New York Hall of Science: Computational Thinking School Strategy Guide

This guide is made possible with support from the Robin Hood Foundation Learning + Technology Fund.

Computational Thinking School Strategy Guide

This Computational Thinking School Strategy Guide provides information and resources on how elementary school teachers can integrate the core constructs of computational thinking into their classroom curriculum to foster higher order thinking and problem-solving skills.

The guide is based on the *Integrated Computational Thinking* project funded by the Robin Hood Learning + Technology Fund to pilot an innovative professional development model for elementary teachers in collaboration with the New York Hall of Science (NYSCI) and Public School 13 Clement C. Moore in Queens, NY. The primary focus of the project was to explore and develop effective strategies for building elementary teachers' capacity to recognize and support the practice of computational thinking within the context of their existing priorities and instructional constraints. As a result of this collaboration, the following guide serves as a road map for teachers to build foundational steps towards integrating computational thinking across curriculum subjects through *unplugged* (without technology) and *plugged* (with technology) activities.

What Is Computational Thinking?

There are numerous existing definitions of computational thinking (CT) that can be quite confusing or complicated. A modified definition that resonated with our school project team is:

Computational thinking is a problem-solving strategy that is derived from computer science but is also applicable in any domain. This strategy includes the following four core constructs:

> *"When you think of CT, a lot of people are talking about tech and I think the important thing about CT is that it's a way of critically thinking and you can apply it to any situation that you come across."*
> *— 5th Grade ELA Teacher*

Decomposition —
Breaking a problem into smaller, more manageable parts.

Pattern Recognition —
The ability to use prior knowledge to find patterns within the smaller problem that will help solve the complex problem more efficiently.

Abstraction —
Removing unnecessary information and focusing on what is truly important in a given situation.

Algorithm/Debugging —
Developing a series of instructions to solve the original problem, and evaluating the solution to address any errors.

Although there can be more core constructs to computational thinking, these four were determined by this project team to be the core constructs that could be applied toward any subject matter with or without the use of technology. In addition, these core constructs are not always implemented in a linear or step-by-step process, but do tend to have the algorithmic design and debugging components achieved as the last steps to solving a problem.

"Using these skills made students more accountable for their work and for their learning. They were able to problem solve, come up with a plan, and check their work and not always rely on the teacher to give answers."
— 4th Grade Teacher

Why Is Computational Thinking Important?

The term computational thinking has been around for quite some time and most teachers have already been embedding certain elements of computational thinking into their practice without being aware of it. With the development of explicit computational thinking integration strategies, the push for computational thinking into the classroom has grown tremendously. Educators, researchers and policymakers are realizing more and more how valuable understanding and integrating computational thinking is for a multitude of reasons. This includes:

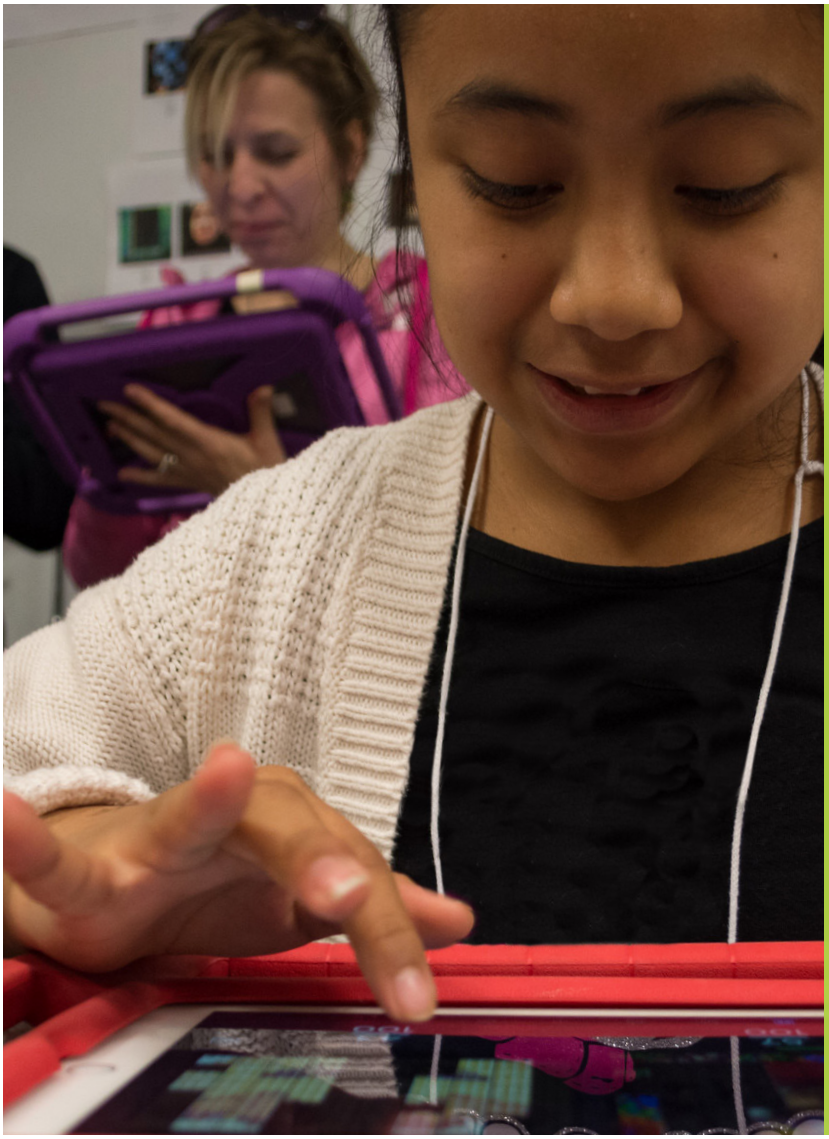
1. Computational thinking is a problem-solving strategy that spans across all subjects and can help students of all grades and abilities tackle complex challenges with or without the use of technology.
2. Computational thinking helps students become active participants in this growing digital age and provides them with the skills necessary to be content creators instead of content consumers.
3. Employee skill requirements are evolving and computational thinking is seen as a vital skills needed for future jobs in any industry.

Additional reasons that evolved through the Integrated Computational Thinking project at PS 13:

4. Computational thinking is a strategy that can help develop student perseverance relating to various school subjects AND is applicable towards problem solving in everyday life.
5. Integrating computational thinking in early elementary provides students with a foundation for learning computer science concepts.
6. Computational thinking can support the learning and development of students' 21st century learning skills and science, technology, engineering and math (STEM) practices.

How Might I Already Be Doing Computational Thinking?

Computational thinking is a process that we all use as we tackle any task or problem in our lives. Even what may seem like a simple classroom task incorporates elements of computational thinking. For instance, kindergarten students learning the routine of entering the classroom is equivalent to learning an algorithm. Teachers are already using computational thinking within their classes, but not necessarily associating it with specific computational thinking vocabulary. At P.S. 13, teachers recognized that computational thinking was not an additional topic that they were expected to teach. Instead they understood computational thinking as a valuable tool that they used more explicitly within their existing lessons, and that would result in students better understanding the content they were learning, feeling more confident in their problem-solving abilities, as well as helping students make connections between classroom learning and their everyday lives.



"I just realized that in computational thinking, it's not necessarily something new or different that you're doing. It's what you're already doing, but more so labeling it, defining it for the kids and teaching them strategies on how to do what they're already doing using computational thinking."

— 5th Grade Math Teacher

Computational Thinking Examples

Below is a chart that demonstrates the computational thinking process and how it relates in everyday tasks or problems.

Everyday Life Example	Decomposition	Pattern Recognition	Abstraction	Algorithm/ Debugging
Complex Task: Clean up a messy room	The floor has to be cleaned, my bed has to be made, and the trash has to be emptied. *The complex task has been broken down into smaller, more manageable tasks.*	Clean clothes go in the dresser. Dirty clothes go in the hamper. *Recognizing how to sort clothes based on patterns or prior experience enables you to accomplish the task more efficiently and reduces errors.*	It doesn't matter what order my books are in as long as they are neatly placed on the shelf. *Focus on the factors that are important to know or to help eliminate unnecessary steps.*	Pick up clean clothes and put them in a closet or dresser. Pick up dirty clothes and put them in the hamper. Sweep the floor, use a dustpan to pick up dirt, and place it in the trash. Throw out the trash and place a new garbage bag in the trash can. Check over everything to ensure that it is clean. *A series of steps to take that optimizes the speed and efficiency of completing the task. This algorithm can also be used to address similar problems.*

Whether you are brushing your teeth, preparing a sandwich, or fixing a tire, you are constantly utilizing your own computational thinking skills to accomplish tasks throughout your day.

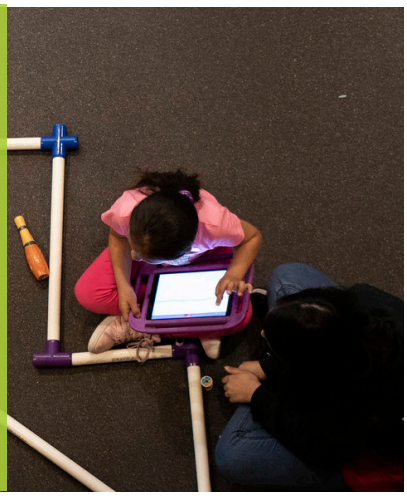
Similarly, students are already using these computational thinking core constructs at school in various subjects, whether it is solving a multiplication problem in math class, creating an essay on World War I for history class, constructing a model of a butterfly lifecycle for science class, or drawing a landscape scene for art class. These tasks all require students to apply computational thinking strategies as they work toward accomplishing each task.

Below is an example of how computational thinking might be used by an individual to perform a task per subject.

Subject/ Assignment	Decomposition	Pattern Recognition	Abstraction	Algorithm/ Debugging
Math — Solve 6(1+2)	What are the operations being used here and which should I start with first?	What are the set of rules that should be applied to solve this problem?	What are math properties that I don't need to be concerned with?	Solve what is in the parenthesis first. Then multiply by 6. Double check answer.
History/ELA — World War I essay	What are the parts of an essay?	What resources can I use to research information for my essay based on past experience (library, internet, etc.)?	What information is irrelevant to include in the essay?	Write the topic introduction paragraph. Write the main body of the essay to support the main point. Write the conclusion to connect all points together. Double check facts, references and grammar.
Science — Design butterfly life cycle model	What are the lifecycle stages and what are the materials that I need to create the model?	What do I already know about using these materials?	What specific criteria did the teacher give me for the model?	Research stages of lifecycle. Draw a circle diagram. Add pictures of each stage and label. Color in model. Double check for accuracy and meeting criteria.

Integrating Computational Thinking at Your School

"[Computational thinking has] helped [students] tremendously with multi-step [math] word problems. Instead of being overwhelmed by the huge problem, they're able to break it into pieces and abstract what's important, and notice patterns within the problem." — 2nd Grade Teacher



Below are recommended steps that will help guide teachers or school leaders on how to integrate computational thinking at their school based on the effective strategies demonstrated in our Integrated Computational Thinking project.

1 Familiarize Yourself With Computational Thinking

Computational thinking can be a complex approach to understand. Begin by familiarizing yourself with examples or resources that are relevant to your work and are easier to understand. This step proved to be beneficial for the project team and helped begin to lay a foundation of understanding around computational thinking.

Below are some resources that we recommend to help get you started:

- *Computational Thinking by JULES*
  This animated video provides a very manageable definition of computational thinking and its core constructs. It also provides an example of applying a computational thinking approach to fixing a flat tire.
  https://www.youtube.com/watch?time_continue=2&v=mUXo-S7gzds&feature=emb_logo
- *Google Computational Thinking for Educators*
  The 5-minute video on the homepage provides an overview of computational thinking and explains the difference between computer science and computational thinking.
  https://www.youtube.com/watch?v=sxUJKn6TJOI&feature=youtu.be
  https://computationalthinkingcourse.withgoogle.com/unit
- *Robin Hood Learning + Technology Fund | Computational Thinking*
  View the impact of computational thinking integration across subjects at PS 13:
  https://www.youtube.com/watch?v=QcIE7ovH-Ck

Computational Thinking Research Articles:

> Six Research Takeaways to Help You Understand Computational Thinking
https://medium.com/tech-based-teaching/six-research-takeaways-to-help-you-understand-computational-thinking-53233d00018

> Research Notebook: Computational Thinking, What and Why? By Jeannette M. Wing
http://people.cs.vt.edu/~kafura/CS6604/Papers/CT-What-And-Why.pdf

2 Dive Deeper With Professional Learning Opportunities

Participating in professional learning experiences will provide you with more in-depth tools and ideas for integrating computational thinking. Check with your school representatives, local or national professional learning providers (*including science centers such as the New York Hall of Science*) for computational thinking workshops that align with your needs and interests.

> *ISTE Introduction to Computational Thinking for Every Educator*
The International Society for Technology in Education (ISTE) provides a free 15-hour online course on computational thinking integration.
https://www.iste.org/learn/iste-u/computational-thinking

> *Google Computational Thinking for Educators*
Google provides a free online course for educators on how to enhance your lessons with computational thinking strategies.
https://computationalthinkingcourse.withgoogle.com/unit

> *New York Hall of Science*
NYSCI provides professional learning workshops on various topics including computational thinking.
www.nysci.org

3 Introduce the Computational Thinking Approach to Your Students

It is important to build a foundational understanding with students on the computational thinking approach in ways that students can relate to and that are meaningful to them. Reviewing early in the school year the computational thinking approach through unplugged methods such as demonstrations or reflective conversations helps students understand the process and how they can apply it to their daily lives and in school. Focusing on computational thinking integration with unplugged activities also helps students transition the computational thinking process to plugged activities once technology-based activities are phased into their class curricula. Below are a few recommended methods that were successful at PS 13:

- Demonstrate to students how to use the computational thinking approach to tackle a complex task that students can identify with on a daily basis (e.g. cleaning a messy room). Providing everyday examples and utilization of the computational thinking approach will create a pathway for students on how to use computational thinking in school. If you need to make specific accommodations for your students, you might want to consider focusing on introducing each computational thinking component individually so they have the opportunity to practice that one skill set at a time.

- For early learners (K – Grade 1), focus on building positive associations and familiarity with computational thinking in relationship with their regular set of activities at home, school and during play. For example, introduce algorithms as a vocabulary term when they are learning to wash their hands after an art activity by asking students to create the algorithm (series of steps) for properly washing their hands. Look for opportunities to practice foundational computational thinking skills within the structure of your normal classroom day or within your lessons targeting early childhood learning skills such as sequencing, sorting and pattern identification.
- Display computational thinking vocabulary charts or posters throughout your classroom or school. These provide constant reference points and reminders for students. (Robbotresources.com provides free computational thinking posters.) Teachers can also develop their own posters or charts that connect computational thinking vocabulary to specific classroom activities or daily tasks.
- Incorporate the computational thinking questioning strategies into your daily classroom facilitation (regardless of class subject) to reinforce a computational thinking mindset with your students. Use the questions below when conducting an activity or assignment in class to help guide students through the computational thinking process and build a computational thinking growth mindset.

CT Core Constructs	CT Facilitation Questions
Decomposition	*How can you break this problem down into smaller steps?*
Pattern Recognition	*What patterns do you notice that may be helpful in solving this problem? What do you already know about this subject that will help?*
Abstraction	*What are the important and unimportant parts to know to solve the problem?*
Algorithm / Debugging	*What are the specific steps that you need to take to solve this problem? How do you test it out to make sure these steps are correct and fix it if needed?*

4 Identify and Integrate Computational Thinking Into Your Lessons

Computational thinking is already happening in your classroom in different ways. Students are utilizing these core constructs throughout their day — in homework assignments, hands-on activities, engineering projects, etc. It is really just a matter of identifying where in your lesson it is already happening, and integrating additional opportunities to best suit your needs and pacing calendar.

The New York Hall of Science developed a Computational Thinking Lesson Planning Tool to help teachers identify where they see computational thinking alignment in their existing lessons or to help teachers begin planning a new computational thinking centered lesson. The Computational Thinking Lesson Planning Tool below is pre-filled with an example of how a teacher would fill out this worksheet as they are comparing it to an existing math lesson on adding and subtracting of decimals (also included below for reference). A blank template of the Computational Thinking Lesson Planning Tool is available at the end of the guide for future use by teachers.

The following are 2 sample activities of a lesson developed by the NYSCI Digital Programming team and is being shared as a precursor to the CT Lesson Planning and Observation tool Procedure.

Adding and Subtracting With Decimals Lesson Sample — 5th Grade Math

Activity 1: Jumbled Algorithm

TEACHING POINT

How can we use the algorithm for addition and/or subtraction of decimal numbers to solve a complex math problem?

STANDARDS ADDRESSED

5.NBT.B.7 – Perform operations with multi-digit whole numbers and with decimals to hundredths.

OBJECTIVE

Upon completing activities, students will:

- Determine an algorithm for adding/subtracting decimal numbers.
- Use their algorithm to solve a complex math problem.
- Identify ways in which elements of computational thinking are utilized towards solving problems.

MATERIALS

- Scissors
- Tape or Glue Stick
- Jumbled Algorithm (Worksheet)

PROCEDURE

This lesson should follow an introduction on the addition/subtraction of decimals and a conversation defining the elements of computational thinking, as this will help students begin to think of the elements of computational thinking as tools for solving problems.

Provide each student with a worksheet that randomly lists all the steps involved in the addition and subtraction of decimal problem such as the items listed below. Once distributed, have students cut the jumbled steps of the algorithm and place them in the correct order.

If you are adding and you get a two-digit answer in one column, you need to "carry the 1" into the next column on the left. If you are subtracting, and the digit being subtracted in a column is larger than the digit above it, you must “borrow” from the next column over on the left.

If the decimal number has less digits on the right side of the decimal point than the other numbers, put zero (0) at the end of the last digit until the number would have the same number of digits at the right side of the decimal point.

Going from right to left, add/subtract each column of numbers.

Insert the decimal point in the answer directly beneath the decimal points in the numbers being added or subtracted.

Vertically stack the numbers you are working with.

Align the decimal points.

Activity 2: The Case of the Missing Winner

MATERIALS

- The Case of the Missing Winner (Worksheet)
- Pen or Pencil

Have students read the problem description in the worksheet:

The sales manager at Dylan's Toys was having a forgetful day. She was late to work because she left her keys at home and had to go back to get them. She also forgot to tie her shoes and tripped on her way to the cash register. She even forgot to give one of her customers their change as that customer left in a hurry. The worst part was that she was supposed to award the toy store's 1,000,000th customer with $100,000, but forgot to look at the receipts as she was finishing each sale! Now the award money could belong to any of the five customers she took care of during her shift, and she has no idea who. There was no way she was going to figure this out on her own, so she hired a detective to help. The only clues we have is the manager's story, a toy catalog with item prices, and the poorly printed receipt that belongs to the 1,000,000th customer. Can you figure out who the 1,000,000th customer was and how much change that person is supposed to get?

After reading the problem description, students should examine the evidence provided in the worksheet (winner's receipt, list of five customers, number of items purchased by each, and cost of each type of item) to then determine the best course of action to solve the problem and identify the missing customer. Ideally, students will note that the receipt belonging to the person we are looking for indicates four items purchased. This allows us to exclude two people from our investigation who purchased a different amount of items. The items purchased by the three remaining customers can be tallied respectively using the prices listed in the catalog. Whichever customer's tallied purchases are equal to the total on the receipt is the winner of the award.

Corresponding Computational Thinking Lesson Planning Tool Example

Teacher Notes Highlighted in Red

Lesson: *Addition and Subtraction of Decimals*	**Objective:** *Understanding and applying the rules for adding and subtracting decimals.* *Describe how your lesson reflects each computational thinking core construct. Use the questions in italics to help guide your answer.*
Decomposition: Breaking a problem into smaller, more manageable parts.	How will your lesson/activity require students to break down a problem into smaller steps? *In activity 1, students have to read over each step, cut each step out, and then sort each step into the proper sequence. In activity 2, students have to read the mystery, review the evidence, and perform several calculations before they can get to their final answer.*
Pattern Recognition: The ability to find patterns within the smaller problem that will help solve the complex problem more efficiently.	What prior knowledge can students use to help solve the problem? What patterns will they notice within the problem? *Students will be using their prior knowledge on adding and subtracting decimals such as lining up decimals correctly and general rules of addition. In the second activity, students will use their familiarity with sales transactions and recognize patterns between the items purchased and those found on the winner's receipt.*
Abstraction: Removing unnecessary information and focusing on what is truly important in a given situation.	What is unnecessary information that students might remove? What should the student be focusing on to solve the problem? *Using the information provided in the problem, students could eliminate one of the potential answers and can focus on analyzing the other receipts.*
Algorithm/Debugging: Developing a series of instructions to solve the original problem, and evaluating the solution to address any errors.	How will students use step by step instructions to come to a solution? How will students check and fix their work? *Students will reorganize an algorithm for adding and subtracting decimals and then test their algorithm to solve the problem. Students will be paired up to check their answers with each other.*
CT Student Reflection: A debrief of the elements used to solve the problem.	Where are opportunities in the lesson for students to explain their solutions using computational thinking terms (decomposition, pattern recognition, abstraction, algorithm, debugging, etc.)? *Students will be given a worksheet where they will need to describe how they used computational thinking as an approach to solve this problem. This will then be reviewed in small groups before a bigger classroom discussion.*

Utilizing this tool will enable teachers to quickly identify areas where they see computational thinking processes happening within their existing lessons and provide guiding questions for teachers to think about practical modifications that they could make in the framing or facilitation of the lesson.

Once you have planned your computational thinking integrated lesson and are ready to implement, you can use the Computational Thinking Observation Tool (example below) to reflect on the

experience and pinpoint areas in need of modification or rethinking. This tool not only captures reflecting on computational thinking integration, but also ensures that the original objectives of the content learning and student engagement are met within your lesson. The tool can be used during the lesson or directly after the lesson when events are fresh in your mind. Below is a pre-filled example based on the same lesson applied in the Computational Thinking Lesson Planning Tool above. A blank template of this tool is also available at the end of the guide.

Computational Thinking Observation Tool

Observation Areas	Checklist	Evidence
Students Understanding of Content Area	☑ Are students using specific content area vocabulary? ☑ Are students correctly applying content area concepts to activity? ☑ Are they able to summarize what they learn?	*• Students are using key terms (sum, difference, place value, etc.) in their conversations about their algorithm for adding and subtracting decimals.* *• They are able to apply their previous knowledge on the topic to all the activities presented today.* *• Students reviewed and discussed what they learn at the end of each activity.*
Students Understanding of Computational Thinking Concepts	☑ Are students correctly using specific computational thinking vocabulary? ☑ Are they applying computational thinking concepts to the activity? Are they specifically grasping: ☑ pattern recognition, ☑ abstraction and ☑ algorithm? ☑ Are they debugging?	*• Students are specifically referencing their algorithm during one of the activities.* *• Students are referencing how they did each of the computational thinking components at the end of the lesson.* *• Some students were able to make connections of certain computational thinking components to situations outside the classroom.*
Student Engagement	☑ Are students paying attention? ☑ Are they responding to questions? ☑ Are they asking questions? ☑ Are they productively collaborating?	*• Students are actively participating in small group and classroom conversations, as well as asking clarifying questions.*
Activity Differentiation	☑ Are there modifications to the lesson for different learners in your classroom?	*• Once students reach the final multi-step problem, they are assigned to specific groups that can allow for specific scaffolding to meet students' needs.*

Once you have completed the Computational Thinking Observation Tool, you can determine the specific areas that you will need to address for improvement and where your lesson was most successful in regards to computational thinking implementation.

5 Incorporate Technology Into Your Classroom

Thinking of ways to utilize technology within your lessons may be daunting due to a variety of reasons including time management, access to technology, or aligning to specific standards. However, investing time in planning how to incorporate technology into your class units will not only help build computational thinking skills, it can also help your students be more engaged in the activity and better understand the subject material or content in your lesson. Below are a few recommended tech tools:

Tech Tool & Online Resource	Recommended Grade	Activity Ideas
Sphero Edu.sphero.com	Grade 4 and up	*Math — Students navigate Sphero through a floor maze avoiding incorrect answers to a series of math problems.* *Art — Using a kiddie-sized pool with the inside floor covered in butcher paper and different spots of acrylic paint, students can program or use sphero basic navigation interface to create an art piece. Sphero must be covered in a protective case and plastic wrap.*
Code & Go blog.learningresources.com/coding/	K – Grade 3	*Math — Place pictures of geometric shapes in the maze and have students select matching shape cards and program the mouse to reach those shapes in the maze.*
Scratch Jr. scratchjr.org	K – Grade 3	*ELA — Create an animation that reflects narrative structure to a story.* *Math — Create a visual math story that demonstrates application of math.*
Scratch scratch.mit.edu/educators	Grade 4 and up	*Math — Design a game utilizing math operations blocks.* *ELA — Design an animation that demonstrates the theme or main points in a story.*
Botley http://blog.learningresources.com/wp-content/uploads/2018/03/Botleylessonplan.pdf	Grade 2 and up	*Science — Use Botley to investigate push and pulls on objects.* *ELA — Create a story based on Botley's obstacle course adventure.*

6 Pace Yourself!

It is important that teachers feel comfortable with how and when they are integrating computational thinking that best suits their classroom needs as well as available resources. Computational thinking can be integrated in many different ways with different levels of intensity. Create a model for computational thinking integration in your classroom or at your school that accounts for available

"I'm not afraid to use the words decomposition and algorithm, because kids feel so smart for being like, 'Oh, my algorithm was...' It's good to use that vocabulary. Some people might be afraid to introduce those types of words to a child, but I use the words interchangeably. Let's decompose the problem. Let's break the problem apart.' And that really helps the kids remember, 'Okay, that's what it means.' By the end of the year, they know." — 5th Grade STEM teacher

resources, time management and administrative support, with opportunities to grow and expand. This may be completely unplugged, beginning with your current lessons to build computational thinking understanding among students, with the goal of working towards plugged activities where students can really apply their computational thinking skills toward coding or programming. The main goal is to apply computational thinking concepts, practices and perspectives that are developmentally appropriate to support student learning.

Additional Resources

The New York Hall of Science offers professional learning opportunities for teachers that provide training and support on computational thinking. Professional learning opportunities range from a one-day workshop for foundational computational thinking training to yearlong intermediate computational thinking coaching programs for extensive training and lesson modeling within the school environment.

Computational Thinking Introductory Course

During this six-hour workshop, participants learn the definition and fundamentals of computational thinking through lesson modeling and reflection of several unplugged activities. Participants also receive training on how to use specific tools that can help them brainstorm and monitor the progress of their own computational thinking unplugged lesson.

Computational Thinking Unplugged Lesson Writing

During this six-hour workshop, teachers collaborate to design and share a computational thinking unplugged lesson draft based on a grade-appropriate lesson goal provided by a NYSCI instructor. Teachers will have the opportunity to have more individualized consultation sessions to address any specific needs or challenges with the audience they serve.

Computational Thinking Technology Introductory Course

During this six-hour workshop, teachers will engage in a computational thinking plugged-based lesson while learning best practices for when and how to use tech in the classroom. Teachers will then be introduced and trained on how to use a grade-appropriate tech tool through a design challenge activity.

Computational Thinking Plugged Lesson Writing

In this six-hour workshop, teachers collaborate to design and share a computational thinking plugged lesson draft based on a grade-appropriate tech tool and lesson goal provided by a NYSCI instructor. Teachers will have the opportunity to have more individualized consultation sessions to address any specific needs or challenges with the audience they serve.

Computational Thinking Coaching

In this school yearlong coaching model, NYSCI staff will work with school leaders to customize professional learning objectives and experiences for a cohort of teachers within your school. Through a combination of group workshops, individual coaching sessions and classroom lesson modeling, teachers receive training and support to ensure computational thinking integration meets the specific needs of their own unique classrooms.

For more information on these professional development programs, contact Anthony Negron, NYSCI's Manager of Digital Programming at anegron@nysci.org.

Computational Thinking Lesson Planning Tool

Use this tool to help guide how you incorporate computational thinking into your lesson.

Lesson Name:	**Objective:**
	Describe how your lesson reflects each computational thinking core construct. Use the questions in italics to help guide your answer.
Decomposition: *Breaking a problem into smaller, more manageable parts.*	**How will your lesson/activity require students to break down a problem into smaller steps?**
Pattern Recognition: *The ability to find patterns within the smaller problem that will help solve the complex problem more efficiently.*	**What prior knowledge can students use to help solve the problem? What patterns will they notice within the problem?**
Abstraction: *Removing unnecessary information and focusing on what is truly important in a given situation.*	**What is unnecessary information that students might remove? What should the student be focusing on to solve the problem?**
Algorithm/Debugging: *Developing a series of instructions to solve the original problem, and evaluating the solution to address any errors.*	**How will students use step by step instructions to come to a solution? How will students check and fix their work?**
CT Student Reflection: *A debrief of the elements used to solve the problem.*	**Where are opportunities in the lesson for students to explain their solutions using computational thinking terms (decomposition, pattern recognition, abstraction, algorithm, debugging, etc.)?**

Computational Thinking Observation Tool

Use this tool to reflect on how well students engaged in computational thinking during your lesson and what modifications might be needed after implementing the lesson.

Observation Areas	Checklist	Evidence
Students Understanding of Content Area	☐ Are students using specific content area vocabulary? ☐ Are students correctly applying content area concepts to activity? ☐ Are they able to summarize what they learn?	
Students Understanding of Computational Thinking Concepts	☐ Are students correctly using specific computational thinking vocabulary? ☐ Are they applying computational thinking concepts to the activity? Are they specifically grasping: ☐ pattern recognition, ☐ abstraction and ☐ algorithm? ☐ Are they debugging?	
Student Engagement	☐ Are students paying attention? ☐ Are they responding to questions? ☐ Are they asking questions? ☐ Are they productively collaborating?	
Activity Differentiation	☐ Are there modifications to the lesson for different learners in your classroom?	