

# LESSON 4

## ALGORITHMS

1ST GRADE



Lesson created by the GMU-ODU CSforAll Team. For more information about this lesson and our CSforAll initiative, contact Dr. Amy Hutchison at [achutchison1@ua.edu](mailto:achutchison1@ua.edu)

# SUMMARY AND STANDARDS

Summary: In this lesson, students will sequence a story with a beginning, middle, and end, and write an algorithm for how the wolf got from house to house in the story of the “Three Little Pigs.”

## ELA Standards:

### Communication and Multimodal Literacies:

- 1.1 The student will continue to demonstrate growth in the use of oral language.
- a) Listen and respond to a variety of electronic media and other age-appropriate materials.
  - b) Tell and retell stories and events in logical order.
- 1.9 The student will read and demonstrate comprehension of a variety of fictional texts.

## CS Standards:

- 1.1 The student will construct sets of step-by-step instructions (algorithms) a. using sequencing;
- 1.2 The student will construct programs to accomplish tasks as a means of creative expression using a block based programming language or unplugged activities
- 1.3 The student will analyze, correct, and improve (debug) an algorithm that includes sequencing and simple loops, with or without a computing device.

## **MATERIALS AND RESOURCES NEEDED FOR THIS LESSON:**

- Teacher slide deck (see website)
- [Student slide deck](#)
- Video read aloud of “The Three Little Pigs”:  
[https://www.youtube.com/watch?v=FNYBQsay\\_Ek](https://www.youtube.com/watch?v=FNYBQsay_Ek)

## **LESSON OBJECTIVES: I CAN...**

- Define and give examples of “algorithms”
- Write an algorithm
- Debug an algorithm

### **Vocab:**

- Algorithms



**REVIEW**

## COMPUTER SCIENCE



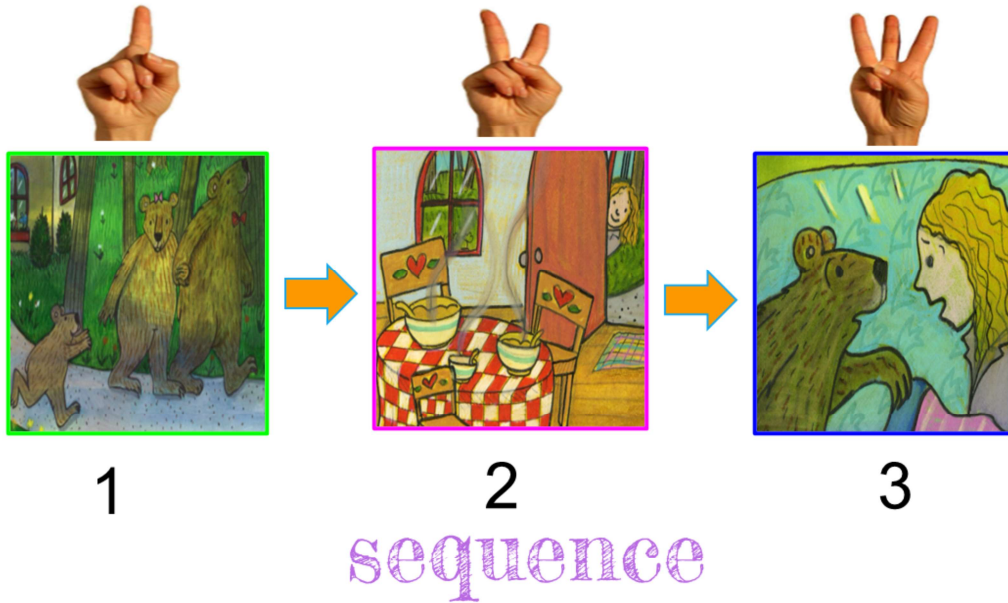
## Review key academic vocabulary and link concepts

- Computer science

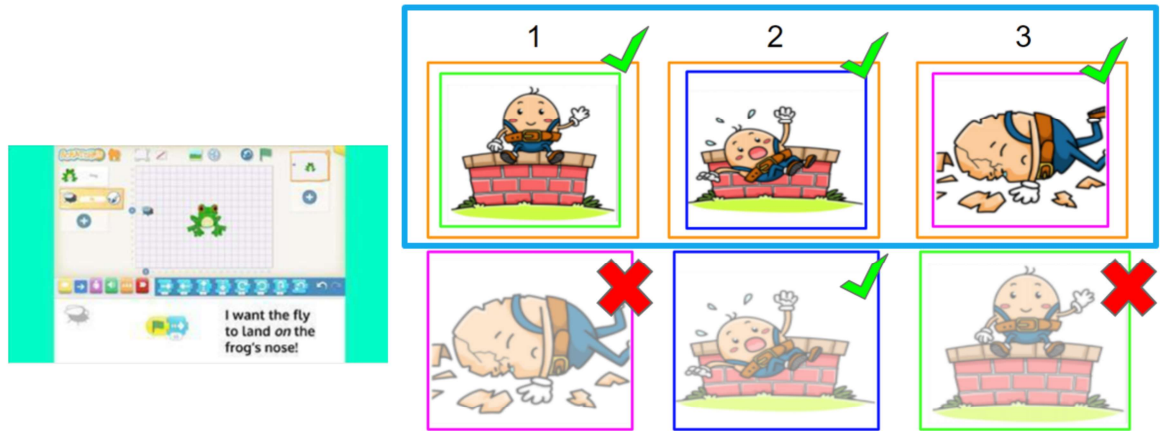
“Welcome back to **computer science** in Second Grade!

**Computer science** is using the power of **computers** (show “computer” word wall card) to solve our problems and express ourselves.

## LET'S THINK ABOUT OUR LAST COMPUTER SCIENCE LESSON...

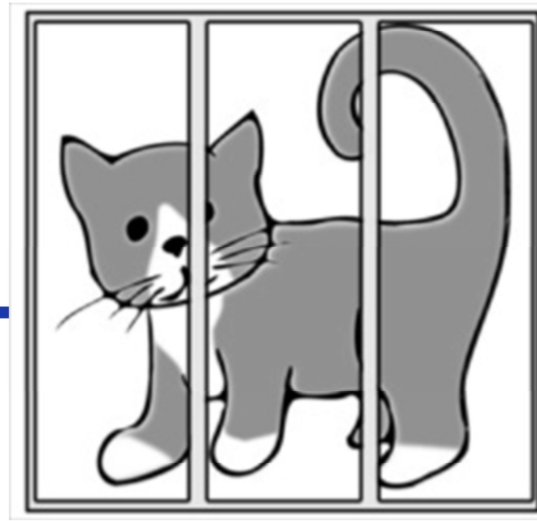


Last time, we learned that computer scientists put code in the correct **sequences** to tell the computer what to do.



debug

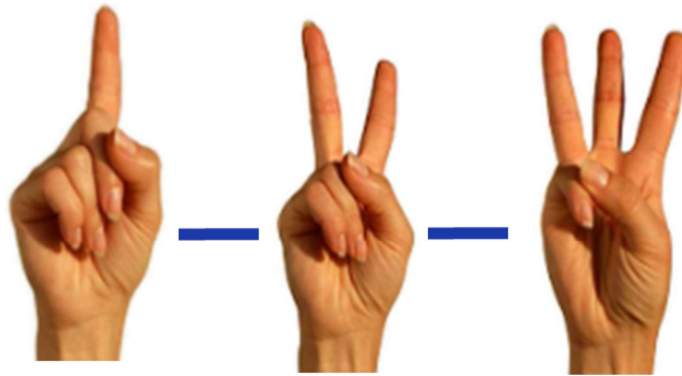
We also learned that if there's a problem in the sequence, we can **debug** it by finding the problem and fixing it.



abstraction

Today, we're going to learn two new vocabulary words. The first one is "abstraction."

## REMEMBER THIS? A SUMMARY IS AN ABSTRACTION!



1

2

3

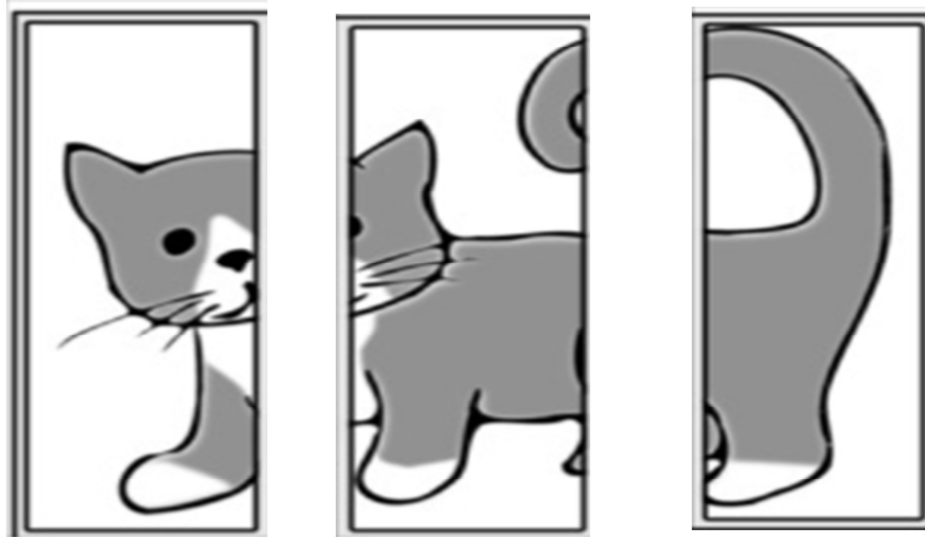
Beginning 🖐️

Middle 🖐️

End 🖐️

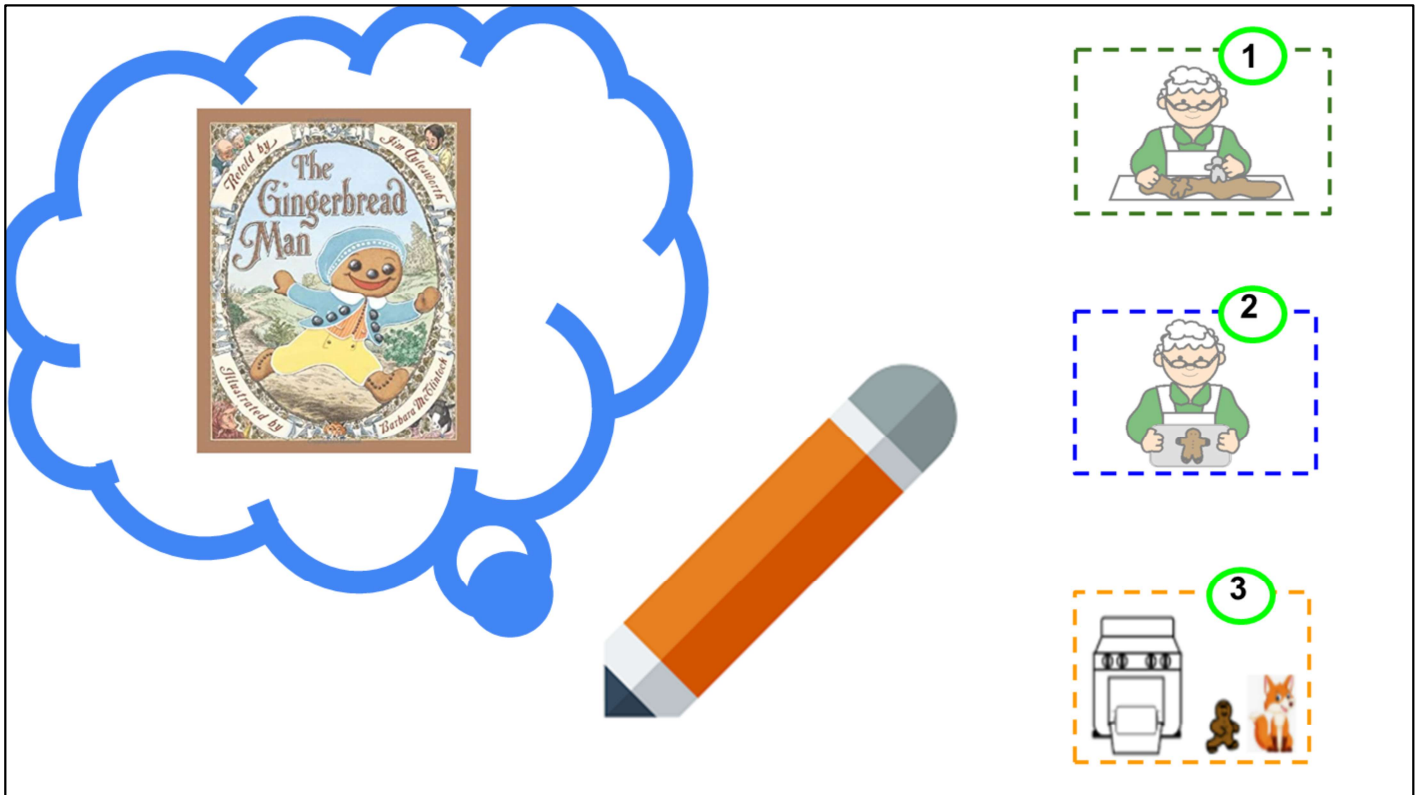
When we summarized our stories last time, we actually used abstraction! It would not be efficient to retell the entire story to someone. So when they ask you what a story or movie was about, you pick out the most important details.

Computer scientists are like this, too! They want the most efficient code that only tells the computer the necessary information. That means that they find *the most important parts*. This helps computer scientists focus on finding a solution that can help fix more than just a single problem.



## decomposition

In abstraction, you filter out unnecessary information. You make decisions about all the details and decide which ones are the most important. Decomposition is a problem-solving strategy that simply means “breaking a problem or process down into smaller parts.”



So a great example of decomposition in class is when we try to write paragraphs of our own! It can be overwhelming to write a full paragraph all at once so we break it down into our topic sentence and main idea. If we were going to WRITE our own story we'd have to plan the beginning, middle, and end separately rather than writing it all at once.

Jim Aylesworth, the author of *The Gingerbread Man*, didn't come up with the idea for his story all at once! He had to write it in multiple parts.

Computer scientists don't always tell stories or decompose problems across their fingers, but they use the same idea of **decomposition** that we used for the *Gingerbread Man*, by thinking about breaking their code down into smaller projects.



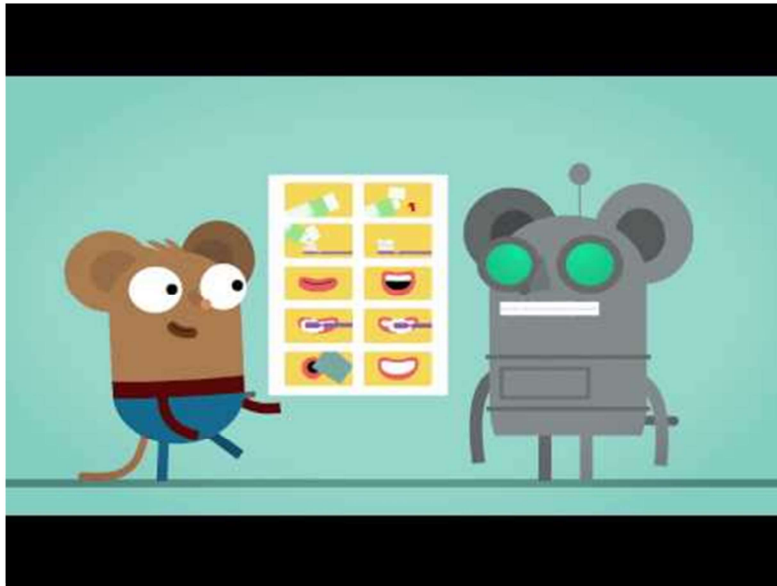
**DIRECT INSTRUCTION**



algorithm

Today we are going to learn about **algorithms!** **Computer scientists write algorithms in code, like the one on the screen, but we apply the same concept in lots of other areas of our lives too.**

## WHAT IS AN ALGORITHM? -CLICK ON IMAGE



“Today, we are going to watch a video. While we are watching, I want you to think about the computer science words that you hear and what they mean. Do you hear words we have talked about when we have been learning about computer science? Do you hear new words?”

What new word did you hear? *Provide students an opportunity to share.* That’s right.

Yes, like we mentioned earlier, today is all about **algorithms**. And, the video was ALL about **algorithms**! An **algorithm** is a set of directions!

Video link:

<https://www.youtube.com/watch?v=Da5TOXCwLSg>

# GUIDED INSTRUCTION

# **ALGORITHM: SET OF DIRECTIONS; A LIST OF STEPS TO FINISH A TASK**

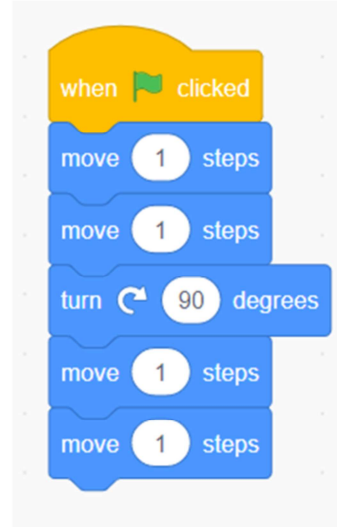
An Algorithm is a list of steps or commands to finish a task.

## ALGORITHMS

- Use the correct sequence
- Be clear and precise



**Code:** The language that computer scientists create and use to tell a computer what to do.



Algorithms must be written in the correct sequence so that others, humans or a computer, can follow the orders to complete a task. This task can be anything, so long as you can give clear instructions for it. Think about an algorithm for getting your shoes on in the morning, you must put on your socks before you put on your shoes!

In computer science, we write our algorithms using code. Code is the instructions that computer scientists create and use to tell a computer what to do.

Writing code is like writing the commands for a computer. When you know how to write code, you can tell computers what to do.



But you can write instructions for humans as well as computers! Recipes are one great example. You can write recipes for:

- how to make hot chocolate
- how to make a yummy treat
- how to make a chocolate, vanilla, or chocolate milkshake!
- how to make koolaid
- how to make lemonade



This is an example of a book where a little girl writes an algorithm for how to build a sandcastle! Our recipes and instructions are like algorithms. Just like in computer science, they were all focused on explaining a task to the reader.



# ALGORITHMS = SET OF INSTRUCTIONS

We may wish to write algorithms for how to....

- Get somewhere (directions)
  - To the cafeteria
  - The park in your neighborhood
- Do something (instructions)
  - Build a fort in your living room
  - Do a dance
  - Shoot a basketball or kick a soccer ball
  - Create a craft
- Explain something
  - How your family celebrates the holidays
  - About someone important to you or someone famous
  - How something happens, such as photosynthesis or the water cycle



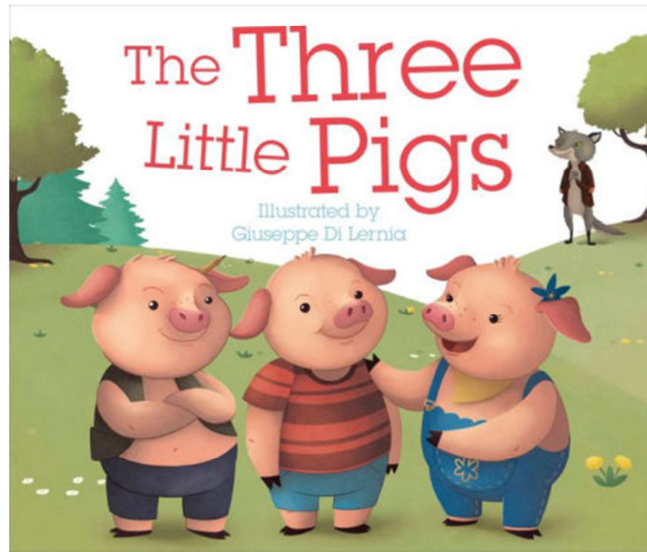
There are lots of times when you may wish to explain something to someone! [read slide]

**TURN & TALK: HOW WOULD YOU  
WRITE AN ALGORITHM FOR  
“GETTING READY FOR SCHOOL”?**

Give students time to discuss. Call on a few students for examples.


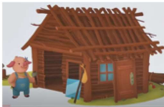
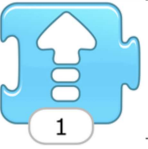

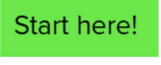
# INDEPENDENT PRACTICE


## **“THE THREE LITTLE PIGS” READ ALOUD**



[https://www.youtube.com/watch?v=FNYBQsay\\_Ek](https://www.youtube.com/watch?v=FNYBQsay_Ek)

Optional: Play this 5-minute read-aloud of the Three Little Pigs **OR** just remind students of the story.

 Each square equals a single step

Today you are going to write an **algorithm** to guide the Big Bad Wolf to visit each of the three little pigs' houses following the **sequence** in the story. You will do this by using code blocks like the blue block on your screen or by drawing arrows on your paper. The instructions are on the next screen.

## INSTRUCTIONS: BIG BAD WOLF'S ALGORITHM

Write an **algorithm** to guide the Big Bad Wolf to visit each of the three little pigs' houses following the **sequence** in the story.

You may choose to use either words or coding blocks to write your algorithm.

**BONUS:** can you find a way to **abstract** any steps in your algorithm to make it shorter and more efficient?

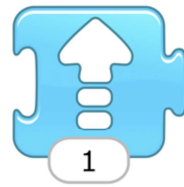
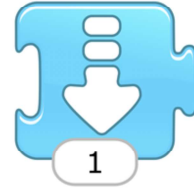
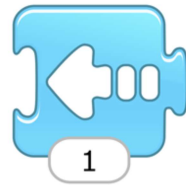
Read slide.

## WORD & CODE BANKS

### Word bank:

- Move left one step
- Move right one step
- Move up one step
- Move down one step
- Turn right one time
- Turn left one time

### Code bank:















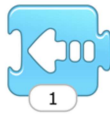




These are the blocks you can use to write your algorithm, or you can draw arrows like these on your paper.

# PAIR DEBUGGING!

Swap your neighbor and see if you can spot and debug any errors in their algorithm.



						
	 1	 1	 1	 1	 1	 1
	 1					
	 1					
	 1	 1	 1	 1	 1	Start here!

 Each square equals a single step

Here is my solution. Does yours like like this? Did you choose a different path? Did you remember to turn? What would happen if you followed the code you created?

# VOLUNTEERS?

Would anyone like to share their algorithm with the class?

**WRAP UP**

## LET'S REVIEW



## ALGORITHM

### Review algorithms

- Ask students to explain what **algorithms** are to a friend or family member *and* think about another **algorithm** you may want to try.

“Great job, computer scientists! You were working to create algorithms in ScratchJr to make a part of our story come to life. Remember, we are learning to be computer scientists. We can use **algorithms** every day to give computers directions about what to do or to solve problems. Tonight, I want you to explain what **algorithms** are to a friend or family member *and* think about another **algorithm** you may want to try.

## YOU CAN BE A COMPUTER SCIENTIST!



Using your **pattern recognition**, **sequencing**, **abstraction**, **decomposition** and **algorithm-writing** skills, you can be a computer scientist!

Remind students that they are computer scientists and writers. They can use computer science skills like finding **patterns**, **debugging**, **decomposing**, **abstracting**, and **coding algorithms** to do important work, like reading and writing stories.

“Today and every day, you are computer scientists and writers. We can use our computer science skills, like finding **patterns**, **debugging**, **decomposing**, **abstracting**, and **coding algorithms** to do important work, like reading and writing stories. You are experts!”

## TODAY'S CAREER IN TECH: MEET TESS



Play video