

Course E

Created with fourth grade students in mind, this course begins with a brief review of concepts previously taught in courses C and D. This introduction is intended to inspire beginners and remind the experts of the wonders of computer science. Students will practice coding with algorithms, loops, conditionals, and events before they are introduced to functions. At the end of the course, students will have the opportunity to create a capstone project that they can proudly share with peers and loved ones.

Teacher Links: [Teacher Videos Playlist](#)

Lesson 1: Programming: My Robotic Friends

Algorithms | Debugging | Unplugged

Turn your friends into robots and tell them what to do!

Lesson 2: Sequences in Maze

Sequencing | Debugging | Maze

In this lesson, you will learn how to write your very own programs!

Lesson 3: Building a Foundation

Unplugged | Persistence | Frustration

Build a structure that can hold a textbook. You might feel frustrated- remember to be persistent!

Lesson 4: Debugging in Scrat

Debugging | Ice Age

Find problems in Ice Age puzzles and practice your debugging skills.

Lesson 5: Programming in Artist

Program | Programming | Artist

Create beautiful images by programming the Artist.

Lesson 6: My Loopy Robotic Friends

Unplugged | Loop | Repeat

Turn your friends into robots and tell them what to do using loops!

Lesson 7: Loops in Artist

Loop | Artist

In this lesson, loops make it easy to make even cooler images with Artist!

Lesson 8: Nested Loops

Nested Loops | Loops | Bee | Maze

Loops inside loops inside loops. What does this mean? This lesson will teach you what happens when you place a loop inside another loop.

Lesson 9: Nested Loops with Frozen

Loop | Nested Loop | Frozen

Anna and Elsa have excellent ice-skating skills, but need your help to create patterns in the ice. Use nested loops to create something super COOL.

Lesson 10: Algorithms: Dice Race Unplugged

Unplugged | Dice Race | Algorithms

Learn how to describe a game from the computer's point of view in this lesson on algorithms.

Lesson 11: Introduction to Online Puzzles

Sequencing | Debugging | Loop | Ice Age | Maze | Artist

This lesson will give you practice in the skills you will need for this course.

Lesson 12: Conditionals in Farmer

Conditionals | Farmer

You will get to tell the computer what to do under certain conditions in this fun and challenging series.

Lesson 13: Digital Citizenship

Common Sense Education | Personal Information | Private Information | Identity Theft

The internet is fun and exciting, but it's important to stay safe too. This lesson teaches you the difference between information that is safe to share and information that is private.

Lesson 14: Build a Star Wars Game

Star Wars | Event

Feel the force as you build your own Star Wars game in this lesson.

Lesson 15: Functions: Songwriting Unplugged

Unplugged | Function

Even rockstars need programming skills. This lesson will teach you about functions using lyrics from songs.

Lesson 16: Functions in Artist

Function | Artist

Make complex drawings more easily with functions!

Lesson 17: Functions in Bee

Function | Bee

Don't write too much code to gather all of the nectar and honey. Use functions instead!

Lesson 18: Functions in Harvester

Function | Harvester

Functions will save you lots of work as you help the farmer with her harvest!

Lesson 19: Determine the Concept

Bee

We aren't giving away any secrets! This lesson could use any of the skills you've learned so far.

Lesson 20: Build a Play Lab Game

Event | Play Lab

Practice making games to share with your friends and family.

Lesson 21: Explore Project Ideas

Project | Define | Prepare | Try | Reflect

Here are several games and drawings. Play with each of them to get ideas for projects of your own!

Lesson 22: The Design Process

Project

Projects this big take time and plenty planning. Here, you will learn about the design process that you'll use to build your own creation.

Lesson 23: Build Your Project

Project

Get those hands ready for plenty of coding! It's time to start building your project.

Lesson 24: Present Your Project

Project

Get ready to show off! It's time to present your finished project to your peers.

Lesson 25: Beyond Programming: The Internet

Unplugged | Internet

Ever wondered how information travels across the internet? It's not magic! This lesson will teach you the basics of how the internet works.

Lesson 26: Beyond Programming: Crowdsourcing

Unplugged | Crowdsourcing

This lesson will teach you about crowdsourcing, the process of building a project with a team.



This curriculum is available under a
Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 1: Programming: My Robotic Friends

Algorithms | Debugging | Unplugged

Overview

Using a special set of offline commands, students will design algorithms to instruct a "robot" to stack cups in different patterns. Students will take turns participating as the robot, responding only to the algorithm defined by their peers. This segment teaches students the connection between symbols and actions, the difference between an algorithm and a program, and the valuable skill of debugging.

Purpose

This unplugged lesson brings the class together as a team with a simple task to complete: get a "robot" to stack cups in a specific design. Students will work to recognize real world actions as potential instructions in code. The art of following precise instructions will also be practiced, as students work to translate algorithms into code, using the symbols provided. If problems arise in the code, students should work together to recognize bugs and build solutions. This activity lays the groundwork for the programming that students will do throughout the course as they learn the importance of defining a clearly communicated algorithm.

Agenda

Warm Up (5 min)

Talking to Robots

Activity (45 min)

Introduction and Modeling

Differentiation Options:

Programming Your Robots

Wrap Up (10 min)

Journaling

Objectives

Students will be able to:

- Reframe a sequence of steps as an encoded program
- Identify and address bugs or errors in sequenced instructions

Preparation

Watch the **My Robotic Friends - Teacher Video**.

(Optional) Print out one **My Robotic Friends - Symbol Key** per group or 4 students. Alternatively, find a place to display this information where students can reference throughout the lesson.

Prepare a stack of 10 disposable cups per group of 4 students, OR

(Optional) print and cut out **Paper Trapezoid Template - Manipulatives** for each group if your class is not going to use cups.

Print out one set of **Stacking Cup Ideas - Manipulatives** per group.

Make sure each student has a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **My Robotic Friends** - Teacher Video

For the Students

- **Stacking Cup Ideas** - Manipulatives
- **Paper Trapezoid Template** - Manipulatives
- **My Robotic Friends** - Unplugged Video ([download](#))

- **My Robotic Friends** - Symbol Key

Vocabulary

- **Algorithm** - A list of steps to finish a task.
- **Bug** - Part of a program that does not work correctly.
- **Debugging** - Finding and fixing problems in an algorithm or program.
- **Loop** - The action of doing something over and over again.
- **Program** - An algorithm that has been coded into something that can be run by a machine.
- **Repeat** - Do something again

Teaching Guide

Warm Up (5 min)

Talking to Robots

🗣️ **Discuss:** Start by asking the class if anyone has heard of robotics. Has anyone seen a robot or touched one? Does a robot really “hear” you speak? Does it really “understand” what you say?

Say: Robots can only do what they've been instructed or programmed to do. In order to accomplish a task, a robot needs to have a list of instructions (sometimes called an algorithm) that it can read. Today, we are going to learn what it takes to make that happen.

Discussion

The goal of this quick discussion is to call out that while robots may seem to behave like people, they're actually responding only to their programming. Students will likely refer to robots from movies and TV that behave more like humans. Push them to consider robots that they've seen or heard of in real life, like Roombas, or even digital assistants like Amazon Alexa.

Activity (45 min)

Introduction and Modeling



Pick Up Cup



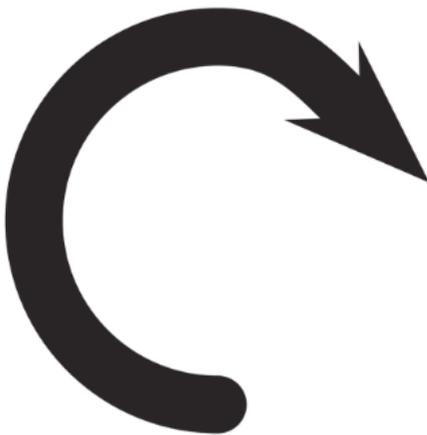
Put Down Cup



Step Forward



Step Backward



Turn Cup Right 90°



Turn Cup Left 90°

📍 **Display:** Display the **My Robotic Friends - Symbol Key** or write the allowed actions on the board - make sure these are in a place where they can be seen for the whole activity. Explain to the class that these will be the only six actions that they will be using for this exercise. For this task, they will instruct their “robot” friend to build a specific cup stack using only the commands listed on the key.

Model: In order to explain how the instructions are intended to work, model for the class how to create and follow an algorithm for replicating a simple pattern. Place a single stack of cups in front of you to start.

Display: Hold up the pattern you plan to model. A simple three cup pattern is a great place to start.



Initiation Options:

es this all feel a little complicated for your
ere are a couple of tips to simplify the

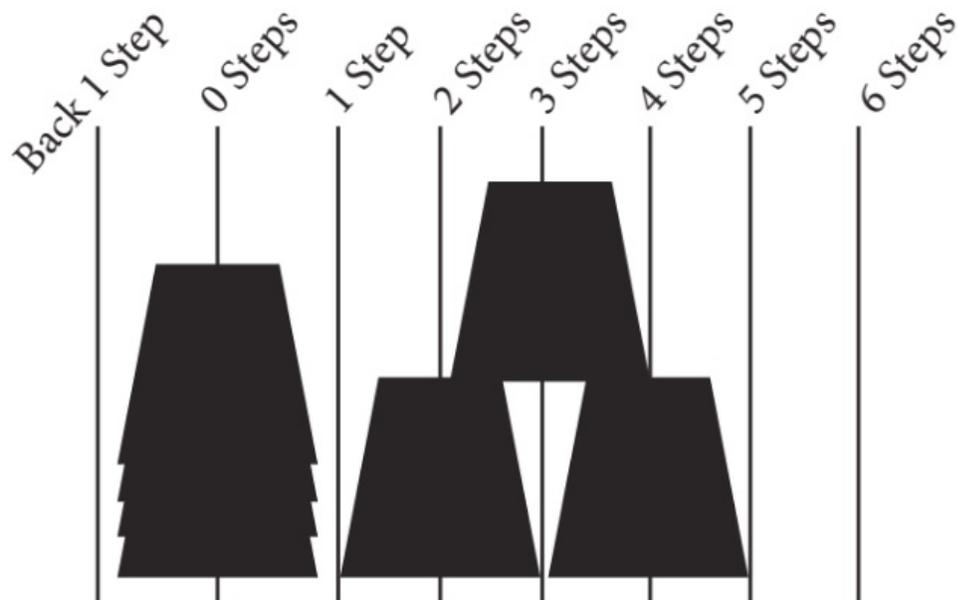
of steps into the ups and downs
ins that the cup automatically goes up as high
eds to
eans that it automatically goes down until it
n something
utomatically returns to cup stack after setting
cup

ie your students more advanced? Do you
ion to relate more closely to the online
e are some modifications that you can make:

Prompt: Ask the class what the first instruction should be, using **only the six instructions allowed**. The first move should be to "pick up cup." If students suggest something else from the list, perform that action and allow them to see their error. If they suggest something not from the list, make a clear malfunction reaction and let them know that the command is not understood.

With cup in hand, ask the class to continue giving you instructions until the first cup is placed. This is a great place to clarify that a "step forward" and "step backward" each imply moving half a cup width. See the image below for reference.

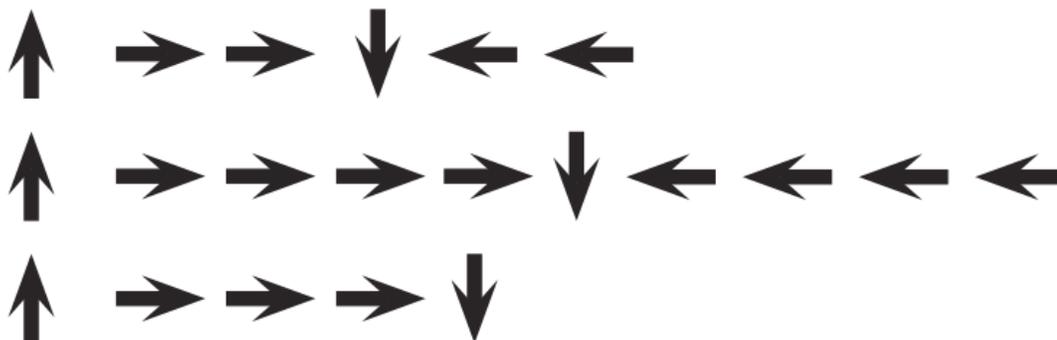
- One arrow corresponds to one movement
 - When a cup is removed from the stack, it returns to table-level before moving
 - Students need to use multiple "up" arrows to lift the cup multiple levels
 - Students need to use multiple "down" arrows to lower the cups multiple levels
 - Students need to use the "back" arrows to get back to the cup stack
 - **Set Up:** Have stacks of cups or cut paper trapezoids available for groups.



Continue asking for instructions from the classroom until you have completed the entire design.

Once your stack is complete, point out that they just gave you a list of instructions for completing a task. That's an algorithm. Algorithms are great for sharing ideas, but spelling them out word by word can take a long time. That's what the symbols are for! When you encode an algorithm into symbols that a robot (or computer) understands, that's called programming.

Ask the class to help you write the "program" for that first move, and then the rest of the moves necessary to complete the pattern. Depending on the confidence of your students, you might switch back and forth frequently between acting as the "robot" and writing down the code, or you might push them to write the whole program before you will implement it. One possible solution looks like this:



Volunteer: Once the class has completed the model program, ask one of the students to come up and act as the "robot" to ensure that the program really works. Encourage them to say the instructions out loud as they "run" the code.

Programming Your Robots

Group: Place students into groups of 4. Each group should then further break down into two pairs - each pair will develop their own program to be "run" by the other pair.

Distribute: Give each group one stack of cups or paper cutouts.

Display: Show **Stacking Cup Ideas - Manipulatives** to the class or hand out individual copies for groups to use. Have each pair (not group) choose which idea they would like their robots to do. Try to push for an easier idea for the first time, then have them choose a more complex design later on. Encourage pairs to keep their choice secret from the other half of their group.

Discuss: Give each pair time to discuss how the stack should be built, using only the provided symbols. Make sure each group writes down the "program" somewhere for the "robot" to read later.

🔗 **Do:** Once both of the group's pairs have decided on their algorithms, they can take turns being "robots" for each other by following the instructions each pair wrote. Encourage students to watch their "robot" closely to ensure that they are following instructions. If a student sees a bug and raises their hand, have the robot finish the instructions to the best of their ability. Afterward, have the students discuss the potential bug and come up with a solution. Continue repeating until the stack is built properly.

🔗 Teaching Tip

Enforcing the rules: While the robot is working on the stack make sure that the class knows:

- Programmers are not allowed to talk when the robot is working. This includes blurting out answers or pointing out when the robot has done something wrong.
- Programmers should raise their hand if they see a bug.

Circulate: Look for groups who are trying to take shortcuts by adding things (like numbers) to their code. Praise them for their ingenuity, but remind them that for this exercise, the robots do not understand **anything** but the provided symbols. If you like, you can hint that they should save their brilliant solution for the next time they play this game, since they might get the chance to use it soon!

Iterate: Depending on your time available, mix up the pairs and give them a chance to do a different pattern. Each time

groups repeat the process, encourage them to choose a more challenging pattern.

🗣️ **Discuss:** After everyone has had a chance to be the robot, bring the class back together to discuss their experience. In particular, discuss as a class:

- What was the most difficult part of coming up with the instructions?
- Did anyone find a bug in your instructions once your robot was following them?
 - What was the bug?
 - Why do you think you didn't notice it when writing the program?
- When you were the robot, what was the hardest part of following the instructions you were given?

Discussion

Sense making: The goal of this discussion is to give students space to make sense of their experience both as robot and programmer. The questions are intentionally broad, but designed to get students thinking about the challenges of writing a clear program and the constraints of a robot or computer in interpreting your instructions.

Wrap Up (10 min)

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How did you feel during today's lesson?
- Draw a stack of cups that the robot made today.
- Draw a stack of cups that you would like a robot to make someday!

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ **AP** - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 2: Sequences in Maze

Sequencing | Debugging | Maze

Overview

In this set of puzzles, students will begin with an introduction (or review depending on the experience of your class) of Code.org's online workspace. There will be videos pointing out the basic functionality of the workspace including the Run, Reset, and Step buttons. Also discussed in these videos: dragging Blockly blocks, deleting Blockly blocks, and connecting Blockly blocks. Next, students will practice their **sequencing** and **debugging** skills in maze.

Purpose

We recognize that every classroom has a spectrum of understanding for every subject. Some students in your class may be computer wizards, while others haven't had much experience at all. In order to create an equal playing (and learning) field, we have developed this "Ramp Up Stage" for Course E. This can be used as either an introduction or a review of how to use Code.org and basic computer science concepts.

Agenda

Warm Up (10 min)

Introduction
Vocabulary

Bridging Activities - Programming (10 min)

Unplugged Activity Using Paper Blocks
Preview of Online Puzzles as a Class

Main Activity (30 min)

Course E Online Puzzles - Website

Wrap Up (10 min)

Journaling

Objectives

Students will be able to:

- Order movement commands as sequential steps in a program.
- Modify an existing program to solve errors.
- Break down a long sequence of instructions into the largest repeatable sequence.

Preparation

Play through the **Course E Online Puzzles - Website** to find any potential problem areas for your class.

Make sure every student has a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Course E Online Puzzles - Website**
- **CS Fundamentals Main Activity Tips - Lesson Recommendations**

For the Students

- **Unplugged Blockly Blocks (Grades 2 - 5) - Manipulatives (download)**

Vocabulary

- **Bug** - Part of a program that does not work correctly.
- **Debugging** - Finding and fixing problems in an algorithm or program.
- **Loop** - The action of doing something over and over again.
- **Program** - An algorithm that has been coded into something that can be run by a machine.
- **Programming** - The art of creating a program.

Teaching Guide

Warm Up (10 min)

Introduction

Students will either be learning a lot of new concepts or reviewing a lot of basic concepts. Based on your class's experience, you can cover the following vocabulary or move on to a bridging activity. We recommend using the following words in sentences if the definitions aren't explicitly covered.

Vocabulary

This lesson has four new and important vocabulary words:

- **Program** - Say it with me: Pro - Gram An algorithm that has been coded into something that can be run by a machine.
- **Programming** - Say it with me: Pro - Gramm - ing The art of creating a program.
- **Bug** - Say it with me: Bug An error in a program that prevents the program from running as expected.
- **Debugging** - Say it with me: De - Bugg - ing Finding and fixing errors in programs.
- **Loop** - Say it with me: Loo-p The action of doing something over and over again

Bridging Activities - Programming (10 min)

This activity will help bring the unplugged concepts from "My Robotic Friends" into the online world that the students are moving into. Choose **one** of the following to do with your class:

Unplugged Activity Using Paper Blocks

Similar to "My Robotic Friends", have the students in your class pair up. Pass out multiple fill 1 and move ___ blocks from the **Unplugged Blockly Blocks (Grades 2 - 5) - Manipulatives** to each pair. Have each pair of students draw a design on a four by four graph from **Graph Paper Programming - Worksheet**. Next, have the students work together to write the program needed to draw this design using the paper Blockly blocks. The students will need to write up, down, right, or left on the move ___ block. Make sure the students know that the program goes from top to bottom and the blocks need to touch!

Preview of Online Puzzles as a Class

Pull up a puzzle from **Course E Online Puzzles - Website**. We recommend puzzle 6 for this activity. Break up the students into groups of three or four. Have them "program" Red, the angry bird, to get to the pig using arrows from "My Robotic Friends".

↑ - Pick Up Cup

↓ - Put Down Cup

→ - Move 1/2 Cup Width Forward

← - Move 1/2 Cup Width Backward

↻ - Turn Cup Right 90°

↺ - Turn Cup Left 90°

Once all the groups have an answer, discuss the path as a class.

Main Activity (30 min)

Course E Online Puzzles - Website

Teachers play a vital role in computer science education and supporting a collaborative and vibrant classroom environment. During online activities, the role of the teacher is primarily one of encouragement and support. Online lessons are meant to be student-centered, so teachers should avoid stepping in when students get stuck. Some ideas on how to do this are:

- Utilize **Pair Programming - Student Video** whenever possible during the activity.
- Encourage students with questions/challenges to start by asking their partner.
- Unanswered questions can be escalated to a nearby group, who might already know the solution.
- Remind students to use the debugging process before you approach.
- Have students describe the problem that they're seeing. What is it supposed to do? What does it do? What does that tell you?
- Remind frustrated students that frustration is a step on the path to learning, and that persistence will pay off.
- If a student is still stuck after all of this, ask leading questions to get the student to spot an error on their own.

💡 Teacher Tip:

Show the students the right way to help classmates:

- Don't sit in the classmate's chair
- Don't use the classmate's keyboard
- Don't touch the classmate's mouse
- Make sure the classmate can describe the solution to you out loud before you walk away

Wrap Up (10 min)

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How did you feel about today's lesson?
- List some of the bugs you found in your programs today.
- What was your favorite puzzle to complete? Draw your favorite character completing a puzzle.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ **AP** - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 3: Building a Foundation

Unplugged | Persistence | Frustration

Overview

New and unsolved problems are often pretty hard. If we want to have any chance of making something creative, useful, and clever, then we need to be willing to attack hard problems even if it means failing a few times before we succeed. In this lesson, students will be building a structure with common materials. The structure will be tested on its ability to hold a textbook for more than ten seconds. Most students will not get this right the first time, but it's important they push through and keep trying.

Purpose

This lesson teaches that failure is not the end of a journey, but a hint for how to succeed. The majority of students will feel frustrated at some point in this lesson, but it's important to emphasize that failure and frustration are common steps to creativity and success.

Agenda

Warm Up (20 min)

Vocabulary

Try, Try Again

Main Activity (20 min)

Building a Foundation

Wrap Up (10 min)

Flash Chat: What did we learn?

Journaling

Extended Learning

Objectives

Students will be able to:

- Outline steps to complete a structural engineering challenge.
- Predict and discuss potential issues in structure creation.
- Build a structure based on team plan.
- Revise both the plan and the structure until they satisfy the challenge.

Preparation

Watch the **Building a Foundation - Teacher Video**.

Watch the **Building a Foundation - Lesson in Action Video**.

Print **Building a Foundation - Teacher Prep Guide**.

Gather enough building elements (marshmallows or gumdrops with toothpicks or popsicle sticks) for each group. You don't have to give any certain amount; just make sure you put some limit on materials.

Give a **Think Spot Journal - Reflection Journal** to each student.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Building a Foundation** - Unplugged Video ([download](#))
- **Building a Foundation** - Teacher Video
- **Building a Foundation** - Lesson in Action Video
- **Building a Foundation** - Teacher Prep Guide

Vocabulary

- **Frustrated** - Feeling annoyed or angry

because something is not the way you want it.

- **Persistence** - Trying again and again, even when something is very hard.

Teaching Guide

Warm Up (20 min)

Vocabulary

This lesson has one new and important word:

Persistence - Say it with me: Per-sis-tence

Trying again and again, even when something is very hard

Try, Try Again

- Does everyone get everything right the first time?
- When I was a baby learning to walk, did I stand up and run off on my first try?
- Sometimes, the best and most useful things to do are the hardest to learn.
 - It can take a while to learn hard things
 - If you don't do something well at first, does it mean that you never will?
 - Can you think of something that was hard at first, but that you can now do pretty easily?
 - Walking
 - Talking
 - Riding a bike
- When you fail at doing something, you get a hint at what went wrong. You just need to look for it.
 - If your bike tips over, next time you need to work on balance.
 - If you're filling a balloon and it pops, next time you need less air.
- Think of the mistakes as chances to learn how to do something better next time.

💡 Lesson Tip

Here are some great resources to prep your class with the concept of persistence before you turn them loose on this project:

- **Mouse Wants a Cracker**
- **Fall 7 Times, Stand Up 8**
- **Never Ever Give Up**
- **If You Quit Too Soon**

Main Activity (20 min)

Building a Foundation

Have you ever started on a task, then discovered that it was much harder than you thought it would be? Hard tasks can make us want to give up, but if we stick to our goal and keep trying, then we just might make something better than we've ever made before!

In this challenge, we'll work to construct towers that are strong enough to hold a textbook for at least 10 seconds, using everyday materials.

Rules:

- Use only the supplies provided to build a tower.
- The tower can be any shape, but it has to be at least as tall as the paper cup.
- The tower must support the weight of a book for a full 10 seconds.

Directions:

1. Divide students into groups of three or four.
2. Explain the rules of the challenge, given above.
3. Provide each group with limited supplies and make it known that they will get no more.
4. Challenge the class to think ahead to the problem and plan out their method of building their first tower.

5. Encourage students to begin building, then have them alert you when they think they've met the challenge described by the rules.
6. Test each structure. Is it taller than the cup? Does it hold a book?
7. If not, have students enter a cycle of planning, fixing, testing, and planning again until the challenge has been met.
8. Congratulate the students as they succeed and take pictures of the successful towers!

Lesson Tip

The planning stage can be difficult for young students. It may be helpful for you to place some idea "examples" at the front of the room. Do not announce that they are there. Simply encourage students to take a walk if they get frustrated. Try to encourage students to locate the tips on their own if at all possible.

Wrap Up (10 min)

Flash Chat: What did we learn?

- Were you proud of what you made?
- Do you think you could make a tower as tall as a chair that could hold a person?
 - How many gumdrops do you think you would need?
- Was there a time that you thought about giving up?
 - How did you get past that feeling?

Lesson Tip

Flash Chat questions are intended to spark big-picture thinking about how the lesson relates to the greater world and the students' greater future. Use your knowledge of your classroom to decide if you want to discuss these as a class, in groups, or with an elbow partner.

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future. We provide a **Think Spot Journal - Reflection Journal** as a basic template for students to use as their daily journal.

Journal Prompts:

- What was today's lesson about?
- How did you feel during today's lesson?
- Draw a picture of your structure.
- What were some problems you ran into while building? How did you fix these problems?

Extended Learning

Use these activities to enhance student learning. They can be used as outside of class activities or other enrichment.

Try It Again!

Try doing the same activity with different materials.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ AP - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 4: Debugging in Scrat

Debugging | Ice Age

Overview

Debugging is an essential element of learning to program. In this lesson, students will encounter puzzles that have been solved incorrectly. They will need to step through the existing code to identify errors, including incorrect loops, missing blocks, extra blocks, and blocks that are out of order.

Purpose

Students in your class might become frustrated with this lesson because of the essence of debugging. **Debugging** is a concept that is very important to computer programming. Computer scientists have to get really good at facing the bugs in their own programs. Debugging forces the students to recognize problems and overcome them while building critical thinking and problem solving skills.

Agenda

Warm Up (15 min)

Introduction
Vocabulary

Main Activity (30 min)

Course E Online Puzzles - Website

Wrap Up (5 - 10 min)

Journaling

Extended Learning

Objectives

Students will be able to:

- Predict where a program will fail.
- Modify an existing program to solve errors.
- Reflect on the debugging process in an age-appropriate way.

Preparation

Play through the **Course E Online Puzzles - Website** to find any potential problem areas for your class.

(Optional) Pick a couple of puzzles to do as a group with your class.

Review **CS Fundamentals Main Activity Tips - Lesson Recommendations**.

Make sure every student has a **Think Spot Journal - Reflection Journal**.

Review **Debugging Recipe - Student Handout** with the class.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Course E Online Puzzles - Website**
- **CS Fundamentals Main Activity Tips - Lesson Recommendations**

For the Students

- **Think Spot Journal - Reflection Journal**
- **Debugging Recipe - Student Handout**

Vocabulary

- **Bug** - Part of a program that does not work correctly.
- **Debugging** - Finding and fixing problems in an algorithm or program.

Teaching Guide

Warm Up (15 min)

Introduction

Ask students to think about problems they have to solve in everyday life.

- How do you fix something that isn't working?
- Do you follow a specific series of steps?
- The puzzles in this unit have already been solved for you (yay!), but they don't seem to be working (boo!)
- We call the problems in these programs "bugs," and it will be your job to "debug" them.

Vocabulary

This lesson has three new and important vocabulary words:

- **Bug** - Say it with me - Buhh-g. Something that is going wrong. An error.
- **Debugging** - Say it with me: Dee-bug-ing. To find and fix errors.
- **Persistence** - Say it with me: Purr-siss-tense. Not giving up. Persistence works best when you try things many different ways, many different times.

Say:

Debugging is a process. First, you must recognize that there is an error in your program. You then work through the program step by step to find the error. Try the first step, did it work? Then the second, how about now? If you make sure that everything is working line by line, then when you get to the place that your code isn't doing what it's supposed to, you know that you've found a bug. Once you've discovered your bug, you can work to fix (or "debug") it!

If you think it will build excitement in the class you can introduce the character of today's puzzles, Scrat from Ice Age. If students aren't familiar with Scrat, **show some videos** of the quirky squirrel running into trouble.

Main Activity (30 min)

Course E Online Puzzles - Website

Before letting the students start on the computer, remind them of the advantages of **Pair Programming - Student Video** and asking their peers for help. Sit students in pairs and recommend they ask at least two peers for help before they come to a teacher.

As mentioned in the purpose of this lesson, make sure the students are aware that they will face frustrating puzzles. Tell them it is okay to feel frustrated, but it is important to work through the problem and ask for help. As the students work through the puzzles, walk around to make sure no student is feeling so stuck that they aren't willing to continue anymore.

Wrap Up (5 - 10 min)

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How did you feel during today's lesson?
- What kind of bugs did you find today?

- Draw a bug you encountered in one of the puzzles today. What did you do to "debug" the program?

Extended Learning

Use these activities to enhance student learning. They can be used as outside of class activities or other enrichment.

Planting bugs

Have students go back through previous levels, purposefully adding bugs to their solutions. They can then ask other students to debug their work. This can also be done with paper puzzles.

When other students are debugging, make sure that the criticisms are constructive. If this could be a problem for your class, go over respectful debugging before this activity by role playing with another student.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ AP - Algorithms & Programming



This curriculum is available under a
Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 5: Programming in Artist

Program | Programming | Artist

Overview

In this lesson, students will take control of the Artist to complete drawings on the screen. This Artist stage will allow students to create images of increasing complexity using new blocks like `move forward` by 100 pixels and `turn right` by 90 degrees .

Purpose

Building off of the students' previous experience with sequencing, this lesson will work to inspire more creativity with coding. The purpose of this lesson is to solidify knowledge on sequencing by introducing new blocks and goals. In this case, students learn more about pixels and angles using the new blocks, while still practicing their sequencing skills. Also, students will be able to visualize new goals such as coding the Artist to draw a square.

Agenda

Warm Up (10 min)

Introduction

Main Activity (30 min)

Course E Online Puzzles - Website

Wrap Up (10 - 15 min)

Journaling

Extended Learning

Objectives

Students will be able to:

- Create a program to complete an image using sequential steps.
- Break complex shapes into simple parts.

Preparation

Play through the **Course E Online Puzzles - Website** in stage 6 to find any potential problem areas for your class.

Review **CS Fundamentals Main Activity Tips - Lesson Recommendations**.

(Optional) Obtain protractors for your class to visualize the angles they must use to complete the puzzles.

Print one **Turns & Angles - Student Handout** for each student.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Course E Online Puzzles - Website**
- **CS Fundamentals Main Activity Tips - Lesson Recommendations**

For the Students

- **Think Spot Journal** - Reflection Journal
- **Artist Introduction** - Student Video
- **Turns & Angles** - Student Video
- **Turns & Angles** - Student Handout

Vocabulary

- **Program** - An algorithm that has been coded into something that can be run by a machine.
- **Programming** - The art of creating a program.

Teaching Guide

Warm Up (10 min)

Introduction

Show the students one or both of the following videos as an introduction to angles:

Artist Introduction - Student Video (1.5 minutes long)

Turns & Angles - Student Video (2 minutes long)

Use the **Turns & Angles - Student Handout** to show the students interior versus exterior angles for different shapes. This document can be used as a hand out or you can choose to print it out as a poster for students to refer to.

Ask:

Discuss the square and triangle shapes from the document.

- How would you code a computer to draw that shape?
- What order do the instructions need to be in?

Tell the students that in these puzzles they will be moving a character who leaves a line everywhere he goes. The students will be writing code that gets the character to draw various shapes, including a square.

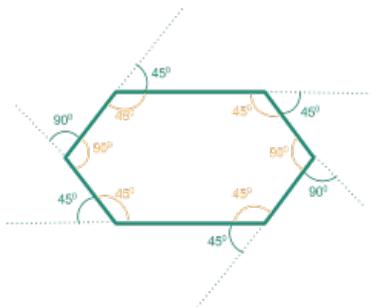
Main Activity (30 min)

Course E Online Puzzles - Website

In this set of puzzles, the artist will no longer be constrained to 90 degree angles. Having physical protractors available can help students better visualize the angles they need. Otherwise, the stage provides images of the angles as the student selects which angle to use. (Please note: Angle choices are limited to two inside of the dropdown menu, reducing the number of options students have to work through.)

Before sending the students to the computers to work on the puzzles, it might be beneficial to give a brief presentation of how to use the tools in this level. We recommend puzzle 5 as a good puzzle to show how to use the protractor online.

The eighth puzzle asks the students to draw a 6 sided polygon. This might be challenging for some students. We recommend getting the students to try a few times, ask a peer, then ask the teacher for help. Below is an image that might be helpful for the students.



Wrap Up (10 - 15 min)

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How did you feel during today's lesson?
- What are the interior angles that make up a square. What about for a triangle?
- Sketch a simple shape on your paper and imagine the code used to draw it. Can you write that code out next to the shape?

Extended Learning

Use these activities to enhance student learning. They can be used as outside of class activities or other enrichment.

The Copy Machine

- Give students two pieces of paper
- On one sheet draw a simple image, using straight lines only.
- On the second sheet draw instructions for recreating that image commands to move straight and turn at various angles.
- Trade instruction sheets and attempt to recreate the image using only the provided instructions.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ **AP** - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 6: My Loopy Robotic Friends

Unplugged | Loop | Repeat

Overview

Building on the initial "My Robotic Friends" activity, students learn to use loops when programming their robots in order to build bigger structures more efficiently.

Purpose

This lesson serves as a reintroduction to loops, using the now familiar set of "robot" programming instructions. Students will develop critical thinking skills by looking for patterns of repetition in the movements of classmates and determining how to simplify those repeated patterns using loops.

Agenda

Warm Up (10 min)

My Robotic Friends Review

Activity (30 min)

Introduction and Modeling Looping Your Robots

Wrap Up (5 min)

Flash Chat / Journaling

Extension Activities

Objectives

Students will be able to:

- Identify repeated patterns in code that could be replaced with a loop
- Write instructions that use loops to repeat patterns

Preparation

Make sure each student has a **Think Spot Journal - Reflection Journal**

Watch the **My Loopy Robotic Friends - Teacher Video**

Prepare a stack of 20 paper cups for each group of 4 students

(Optional) Print out one **My Loopy Robotic Friends Packet - PDF** per group of 4 students

OR

Display the **My Robotic Friends - Symbol Key** where students can reference throughout the lesson.

Print and cut out **Paper Trapezoid Template - Manipulatives** for each group if your class is not going to use cups.

Print out one set of **Stacking Cup Ideas - Manipulatives** per group.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **My Loopy Robotic Friends - Teacher Video**

For the Students

- **My Loopy Robotic Friends Packet - PDF**

Vocabulary

- **Loop** - The action of doing something over and over again.
- **Program** - An algorithm that has been coded into something that can be run by a machine.
- **Programming** - The art of creating a program.
- **Repeat** - Do something again

Teaching Guide

Warm Up (10 min)

My Robotic Friends Review



Pick Up Cup



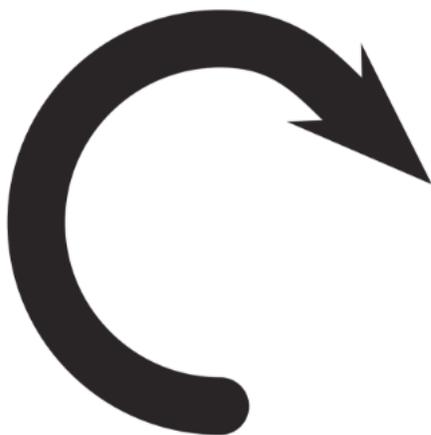
Put Down Cup



Step Forward



Step Backward



Turn Cup Right 90°



Turn Cup Left 90°

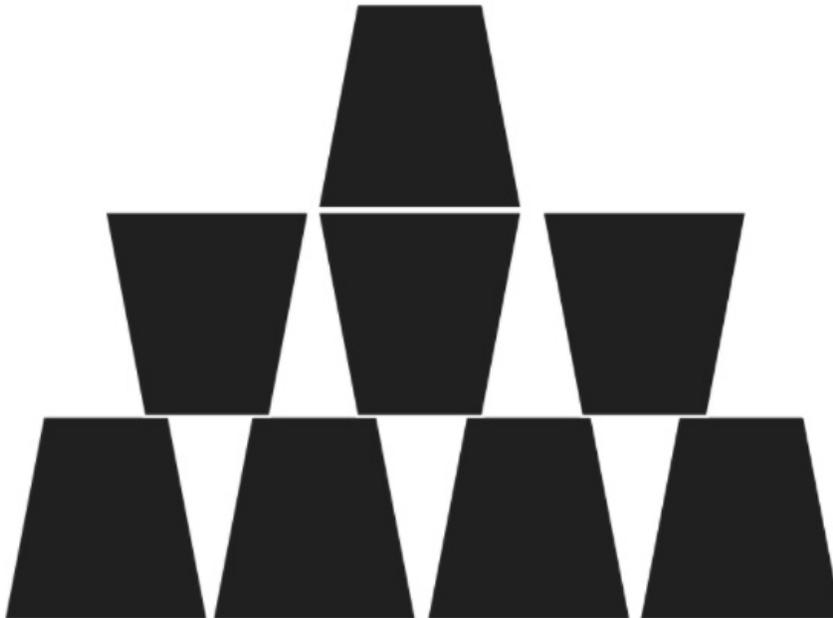
Goal: This review will refresh the students' minds about how quickly programs for the "My Robotic Friends" activity can get

intense.

Display: Show the **My Robotic Friends - Symbol Key** that we used in My Robotic Friends. For each of the six symbols, ask students to show you what it looks like for a robot to follow that instruction.

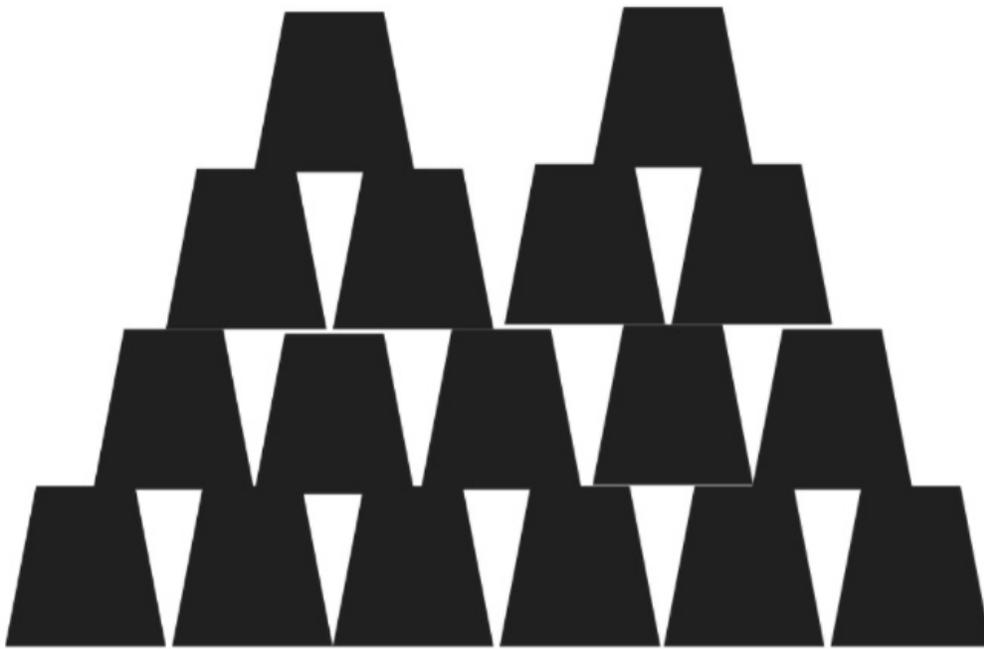
Model: With the class together as a group, pull an easy puzzle from the "My Robotic Friends" Cup Stack Pack and program together as a reminder of rules and terminology.

Next, pull a puzzle that's slightly harder, but also requires a lot of steps like the one below.



Volunteer: Ask a volunteer (or a group of volunteers) to come forward to help program this one on the board. If you make them stick strictly to the “no symbols other than those on the key” rule, it will probably take a while!

Display: Now, bring up this image:



What is the reaction of the class?

Prompt: Give students the opportunity to brainstorm shorter ways to relay the code that they're about to create. (This bit can be skipped over if your students start saying things like: "Move forward 6 times." Since that will open the discussion about how to show "six times" with symbols.)

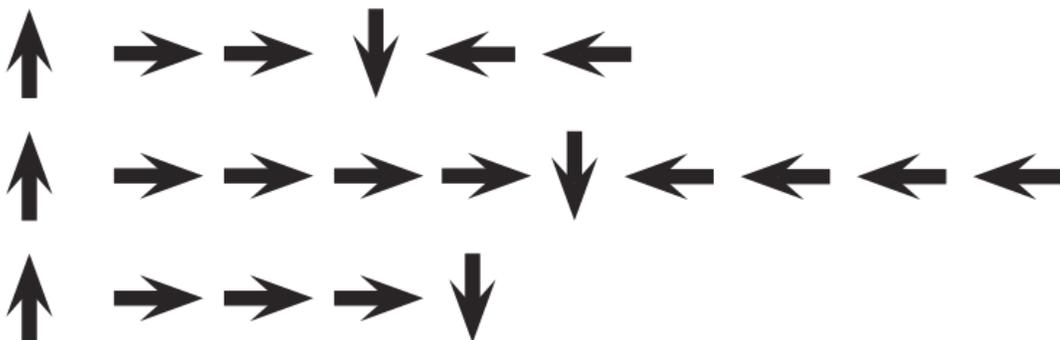
Once students have put together the idea of "repeating" code, give them the vocabulary around it. Make sure to share with them that the terms "repeat something" and "loop something" are often used interchangeably.

Activity (30 min)

Introduction and Modeling

Set Up: Have stacks of cups or cut paper trapezoids available for groups.

Display: Take the program from one of your previous cup stacks and display it for the class, or use the one below.

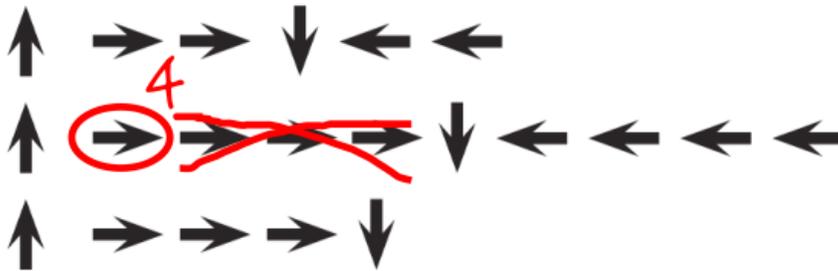


Think: Ask students to think quietly about where in this program they can find a pattern of instructions that repeat uninterrupted (one repetition after another).

Pair: Turn to a neighbor and share one of the repeating patterns you found.

Share: Ask a few students to share out the patterns they identified. Try to pull out different approaches to grouping patterns. For each pattern, ask students to identify how many times the pattern repeats.

Model: Using one of the repeating patterns that the class identified, model how Circle the instruction or pattern that repeats, write the number of loops near that circle, then cross out the rest of the arrows.



Repeat this until the entire program has been shortened, then re-write the program in a way where students can see how much more simple the resulting instructions are.

Looping Your Robots

Group: Place students into groups of 4. Each group should then further break down into two pairs - each pair will develop their own program for the other pair to "run".

Distribute: Give each group one stack of cups or paper cutouts.

Display: Show **Stacking Cup Ideas - Manipulatives** to the class or hand out individual copies for groups to use. Have each pair (not group) choose which idea they would like their robot to do. Encourage pairs to select a more complicated pattern this time around.

💡 **Discuss:** Let each group discuss how the stack should be built, then instruct each group to translate the algorithm into the symbols. Make sure each group writes down the program somewhere for the "robot" to read later. As students are working on their programs, remind them to be on the lookout for opportunities to replace a repeating pattern with a loop.

Do: When groups have finished their instructions, have each pair take turns "running" their code with another pair. Remind students to be on the lookout for bugs in their code, but not to interrupt a robot until it's finished running the program.

Discuss: When all of the pairs have had a chance to run their programs, ask a few to share their solutions with the class. Use this opportunity to discuss how groups came up with different solutions to the same puzzle. In particular, you might ask of each program:

- How did they identify the loops?
- Are there other ways those loops could have been written?

💡 Teaching Tip

Looking for Loops: Be sure to keep your eyes open for students using loops. Try to avoid correcting their overall algorithms or prescribing a solution, but feel free to direct students towards patterns that could be shortened using a repeat circle.

Watch students as they run through the code. Are there any bugs? Use the debugging questions to help them find a solution.

- What does it do?
- What is it supposed to do?
- What does that tell you?
- Does it work at the first step?
- Does it work at the second step?
- Where does it stop working?

- How much shorter is the program with loops than it would be without?
- Is the program easier to understand with loops or written out longhand? Why?

Wrap Up (5 min)

Flash Chat / Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and help to build a review sheet for them to look at in the future.

Flash Chat:

Here are some possible topics:

- Do you feel like loops make programming easier or harder?
- What other kinds of things in life do we repeat?
 - Eating - put food in mouth, chew 20 times
 - Brushing hair - brush through hair 35 times
 - Routines - Wake up, go to school, come home, go to bed
- Write or draw something in your journal that will remind you later what loops are.
 - What does "repeat" mean to you?
 - Draw a picture of you repeating something.

Extension Activities

- Have students draw their own cup stacking creations for someone else to code.
- Provide students with algorithms that utilize repeats, then have them expand the program back out to a full step-by-step version.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ AP - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 7: Loops in Artist

Loop | Artist

Overview

Watch student faces light up as they make their own gorgeous designs using a small number of blocks and digital stickers! This lesson builds on the understanding of loops from previous lessons and gives students a chance to be truly creative. This activity is fantastic for producing artifacts for portfolios or parent/teacher conferences.

Purpose

This series highlights the power of loops with creative and personal designs.

Offered as a project-backed sequence, this progression will allow students to build on top of their own work and create amazing artifacts.

Agenda

Warm Up (15 min)

Introduction

Main Activity (30 min)

Course E Online Puzzles - Website

Wrap Up (15 min)

Journaling

Objectives

Students will be able to:

- Identify the benefits of using a loop structure instead of manual repetition.
- Differentiate between commands that need to be repeated in loops and commands that should be used on their own.

Preparation

Play through the **Course E Online Puzzles - Website** corresponding to this course to find any potential problem areas for your class.

Review **CS Fundamentals Main Activity Tips - Lesson Recommendations**.

Make sure every student has a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Course E Online Puzzles - Website**
- **CS Fundamentals Main Activity Tips - Lesson Recommendations**

For the Students

- **Think Spot Journal - Reflection Journal**

Vocabulary

- **Loop** - The action of doing something over and over again.
- **Repeat** - Do something again

Teaching Guide

Warm Up (15 min)

Introduction

Students should have had plenty of introduction to loops at this point. Based on what you think your class could benefit from, we recommend:

- Creating a new stack design with loops just like in "My Loopy Robotic Friends"
- Previewing a puzzle from this lesson

All of these options will either review loops or the artist, which will help prepare your class for fun with the online puzzles!

Main Activity (30 min)

Course E Online Puzzles - Website

Some students may discover where to add repeat loops by writing out the program without loops then circling sections of repetitions. If the students in your class seem like they could benefit from this, have them keep paper and pencils beside them at their machines. Students might also enjoy drawing some of the shapes and figures on paper before they program it online. (When drawing stamps, it can be easier to symbolize those with simple shapes like circles and squares.)

Wrap Up (15 min)

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How did you feel during today's lesson?
- What was the coolest shape or figure you programmed today? Draw it out!
- What is another shape or figure you would like to program? Can you come up with the code to create it?

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ AP - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 8: Nested Loops

Nested Loops | Loops | Bee | Maze

Overview

In this online activity, students will have the opportunity to push their understanding of loops to a whole new level. Playing with the Bee and Plants vs Zombies, students will learn how to program a loop to be inside of another loop. They will also be encouraged to figure out how little changes in either loop will affect their program when they click Run .

Purpose

In this introduction to **nested loops**, students will go outside of their comfort zone to create more efficient solutions to puzzles.

In earlier puzzles, loops pushed students to recognize repetition. Here, students will learn to recognize patterns **within** repeated patterns to develop these **nested loops**. This stage starts off by encouraging students try to solve a puzzle where the code is irritating and complex to write out the long way. After a video introduces **nested loops**, students are shown an example and asked to predict what will happen when a loop is put inside of another loop. This progression leads into plenty of practice for students to solidify and build on their understanding of looping in programming.

Agenda

Warm Up (10 min)

Introduction

Main Activity (30 min)

Course E Online Puzzles - Website

Wrap Up (15 min)

Journaling

Objectives

Students will be able to:

- Break complex tasks into smaller repeatable sections.
- Recognize large repeated patterns as made from smaller repeated patterns.
- Identify the benefits of using a loop structure instead of manual repetition.

Preparation

Play through **Course E Online Puzzles - Website** to find any potential problem areas for your class.

Review **CS Fundamentals Main Activity Tips - Lesson Recommendations**.

Make sure every student has a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Course E Online Puzzles - Website**
- **CS Fundamentals Main Activity Tips - Lesson Recommendations**

For the Students

- **Think Spot Journal - Reflection Journal**

Vocabulary

- **Loop** - The action of doing something over and over again.
- **Repeat** - Do something again

Teaching Guide

Warm Up (10 min)

Introduction

Briefly review with the class what loops are and why we use them.

- What do loops do?
 - Loops repeat a set of commands. (see vocabulary on command if students don't recognize it)
- How do we use loops?
 - We use loops to create a pattern made of repeated actions.

Tell the class that they will now be doing something super cool: using loops inside loops. Ask the class to predict what kinds of things we would be using a loop inside of a loop for.

"If a loop repeats a pattern, then looping a loop would repeat a pattern of patterns!"

Students don't need to understand this right away, so feel free to move on to the online puzzles even if students still seem a little confused.

Main Activity (30 min)

Course E Online Puzzles - Website

We highly recommend **Pair Programming - Student Video** in this lesson. This may not be an easy topic for the majority of your students. Working with a partner and discussing potential solutions to the puzzles might ease the students' minds.

Also, have paper and pencils nearby for students to write out their plan before coding. Some puzzles have a limit on the number of certain blocks you can use, so if students like to write out the long answer to find the repeats, paper can be useful.

Wrap Up (15 min)

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How did you feel about today's lesson?
- What is a nested loop?
- Can you draw a puzzle that would use a nested loop? Try coding the solution to your own puzzle.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ AP - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 9: Nested Loops with Frozen

Loop | Nested Loop | Frozen

Overview

Now that students know how to layer their loops, they can create so many beautiful things. This lesson will take students through a series of exercises to help them create their own portfolio-ready images using Anna and Elsa's excellent ice-skating skills!

Purpose

In this series, students will get practice nesting loops while creating images that they will be excited to share.

Beginning with a handful of instructions, students will make their own decisions when it comes to creating designs for repetition. They will then spin those around a variety of ways to end up with a work of art that is truly unique.

Agenda

Warm Up (15 min)

Introduction

Main Activity (30 min)

Course E Online Puzzles - Website

Wrap Up (15 min)

Journaling

Objectives

Students will be able to:

- Describe when a loop, nested loop, or no loop is needed.
- Recognize the difference between using a loop and a nested loop.
- Break apart code into the largest repeatable sequences using both loops and nested loops.

Preparation

- Play through the **Course E Online Puzzles - Website** corresponding to this lesson to find and potential problem areas for your class.
- Review **CS Fundamentals Main Activity Tips - Lesson Recommendations**.
- Make sure every student has a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Course E Online Puzzles - Website**
- **CS Fundamentals Main Activity Tips - Lesson Recommendations**

For the Students

- **Think Spot Journal - Reflection Journal**

Vocabulary

- **Loop** - The action of doing something over and over again.
- **Repeat** - Do something again

Teaching Guide

Warm Up (15 min)

Introduction

Ask the class to discuss the last set of puzzles.

- What did they like/dislike?
- Which puzzles were hard? Why?
- Which puzzles were easy? Why?
- If you were to teach nested loops to a friend, what would you say to help them understand?

If there's time, give an introduction to the main characters of today's puzzles, Anna and Elsa from Frozen. Give the class the sister's back story if the class doesn't already know. To build excitement, tell the class they will be using nested loops to make some fantastic drawings with Anna and Elsa's ice skates!

Main Activity (30 min)

Course E Online Puzzles - Website

This set of puzzles is set up as a progression. This means every puzzle builds a foundation for the next puzzle. Students will enjoy making more and more interesting designs by making small and simple changes to code they have already written.

Wrap Up (15 min)

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How did you feel during today's lesson?
- When do you use a loop? When do you use a nested loop?
- Thought exercise: Can you make everything a nested loop can with just a normal loop? Can you draw out an example?
 - Answer: Yes, you can, but it is a lot more difficult. Nested loops make programs simpler.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ AP - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 10: Algorithms: Dice Race Unplugged

Unplugged | Dice Race | Algorithms

Overview

In this lesson, students will relate the concept of algorithms back to real-life activities by playing the Dice Race game. The goal here is to start building the skills to translate real-world situations to online scenarios and vice versa.

Purpose

By introducing a basic concept like **algorithms** to the class in an unplugged activity, students who are intimidated by computers can still build a foundation of understanding on these topics. Algorithms are essential to computer science. In this lesson, students will learn how to translate instructions into an algorithm and how that plays a role in programming.

Agenda

Warm Up (15 min)

Introduction
Vocabulary

Main Activity (30 min)

Real-Life Algorithms: Dice Race - Worksheet

Wrap Up (15 min)

Flash Chat: What did we learn?
Journaling

Extended Learning

Objectives

Students will be able to:

- Decompose large activities into a series of smaller events.
- Arrange sequential events into their logical order.

Preparation

- Watch the **Real-Life Algorithms: Dice Race - Teacher Video**.
- Print one **Real-Life Algorithms: Dice Race - Worksheet** per group.
- Print one **Real-Life Algorithms: Dice Race - Assessment** per student.
- Give every student a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Real-Life Algorithms: Dice Race - Unplugged Video (download)**
- **Real-Life Algorithms: Dice Race - Teacher Video**
- **Real-Life Algorithms: Dice Race - Worksheet**
- **Real-Life Algorithms: Dice Race - Worksheet Answer Key**
- **Real-Life Algorithms: Dice Race - Assessment**
- **Real-Life Algorithms: Dice Race - Assessment Answer Key**

For the Students

- **Think Spot Journal - Reflection Journal**

Vocabulary

- **Algorithm** - A precise sequence of instructions for processes that can be executed by a computer

Teaching Guide

Warm Up (15 min)

Introduction

- Ask your students what they did to get ready for school this morning.
 - Write their answers on the board.
 - If possible, put numbers next to their responses to indicate the order that they happen.
 - If students give responses out of order, have them help you put them in some kind of logical order.
 - Point out places where order matters and places where it doesn't.
- Introduce students to the idea that it is possible to create algorithms for the things that we do everyday.
 - Give them a couple of examples, such as making breakfast, brushing teeth, planting a flower, and making paper airplanes.
- Computers need algorithms and programs to show them how to do even simple things that we can do without thinking about them.
 - It can be challenging to describe something that comes naturally in enough detail for a computer to replicate.
- Let's try doing this with a new and fun activity, like playing the Dice Race Game!

Vocabulary

This lesson has one vocabulary word that is important to review:

- **Algorithm** - Say it with me: Al-go-ri-thm

A list of steps to finish a task.

Main Activity (30 min)

Real-Life Algorithms: Dice Race - Worksheet

- You can use algorithms to help describe things that people do every day. In this activity, we will create an algorithm to describe how we play the Dice Race Game.
- The hardest part about getting a problem ready for a computer can be figuring out how to describe real-life activities. We're going to get some practice by playing and describing the Dice Race game.

💡 Lesson Tip

You know your classroom best. As the teacher, decide if students should do this in pairs or small groups.

Directions:

- Read the rules below.
- Play a couple rounds of the Dice Race game.
 - As you're playing, think about how you would describe everything that you're doing.
 - What would it look like from the computer's point of view?

Rules:

- Set each player's score to 0
- Have the first player roll
- Add points from that roll to player one's total score
- Have the next player roll
- Add points from that roll to player two's total score
- Each player should go again two more times
- Check each player's total score to see who has the

💡 Lesson Tip

Help the students see the game from a computer's point of view. If they need to roll the dice, then the computer needs to provide dice. If the student needs to play three turns, then the computer needs to loop through the steps multiple times.

most points

- Declare Winner

| Game 1 | Turn 1 | Turn 2 | Turn 3 | Total |
|----------|--------|--------|--------|-------|
| Player 1 | | | | |
| Player 2 | | | | |

Circle the Winner!

Gather the class together and have each student complete the **Real-Life Algorithms: Dice Race - Assessment**. Once the students have completed the worksheet, have students share out their algorithms to the class. Open a discussion on the difference between an algorithm from a human's point of view and a computer's point of view.

Wrap Up (15 min)

Flash Chat: What did we learn?

- How many of you were able to follow your classmates' algorithms to play the Dice Race Game?
- What's the difference between an algorithm and a program?
 - An algorithm is the thinking behind what needs to happen, while the program is the actual instruction set that makes it happen.
 - An algorithm has to be translated into a program before a computer can run it.
- Did the exercise leave anything out?
 - What would you have added to make the algorithm even better?
 - What if the algorithm had been only one step: "Play Dice Race"?
 - Would it have been easier or harder?
 - What if it were forty steps?
- What was your favorite part about that activity?

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about
- How do you feel about today's lesson?
- What is an algorithm?
- What are some algorithms you use in your daily life?

Extended Learning

Use these activities to enhance student learning. They can be used as outside of class activities or other enrichment.

Go Figure

- Break the class up into teams.
- Have each team come up with several steps that they can think of to complete a task.
- Gather teams back together into one big group and have one team share their steps, without letting anyone know what the activity was that they had chosen.
- Allow the rest of the class to try to guess what activity the algorithm is for.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ **AP** - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 11: Introduction to Online Puzzles

Sequencing | Debugging | Loop | Ice Age | Maze | Artist

Overview

In this progression, students will begin with an introduction (or review depending on the experience of your class) of Code.org's online workspace. Students will learn the basic functionality of the interface including the Run, Reset, and Step buttons. Dragging, deleting, and connecting Blockly blocks is also introduced in the beginning video. In the puzzles, students will practice their sequencing and debugging skills in Maze and Artist.

Purpose

We recognize that every classroom has a spectrum of understanding for every subject. Some students in your class may be computer wizards, while others haven't had much experience at all. In order to create an equal playing (and learning) field, we have developed this "Ramp-Up Stage" for Course E. This can be used as either an introduction or a review to the Code.org interface and basic computer science concepts. This stage, along with the three that follow, cover all prerequisites needed to start Course E.

Agenda

Warm Up (15 min)

Introduction

Bridging Activity (15 min)

Preview of Online Puzzles

Main Activity (30 min)

Course E Online Puzzles - Website

Wrap Up (15 min)

Journaling

Objectives

Students will be able to:

- Order movement commands as sequential steps in a program.
- Modify an existing program to solve errors.
- Break down a long sequence of instructions into the largest repeatable sequence.

Preparation

Play through puzzles on **Course E Online Puzzles - Website** to find any potential problem areas for your class.

Make sure every student has a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Course E Online Puzzles - Website**
- **CS Fundamentals Main Activity Tips - Lesson Recommendations**

For the Students

- **Unplugged Blockly Blocks (Grades 2 - 5) - Manipulatives (download)**
- **Think Spot Journal - Reflection Journal**

Vocabulary

- **Bug** - Part of a program that does not work correctly.
- **Debugging** - Finding and fixing problems in an algorithm or program.
- **Loop** - The action of doing something over and over again.
- **Program** - An algorithm that has been coded into something that can be run by a machine.
- **Programming** - The art of creating a program.

Teaching Guide

Warm Up (15 min)

Introduction

Today will be largely spent on getting students introduced to Code.org or providing a review. Briefly survey your class on their experience with Code.org. If there is a wide range, pair experienced students with non-experienced students for their first lesson or two. If there isn't a wide range of experience, offer up a discussion on what Code.org is and why it is being taught.

Bridging Activity (15 min)

This activity will help bring the unplugged concepts from "Dice Race" into the online world that the students are moving into. We recommend doing the following activity with your class:

Preview of Online Puzzles

Pull up a puzzle from **Course E Online Puzzles - Website**. We recommend puzzle 9. Tell the class that Scrat from Ice Age needs to get to the acorn. To get Scrat to his goal, the class needs to come up with a program. Make sure to show the class how to use the hints and the step button while completing this puzzle.

Main Activity (30 min)

Course E Online Puzzles - Website

Teachers play a vital role in computer science education and supporting a collaborative and vibrant classroom environment. During online activities, the role of the teacher is primarily one of encouragement and support. Online lessons are meant to be student-centered, so teachers should avoid stepping in when students get stuck. Some ideas on how to do this are:

- Utilize pair programming whenever possible during the activity.
- Encourage students with questions/challenges to start by asking their partner.
- Unanswered questions can be escalated to a nearby group, who might already know the solution.
- Remind students to use the debugging process before you approach.
- Have students describe the problem that they're seeing. What is it supposed to do? What does it do? What does that tell you?
- Remind frustrated students that frustration is a step on the path to learning, and that persistence will pay off.
- If a student is still stuck after all of this, ask leading questions to get the student to spot an error on their own.

💡 Teacher Tip

Show the students the **right** way to help classmates:

- Don't sit in the classmate's chair
- Don't use the classmate's keyboard
- Don't touch the classmate's mouse
- Make sure the classmate can describe the solution to you out loud before you walk away

Wrap Up (15 min)

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How do you feel about today's lesson?
- What does it mean to "program"?
- Why is programming important?
- What's something about computers that you would like learn more about?

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ **AP** - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 12: Conditionals in Farmer

Conditionals | Farmer

Overview

This lesson introduces students to while loops and if / else statements. **While loops** are loops that continue to repeat commands as long as a condition is true. While loops are used when the programmer doesn't know the exact number of times the commands need to be repeated, but the programmer does know what condition needs to be true in order for the loop to continue looping. If / Else statements offer flexibility in programming by running entire sections of code only if something is true, otherwise it runs something else.

Purpose

A basic understanding of conditionals is a recommended prerequisite for Course E. We created this introduction to give a review for the students already familiar to conditionals and allow practice for the students that are just learning. If you find that the understanding of conditionals varies widely in your classroom, we recommend a strategic pairing of students when completing this online lesson.

Agenda

Warm Up (15 min)

Introduction

Main Activity (30 min)

Course E Online Puzzles - Website

Wrap Up (15 min)

Journaling

Extended Learning

Objectives

Students will be able to:

- Define circumstances when certain parts of a program should run and when they shouldn't.
- Determine whether a conditional is met based on criteria.

Preparation

Play through puzzles in **Course E Online Puzzles - Website** to find any potential problem areas for your class.

Review **CS Fundamentals Main Activity Tips - Lesson Recommendations**.

Make sure every student has a **Think Spot Journal - Reflection Journal**

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Course E Online Puzzles - Website**
- **CS Fundamentals Main Activity Tips - Lesson Recommendations**

For the Students

- **Think Spot Journal - Reflection Journal**

Vocabulary

- **Condition** - A statement that a program checks to see if it is true or false. If true, an action is taken. Otherwise, the action is ignored.
- **Conditionals** - Statements that only run under certain conditions.
- **While Loop** - A loop that continues to repeat while a condition is true.

Teaching Guide

Warm Up (15 min)

Introduction

Gather the class together and ask two volunteers to walk straight in some direction in the classroom. If they encounter a chair out of place, they must step over it. If they reach a wall, they must sit down.

Once all of the students are sitting down, ask how you would program a robot to respond to a wall or a chair. Remind students that you cannot simply say "Step over chair" unless you know there is a chair, and you will not always know there is a chair. It might be helpful to translate the task into instructions like:

- while there is a path ahead
 - walk forward
 - if there is a chair, step over it
- sit down

Tell students they will be using **conditionals** to solve this problem on Code.org. Give the definition of:

- **Condition:** A statement that a program checks to see if it is true or false. If true, an action is taken. Otherwise, the action is ignored.
- **Conditionals:** Statements that only run under certain conditions.

Open up a discussion of when you might use a conditional in your code.

Main Activity (30 min)

Course E Online Puzzles - Website

The patterns in these puzzles may not be obvious to every student. We recommend that you play through these levels beforehand to best understand any problem areas for your class. Also, watching and using the techniques from **Pair Programming - Student Video** may be helpful for your class.

Wrap Up (15 min)

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How do you feel about today's lesson?
- What is a conditional? Why would you program a conditional?
- Give an example of using a conditional during your day? (ex. **If** I am hungry, I eat some food; **While** I am walking across the street, I keep an eye out for cars)

Extended Learning

While We Play

Gather the class for some fun outside or in a gym with a ball! This could be done in a circle or as a team in a court

Rules:

- **While** the ball is in play, we must all be ready to hit it
- **If** the ball is hit to you, you must keep it in the air
- **If** you hit the ball once, you cannot hit it again (only one touch per person per turn, no double hitting)
- **If** the ball goes out of bound, every student must fall to the ground dramatically. The last kid to fall has to retrieve the ball.

At the end of the first round, ask the students if they can identify the conditionals in the game. Can they come up with others they might want in the game?

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ **AP** - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 13: Digital Citizenship

Common Sense Education | Personal Information | Private Information | Identity Theft

Overview

Developed by Common Sense Education, this lesson is about the difference between information that is safe to share online and information that is not.

As students visit sites that request information about their identities, they learn to adopt a critical inquiry process that empowers them to protect themselves and their families from identity theft. In this lesson, students learn to think critically about the user information that some websites request or require. They learn the difference between private information and personal information, as well as how to distinguish what is safe or unsafe to share online.

Purpose

Common Sense Education has created this lesson to teach kids the importance of security on the internet. By discussing the difference between personal and private information, students will be able to recognize what information should and shouldn't be shared. Students will also learn what signs you should look for to determine if a website is safe or not.

Agenda

Warm Up (5 min)

Introduction

Main Activity (35 min)

Log In

Private and Personal

What's Safe to Share Online?

Wrap Up (15 min)

Flash Chat: What did we learn about today?

Journaling

Assessment (10 min)

Private and Personal Information

Objectives

Students will be able to:

- Learn about the benefits and risks of sharing information online.
- Understand what type of information can put them at risk for identity theft and other scams.

Preparation

Copy the **Protect Yourself Student Handout** (7th page of the teacher prep guide), one for each student.

Copy the **All About Me Student Handout** (6th page of the teacher prep guide), one for each student.

Print out an assessment (8th page of the teacher prep guide) for each student. Teacher version is the page after the student assessment.

Preview websites like **Neopets**, **Nickelodeon**, and **BookAdventure** and prepare to show them to the class.

Review **CSF Digital Citizenship - Resource List** for more online safety content.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Common Sense Education - Private and Personal Information** - Teacher Prep Guide
- **Common Sense Education** - Website
- **CSF Digital Citizenship** - Resource List

For the Students

- **Think Spot Journal** - Reflection Journal

Vocabulary

- **Identity Theft** - When a thief steals someone's private information in order to pretend to be that person.
- **Personal Information** - Information that can't be used to identify you.
- **Private Information** - Information that can be used to identify you.
- **Register (Online)** - To enter your information in order to sign up and get access to a website.

Teaching Guide

Warm Up (5 min)

Introduction

Ask:

- What types of information do you think are okay to share publicly online such as on an online profile that others will see?
 - Interests and favorite activities
 - Opinions about a movie
 - First name
- What are some examples of websites where you must register in order to participate?
 - Social networking sites
 - Video-sharing sites
 - Youth discussion sites
 - Ask-an-expert sites
 - Game sites

Write the names of the websites on the board. Explain that it's important to know that sharing some kinds of user information can put you and your family's privacy at risk.

Main Activity (35 min)



Log In

Project for the class, or have students go online to **Neopets**, **Nickelodeon**, or **BookAdventure**. **Do not ask the students to sign up for these sites!**

Discuss with the students the kinds of information that each website requires or requests before the users can participate.

Ask:

- What information is required? Why do you think it is required?
 - First name, username, password, password hint, gender, the state you live in, parent's permission, etc. This information is required because it helps distinguish one person from another. Or perhaps the website is keeping a record of who uses it.
- What information is optional? Why do you think it is optional?
 - Parent's email, birthday, state, country, gender, etc. This information is likely optional because the website does not require it for payment or to distinguish people. Or perhaps the website wants to keep track of this kind of information.
- Why do you think websites ask for this kind of information?
 - They want to get people to pay in order to use the site, they want to send messages to people who are signing up, or they want to try to sell things to those people.

Point Out that you do not have to fill out fields on websites if they are not required. Required fields are usually marked by an asterisk (*) or are highlighted in red.

Teacher Tip

As an offline alternative, print out and copy the website pages that ask for registration and log-in information. Distribute these to the students.

Private and Personal

Explain to the students that some kinds of information are generally safe to share on the internet and some are not. However, the information that's considered safe should not be shared one-on-one with people the students don't already know offline.

Define:

- **Personal Information:** Information that can't be used to identify you.
- **Private Information:** Information that is about you and can be used to identify you.

Emphasize that personal information is usually safe to share online. Private information is usually unsafe to share online, meaning students should get permission from a parent or guardian before sharing this kind of information.

Teacher Tip

If you'd like a more clear distinction between "personal" and "private" information in these definitions, you can use other phrases like "friendly information" or "sharable information" to better define the line that the students should recognize. We chose to keep "personal" and "private" to stay true to Common Sense Education's lesson plan.

Share the following examples of information that is safe or unsafe to share:

| SAFE - Personal Information | UNSAFE - Private Information |
|--|--|
| <ul style="list-style-type: none">- Your favorite food- Your opinion (though it should be done respectfully)- First name (with permission) | <ul style="list-style-type: none">- Mother's maiden name- Social Security number- Your date of birth- Parents' credit card information- Phone number |

Ask:

- Why would someone want to steal someone else's identity on the internet?
 - To steal money
 - To do something bad or mean
 - To hide their real identity

Define:

- **Identity Theft:** When a thief steals someone's private information in order to pretend to be that person.

Explain that an identity thief uses private information to pretend to be the person whose identity he or she has stolen. Once the thief has taken someone's identity, he or she can use that person's name to get a driver's license or buy things, even if the person whose identity they stole isn't old enough to do these things! It's often not until much later that people realize their identity has been stolen. Identity thieves may also apply for credit cards in other people's names and run up big bills that they don't pay off. Let students know that identity thieves often target children and teens because they have a clean credit history and their parents are unlikely to be aware that someone is taking on their child's identity.

Emphasize the difference between private information (which can be used to steal your identity) and personal information (which cannot be used to steal your identity). Invite students to answer the following questions (write their answers on the board):

Ask:

- What kinds of private information could an identity thief use to find out and steal your identity?
 - First and last name, postal address, email address, phone numbers, passwords, credit card numbers, Social Security number, mother's maiden name.
- What kinds of personal information could you share about yourself without showing your identity?
 - Your age, gender, how many siblings you have, your favorite music, your favorite food, what pets you have, the name of your pet, your opinion about something.

Explain to students that on the internet, people you interact with could be your friends next door or strangers who live on

the other side of the world. Because it's hard to know the intentions of people who you've never met before, it is best to remain cautious when sharing your information. You wouldn't give strangers your private information in the real world, and you need to be just as careful when you're online.

Remind students how important it is each time they share information online to stop and think: "Am I giving out information that I should keep private?" Point out that it can sometimes be safe to give out some private information. For example, a website might ask for your birth date or email address. But students should always ask their parent or guardian before giving out private information.

Distribute the **Protect Yourself Student Handout** and have students complete the activity. Review the answers as a class.

What's Safe to Share Online?

Distribute the **All About Me Handout**. Have students write down all the personal information they would like to share on a public profile in an online community. Emphasize that even though personal information is safe to share online, it is okay to choose not to share it. Remind students that everything on the list should be safe to share; none of it should be private information that can put their identity at risk.

Encourage students to share their lists with the class.

Ask:

- Is there anything on the lists that could be used by an identity thief? Why?
 - Guide students to explain their answers and encourage them to use the vocabulary terms.

Wrap Up (15 min)

Flash Chat: What did we learn about today?

You can use these questions to assess your students' understanding of the lesson objectives. You may want to ask students to reflect in writing on one of the questions, using a journal or an online blog/wiki.

Ask:

- What is identity theft?
 - Using someone else's private information to pretend to be that person.
- How do personal information and private information differ?
 - Private information, such as a Social Security number, is unsafe to share. It should be kept private so that identity thieves cannot use it. Personal information, such as your favorite food, cannot be used by identity thieves and is safe to share. Even though personal information is usually safe to share online, you might choose not to share this information, and that's fine.
- What would be a good rule for kids about giving out private information?
 - They should not share it online without the permission of a teacher, parent, or guardian.

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What did you learn in today's lesson?
- How do you feel about today's lesson?
- Give an example of personal information and private information.
- What's a website that you use often? How do you know it is a safe website to use?

Assessment (10 min)

Private and Personal Information

Hand out the assessment to students. Allow students time to complete the assessment. If there is time left over, go over the answers with the students.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ **NI** - Networks & the Internet



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 14: Build a Star Wars Game

Star Wars | Event

Overview

In this lesson, students will practice using events to build a game that they can share online. Featuring R2-D2 and other Star Wars characters, students will be guided through events, then given space to create their own game.

Purpose

CS Fundamentals is not simply about teaching computer science, it is about making computer science fun and exciting. In this series, students will learn about events using popular characters from Star Wars. These puzzles blur the lines between "learning" and "fun". Also, students will learn to recognize regular programming practices in games so that when they play games at home, they can see common computer science principles being used.

Agenda

Warm Up (15 min)

Introduction

Main Activity (30 min)

Course E Online Puzzles - Website

Wrap Up (15 min)

Journaling

Objectives

Students will be able to:

- Create an animated, interactive game using sequence and events.
- Identify actions that correlate to input events.

Preparation

Play through puzzles in **Course E Online Puzzles - Website** and find any potential problem areas.

Review **CS Fundamentals Main Activity Tips - Lesson Recommendations**.

Make sure every student has a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Course E Online Puzzles - Website**
- **CS Fundamentals Main Activity Tips - Lesson Recommendations**

For the Students

- **Think Spot Journal - Reflection Journal**

Vocabulary

- **Event** - An action that causes something to happen.

Teaching Guide

Warm Up (15 min)

Introduction

In a class discussion, ask the students what their favorite video game is (you might need to remind the students to only use games that are classroom appropriate). Ask the students what their favorite part of the game is.

Most of the time, students will respond with some kind of event. When you recognize a student response that describes an event, ask the student to describe it further.

Once the student is done describing their fun, take a minute to relate it back to the definition of an event.

- **Event:** An action that causes something to happen.

Ask the students to try and relate some of their favorite parts of video games and how they can be described as events. Have them pair share and discuss the differences between their events and their partner's.

💡 Teacher Tip

If you're not quite sure if a student's response describes an event, try to break down the response. Is there an action and a response?

For example:

- Crossing the finish line and having on screen characters congratulate you
- Finding a big pot of treasure (or other item) and watching your inventory grow
- Buying new items from the game's store and having the item to use
- Pressing the buttons on a game controller and having your character do something cool

Main Activity (30 min)

Course E Online Puzzles - Website

Students will likely be very excited to make their own Star Wars game at the end of this set of puzzles. If there's time, ask them to plan out what they want the game to do. The planning and preparation will help the students better recognize the key concepts this lesson is trying to teach. Encourage the students to share and remix each other's games at the end of this lesson.

💡 Teacher Tip

Remind the students to only share their work with their close friends or family. For more information watch or show the class:

Pause and Think Online - Video .

Wrap Up (15 min)

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How do you feel about today's lesson?
- Give an example of an event you used in your program today?
- Why is it important not to share private information online? How do you know if information is private?

Standards Alignment



This curriculum is available under a
Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 15: Functions: Songwriting Unplugged

Unplugged | Function

Overview

One of the most magnificent structures in the computer science world is the function. Functions (sometimes called procedures) are mini programs that you can use over and over inside of your bigger program. This lesson will help students intuitively understand why combining chunks of code into functions can be such a helpful practice.

Purpose

The use of functions helps simplify code and develop the student's ability to organize their program. Students will quickly recognize that writing functions can make their long programs easier to read and easier to debug if something goes wrong.

Agenda

Warm Up (20 min)

Vocabulary
Sing a Song

Main Activity (20 min)

Functions Unplugged: Songwriting - Worksheet

Wrap Up (5 min)

Flash Chat: What did we learn?
Journaling

Assessment (5 min)

Functions Unplugged: Songwriting - Assessment

Extended Learning

Objectives

Students will be able to:

- Locate repeating phrases inside song lyrics.
- Identify sections of a song to pull into a function.
- Describe how functions can make programs easier to write.

Preparation

Watch the **Songwriting and Songwriting with Parameters (Functions) - Teacher Video**.

Watch the **Functions Unplugged: Songwriting - Lesson in Action Video**.

Print several **Functions Unplugged: Songwriting - Worksheet** for each group.

Print one **Functions Unplugged: Songwriting - Assessment** for each student.

Access to the internet, or pre-downloaded songs and lyrics for activity.

Make sure every student has a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Songwriting with Functions** - Unplugged Video ([download](#))
- **Songwriting and Songwriting with Parameters (Functions) - Teacher Video**
- **Functions Unplugged: Songwriting - Lesson in Action Video**
- **Functions Unplugged: Songwriting - Worksheet**
- **Functions Unplugged: Songwriting - Assessment**

- **Functions Unplugged: Songwriting** - Assessment Answer Key

For the Students

- **Think Spot Journal** - Reflection Journal

Vocabulary

- **Function** - A named group of programming instructions. Functions are reusable abstractions that reduce the complexity of writing and maintaining programs.

Teaching Guide

Warm Up (20 min)

Vocabulary

This lesson has one new and important word:

- **Function** - Say it with me: Func-shun

A piece of code that you can call over and over again.

Sing a Song

- Let the class know that today is song day!
- We're going to learn a song together.
 - Start with a simple song, either written out or projected on the screen.
 - Point to the chorus and be sure that the class knows how it goes before you begin on the song.
 - Blast through the song, singing it with them in the beginning, then see what happens when you get to the part where it calls the chorus.

💡 Chorus:

Little bunny Foo Foo
Hopping through the forest
Scooping up the field mice
And bopping 'em on the head
Down came the Fairy
And she said
"Little bunny Foo Foo
I don't wanna see you
Scooping up the field mice
And bopping 'em on the head"*

Song:

Chorus

**I'll give you 3 chances.
Then I'll turn you into a goon!
The next day. . .**

Chorus

**I'll give you 2 chances.
Then I'll turn you into a goon!
The next day. . .**

Chorus

**I'll give you 1 chance.
Then I'll turn you into a goon!
The next day. . .**

Chorus

**"I gave you two chances.
Now I'll turn you into a goon!"
(POOF!)**

💡 Teaching Tip

Little Bunny Foo Foo is being used here as an example only. If your students know this song, feel free to use it. Otherwise, choose an appropriate song that they might be more familiar with (either from music class or the radio.)

And the moral of the story is:

Hare today, goon tomorrow!

- It's quite likely that the majority of the class will sing the lyrics for the chorus when you point to that bit.
 - Stop the song once that happens, and explicitly highlight what just happened.
 - You defined the chorus.
 - You called the chorus.
 - They sang the chorus.
- Ask the class why they suppose you only wrote the chorus once at the top of the paper instead of writing it over and over in each place where it is supposed to be sung.
 - What are other benefits of only writing the chorus once when you sing it many times?

Now, imagine that this song is a computer program.

Defining a title (like "chorus") for a little piece of code that you use over and over again is called creating a function.

This is helpful to computer scientists for some of the same reasons that it is helpful to songwriters.

- It saves time not having to write all the code over and over in the program.
- If you make a mistake, you only have to change it one place.
- The program feels less complicated with the repeating pieces defined just once at the top.

We are going to play with songs a little more, to try to really understand how often this technique is used!

💡 Lesson Tip

To hit this point home, you can look up the lyrics for some popular songs on the Internet. Show the students that the standard for repeating lyrics is to define the chorus at the top and call it from within the body of the song.

Main Activity (20 min)

Functions Unplugged: Songwriting - Worksheet

A fantastic way to compare functions to something we see in our everyday lives is to look at songs. Songs often have certain groups of lyrics that repeat over and over. We call such a group a "chorus."

Directions:

- Divide into groups of 4, 5, or 6.
- Give each group several copies of the Songwriting Worksheet.
- Play a short song for the class that contains a clear chorus that does not change from verse to verse.
- Challenge the class to identify (and write down) the chorus.
- Compare results from each group.

Did everyone get the same thing? Sing your choruses together to find out! Play this game over and over until the class has little trouble identifying the choruses.

- It is often easier just to have the class listen to (or watch) the song, then vote on what the chorus is by singing it together, rather than writing the whole thing down. If you choose this method, consider having the class do a written chorus for the final song selection to be sure that the visual learners get proper reinforcement.

💡 Lesson Tip

It's most exciting for students to do this lesson with popular music from the radio, but if you're having a hard time finding appropriate songs where the lyrics repeat exactly, here are a few timeless options:

- **You Are My Sunshine**
- **Boom, Boom, Ain't it Great**
- **How Much Is That Doggie in the Window**
- **I Love Trash**

Wrap Up (5 min)

Flash Chat: What did we learn?

- Would you rather write lyrics over and over again or define a chorus?

- Do you think it's possible to make multiple choruses for the same song?
- Does it make sense to make a new chorus for every time it's needed in a song?

💡 Lesson Tip

Flash Chat questions are intended to spark big-picture thinking about how the lesson relates to the greater world and the students' greater future. Use your knowledge of your classroom to decide if you want to discuss these as a class, in groups, or with an elbow partner.

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How do you feel about today's lesson?
- What is a function and how do you use it?
- Can you think of another activity where you might want to call a special group of instructions several times?

Assessment (5 min)

Functions Unplugged: Songwriting - Assessment

Hand out the assessment worksheet and allow students to complete the activity independently after the instructions have been well explained. This should feel familiar, thanks to the previous activities.

Extended Learning

Use these activities to enhance student learning. They can be used as outside of class activities or other enrichment.

Functional Suncatchers Visit the **CS Fundamentals Unplugged Table** or click on the link for **Functional Suncatchers**. This activity does take a few supplies from the craft store, but it helps students to see the value of calling multiple functions.

Create Your Song

- Start by creating a chorus together, then repeat it between verses of a song that you develop around it.
- Make a change to the chorus, and ponder how much easier it is to change in just one place.
- Change the chorus again, making it much longer than it was originally.
- Add a second chorus and alternate between them in your verses.

Songwriting a Program

- What if we acted out songs instead of singing them? All of a sudden, our chorus would be a function of repeated actions, rather than words.
- Use the concepts of the arrows from the Graph Paper Programming lesson and create a program with lots of repeating instructions.
 - Circle those repeating actions so that the class can see where they are.
 - Define a function called "Chorus" above the program.
 - Cross out everywhere the repeating actions appear in the program and write "Chorus" instead.
- Repeat until the class can go through this process with little direction.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ **AP** - Algorithms & Programming



This curriculum is available under a
Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 16: Functions in Artist

Function | Artist

Overview

Students will be introduced to using functions on Code.org. Magnificent images will be created and modified with functions in Artist. For more complicated patterns, students will learn about nesting functions by calling one function from inside another.

Purpose

One of the most important components to this lesson is providing students with a space to create something they are proud of. These puzzles progress to more and more complex images, but each new puzzle only builds off the previous puzzle. At the end of this lesson, students will feel confident with themselves and proud of their hard work.

Agenda

Warm Up (15 min)

Introduction

Bridging Activity - Functions (15 min)

Unplugged Activity Using Some Blockly
Preview of Online Puzzles

Main Activity (30 min)

Course E Online Puzzles - Website

Wrap Up (15 min)

Journaling

Extended Learning

Objectives

Students will be able to:

- Categorize and generalize code into useful functions.
- Recognize when a function could help to simplify a program.

Preparation

Play through puzzles in **Course E Online Puzzles - Website** to find any potential problem areas for your class.

Review **CS Fundamentals Main Activity Tips - Lesson Recommendations**.

Make sure every student has a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Course E Online Puzzles - Website**
- **CS Fundamentals Main Activity Tips - Lesson Recommendations**

For the Students

- **Unplugged Blockly Blocks (Grades 2 - 5) - Manipulatives (download)**
- **Think Spot Journal - Reflection Journal**

Vocabulary

- **Function** - A named group of programming instructions. Functions are reusable abstractions that reduce the complexity of writing and maintaining programs.

Teaching Guide

Warm Up (15 min)

Introduction

Ask the class to think back to "Functions Unplugged: Songwriting" and recall what a function is. Open a discussion about when to use a function when writing a song.

Tell the class that there are two main components to using functions.

1. **The Declaration:** Function declarations are what create a function. In a function declaration, you fill in the function with code and you give the function a name. You must declare a function before you can use it.
2. **The Call:** Function calls are what makes the program run the code in the function. To call a function, you place the name of the function in your program. Make sure your function is properly defined before calling it in your program.

The class can use songwriting as an example to understand these two components. In the unplugged activity, the function containing the lyrics to the chorus was named "chorus". When we first made this function, we circled the lyrics that would go in the function. Once we named the function, we could read through the lyrics and replace the repeated chorus lyrics with a function call to "chorus".

Continue the conversation until students have a basic understanding of functions being declared and called. If students don't get to this point, make sure to do one of the bridging activities before moving into the Code.org puzzles.

Bridging Activity - Functions (15 min)

This activity will help bring the unplugged concepts from "Functions Unplugged: Songwriting" into the online world that the students are moving into. Choose **one** of the following to do with your class:

Unplugged Activity Using Some Blockly

Pick a song to play that the students enjoy and print out the lyrics. You can use the same song from "Functions Unplugged: Songwriting." Break your class into groups or pairs. Pass out the printed out lyrics (including the repeated chorus) and the basic function blocks from **Unplugged Blockly Blocks (Grades 2 - 5) - Manipulatives** to each group or pair of students. See lesson tip for details.

Ask the students to cross out any part of the song that can be made into a function (the chorus is a good example) and put it into the function blocks provided. Students should fill in the function declaration with a function name and the words of the repeated lyrics. Once the function declaration is done, ask the students to fill in the function calls and place them on top of the crossed out lyrics.

Once every group or pair is done, ask the class where they put their functions and why. Did everyone make the same function? How often is the function repeated?

Preview of Online Puzzles

Pull up a puzzle from **Course E Online Puzzles - Website**. We recommend the first puzzle for this activity. As a class, work through the puzzle without using functions. Once you have gotten the solution, display it on a white board or overhead. Ask the class to point to the repeated code. Ask the class how they would simplify the program.

Lesson Tip

Function blocks:



The block to the left is a function declaration, a block that students will name and use to fill in the function. The block to the right is a function call, a block that makes the function code run. Students will need multiple of the function call blocks.

On the white board or overhead, rewrite the program without the repeated code, but leaving one line space. In that/those line space(s), call a function. Off to the side, declare the function like the left example block in the lesson tip. Ask the class what they think the code will do now.

Open up a discussion with the class on why functions could be useful in programming. Invite students to discuss the difference between functions and loops.

Main Activity (30 min)

Course E Online Puzzles - Website

Students may benefit from writing code without functions then creating functions from the repeated code. If students don't enjoy doing this in the Code.org workspace, we recommend providing paper and pencils for students to write (or draw) out their ideas.

Wrap Up (15 min)

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How do you feel about today's lesson?
- What are some differences between functions and loops?
- Sketch out a drawing you made today. Can you write the code needed to create this?
- Draw a picture you would like to create with code. Try writing or drafting the code that would make that drawing.

Extended Learning

Draw by Functions

Break the class into groups of 2-3 students. Have each group write a function that draws some kind of shape and a program that uses that function. Depending on the creativity or focus the groups, students might need to be assigned a shape to create. Once every group is done, have the groups switch programs. On a separate piece of paper, each group should draw what the program creates. The groups should then return the programs and drawings to the original group.

Did every group get the drawing they expected? If not, what went wrong? Have the class go through the debugging process and try again.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ AP - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 17: Functions in Bee

Function | Bee

Overview

In the second round of practice with online functions, students will navigate complex paths, collect plenty of nectar, and make lots of honey.

Purpose

Students will discover the versatility of programming by practicing functions in different environments. Here, students will recognize patterns in the bee's maze. The bee will need to navigate the play area, collect nectar, and make honey. Students will learn to organize their programs and create functions for repeated code.

Agenda

Warm Up (10 min)

Introduction

Main Activity (30 min)

Course E Online Puzzles - Website

Wrap Up (15 min)

Journaling

Objectives

Students will be able to:

- Use functions to simplify complex programs.
- Use pre-determined functions to complete commonly repeated tasks.

Preparation

Play through **Course E Online Puzzles - Website** to find any potential problem areas for your class.

Review **CS Fundamentals Main Activity Tips - Lesson Recommendations**.

Make sure every student has a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Course E Online Puzzles - Website**
- **CS Fundamentals Main Activity Tips - Lesson Recommendations**

For the Students

- **Think Spot Journal - Reflection Journal**

Vocabulary

- **Function** - A named group of programming instructions. Functions are reusable abstractions that reduce the complexity of writing and maintaining programs.

Teaching Guide

Warm Up (10 min)

Introduction

Ask the students if they enjoyed creating images in Artist during the last lesson. Open up a class discussion on the puzzles they encountered (both hard and easy).

Tell the class that they will now be using functions to simplify code with a bee. The bee needs to traverse a maze, collect nectar, and make honey. Functions will make their program cleaner by compartmentalizing complex chunks of code.

This might be a good time to discuss the differences between using functions and using loops.

- Use loops when you need to do something several times in a row, exactly the same way
- Use a function when you need to do something at different times in the same program

Main Activity (30 min)

Course E Online Puzzles - Website

Similar to the last lesson, students may benefit from programming without functions then creating functions from repeated code. We recommend providing paper and pencils for students to write (or draw) out their ideas. Also, if students are having trouble recognizing patterns, have them work with a partner on the harder puzzles.

Wrap Up (15 min)

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

- What was today's lesson about?
- How do you feel about today's lesson?
- What did your functions do in the programs you wrote today? How did that help you?
- When should you use a function instead of a loop?

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ AP - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 18: Functions in Harvester

Function | Harvester

Overview

Students have practiced creating impressive designs in Artist and navigating mazes in Bee, but today they will use functions to harvest crops in Harvester. This lesson will push students to use functions in the new ways by combining them with `while` loops and `if / else` statements.

Purpose

This lesson is meant to further push students to use functions in more creative ways. By also using conditionals and loops, students will learn there are many ways to approach a problem, but some are more efficient than others. These puzzles are intended to increase problem solving and critical thinking skills.

Agenda

Warm Up (10 min)

Introduction

Main Activity (30 min)

Course E Online Puzzles - Website

Wrap Up (15 min)

Journaling

Objectives

Students will be able to:

- Recognize when a function could help to simplify a program.
- Use pre-determined functions to complete commonly repeated tasks.

Preparation

Play through **Course E Online Puzzles - Website** to find any potential problem areas for your class.

Review **CS Fundamentals Main Activity Tips - Lesson Recommendations**.

Make sure every student has a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Course E Online Puzzles - Website**
- **CS Fundamentals Main Activity Tips - Lesson Recommendations**

For the Students

- **Think Spot Journal - Reflection Journal**

Vocabulary

- **Function** - A named group of programming instructions. Functions are reusable abstractions that reduce the complexity of writing and maintaining programs.

Teaching Guide

Warm Up (10 min)

Introduction

At this point, your students should already be introduced to functions. Take this time to have them discuss the advantages and disadvantages of using functions in a program. Either have them pair share or discuss as a class. Try using examples of hard or easy puzzles in either Artist or Bee.

Ask the class:

- When would you use a function?
- Why does a function help to simplify your program?
- Do you think functions make programming easier or harder? Why?

Main Activity (30 min)

Course E Online Puzzles - Website

Some puzzles will have a function pre-declared for the students to fill in. It may be helpful for the students to write the entire program without a function first, then determine where a function would be useful in the program.

It's important to make sure that every student is completing each puzzle with a dark green dot. If some of your students are struggling to simplify code and use functions, set up teams of expert students within your class to go around and answer questions.

Don't forget to provide pencils and paper to help students sketch out possible solutions.

Wrap Up (15 min)

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How did you feel about today's lesson?
- What makes you realize a function could help your program?
- How do while loops and if / else statements help your program?

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ AP - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 19: Determine the Concept

Bee

Overview

This series brings together concepts from previous lessons and gives students a chance to think critically about how they would solve each problem, but without telling them which concept to apply. Students will review basic algorithms, debugging, repeat loops, conditionals, while loops, and functions.

Purpose

It's important for students to remember that computer science provides plenty of opportunities to be creative. Every topic can be combined with another to make something bigger and better.

In this lesson, students will use previously learned concepts together, allowing for a "big picture" view of programming projects. This lesson will also bridge any gaps in understanding of when to use certain programming tools over others.

Agenda

Warm Up (10 min)

Introduction

Main Activity (30 min)

Course E Online Puzzles - Website

Wrap Up (15 min)

Journaling

Objectives

Students will be able to:

- Recognize which programming concept to use to solve a given problem.
- Describe the different ways one could solve a given problem.

Preparation

Play through the **Course E Online Puzzles - Website** associated with this level to find any potential problem areas for your class.

Review **CS Fundamentals Main Activity Tips - Lesson Recommendations**.

Make sure every student has a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Course E Online Puzzles - Website**
- **CS Fundamentals Main Activity Tips - Lesson Recommendations**

For the Students

- **Think Spot Journal - Reflection Journal**

Teaching Guide

Warm Up (10 min)

Introduction

This lesson is meant to review previous concepts by mixing them together.

Take this time to ask students to discuss their favorite parts of the course so far.

- What did they like learning about? Why?
- What did they not like learning about? Why not?
 - If it was too difficult, would they be interested in doing more activities to better understand those concepts?
- What was something really cool they got to make?
- What is something they hope to make with code someday? What kinds of programming tools might they use to make that?

This discussion should help the students get into the "bigger picture" framework that this lesson is trying to teach. Ask the students how they would pair certain programming tools together like:

- Loops and conditionals
- Functions and events

Main Activity (30 min)

Course E Online Puzzles - Website

Because the concepts in this set of lessons are not explicitly given, students may face more frustration than usual. Make sure to walk around and do a "wellness check" on each student. If one student is struggling in some manner, pair them with another student in order to stimulate discussion between the two.

Wrap Up (15 min)

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How do you feel about today's lesson?
- What is your favorite thing you have learned in your coding class so far? Why? How have you used that in programs lately?

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ AP - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 20: Build a Play Lab Game

Event | Play Lab

Overview

This lesson features Play Lab, a platform where students can create their own games and have interactions between characters and user input. Students will work with events to create keyboard controls. This set of puzzles will also loosely guide students through game development, but with freedom to add their own ideas.

Purpose

Students will use events to make characters move around the screen, make noises, and change backgrounds based on user input. This lesson offers a great introduction to events in programming and even gives a chance to show creativity! At the end of the puzzle sequence, students will be presented with the opportunity to share their projects.

Agenda

Warm Up (15 min)

Introduction

Review of "Personal and Private information"

Main Activity (30 min)

Course E Online Puzzles - Website

Wrap Up (15 min)

Journaling

Objectives

Students will be able to:

- Identify actions that correlate to input events.
- Create an animated, interactive game using sequence and events.

Preparation

Play through the **Course E Online Puzzles - Website** associated with this lesson to find any potential problem areas for your class.

Review **CS Fundamentals Main Activity Tips - Lesson Recommendations**.

Make sure every student has a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Course E Online Puzzles - Website**
- **CS Fundamentals Main Activity Tips - Lesson Recommendations**

For the Students

- **Think Spot Journal - Reflection Journal**

Teaching Guide

Warm Up (15 min)

Introduction

Today students will revisit **events** in programming.

- **Event:** An action that causes something to happen.

Ask the students to raise their hands in the air.

What you did was declare an event. When you say "raise your hands in the air" the students responded by raising their hands. In coding, you would declare this by saying "when I say 'raise your hands' you will raise your hands".

You can also think of cities also declaring events. There are laws that say "when there is a green light, the cars can move through the intersection". Ask the students why they think this is an event.

Today, students will play in Play Lab, but the events they will be working on will be more like the video games they are used to playing. Events will take the form of actions, such as pressing the up arrow or two characters running into each other.

Review of "Personal and Private information"

Remind students of information that is safe to share online and information that is strictly private.

| SAFE - Personal Information | UNSAFE - Private Information |
|--|--|
| Your favorite food Your opinion (though it should be done respectfully) First name (with permission) | Mother's maiden name Social Security number Your date of birth Parents' credit card information Phone number |

Discuss other examples of the two categories above.

Main Activity (30 min)

Course E Online Puzzles - Website

This is one of the most free-form online activities of the course. At the final stage students have the freedom to create a game of their own. You may want to provide structured guidelines around what kind of game to make, particularly for students who are overwhelmed by too many options.

Teacher Tip

Remind the students to only share their work with their close friends or family. For more information watch or show the class **Pause and Think Online - Video**.

Wrap Up (15 min)

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How do you feel about today's lesson?
- What is your game about? What are some cool features it has? What kind of code was needed to make these cool features?
- What's something you think is really cool in games you play? Create a hypothesis around what code would be needed to create those features.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ **AP** - Algorithms & Programming



This curriculum is available under a
Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 21: Explore Project Ideas

Project | Define | Prepare | Try | Reflect

Overview

The next four lessons provide an opportunity for students to put their coding skills to use in a capstone project. This project will help individuals gain experience with coding and produce an exemplar to share with peers and loved ones. Intended to be a multi-lesson or multi-week experience, students will spend time exploring brainstorming, learning about the design process, building, and presenting their final work.

In the explore stage, students will play with pre-built examples of projects in both Artist and Play Lab for inspiration. Next, students will learn about the design process and how to implement it in their own projects. They will then be given the space to create their own project in Artist, Play Lab, or another interface that they have become familiar with (this is likely the longest stage of the project). Finally, students will be able to present their finished work to their peers.

Purpose

Exploring project ideas is meant to inspire students with realistic and entertaining ideas for their culminating projects.

Agenda

Day 1 - Explore Project Ideas (45 min)

Example Projects

Day 2 - The Design Process (45 min)

Define and Prepare

Day 3 - Build Your Project (45 min)

Try

Day 4 & 5 - Present Your Project (45 min each)

Presentations

Extension Activity

Reflect and Try Again (45 min)

Other

Objectives

Students will be able to:

- Learn to plan in advance for an ongoing assignment.
- Explain how system limitations can affect project design.
- Describe how compromise can help keep a project on track and inspire creativity.

Preparation

Play through the online **Course E Project - Examples** to get an idea of the strengths, weaknesses, and limitations of the tool.

Print one copy of the **Design Process - Teacher Prep Guide** for each student.

Modify the **CS Fundamentals Final Project - Rubric** to fit your class goals and print out a copy for each student.

Modify the **Final Project Design - Worksheet** to fit your class and print one packet for each student.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Final Project Design** - Worksheet

[Make a Copy](#)

- **CS Fundamentals Final Project** - Rubric
- **Design Process** - Teacher Prep Guide

[Make a Copy](#)

For the Students

- **Think Spot Journal** - Reflection Journal

Vocabulary

- **Define** - Figure out the details of the problems that you are trying to solve
- **Prepare** - Research, plan, and acquire

materials for the activity you are about to do

- **Reflect** - Carefully think back on something with the intention of improving the outcome in the future
- **Try** - Attempt to do something

Teaching Guide

Day 1 - Explore Project Ideas (45 min)

Example Projects

Goal: This part of the process is an exploration. Students will sit down with a stage full of example projects to remix and learn. Not only will this give students an idea of what is possible, it will also help them see the limitations of the tool.

Give students a day to play with and remix the projects found in **Course E Project - Examples**. Have them use their journals (or notebook paper) to keep track of thoughts and ideas as they go.

This activity should be done in the same pairs/groups that will be working on projects together over the next several lessons.

Make sure your class understands that they will be spending the next several weeks working with projects of their own, so they should pay close attention to how these programs were written, as well as the concepts that they use.

Day 2 - The Design Process (45 min)

Define and Prepare

Students will come up with a project and plan their strategy for programming that project in a single day. Students should have a project sketch and a description by the time the day is done.

Preparing Students for the Process:

The most important responsibility you have in kicking off this segment is to help your class understand the scope of this project. Students should be clear about the various expectations over the coming weeks so that they can prepare for their presentations appropriately.

To help your class manage this multi-stage undertaking, they should be given both the **Final Project Design - Worksheet** and the **CS Fundamentals Final Project - Rubric** on the first day of planning. Students will then be able to follow the rubric each step of the way to predict what their project grade will be in the end.

The **Final Project Design - Worksheet** will provide a place for students to capture relevant thoughts and processes as they go, so they are more prepared for their presentations in the end.

As the teacher, you should decide which elements of these documents are important to you and be sure to edit or remove anything that you do not intend to draw student focus.

Define and Prepare:

Now that the class has their **Final Project Design - Worksheet** in hand, they should start filling out the questions under **Day 1**.

Students will likely need to refer back to their notes from playing with the example projects, especially if they don't have access to online Artist or Play Lab project levels while they plan.

Students should focus on defining and planning their project during Day 1, and not cross over into building until their ideas have been written up and/or drawn out.

If students get stuck, help them work through ideas by asking questions and recalling examples, rather than offering solutions.

💡 Lesson Tip

Save 5 minutes or so at the end of the day to have students trade their **Final Project Design - Worksheet** to look at each other's work. This will help make sure that nothing is omitted or overlooked.

Day 3 - Build Your Project (45 min)

Try

Students will use this day to build an initial version of their project.

Equipped with their **Final Project Design - Worksheet**, students should head to the computers to start bringing their projects to life.

This process will come complete with plenty of trial and error. Projects are likely to become truncated versions of the original scope (if not morphed altogether). Remind students that this kind of compromise is common in software design, but they need to be sure to document the reasons for the changes in their product.

Don't let the class forget to fill out their **Final Project Design - Worksheet** as they go. It might be helpful to suggest that pairs/groups take a worksheet break to begin discussing these questions about halfway through their lab time. Alternatively, the navigator can keep their eyes open for pertinent answers while the driver codes.

Be sure that each team member has their own Final Project Design Worksheet, as there are questions about each student's own individual thoughts and behaviors that need to get captured along the way.

Day 4 & 5 - Present Your Project (45 min each)

Presentations

Students will create and present their projects in an approved manner (written, oral, or using multimedia).

Create:

Ideally, you will have class time available to give students to work on their presentations. This will allow them to incorporate rich multimedia components, like **Google Slides**. For other presentation ideas, visit **72 Creative Ways for Your Students to Show What They Know - Website**.

Encourage students to include all of the information from Section J of the Final Project Design Worksheet into their presentation, as well as two or more questions from Section K.

Present:

Students should showcase their apps first, then they can discuss the questions that they covered in their presentations.

It can be very helpful to have students sign up for a specific order in which to give their presentations, so that they are able to enjoy the demonstrations of their classmates without worrying about whether they will be called on next.

Lesson Tip:

If you are looking for a section of this series to assign as homework, this is it! Projects do not have to be presented in electronic form, so this is a great offline option. Other ways to present projects (both online and offline) include:

- Report
- Blog post
- Online
- In front of the class with a poster

Extension Activity

Reflect and Try Again (45 min)

Students will work with another group to give and receive feedback in an effort to make each other's projects stronger.

Reflect:

For reflections, have each group pair up with another group to try each other's projects. After about 10 minutes, have the groups discuss the questions in the Final Project Design Worksheet.

Encourage students to ask the questions on the Final Project Design Worksheet and write down feedback provided by their reviewing teams so that they can refer back to it later. This portion should take approximately 15 more minutes.

Try Again:

With their new reflections in hand, students can head back to their machines to make a handful of edits. With just 10 minutes left, they will likely have to select only the most important feedback to incorporate.

Other

If your students are already comfortable with coding concepts, try having them create their projects in another platform, like **Scratch** or **Alice**.

💡 Lesson Tip:

Teachers should avoid assigning the final bit of project work as homework unless they are certain that students both live within a close proximity to one another **and** have internet access at home.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ **AP** - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 22: The Design Process

Project

Overview

Over the course of four lessons, students will be building up to programming a project of their own design using either Play Lab or Artist as their programming environment. In this portion of the project, students will learn about the design process and how to implement it in their own projects. The lesson guide for all four stages of the process can be found in the first stage of this project process [here](#).

Purpose

Students may be ready to jump straight into building their projects, but this lesson will help shape their ideas into plans. This structure will keep the dreamers grounded and illuminate a path for those feeling left in the dark.

Agenda

Day 2 - The Design Process (45 min)

Define and Prepare

Objectives

Students will be able to:

- Shape ideas into reasonable goals and plans.
- Recognize any potential obstacles such as time constraints or bugs.

Teaching Guide

Day 2 - The Design Process (45 min)

Define and Prepare

Students will come up with a project and plan their strategy for programming that project in a single day. Students should have a project sketch and a description by the time the day is done.

Preparing Students for the Process:

The most important responsibility you have in kicking off this segment is to help your class understand the scope of this project. Students should be clear about the various expectations over the coming weeks so that they can prepare for their presentations appropriately.

To help your class manage this multi-stage undertaking, they should be given both the **Final Project Design - Worksheet** and the **CS Fundamentals Final Project - Rubric** on the first day of planning. Students will then be able to follow the rubric each step of the way to predict what their project grade will be in the end.

The **Final Project Design - Worksheet** will provide a place for students to capture relevant thoughts and processes as they go, so they are more prepared for their presentations in the end.

As the teacher, you should decide which elements of these documents are important to you and be sure to edit or remove anything that you do not intend to draw student focus.

Define and Prepare:

Now that the class has their **Final Project Design - Worksheet** in hand, they should start filling out the questions under **Day 1**.

Students will likely need to refer back to their notes from playing with the example projects, especially if they don't have access to online Artist or Play Lab project levels while they plan.

Students should focus on defining and planning their project during Day 1, and not cross over into building until their ideas have been written up and/or drawn out.

If students get stuck, help them work through ideas by asking questions and recalling examples, rather than offering solutions.

Lesson Tip

Save 5 minutes or so at the end of the day to have students trade their **Final Project Design - Worksheet** to look at each other's work. This will help make sure that nothing is omitted or overlooked.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ AP - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 23: Build Your Project

Project

Overview

Over the course of four lessons, students will be building up to programming a project of their own design using either Play Lab or Artist as their programming environment. Now the students will be given their own space to create their project with either Artist or Play Lab. This is likely to be the longest stage of the project. The lesson guide for all four stages of the process can be found in the first stage of this project process [here](#).

Purpose

This lesson provides students with ample time to build and revise their projects. The trial and error inevitably involved in this lesson will teach problem solving and persistence.

Agenda

Day 3 - Build Your Project (45 min)

Try

Objectives

Students will be able to:

- Use the planned design as a blueprint for creation.
- Overcome obstacles such as time constraints or bugs.

Teaching Guide

Day 3 - Build Your Project (45 min)

Try

Students will use this day to build an initial version of their project.

Equipped with their **Final Project Design - Worksheet**, students should head to the computers to start bringing their projects to life.

This process will come complete with plenty of trial and error. Projects are likely to become truncated versions of the original scope (if not morphed altogether). Remind students that this kind of compromise is common in software design, but they need to be sure to document the reasons for the changes in their product.

Don't let the class forget to fill out their **Final Project Design - Worksheet** as they go. It might be helpful to suggest that pairs/groups take a worksheet break to begin discussing these questions about halfway through their lab time. Alternatively, the navigator can keep their eyes open for pertinent answers while the driver codes.

Be sure that each team member has their own Final Project Design Worksheet, as there are questions about each student's own individual thoughts and behaviors that need to get captured along the way.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ AP - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 24: Present Your Project

Project

Overview

Over the course of four lessons, students will be building up to programming a project of their own design using either Play Lab or Artist as their programming environment. Finally, students will be able to present their finished work to their peers or share with their loved ones with a special link. The lesson guide for all four stages of the process can be found in the first stage of this project process [here](#).

Purpose

At this point, students have worked very hard on their projects, so this lesson is meant to offer a space for the students to share their projects. This lesson will build a supportive community where students will build their own confidence and feel connected to their hardworking peers.

Agenda

Day 4 & 5 - Present Your Project (45 min each)

Presentations

Objectives

Students will be able to:

- Clearly indicate where each criteria point from the rubric is satisfied in the code for the finished culminating project.
- Articulate the design process and how it helped shape the finished culminating project.

Teaching Guide

Day 4 & 5 - Present Your Project (45 min each)

Presentations

Students will create and present their projects in an approved manner (written, oral, or using multimedia).

Create:

Ideally, you will have class time available to give students to work on their presentations. This will allow them to incorporate rich multimedia components, like **Google Slides**. For other presentation ideas, visit **72 Creative Ways for Your Students to Show What They Know - Website**.

Encourage students to include all of the information from Section J of the Final Project Design Worksheet into their presentation, as well as two or more questions from Section K.

Present:

Students should showcase their apps first, then they can discuss the questions that they covered in their presentations.

It can be very helpful to have students sign up for a specific order in which to give their presentations, so that they are able to enjoy the demonstrations of their classmates without worrying about whether they will be called on next.

💡 Lesson Tip:

If you are looking for a section of this series to assign as homework, this is it! Projects do not have to be presented in electronic form, so this is a great offline option. Other ways to present projects (both online and offline) include:

- Report
- Blog post
- Online
- In front of the class with a poster

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ AP - Algorithms & Programming



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 25: Beyond Programming: The Internet

Unplugged | Internet

Overview

Even though many people use the internet daily, not very many know how it works. In this lesson, students will pretend to flow through the internet, all the while learning about connections, URLs, IP Addresses, and the DNS.

Purpose

If you have been doing every lesson in this course, then each student in your classroom has used the internet...but how many know how it works? Learning more about the internet will help students develop a better understanding of its endless possibilities.

Agenda

Warm Up (20 min)

Vocabulary
Getting the Message

Main Activity (20 min)

The Internet

Wrap Up (15 min)

Flash Chat: What did we learn?
Journaling

Assessment (5 min)

Internet - Assessment

Objectives

Students will be able to:

- Learn about the complexity of sending messages over the internet.
- Translate URLs into IP Addresses.

Preparation

Watch the **Internet - Teacher Video**.

Print enough **IP Address Cards and Delivery Type Cards - Manipulatives** for each group.

Print one **Internet - Assessment** for each student.

Access to the internet (such as **get-site-ip.com**).

Make sure every student has a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **The Internet** - Unplugged Video ([download](#))
- **Internet** - Teacher Video
- **IP Address Cards and Delivery Type Cards** - Manipulatives
- **Internet** - Assessment
- **Internet** - Assessment Answer Key

For the Students

- **Think Spot Journal** - Reflection Journal

Vocabulary

- **DNS** - The service that translates URLs to IP addresses.
- **DSL/Cable** - A method of sending

information using telephone or television cables.

- **Fiber Optic Cable** - A connection that uses light to transmit information
- **Internet** - A group of computers and servers that are connected to each other.
- **IP Address** - A number assigned to any item that is connected to the Internet.
- **Packets** - Small chunks of information that have been carefully formed from larger chunks of information.
- **Servers** - Computers that exist only to provide things to others.
- **URL** - An easy-to-remember address for calling a web page (like www.code.org).
- **Wi-Fi** - A wireless method of sending information using radio waves.

Teaching Guide

Warm Up (20 min)

Vocabulary

This lesson has several new and important words:

- **IP Address** - Say it with me: I-P Add-ress

A number assigned to any item that is connected to the Internet

- **DNS (Domain Name Service)** - Say it with me: D-N-S

The service that translates URLs to IP addresses

- **URL (Universal Resource Locator)** - Say it with me: U-R-L

An easy-to-remember address for calling a web page (like www.code.org)

- **Internet** - Say it with me: In-ter-net

A group of computers and servers that are connected to each other

- **Servers** - Say it with me: Ser-vers

Computers that exist only to provide things to others

- **Fiber Optic Cable** - Say it with me: Fye-ber Op-tic Cay-bl

A connection that uses light to transmit information

- **Wi-Fi** - Say it with me: Wye-Fye

A wireless method of sending information using radio waves

- **DSL/Cable** - Say it with me: D-S-L / Cay-bl

A method of sending information using telephone or television cables

- **Packets** - Say it with me: Pack-ets

Small chunks of information that have been carefully formed from larger chunks of information

💡 Lesson Tip

A quick preview is all you need here. These words will all be explained as part of the lesson, so it would be far less confusing to do a brief intro to the words as a "see if you can spot these during the day" type of heads-up.

Getting the Message

- It's quite likely that your students are aware of what the internet is, but they may not really understand what the internet does.

- Ask "What is the internet?"
- Is the internet a public place or a private place?
- (Truthfully, many people think it can be both, but it should be viewed as a public space no matter what settings you think you've mastered.)
- How does information get from place to place?

- Let's say that I want to look at the webpage for

Code.org. What do you suppose the process would be like for me to send a message to request that page?

- What do I do as a user?
- What do you think happens inside the internet?

Sending a message over the internet is a lot like sending a message through the mail...if every letter we sent required thousands of envelopes!

💡 Lesson Tip

There are some great YouTube videos on this subject that can make this lesson a little easier to understand. You can show them to the class in advance, or just watch them yourself. **Here is one of the videos in the Code.org video series on "How the Internet Works"**. (We recommend watching from 1:44 - 5:13, if possible.) The rest of the playlist is available **here**.

Every message we send through the internet gets chopped up and each piece is wrapped in its own version of an envelope. We call those "packets." Packets are specially formed chunks of information that are able to easily flow through any of the internet's channels.

Sometimes, a few of those packets will get lost, because the internet is a crazy place. In that case, the packets need to be resent, and the whole message has to get put on hold until they arrive.

Where do you think those packets are headed?

- Even if you're sending messages to another person, they first have to go to at least one "server."
 - A server is a special computer that is supposed to be always on and ready to send and receive information.
 - Every website has a server.
 - Even email goes through servers.

Servers don't have names like you and I do. They're actually addressed using numbers. These numbers are called IP addresses, and they look a little strange.

- For example: One of Code.org's IP addresses used to be 54.243.71.82
 - (Please be sure to check this out in advance. Most IP addresses change from time to time and they are then reused for other sites.)

There are many ways to reach the internet from your house, school, or place of business.

- You can connect directly using a cable (that might be DSL, Cable, or Fiber Optic)
- Or you can connect using radio waves over the air through Wi-Fi

Direct connections are most reliable, but they can be inconvenient.

- Can you figure out why?
 - (You have to be attached to a cable!)

Wi-Fi connections are super convenient, but they aren't always reliable.

- Can you figure out why not?
 - (Radio waves bounce all over the place and can get lost.)

So, if you're used to sending information to URLs (like **www.code.org**) and the servers actually have IP addresses for names (like 54.243.71.82) how does the Internet change from one to the other? That's what the DNS is for. The DNS (Domain Name Server) has tables that allow the system to go back and forth between URLs and IP addresses. If the Domain Name Servers ever stopped working, it would shut down the internet as we know it!

With that said, let's try to understand what the DNS does by making a little DNS table ourselves.

Pull out a piece of paper and draw a grid similar to that in the internet activity:

Sample of DNS Table:

| # | URL | IP Address |
|---|----------|--------------|
| 1 | code.org | 54.243.71.82 |
| 2 | | |
| 3 | | |

💡 Lesson Tip

If you're thinking that this is a lot of text and it would be extremely boring to try to lecture this to a class full of elementary school kids, you're absolutely right! If you're unable to show a YouTube video in class to help explain it all, I highly recommend drawing pictures to explain each idea above, or choosing students as volunteers to act out what you describe while you're explaining. They're not expected to get every detail and definition at this point, only to gain exposure.

| # | URL | IP Address |
|---|-----|------------|
| 4 | | |
| 5 | | |

First, we need to fill in this table.

- Survey the class for their favorite websites and write the URLs in the left column
- Use a site like **get-site-ip.com** to find the IP addresses for those sites and write them in the corresponding rows of the right column.

Now let's take this DNS Table and pretend to send messages through the internet!

Main Activity (20 min)

The Internet

Directions:

- Create your own DNS table, similar to what is shown above.
- Have the class help you fill in the blank spots in the table. Pick your favorite URLs and find their IP addresses using a site like **www.get-site-ip.com**.
- Divide into groups of 3 to 5.
- Assign each group an IP address from the newly created table, and assign each person in the group a position:
 - The Message Writer
 - The Internet
 - The Server (carries the IP address)
 - The Return Internet (optional)
 - The Message Receiver (optional)
- Each group will draw an **IP Address Cards and Delivery Type Cards - Manipulatives** to find out where their message is going and what their method of message delivery (Wi-Fi, Cable/DSL, or Fiber Optic Cable) will be.
- The Message Writer will craft a note to send to the server.
- The Internet will rip the message up into 4 small pieces called packets, then deliver each packet one at a time to the Server with the IP address that was drawn from the IP Address Card stack.
- The Server will make sure that the message arrives in order, then will send each packet off one at a time with the Return Internet (can be the same person or different person than the original Internet).
- The Return Internet will deliver each piece back to the Message Receiver (can be the same person or different person than the Message Writer) and put it back together.
- The Message Receiver will wait for all of the pieces to arrive, then read the message to be sure it arrived correctly!

Rules:

- The Internet must rip the message into exactly four packets.
- If the Internet drops a packet, they have to pick it up and go back to the start to deliver it again.
- The server has to wait for all of the message pieces to arrive before it can begin to send the message along.

Info:

- Wi-Fi: Convenient, but spotty. Wi-Fi doesn't require cables, but since the signal bounces all over the place, packets can get lost pretty easily.
 - Simulation: Internet must carry each packet on their shoulder (no hands).
- Cable/DSL: Fairly good at delivering messages, but you must be connected to a wire.
 - Simulation: Internet must carry each packet on the back of one hand and must keep the other hand touching a wall, desk, chair or the floor at all times.

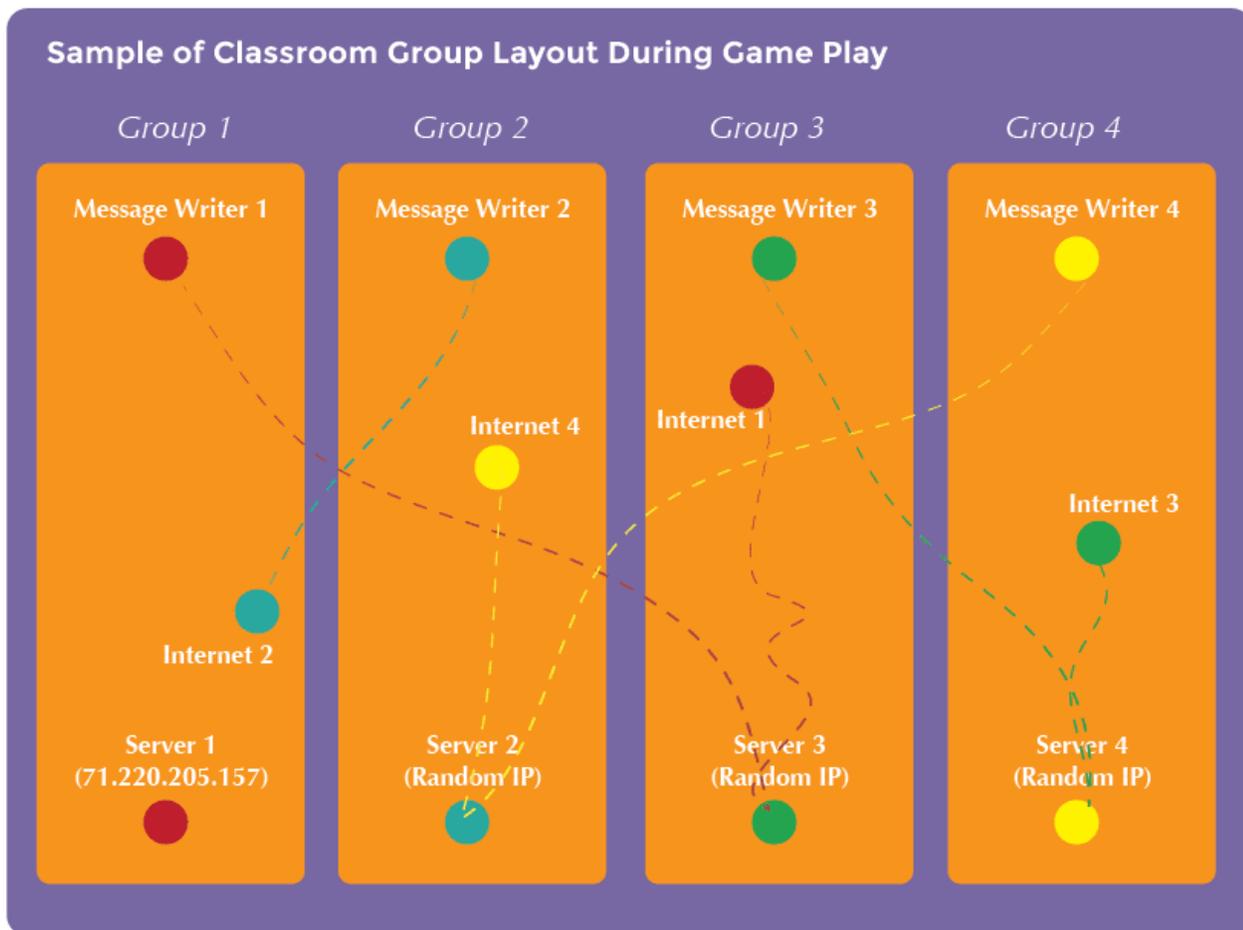
- Fiber Optic Cable: The best at delivering messages, but you must be connected to a wire.
 - Simulation: Internet can carry packets in hand, but must keep the other hand touching a wall, desk, chair or the floor at all times.

To play this game, you can have your groups cluster anywhere, but for the first time it can be less confusing to have groups play in a line.

💡 Lesson Tip

If it feels like there are too many rules to explain outright, feel free to post them on the board and just explain the game as you go. You can play multiple rounds until the class really understands.

- Line up the "Servers" on one end of the room (holding their IP addresses). The Return Internet players can be over there as well (if you have that many people in each group).
- Have the everyone else line up across from their server at the other side of the room.
- The Message Senders will likely be sending their messages to a server other than their own, so the Internet players will likely cross over from group to group. It may look something like the diagram below (in English):



Wrap Up (15 min)

Flash Chat: What did we learn?

- What kind of connection would you rather have (Wi-Fi, DSL/Cable, or Fiber Optic)? Why?
- Why might it take your message a long time to get somewhere?

💡 Lesson Tip

Flash Chat questions are intended to spark big-picture thinking about how the lesson relates to the greater world and the students' greater future. Use your knowledge of your classroom to decide if you want to discuss these as a class, in groups, or with an elbow partner.

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How do you feel about today's lesson?
- What's something you learned about the internet today?
- Why is learning about the internet important?

Assessment (5 min)

Internet - Assessment

Hand out the assessment worksheet and allow students to complete the activity independently after the instructions have been well explained. This should feel familiar, thanks to the previous activities.

Standards Alignment

CSTA K-12 Computer Science Standards

- ▶ **NI** - Networks & the Internet



This curriculum is available under a Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.

Lesson 26: Beyond Programming: Crowdsourcing

Unplugged | Crowdsourcing

Overview

In computer science, we face some big, daunting problems. Challenges such as finding large prime numbers or sequencing DNA are almost impossible to do as an individual. Adding the power of others makes these tasks manageable. This lesson will show your students how helpful teamwork can be in the industry of computer science.

Purpose

It's very rare that one computer scientist works completely alone on a project. Even when that does happen, there is always benefit in numbers. Today, students will learn what it means to crowdsource a project. This activity builds teamwork and creates an efficient environment for students to solve problems.

Agenda

Warm Up (20 min)

Vocabulary
Introduction

Main Activity (20 min)

Crowdsourcing - Worksheet

Wrap Up (15 min)

Flash Chat: What did we learn?
Journaling

Extended Learning

Objectives

Students will be able to:

- Identify a large task that needs to be done.
- Rearrange a large task into several smaller tasks.
- Build a complete solution from several smaller solutions.

Preparation

Watch the **Crowdsourcing - Teacher Video**.

Review **Crowdsourcing - Worksheet**.

Obtain a jar of lots of something (pennies, buttons, slips of paper, etc) and a deck of cards.

Make sure every student has a **Think Spot Journal - Reflection Journal**.

Links

Heads Up! Please make a copy of any documents you plan to share with students.

For the Teacher

- **Crowdsourcing** - Unplugged Video ([download](#))
- **Crowdsourcing** - Teacher Video
- **Crowdsourcing** - Worksheet

For the Students

- **Think Spot Journal** - Reflection Journal

Vocabulary

- **Crowdsourcing** - Getting help from a large group of people to finish something faster.

Teaching Guide

Warm Up (20 min)

Vocabulary

This lesson has one new and important word:

Crowdsourcing - Say it with me: Crowd-sore-sing

Getting help from a large group of people to finish something faster.

Introduction

- Show your students your jar full of something.
 - "Look at this jar. I have a lot of buttons in here, and I need to tell the principal how many there are before the end of class."
 - "Can you think of a way I could get these counted quickly?"
- Your students may guide you toward seeking help, but if they don't, you can suggest it, too.
 - Pour all of the buttons (or pennies, etc.) into a pile on the floor.
 - Invite all of the students to come up and grab a small number (ten is good, but you can do more if your students can handle it).
 - Once they've counted out their ten, have them report to you, drop their buttons back in the jar, and go again until the pile is gone.
- Comment on how fast the task went.
 - Have the class reflect on how long it might have taken or how hard it may have felt to do alone.

💡 Lesson Tip

Jars of buttons and pennies work nicely, but if you find yourself with little time to prepare, you can cut slips of paper and put them in a ziplock bag or even a pencil box.

Main Activity (20 min)

Crowdsourcing - Worksheet

Sometimes you have a big job that needs to get done, but it feels like it will take forever. Crowdsourcing is a way of using teamwork to make the job go much faster! In this game, we'll use crowdsourcing to sort decks of playing cards.

Directions:

1. Divide into groups of 4, 5, or 6.
2. Grab your deck of playing cards and dump it into a bag, bucket, or even a loose pocket that you can make with the bottom of your shirt.
3. Shake the cards until they're all mixed up.
4. Dump the cards out onto a table or desk where the whole group can see them.
5. Decide how to break up the task of sorting the deck so that every person has something to do and no one is doing too much.
6. Time yourself sorting the cards. Can you figure out a way to do it faster?
7. Repeat the game over and over until you think you have found the fastest way of crowdsourcing the card sorting activity.

Wrap Up (15 min)

Flash Chat: What did we learn?

- Have you ever tried to sort a pile of cards by yourself?
- Do you think it was easier or harder to have help?
- What other things do you have to do in life that could be easier with help?

Journaling

Having students write about what they learned, why it's useful, and how they feel about it can help solidify any knowledge they obtained today and build a review sheet for them to look to in the future.

Journal Prompts:

- What was today's lesson about?
- How do you feel about today's lesson?
- What are the benefits of crowdsourcing?
- What kind of things do you want to make with computer science? How do you see crowdsourcing being beneficial in those projects?

💡 Lesson Tip

It can be challenging for students to figure out how to break apart large tasks at first. Students might find it helpful to have some ideas handed to them after working for a while. One great division for sorting cards is as follows:

- One person picks up the cards and determines the suit of each one.
- One person manages Hearts.
- One person manages Diamonds.
- One person manages Clubs.
- One person manages Spades.
- (If there's another, they can put all sorted suits back together again.)

💡 Lesson Tip

Flash Chat questions are intended to spark big-picture thinking about how the lesson relates to the greater world and the students' greater future. Use your knowledge of your classroom to decide if you want to discuss these as a class, in groups, or with an elbow partner.

Extended Learning

Use these activities to enhance student learning. They can be used as outside of class activities or other enrichment.

Reverse Crowdsourcing

Often we think of crowdsourcing as pulling things apart to make them more simple. You can also make big, beautiful things with the same technique.

Have your students each grab three cards and build one segment of a **card house**. Each student can go one after another to build a grand card tower.

Try with two, or even three students adding their chunk at a time.

- Does crowdsourcing always make a task easier?

Crowdsourcing in the Round

- You can crowdsource all at the same time or you can do it one person at a time. Try having the whole class sort the same deck of cards, one student at a time.
 - Shuffle the cards and place them in a pile in the center of the room.
 - Have each student approach the pile and choose four cards.
 - Have four piles for the students to sort their cards into
 - Spades
 - Clubs
 - Hearts
 - Diamonds
 - Once all cards have been put in their four piles, have the following four students sort the individual piles.
 - The last person will put all four piles together.
- This version may not save a lot of time, but it still divides the work and lets each individual have more free time!

Standards Alignment

CSTA K-12 Computer Science Standards

► **AP** - Algorithms & Programming



This curriculum is available under a
Creative Commons License (CC BY-NC-SA 4.0).

If you are interested in licensing Code.org materials for commercial purposes, **contact us**.