

Robot Teacher

Programming and Algorithms

Teresa Lansford, Patricia Turner, Lindsey Link
Published by *Oklahoma Young Scholars/Javits*

This work is licensed under a Creative Commons CC BY-SA 4.0 License

Grade Level	1st – Kindergarten Grade	**Time Frame**	100
		Duration	3 periods

Essential Question

How do computers think and work? How can we make computers do things we want them to do?

Summary

Computer programming helps young learners to understand how to think things through step-by-step, solve problems, and make adjustments based on feedback. There are many great resources now for both online and offline coding and using algorithms to solve problems for learners of all ages. These lessons give some of our youngest learners the chance to see themselves as computer programmers, understand how computers “think,” and analyze a problem to seek solutions while they learn about the term "algorithm."

Snapshot

Engage

Students control a "robot teacher" to learn how robots can only act on what they are told to do, are literal, and need detailed guidance.

Explore

Using BeeBots, or by watching videos of robots doing work, learners think about how something is programmed to do work.

Explain

Students learn the term "algorithm" and create their own algorithms to control their classmates in a follow-the-leader-style game.

Extend

The class works as a team to solve coding problems using code.org.

Evaluate

Students work in pairs or individually to create their own algorithms to solve coding problems on code.org.

Standards

Oklahoma Academic Standards for Computer Science (First Grade)

K.AP: Algorithms & Programming
K.AP.A: Algorithms
K.AP.A.1: With guidance, model daily processes and follow algorithms (sets of step-by-step instructions) to complete tasks verbally, kinesthetically, with robot devices, or a programing language.
K.AP.C: Control
K.AP.C.1: With guidance, independently or collaboratively create programs to accomplish tasks using a programming language, robot device, or unplugged activity that includes sequencing and repetition.
K.AP.PD: Program Development
K.AP.PD.3: With guidance, independently or collaboratively debug programs using a programming language and/or unplugged activity that includes sequencing and repetition
K.AP.PD.4: Use correct terminology (first, second, third) and explain the choices made in the development of an algorithm to solve a simple problem.

Attachments

- Arrow Cards—Robot Teacher.docx
- Arrow Cards—Robot Teacher.pdf
- Lesson Slides—Robot Teacher.pptx

Materials

- Picture book (any)
- BeeBots or other programmable robots (optional)
- Internet access
- Computer or tablet for video display
- Arrow cards (attached; 1 set per pair of students)
- Student devices for exploring code.org (optional)
- Robot Teacher Lesson Slides (attached)
- Chart paper
- Markers

20 minutes

Engage

Use **slides 1-4** to introduce the lesson.

Teacher's Note

You can opt to be the "Robot Teacher" in this activity or recruit a guest to play along. If you use a volunteer, make sure they understand they can only do the exact actions students tell them to do. They will need to be very literal.

Display **slide 5**.

Introduce Robot Teacher by telling students that today they have a robot teacher. Robot teachers cannot think for themselves and, just like a robot, can only do what they are told to do. Their robot teacher wants to read a book but has never been programmed how to do it. It is the student's job to tell the robot teacher step-by-step what it needs to do to read the book. Have the students tell the "robot teacher" step-by-step what to do. Be very literal in following directions. If a student says to "open the book," open it in the middle or upside down. This is a great way to also integrate book handling and concepts into this lesson and can serve as a formative assessment as to learner knowledge and ability in regard to book skills. Have fun with this and emphasize that robots/computers are only able to do what we tell them to do.

Additional Robot Teacher examples

For "pick up the book" hold it over your head. If you have the book upside down, and your classmates say, "turn it over," turn it to look at the back. If they say, "go to the front," walk to the front of the room.

Students will learn that robots do not think for themselves, and we must be detailed in our directions for success as computer programmers.

Move to **slide 6**. Using a Think-Pair-Share, ask the students: What was fun about commanding the robot teacher? What was hard? Have them think, share with a partner, and then share out with a large group.

20 minutes

Explore

Display **slide 7**. BeeBots are a great concrete way for students to see that their commands matter in programming. If you have access to BeeBots, or other coding robots, let students have time to play, wonder, and ask questions commanding these robots. Record what students notice and still have questions about after practice programming robots in an I Notice I Wonder chart.

For classes without access to robots, use all or some of the videos on **slides 8-12** to explore how robots do work. After each video, ask students: What work is the robot doing? How do you think computer programmers were able to teach these robots to do this work?

Teacher's Note

You can choose to watch all or only some of the videos from the list depending on student interest and attention.

Embedded video
https://youtube.com/watch?v=JMLPhk6b0gA

Robot Construction Worker

Embedded video
https://youtube.com/watch?v=hm9ZSN37jVM

Robot Painter

Embedded video
https://youtube.com/watch?v=uhND7Mvp3f4

Robots Balancing and Moving

Embedded video
https://youtube.com/watch?v=M8YjvHYbZ9w

Robot Dog

20 minutes

Explain

Move to **slide 13**. Share with students that computer programmers have a big word they use for the steps to get a computer or robot to do the work they want it to do. That word is *algorithm.* An algorithm is the list of steps that get a robot or computer program to its goal. Today, the class is going to create an algorithm to get their friends to move around.

Place students in pairs. Give each pair of students a set of arrow cards. Tell them these are going to be the commands they give to their friends. Let them know that the person who decides on the commands is called a programmer. The order of the arrow cards will be their algorithm for getting their friends in the class to move around. Give students time to decide on their algorithm and have them stack their cards in the order they want their classmates to move. Call up one group of "programmers" at a time. Be very deliberate in using the vocabulary and having students use it as well. Call up groups of "programmers" and ask them share their "algorithm." Ask "What are the steps in your algorithm?" and have them respond "Our algorithm is . . ." Have them hold up their first card. The class follows the leader by doing what the card shows. Have the pairs show their cards until they have completed their program. Then call the next group.

Some students may notice that when facing the class, they may plan for everyone to move right but to them it looks like everyone is moving left. This is a great programming skill to notice. When programming, you have to think of the robot's perspective. Sometimes we have to turn around ourselves and stand like the robot in order to understand the directions we are giving.

At the end, ask pairs to share with one another what an algorithm is. Look for responses that include the commands you give to a computer or robot to get work done.

Teacher's Note

This would be a great activity to do with buddies from an older grade. The younger students would enjoy getting to control their big kid robots.

20 minutes

Extend

Teacher's Note

Explore the following activity before doing it with the class so that you are familiar with how to drag and drop code as well as run the program. Don't feel like you have to be an expert coder. Learning together enables you to authentically model problem solving with students.

Move to **slide 14**. Share with students that the class will use this site to learn to use block coding. Exit the slide presentation and visit the introduction to coding at Code.org (https://studio.code.org/hoc/1)

As a class, work together to complete coding puzzles. This coding experience from Code.org lets you see how changes to your code affect your character's movement. It has tutorial videos to guide your class through each step of the coding process. Students can take turns coming up to the board or computer to drag blocks into place or you can move the blocks of code.

Do as many levels as are developmentally appropriate for your class.

When you have finished coding, return to **slide 15** in the presentation. Work as a class to make an Anchor Chart for tips when coding. Use block coding based on what you learned from the game. Make sure to include the definition of *algorithm* in your Anchor Chart.

Teacher's Note

This Anchor Chart will be used as a resource by students in the evaluate phase.

20 minutes

Evaluate

To evaluate students, there are several choices depending upon the developmental level of the students and access to devices.

1. Students can return to https://studio.code.org/hoc/1 and try the tasks again, this time independently or with a partner.
2. Students explore https://studio.code.org/s/dance-2019/stage/1/puzzle/1 to program a dancer on their own devices independently or with a partner.
3. If you are comfortable giving students access to games you are not yet familiar with, you can direct students to this link, https://code.org/learn, to view all coding options and sort by grade level.

Note that not all games will work on all devices. Prepare students by letting them know that if a game does not work or does not seem a good fit for them, then they should try something else.

Teacher's Note

Offline Coding activity: If access to devices is limited, this offline activity can be printed off for students to show their understanding of algorithms:

https://n2y.com/n2y-coding-activity/

Teacher's Note

It is important to share with students that "there is no bad code." If a code does not work, they should not delete it all, but instead look through their steps to see where the code goes wrong. Compare it to a popular video game: If Mario were not jumping to the right height, the gaming coders would not throw away the whole game. They would figure out which part was not working and make small changes until they had it right. This is a great opportunity to emphasize having a growth mindset and grit.

As students are coding, take note of their ability to problem solve, troubleshoot, and think through step-by-step instructions. Those that struggle could join you at a teacher table to work through the steps. Refer them to your class Anchor Charts for ideas as well.

Resources

- AwesomeTech. (March 7, 2020). Boston Dynamics' amazing robots Atlas and Handle. [Video]. Youtube. https://www.youtube.com/watch?v=uhND7Mvp3f4
- Boston Dynamics. (February 9, 2015). Introducing Spot Classic (previously Spot). [Video]. Youtube. https://www.youtube.com/watch?v=M8YjvHYbZ9w
- Code.org. (n.d.). Classic Maze. https://studio.code.org/hoc/1
- Code.org. (n.d.). Hour of Code Activities. https://code.org/learn
- Code.org (2019). Dance Party. https://studio.code.org/s/dance
- K20 Center. (n.d.). Anchor Charts. Strategies. https://learn.k20center.ou.edu/strategy/58
- K20 Center. (n.d.). BeeBots. Tech Tools. https://learn.k20center.ou.edu/tech-tool/606
- K20 Center. (n.d.) I Notice, I Wonder. Strategies. https://learn.k20center.ou.edu/strategy/180
- K20 Center. (n.d.). Think-Pair-Share. Strategies. https://learn.k20center.ou.edu/strategy/139
- Mashable. (November 8, 2018). This robot is training to become a construction worker: Genius moments. [Video]. Youtube. https://www.youtube.com/watch?v=JMLPhk6b0gA
- N2y.com. (2018). Coding Activity. n2y.LLC. https://n2y.com/n2y-coding-activity/
- Okibo. (February 27, 2020). Autonomous Painting Robot. [Video]. Youtube. https://www.youtube.com/watch?v=hm9ZSN37jVM