NATIONAL GEOGRAPHIC

RESOURCE LIBRARY
ACTIVITY : 35 MINS

How to Train Your Robot

Students apply logic and sequencing skills to write instructions, called an algorithm, for a student to complete a simple task acting as a robot.

GRADES

2 - 8

SUBJECTS

Engineering

OVERVIEW

Students apply logic and sequencing skills to write instructions, called an algorithm, for a student to complete a simple task acting as a robot.

For the complete activity with media resources, visit:
http://www.nationalgeographic.org/activity/how-train-your-robot/

Program

NATIONAL GEOGRAPHIC PRESENTS ROBOTS 3D

DIRECTIONS

1. Introduce the concept that robots must be programmed to perform tasks.

Ask: *How does a robot know what to do in order to complete a task?* Explain that robots are preprogrammed with a set of specific instructions—they do not inherently know what to do or learn. They have to be programmed to do each of these things. Explain that these

instructions are just like any other set of instructions. Ask: *How do you know how to use a kitchen appliance, like a microwave or can opener?* (Someone gave them instructions, or they learned on their own.)

2. Demonstrate how robots need specific step-by-step instructions to complete a simple task.

Ask a volunteer from the class to perform a simple task. For instance: *Please bring me a tissue*. The volunteer understands the instruction and brings a tissue to the teacher. Thank the student. Explain that a simple task, like getting a tissue, is actually a series of much smaller instructions. Ask: *What is the first step the volunteer took to bring me a tissue?* (It should be along the lines of stand up. Push students to get all the way to the beginning of the process with the answers they share.) Explain that robots need to be instructed all the way through the process; not even the smallest step can be omitted.

Choose another volunteer. Explain that you will act like a robot and the volunteer will give you a basic task to complete, like sharpening a pencil. If the student provides general instructions, you should pretend not to understand how to complete the task. For example, if the student says, "Please sharpen this pencil," you should hold the pencil, look confused, and ask for more specific instructions. Each time, follow the instructions exactly as given. If you are asked to put the pencil in the hole, find any hole in the room in which to stick the pencil, except the hole in the sharpener. Students should realize this will take carefully thought-out instructions to get the task accomplished. A good set of instructions might go as follows: Stand up, lower your arm toward the pencil, open your hand and place it on the pencil, close your hand around the pencil, raise your arm, take seven steps to your right, lift the arm of the hand that holds the pencil 1 m (3 ft), insert the pencil into the hole on the pencil sharpener, lift your other arm 1 m (3 ft), open your hand, grasp the handle of the pencil sharpener and turn it ten times. This should continue until the student volunteer is successful, or the students understand that instructions are complicated.

3. Discuss what was learned from the definition and define words associated with programming robots.

Ask students to identify the differences between the student's ability to follow the teacher's instruction to bring her a tissue and the teacher's ability, acting as a robot, to follow instructions to sharpen a pencil. Tell students that robots require precise step-by-step instructions to complete a task and that these instructions are known as algorithms.

Tell the students that an algorithm is much like a recipe for their favorite cake; if you do not follow the order of the instructions or use the correct ingredients, the food will not turn out as expected. If you do not tell the robot precisely what to do and in what order, or leave out any steps, the robot will not perform as expected.

Humans can use information stored as knowledge to help them perform tasks with less specific instruction. Ask: *What would happen if there was a problem with the instructions provided to a robot?* (The robot would be unable to complete the task.) Explain that if there is a problem with the instructions, someone must go back and find the error to debug the program. Redefine debugging as finding and fixing problems in the instructions.

4. Students create an algorithm for a robot to perform a specific task.

Separate students into pairs. Each pair will decide on a simple task for a robot to complete. Good examples are task-based activities, like asking the robot to pick up a piece of paper and put it in the wastebasket, or open a notebook. Another option is to give students a choice from a list of tasks predetermined by you. If students pick their own task, have them clear it with you before they begin writing their algorithm.

Each pair will collaborate to create basic instructions for the robot to accomplish the task and write them out in a step-by-step fashion on task cards. Each task should go on a separate card. For example: Take one step forward with your right foot. Take one step forward with your left foot. Repeat commands 1 and 2 five times. Turn your entire body to the right 90 degrees. Take one step forward with your right foot. Take one step forward with your left foot. Repeat command 5 and 6 ten times, and so on. Very specific instructions are important. They must specify how far to lift something, exactly how many steps, and the specific way to

pick up an item. For example, “Pick up the cup with your hand” is not a good command. Writing out, step-by-step, how to open the hand, how to find the cup with the hand, how to grasp the cup and lift it up, is the proper way to write an instruction.

Task cards can be made using notecards, notebook paper, or in student journals. If using notecards to make task cards, using a single card for a specific part of the task works best. If using notebook paper or journals, separate the commands for each step with a line or border of some kind.

5. Students “beta” test their algorithms.

When all pairs are satisfied with their first set of instructions, have one student read the instructions to his or her partner. The partner will do exactly as instructed, and will need to listen carefully and not compensate for missing steps. The student reading the algorithm will make notes on the sheet of instructions next to items that need to be changed or added to.

6. Students debug their algorithms and run a second test.

After the first test, they will debug their algorithms by fixing the problems encountered using the notes they took. Be sure to explain to the students that debugging is considered a normal part of the process and should not be seen as a failure; all programmers expect to have some problems during testing. When finished with the debugging process, have the pair trade roles so the other student will read the instructions while the second follows the commands. Have students take notes as they perform the second test, and then use their notes to debug a second time. This process should continue until the pair is satisfied that the instructions are good.

7. Check for understanding by playing the part of the robot while each pair reads their algorithm.

The teams’ algorithms are tested. One member of the pair will read the instructions to the teacher, who performs the task.

8. Students discuss what they discovered during the process of creating an algorithm.

Ask for students to share what they found most challenging. Ask: *What are the qualities of good and bad instructions?* Ask: *Did you think this activity would be easier or harder than you originally thought? Why?* Ask: *What is the most important thing you learned about programming robots?*

Tip

Do not allow students to pick tasks that are too complicated or involve completing more than one task. The best tasks are extremely simple.

Tip

If the students are writing individual algorithms, the partner who will be completing the task should not be allowed to hear or see what the task is before attempting to follow the algorithm. Consider having the partner step into the hall or wait in another classroom during the creation of the algorithm.

Modification

In Step 1 the teacher can pick any task to assign the volunteer, as long as the teacher sticks with insisting that the volunteer clarify instructions.

Modification

This activity can be led as a demonstration for younger students. The teacher can write the steps on the board as the class offers ideas for each step and for debugging the steps.

Informal Assessment

Working pairs will be assessed on how well the teacher is able to follow their written instructions to perform a given task. Students will turn in their debugged algorithms when the activity is complete.

Extending the Learning

Instead of having pairs work together to write an algorithm, split them so the students write algorithms individually. When the pairs reunite, they can take turns running the algorithm with their partner. This will challenge the older programmers to really think about how much detail is needed to make their robots “understand” their task.

OBJECTIVES

Subjects & Disciplines

- Engineering

Learning Objectives

Students will:

- Understand robots operate to complete a task by following a list of sequential instructions
- Understand that errors in programming, known as bugs, will cause problems with task completion
- Write an algorithm with clear and precise steps

Teaching Approach

- Learning-for-use

Teaching Methods

- Cooperative learning
- Discovery learning
- Discussions
- Experiential learning
- Hands-on learning
- Information organization
- Role playing

Skills Summary

This activity targets the following skills:

- 21st Century Student Outcomes
  - Learning and Innovation Skills
    - Communication and Collaboration
    - Creativity and Innovation
    - Critical Thinking and Problem Solving
- Critical Thinking Skills
  - Analyzing
  - Applying
  - Creating
  - Evaluating
  - Remembering
  - Understanding
- Science and Engineering Practices
  - Asking questions (for science) and defining problems (for engineering)
  - Constructing explanations (for science) and designing solutions (for engineering)
  - Developing and using models
  - Planning and carrying out investigations
  - Using mathematics and computational thinking

National Standards, Principles, and Practices

NATIONAL SCIENCE EDUCATION STANDARDS

• **(5-8) Standard E-1:**
Abilities of technological design
• **(5-8) Standard E-2:**
Understandings about science and technology
• **(K-4) Standard E-1:**
Abilities of technological design
• **(K-4) Standard E-2:**
Understanding about science and technology

NEXT GENERATION SCIENCE STANDARDS

• **Engineering Design:**
MS-ETS1-2. Evaluate competing design solutions using a systematic process to determine how well they meet the criteria and constraints of the problem.
• **Engineering Design:**
MS-ETS1-3. Analyze data from tests to determine similarities and differences among several design solutions to identify the best characteristics of each that can be combined into a new solution to better meet the criteria for success.
• **Engineering Design:**
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.
• **Engineering Design:**
3-5-ETS1-2. Generate and compare multiple possible solutions to a problem based on how well each is likely to meet the criteria and constraints of the problem.
• **Engineering Design:**
3-5-ETS1-1. Define a simple design problem reflecting a need or a want that includes specified criteria for success and constraints on materials, time, or cost.
• **Engineering Design:**
MS-ETS1-1. Define the criteria and constraints of a design problem with sufficient precision to ensure a successful solution, taking into account relevant scientific principles and potential impacts on people and the natural environment that may limit possible solutions.

Preparation

What You'll Need

MATERIALS YOU PROVIDE

- Paper
- Pencils
- Objects needed for tasks

REQUIRED TECHNOLOGY

- Internet Access: Required

PHYSICAL SPACE

- Classroom

SETUP

Set up your classroom so groups have plenty of room to work separately.

GROUPING

- Large-group instruction

BACKGROUND & VOCABULARY

Background Information

When we are given a task to complete, we can often be successful with very little instruction. We use our prior experiences to quickly identify the necessary steps and execute to complete the task. In identifying the needed steps, we are creating an algorithm in our head. An algorithm is a series of instructions on how to complete a task. Because robots require detailed instructions, developing an algorithm is typically the first step, and is used as a framework on which to build the program.

Prior Knowledge

[]

Recommended Prior Activities

- None

Vocabulary

Term	Part of Speech	Definition
algorithm	*noun*	set of steps for solving a mathematical problem.
bug	*noun*	mistake or flaw in a computer program.
code	*noun*	instructions written in symbolic language used in programing a computer or robot.
debug	*verb*	to detect and remove errors from a computer program.

Term	Part of Speech	Definition
program	*noun*	set of coded instructions for the automatic performance of a task provided to a robot or computer.

For Further Exploration

Video

- Robot Pals

FUNDER