSTA K–12 CS Standard	1 Moon Base	2 Grabbing Objects	3 Send Messages	4 Volcano Alert	5 Inspection	6 Emotional Design	7 City Safety	8 Animal Senses
K–2	1A-A-3-7	Construct and execute algorithms (sets of step-by-step instructions) that include sequencing and simple loops to accomplish a task, both independently and collaboratively, with or without a computing device.	●	●	●	●	●	●	●	●
K–2	1A-A-6-8	Analyze and debug (fix) an algorithm that includes sequencing and simple loops, with or without a computing device.	●	●	●	●	●	●	●	●
K–2	1A-C-7-9	Identify and use software that controls computational devices (e.g., use an app to draw on the screen, use software to write a story or control robots).	●	●	●	●	●	●	●	●
K–2	1A-C-7-10	Use appropriate terminology in naming and describing the function of common computing devices and components (e.g., desktop computer, laptop computer, tablet device, monitor, keyboard, mouse, printer).								
K–2	1A-C-6-11	Identify, using accurate terminology, simple hardware and software problems that may occur during use (e.g., app or program not working as expected, no sound, device won't turn on).	●	●	●	●	●	●	●	●
K–2	1A-D-7-12	Collect data over time and organize it in a chart or graph in order to make a prediction.								
K–2	1A-D-4-13	Use a computing device to store, search, retrieve, modify, and delete information and define the information stored as data.								
K–2	1A-D-4-14	Create a model of an object or process in order to identify patterns and essential elements (e.g., water cycle, butterfly life cycle, seasonal weather patterns).	●	●	●	●	●	●	●	●

Curriculum Overview of Guided and Open Projects Organized by CSTA Standards

Identifier		Interim CSTA K–12 CS Standard	1 Moon Base	2 Grabbing Objects	3 Send Messages	4 Volcano Alert	5 Inspection	6 Emotional Design	7 City Safety	8 Animal Senses
3–5	1B-A-2-1	Apply collaboration strategies to support problem solving within the design cycle of a program.	●	●	●	●	●	●	●	●
3–5	1B-A-7-2	Use proper citations and document when ideas are borrowed and changed for their own use (e.g., using pictures created by others, using music created by others, remixing programming projects).	●	●	●	●	●	●	●	●
3–5	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, pseudocode, story map).	●	●	●	●	●	●	●	●
3–5	1B-A-5-4	Construct programs, in order to solve a problem or for creative expression, that includes 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).	●	●	●	●	●	●	●	●
3–5	1B-A-5-5	Use mathematical operations to change a value stored in a variable.				●				
3–5	1B-A-3-6	Decompose (break down) a larger problem into smaller sub-problems, independently or in a collaborative group.	●	●	●	●	●	●	●	●

Curriculum Overview of Guided and Open Projects Organized by CSTA Standards

Identifier		Interim CSTA K–12 CS Standard	1 Moon Base	2 Grabbing Objects	3 Send Messages	4 Volcano Alert	5 Inspection	6 Emotional Design	7 City Safety	8 Animal Senses
3–5	1B-A-3-7	Construct and execute an algorithm (set of step-by-step instructions) that includes sequencing, loops, and conditionals to accomplish a task, both independently and collaboratively, with or without a computing device.	●	●	●	●	●	●	●	●
3–5	1B-A-6-8	Analyze and debug (fix) an algorithm that includes sequencing, events, loops, conditionals, parallelism, and variables.	●	●	●	●	●	●	●	●
3–5	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.)								
3–5	1B-C-7-10	Use appropriate terminology in naming internal and external components of computing devices and describing their relationships, capabilities, and limitations.								
3–5	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).								
3–5	1B-D-5-12	Create a computational artifact to model the attributes and behaviors associated with a concept (e.g., solar system, life cycle of a plant).	●	●	●	●	●	●	●	●
3–5	1B-D-5-13	Answer a question by using a computer to manipulate (e.g., sort, total and/or average, chart, graph) and analyze data that has been collected by the class or student.								

Curriculum Overview of Guided and Open Projects Organized by CSTA Standards

Identifier		Interim CSTA Standard	1 Moon Base	2 Grabbing Objects	3 Send Messages	4 Volcano Alert	5 Inspection	6 Emotional Design	7 City Safety	8 Animal Senses
3–5	1B-D-4-14	Use numeric values to represent non-numeric ideas in the computer (binary, ASCII, pixel attributes such as RGB).			●					
3–5	1B-I-7-15	Evaluate and describe the positive and negative impacts of the pervasiveness of computers and computing in daily life (e.g., downloading videos and audio files, electronic appliances, wireless internet, mobile computing devices, GPS systems, wearable computing).								
3–5	1B-I-7-16	Generate examples of how computing can affect society, and also how societal values can shape computing choices.								
3–5	1B-I-1-17	Seek out and compare diverse perspectives, synchronously or asynchronously, to improve a project.								
3–5	1B-I-1-18	Brainstorm ways in which computing devices could be made more accessible to all users.								
3–5	1B-I-1-19	Explain problems that relate to using computing devices and networks (e.g., logging out to deter others from using your account, cyberbullying, privacy of personal information, and ownership).								
3–5	1B-N-7-20	Create examples of strong passwords, explain why strong passwords should be used, and demonstrate proper use and protection of personal passwords.								
3–5	1B-N-4-21	Model how a device on a network sends a message from one device (sender) to another (receiver) while following specific rules.			●					

Assessing Computational Thinking Skills

There are many ways in which you can monitor and assess your students' progress through a WeDo 2.0 project.

This section offers the following tools to help you in your assessments:

- Documentation pages
- Self-assessment statements
- Anecdotal record grid
- Observation rubrics grid

Student-Led Assessment

Documentation Pages

Each project will ask students to create documents to summarize their work.
To have a complete science report, it is essential that students:

- Document their work using various types of media
- Document every step of the process
- Take the time to organize and complete their document

It is most likely that the first document your students will complete will not be as good as the next one. You can support them by:

- Giving feedback and allowing them time to see where and how they can improve some parts of their document.
- Allowing them to share their documents with each other. By communicating their scientific findings, students will be engaged in the work of scientists.

Self-Assessment Statements

After each project, students should reflect on the work they have done. Use the following page to encourage reflection and set goals for the next project.

Student Self-Assessment Rubric

Name: Class: Project:

Directions: Circle the brick that shows how well you did. The bigger brick, the better you did.				
I defined the question or problem.				
I built a LEGO® model and programmed a solution.				
I tested my solution and made improvements.				
I documented and shared my ideas.				

Project Reflection

One thing I did really well was:

One thing I want to improve on for next time is:

Teacher-Led Assessment

Developing students' science, engineering, and computational thinking skills requires time and feedback. Just as in the design cycle, in which students should understand that failure is part of the process, assessment should provide feedback in terms of what students did well and where they can improve. Problem-based learning is not about succeeding or failing. It is about being an active learner and continually building upon and testing ideas.

Giving feedback to students in order to help them develop their skills can be done in various ways. At each phase of the WeDo 2.0 projects, we have provided examples of rubrics that can be used by:

- Observing students' behavior, reaction, and strategies
- Asking questions about their thought processes

As students often work in groups, you can give feedback both on a team level and on an individual level.

Anecdotal Record Grid

The anecdotal record grid lets you record any type of observation you believe is important for each student. Use the template on the next page to provide feedback to students as needed.

Anecdotal Record Grid

Name: Class: Project:

1. Emerging	2. Developing	3. Proficient	4. Accomplished

Notes:

Teacher-Led Assessment

Observation Rubrics

Examples of rubrics have been provided for every Guided Project. For every student, or every team, you can use the observation rubrics grid to:

- Evaluate student performance at each step of the process
- Provide constructive feedback to help the student progress

The observation rubrics provided in the Guided Projects can be adapted to fit your needs. The rubrics are based on these progressive stages:

1. Emerging

The student is at the beginning stages of development in terms of content knowledge, ability to understand and apply content, and/or demonstration of coherent thoughts about a given topic.

2. Developing

The student is able to present basic knowledge only (e.g., vocabulary), and cannot yet apply content knowledge or demonstrate comprehension of the concepts being presented.

3. Proficient

The student has concrete levels of comprehension of the content and concepts and can adequately demonstrate the topics, content, or concepts being taught. The ability to discuss and apply this knowledge outside the required assignment is lacking.

4. Accomplished

The student can take concepts and ideas to the next level, apply concepts to other situations, and synthesize, apply, and extend knowledge to discussions that include extensions of ideas.

Suggestion

Use the observation rubrics grid on the next page to keep track of your students' progress.

Observation Rubrics Grid

Class:		Project:			
Students' Names		NGSS			
		Explore	Create	Test	Share
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					

To be used with the rubrics described on the following page: (1) emerging, (2) developing, (3) proficient, (4) accomplished.

Assessing Project Phases - General Rubrics

You can use these assessment rubrics to give general feedback on a scale of 1 to 4 at the end of each phase of a project.

Explore Phase

In the Explore phase, feedback should relate to whether or not the student is actively involved in the discussion by asking and answering questions, and their level of understanding of the problem.

1. The student is unable to provide answers to questions or adequately participate in discussions.
2. The student is able, with prompting, to provide answers to questions or adequately participate in discussions.
3. The student is able to provide adequate answers to questions and participate in class discussions.
4. The student is able to extend explanations in class discussions.

Test Phase

During the Test phase, make sure that the student works well on a team, justifies his/her best solution, and uses the information collected in the Explore phase.

1. The student is unable to work well on a team, justify solutions, and use information collected for further development.
2. The student is able to work on a team, collect and use information with guidance, or, with help, to justify solutions.
3. The student is able to work on a team and contribute to the team discussions, justify solutions, and collect and use information about the content.
4. The student can justify and discuss solutions that allow for the collection and use of information.

Share Phase

During the Share phase, make sure that the student is able to describe their solution using the right vocabulary and the right level of detail.

1. The student does not use evidence from his/her findings in connection with ideas shared during the presentation and does not follow established guidelines.
2. The student uses some evidence from his/her findings, but the justification is limited. Established guidelines are generally followed but may be lacking in one or more areas.
3. The student provides adequate evidence to justify his/her findings and follows established guidelines for presenting.
4. The student fully discusses his/her findings and thoroughly utilizes appropriate evidence to justify his/her reasoning while following all established guidelines.

Assessing Computational Thinking Skills

Name: Class:

Decomposition	1. Emerging	2. Developing	3. Proficient	4. Accomplished	Notes
Describe the problem in your own words.	The student is unable to describe the problem in their own words.	The student is able, with prompting, to describe the problem in their own words.	The student is able to describe the problem in their own words.	The student is able to describe the problem in their own words and starts to decompose the problem into smaller parts.	
Describe how you will know whether or not you have found a successful solution to the problem.	The student is unable to describe success criteria.	The student is able, with prompting, to describe success criteria.	The student is able to describe success criteria.	The student is able to describe success criteria with a high level of detail.	
Describe how you can break the problem down into smaller parts.	The student is unable to break down the problem.	With prompting, the student is able to break down the problem into smaller parts.	The student is able to break down the problem into smaller parts.	The student is able to break down the problem into smaller parts and can describe the links between each of the parts.	

Assessing Computational Thinking Skills

Name: Class:

Generalization	1. Emerging	2. Developing	3. Proficient	4. Accomplished	Notes
Describe which program you have used from the Program Library (or elsewhere) and why.	The student is unable to describe which program has been used and why. ☐	The student is able to identify which program has been used. ☐	The student is able to describe which program has been used and why. ☐	The student is able to describe, in detail, which program has been used and what modifications have been made to it. ☐	
Observe how your students recognize patterns, or reuse concepts they have seen before.	The student is unable to recognize patterns, or reuse concepts seen before. ☐	With prompting, the student is able to recognize patterns, or reuse concepts seen before. ☐	The student is able to recognize patterns, or reuse concepts seen before. ☐	The student is able to recognize patterns, or reuse concepts of their own. ☐	

Assessing Computational Thinking Skills

Name: Class:

Algorithmic Thinking	1. Emerging	2. Developing	3. Proficient	4. Accomplished	Notes
Describe the list of actions to program.	The student is unable to make a list of actions. ☐	With prompting, the student is able to make a list of actions. ☐	The student is able to make a list of actions. ☐	The student is able to make a detailed list of actions to help them develop their program. ☐	
Describe how you have programmed your solution.	The student is unable to describe the program. ☐	With prompting, the student is able to describe the program. ☐	The student is able to describe the program. ☐	The student is able to describe the program, providing extensive details about each component. ☐	
Describe the programming principles used in your solution (e.g., output, inputs, events, loops, etc.).	The student in unable to describe the programming principles used in their solution. ☐	With prompting, the student in able to describe the programming principles used in their solution. ☐	The student is able to describe the programming principles used in their solution. ☐	The student in able to describe, with extensive comprehension, the programming principles used in their solution. ☐	

Assessing Computational Thinking Skills

Name: Class:

Evaluation	1. Emerging	2. Developing	3. Proficient	4. Accomplished	Notes
Describe what happened when you executed your program, and whether or not it was what you expected.	The student cannot describe what happened.	With prompting, the student is able to describe what happened, and compare it to what was expected.	The student is able to describe what happened, and compare it to what was expected.	The student is able to describe what happened, compare it to what was expected, and is already finding solutions.	
Describe how you have fixed the problems in your program.	The student cannot describe how they have fixed the problems.	With prompting, the student can describe how they have fixed the problems.	The student can describe how they have fixed the problems.	The student can describe, in extensive detail, how they have fixed the problems.	
Describe how your solution is linked to the problem.	The student is unable to describe how their solution is linked to the problem.	With prompting, the student is able to describe how their solution is linked to the problem.	The student is able to describe how thier solution is linked to the problem.	The student is able to describe, in extensive detail, how their solution is linked to the problem.	
Describe how you have tried new ways of solving the problems along the project	Students is unable to describe other ways he has tried along the project.	Students is able, with prompting, to describe other ways he has tried along the project.	Students is able to describe other ways he has tried along the project.	Students is able to describe other ways he has tried along the project and is able to describe why each options has not been considered.	

Assessing Computational Thinking Skills

Name: Class:

Abstraction	1. Emerging	2. Developing	3. Proficient	4. Accomplished	Notes
Describe the most important part of your solution.	The student is not able to describe their solution.	With prompting, the student is able to describe their solution.	The student is able to describe their solution.	The student is able to describe their solution, focusing on the most important part of the solution.	
Describe the most important details of your solution.	The student is not able to provide any details about their solution.	With prompting, the student is able to provide details about their solution.	The student is able to discuss details of their solution, but some of the details are not essential.	The student is able to discuss the most important details of their solution.	
Describe how your solution met the initial criteria.	Their student is unable to describe how their solution met the initial criteria.	With prompting, the student is able to describe how their solution met the initial criteria.	The student is able to describe how their solution met the initial criteria.	The student is able to describe, with extraordinary clarity, how their solution met the initial criteria.	

LEGO® Education WeDo 2.0

LEGOeducation.com

LEGO education