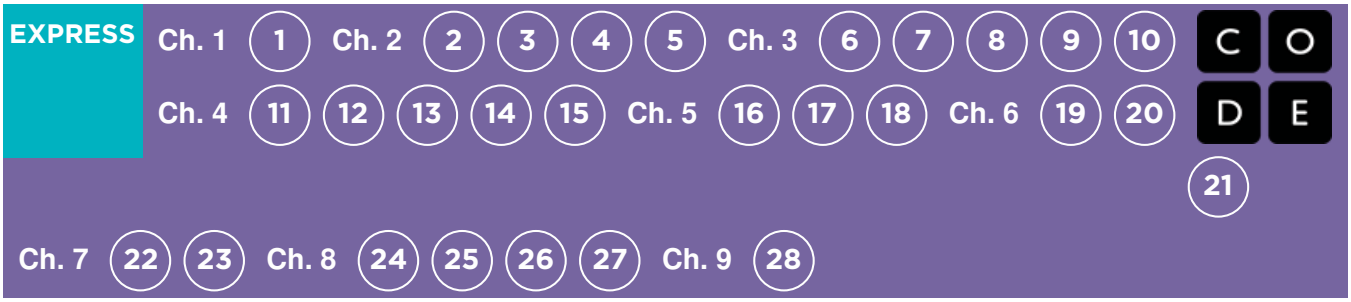


Express

CS Fundamentals Express combines much of Courses A-F into a single self-paced course. We recommend Courses A-F for most K-5 classrooms, If you want to have your students work through plugged lessons from multiple courses at their own paces, the Express course can be a better choice.

Chapter 1: Warm Up

Lesson 1: Dance Party

Warm-up

In this lesson, students will program their own interactive dance party. This activity requires sound as the tool was built to respond to music.

Chapter Commentary

Warm Up

Chapter 2: Sequencing

Lesson 2: Programming with Angry Birds

Sequencing

Learn about sequences and algorithms with Angry Birds.

Lesson 3: Debugging with Scrat

Sequencing

Find problems in puzzles and practice your debugging skills.

Lesson 4: Collecting Treasure with Laurel

Sequencing

Write algorithms to help Laurel the Adventurer collect lots of gems!

Lesson 5: Creating Art with Code

Sequencing

Create beautiful images by programming the Artist.

Chapter Commentary

Sequencing

Chapter 3: Loops

Lesson 6: Loops with Rey and BB-8

Loops

Help BB-8 through mazes using loops!

Lesson 7: Sticker Art with Loops

Loops

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

Lesson 8: Nested Loops in Maze

Loops

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

Lesson 9: Snowflakes with Anna and Elsa

Loops

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: Looking Ahead with Minecraft

Loops

Avoid the lava! Here you will learn about conditionals in the world of Minecraft.

Chapter Commentary

Loops

Chapter 4: Conditionals

Lesson 11: If/Else with Bee

Conditionals

Now that you understand conditionals, it's time to program Bee to use them when collecting honey and nectar.

Lesson 12: While Loops with the Farmer

Conditionals

Loops are so useful in coding. This lesson will teach you about a new kind of loop: while loops!

Lesson 13: Conditionals in Minecraft: Voyage Aquatic

Conditionals

Here you will learn about conditionals in the world of Minecraft.

Lesson 14: Until Loops in Maze

Conditionals

You can do some amazing things when you use `until` loops!

Lesson 15: Harvesting with Conditionals

Conditionals

It's not always clear when to use each conditional. This lesson will help you get practice deciding what to do.

Chapter Commentary

Conditionals

Chapter 5: Functions

Lesson 16: Functions in Minecraft

Functions

Can you figure out how to use functions for the most efficient code?

Lesson 17: Functions with Harvester

Functions

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

Lesson 18: Functions with Artist

Functions

Make complex drawings more easily with functions!

Chapter Commentary

Functions

Chapter 6: Variables

Lesson 19: Variables with Artist

Variables

Don't forget to bring creativity to class! In these puzzles you will be making fantastic drawings using variables.

Lesson 20: Changing Variables with Bee

Variables

This bee loves variables!

Lesson 21: Changing Variables with Artist

Variables

In this lesson, you'll make drawings using variables that change as the program runs.

Chapter Commentary

Variables

Chapter 7: For Loops

Lesson 22: For Loops with Bee

For Loops

Buzz buzz. In these puzzles you will be guiding a bee to nectar and honey using `for` loops!

Lesson 23: For Loops with Artist

For Loops

Get ready to make your next masterpiece. Here you will be using `for` loops to make some jaw-dropping pictures.

Chapter Commentary

For Loops

Chapter 8: Sprites

Lesson 24: Swimming Fish in Sprite Lab

Sprites

Learn how to create and edit sprites.

Lesson 25: Alien Dance Party

Sprites

Practice making games to share with your friends and family.

Lesson 26: Behaviors in Sprite Lab

Sprites

Learn to program your own sprite behaviors!

Lesson 27: Virtual Pet with Sprite Lab

Sprites

In this lesson, students will create an interactive Virtual Pet that looks and behaves how they wish. Students will use Sprite Lab's "Costumes" tool to customize their pet's appearance. They will then use events, behaviors, and other concepts they have learned to give their pet a life of its own!

Chapter Commentary

Sprites

Chapter 9: End of Course Project

Lesson 28: End of Course Project

End of Course Project

Projects this big take time and plenty of planning. Find your inspiration, develop a plan, and unleash your creativity!

Chapter Commentary

End of Course Project

Lesson 1: Dance Party

Overview

In this lesson, students will program their own interactive dance party. This activity requires sound as the tool was built to respond to music.

Purpose

This lesson introduces the core CS concepts of coding and event programming (using blocks).

Agenda

Getting Started (5 minutes)

Setting the Stage

Activity (30-45 minutes)

Code Your Own Dance Party
Level by Level Support

Wrap Up (5 minutes)

Debrief

Assessment (2 minutes)

View on Code Studio

Objectives

Students will be able to:

- Develop programs that respond to timed events
- Develop programs that respond to user input
- Create dance animations with code

Preparation

- [ ] Play through the puzzles to find any potential problem areas for your class.
- [ ] Make sure every student has a journal.
- [ ] Consider the need for headphones. This activity relies on sound.

Links

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

For the Teachers

- **Spotify Playlist (all ages)**

Vocabulary

- **code** - (v) to write code, or to write instructions for a computer.
- **Event** - An action that causes something to happen.
- **Program** - An algorithm that has been coded into something that can be run by a machine.

Teaching Guide

Getting Started (5 minutes)

Setting the Stage

Welcome students to class and very briefly introduce the day's activity.

Remarks

> Today we're going to do something really creative. What's your favorite way to be creative?

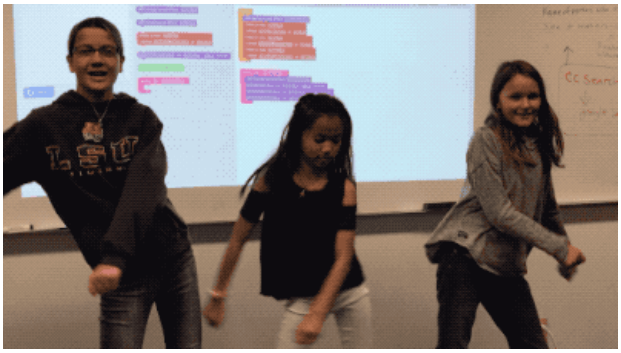
Encourage students to share the ways they express creativity, such as with art, dance, music, writing.

Explain that today we're going to be creative with our code. Just like choosing which type of colors of paint to use, or what kinds of words you write with can be express creativity, choosing what code you write and how people interact with it can be an opportunity to express your creativity too!

Teaching Tip

If you have time and would like to prepare your students with an unplugged activity, consider delivering **"Dance Party: Unplugged"** before this lesson. This brief lesson introduces students to the idea of events triggering different dance moves.

Get up and dance: Announce to the class that today we're going to see how we can combine coding with dancing in a creative way. Ask your kids to floss, dab, or do a creative dance move of their own for 10 seconds to get them in the mood. You can play a song from this **Spotify Playlist (all ages)** to help kick things off. Capture your class's moves on video.



Activity (30-45 minutes)

Code Your Own Dance Party

Music Filtering

This tutorial features songs from popular artists. To get a preview of the song list in this tutorial, check out this **Spotify Playlist**. We are using radio-safe versions of all songs and for students under 13, we limit the music to this filtered list **Spotify Playlist (all ages)**. If you would like to use the filtered list with older students, you can share **this link** with your classroom.

Level by Level Support

Level 1

- Drag the red make a new block from the toolbox on the left to the workspace on the right. Connect it inside the setup block.
- You have now written your first program. Make sure to press Run to see what happens. You should hear music and see a character start to move in the display area.

Level 2

- The green blocks are event blocks. These blocks start a new sequence of code and do not need to be connected inside

the setup block.

- Connecting the purple block under the green event block should make the character perform a dance move after the number of measures you indicate.

Level 3

- Make sure to bring out a second green event block. You should have a after 4 measures block and a after 6 measures block in your workspace. Both should have purple block connected underneath.

Level 4

- Drag out the set background effect block.
- This block can be connected inside setup , under an event block, or in both places!

Level 5

- With the right code, you should see the dancer cycle through a different move every 2 measures.
- Make sure there is a purple do forever block connected inside the every 2 measures block.
- Make sure the do forever block is set to either (Next), (Previous), or (Random). Otherwise, the dancer will just perform the set move repeatedly.

Level 6

- Be careful to avoid putting the red make a new block inside the every 2 measures block. This will cause your program to create multiple identical dancers at the same location.

Level 7

- Many students will be familiar with the idea that you can make something seem to be further away by drawing it a smaller scale. In this level, you'll create this effect by making the two dancers on the ends smaller.
- It is important to make sure that the teal set block is placed somewhere in the program **after** the dancer has been created. To solve this puzzle, place a set backup_dancer2 size to 50 block somewhere under the make a new robot called backup_dancer2 block.

Level 8

- As with the previous level, make sure to only use the set tint to (color) block after you have the made the dancer in your program. For example, placing it as the first step in the setup area of your program will have no effect.

Level 9

- Computer science allows to process input information into interesting types of output. In this level, students can explore how a dancer's properties can be updated automatically based on the actual sound of the music.

Level 10-11

- These levels are about making the dance interactive. Try out the new when pressed event to make the dancers respond to key presses.
- You can use the arrow keys on your computer keyboard or click the orange buttons under the display area.
- Note that the **code students write in these levels is not checked for correctness**. This means they will always pass the level, even if they do not change the program. Encourage students to determine for themselves whether the code is doing what they expect before moving on.

Teaching Tip

By this point in the lesson you may notice that the instructions are less prescriptive. Encourage students to be creative and explore the new blocks that are introduced. From this point on student code is **not checked for correctness** in order to encourage experimentation instead of solving a specific task.

Level 12

- The top six blocks in this tool box look familiar but all work in a different way. Instead of creating or controlling one dancer, they work with groups of dancers.
- Like the previous levels, **code is not checked for correctness here** . Students should feel free to experiment with groups in ways that are interesting to them.

Level 13

- This last level is very open-ended. The tutorial itself is designed to give students ample time to keep working on their own dance.

- **Encourage Sharing:** If students have cell phones with a data plan they can quickly text a link to their projects to their own phone or a friend's. If your school policy allows it, encourage them to do so here.
- **Encourage Creativity:** Creativity is important throughout this lesson, but this is true here more than anywhere else!

Wrap Up (5 minutes)

Debrief

- Pose a prompt that has multiple answers such as “What is something you enjoyed about today's activity?” or “What is the connection between creativity and computer science?"

Assessment (2 minutes)

Ask students to add their “Whip Around” sticky notes or note cards to your "Computer Science" mind map on their way out the door. Try to populate the board with lots of great ideas about what CS is and why it matters.

Lesson 2: Programming with Angry Birds

Overview

Using characters from the game Angry Birds, students will develop sequential algorithms to move a bird from one side of a maze to the pig at the other side. To do this they will stack code blocks together in a linear sequence, making them move straight, turn left, or turn right.

Purpose

In this lesson, students will develop programming and debugging skills on a computer platform. The block-based format of these puzzles help students learn about sequence and concepts, without having to worry about perfecting syntax.

Agenda

Previewing Online Puzzles as a Class (3 min)
- **Teaching this course as a class?**

Main Activity (30 min)
- **Online Puzzles**

Wrap Up (5 - 10 min)
- **Journaling**

Extended Learning

View on Code Studio

Objectives

Students will be able to:

- Translate movements into a series of commands.
- Identify and locate bugs in a program.

Preparation

- [ ] Play through the puzzles to find any potential problem areas for your class.
- [ ] (Optional) Pick a couple of puzzles to do as a group with 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 Students

- **Think Spot Journal** - Reflection Journal

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.
- **Sequencing** - Putting commands in correct order so computers can read the commands.

Teaching Guide

Previewing Online Puzzles as a Class (3 min)

Model: Pull up Puzzle 5 to do in front of the class. While working through this puzzle with the class, remind students that making mistakes is okay and remind them that the only way to be successful is to be persistent.

Discuss: Does anyone know how to solve this puzzle?

As the teacher, you should decide if you will have the students tell you how to solve it from their seats, or come to the computer to drag the actual blocks in one-by-one.

Transition: Now that students have seen an online puzzle in practice, they should be ready to start solving puzzles of their own. Continue to the lab or bring out their classroom machines.

Lesson Tip

Some students may struggle with turning their bird in the correct direction, particularly when the bird isn't facing up. Remind students that when we say turn left or right, we're giving directions from the bird's point of view.

Content Corner

Teaching this course as a class?

Our grade-aligned CS Fundamentals courses use unplugged lessons to build community and introduce tricky computer science concepts, including **sequencing**. Check out the lesson **My Robotic Friends** from **Course C** and **Graph Paper Programming** from **Course D**!

Main Activity (30 min)

Online Puzzles

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

Code Studio levels

Maze Intro: Programming with Blocks 1 2 *(click tabs to see student view)*

Practice 3 4 5 6 7 *(click tabs to see student view)*

Challenge 8 *(click tabs to see student view)*

Practice 9 *(click tabs to see student view)*

Predict 10 *(click tabs to see student view)*

Practice 11 *(click tabs to see student view)*

Levels

Extra | Extra *(click tabs to see student view)*

Circulate: 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.

Discuss: After providing students with end-of-class warnings, grab everyone's attention and get them to reflect on the experiences that they just had.

- Did anyone feel frustrated during any of the puzzles?
- Did anyone notice the need to be persistent?

Transition: Have students grab their Thinkspot Journals and take a moment to leave lessons for themselves.

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?
- Draw an activity you like to do that you struggled with the first time. Draw or describe how you got better.

Extended Learning

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

Create Your Own

In small groups, let students design their own mazes and challenge each other to write programs to solve them. For added fun, make life-size mazes with students as the pig and bird.

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

▸ **AP** - Algorithms & Programming

Lesson 3: Debugging with Scrat

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)

- **Online Puzzles**

Wrap Up (5 - 10 min)

- **Journaling**

Extended Learning

View on Code Studio

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 lesson to find any potential problem areas for your class.
- [ ] (Optional) Pick a couple of puzzles to do as a group with 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 Students

- **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.

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)

Online Puzzles

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 (2017)

- ▶ **AP** - Algorithms & Programming

Lesson 4: Collecting Treasure with Laurel

Overview

In this series of puzzles, students will continue to develop their understanding of algorithms and debugging. With a new character, Laurel the Adventurer, students will create sequential algorithms to get Laurel to pick up treasure as she walks along a path.

Purpose

In this lesson, students will be practicing their programming skills using a new character, Laurel the Adventurer. When someone starts **programming** they piece together instructions in a specific order using something that a machine can read. Through the use of programming, students will develop an understanding of how a computer navigates instructions and order. Using a new character with a different puzzle objective will help students widen their scope of experience with sequencing and algorithms in programming.

Agenda

Warm Up (5 min)
- **Introduction**

Bridging Activity - Programming (10 min)
- **Previewing Online Puzzles as a Class**

Main Activity (30 min)
- **Online Puzzles**

Wrap Up (5 - 10 min)
- **Journaling**

View on Code Studio

Objectives

Students will be able to:

- Order movement commands as sequential steps in a program.
- Represent an algorithm as a computer program.
- Develop problem solving and critical thinking skills by reviewing debugging practices.

Preparation

- [ ] Play through the lesson to find potential problem areas for your class.
- [ ] Make sure every student has a journal.

Links

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

For the Students

- **Think Spot Journal** - Reflection Journal

Vocabulary

- **Algorithm** - A list of steps to finish a task.
- **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 (5 min)

Introduction

This lesson uses most of the same blocks from Programming in Maze and adds the ability to collect . Tell the students that this block will allow Laurel the Adventurer to pick up the treasure that she is standing over. This new block will be discussed more in the bridging activity.

Bridging Activity - Programming (10 min)

Previewing Online Puzzles as a Class

Pull a puzzle from the corresponding online stage. We recommend puzzle 7. Have students discuss a pattern that they think will get Laurel the Adventurer to collect all the treasure. Ask the students to share. See how many other students had the same answer!

Main Activity (30 min)

Online Puzzles

Laurel the Adventurer is looking to collect as much treasure as she can. Instruct the students to traverse the puzzle to collect whatever they can. Some levels will require you to only pick up one piece of treasure, but others will require you to pick up every piece of treasure. Pay attention to the instructions to know what to do!

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?
- Draw a maze that you might solve with the blocks you used today.

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

- ▸ **AP** - Algorithms & Programming

Lesson 5: Creating Art with Code

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)
- **Online Puzzles**

Wrap Up (10 - 15 min)
- **Journaling**

Extended Learning

View on Code Studio

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 lesson to find any potential problem areas for your class.
- ☐ (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 Students

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

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.

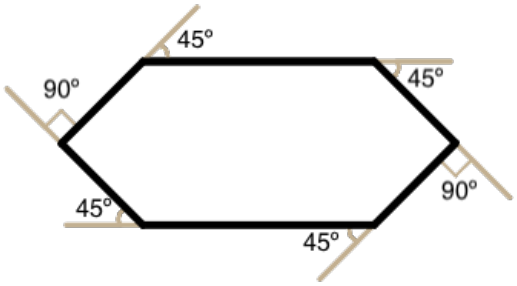
Main Activity (30 min)

Online Puzzles

In this set of puzzles, the artist will no longer be constrained to 90 degree angles. Having physical protractors available can be 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 (2017)

▸ **AP** - Algorithms & Programming

Lesson 6: Loops with Rey and BB-8

Overview

Building on the concept of repeating instructions from "Getting Loopy," this stage will have students using loops to help BB-8 traverse a maze more efficiently than before.

Purpose

In this lesson, students will be learning more about**loops** and how to implement them in Blockly code. Using **loops** is an important skill in programming because manually repeating commands is tedious and inefficient. With the Code.org puzzles, students will learn to add instructions to existing loops, gather repeated code into loops, and recognize patterns that need to be looped. It should be noted that students will face puzzles with many different solutions. This will open up discussions on the various ways to solve puzzles with advantages and disadvantages to each approach.

Agenda

Bridging Activity - Loops (10 min)

- **Teaching this course as a class?**
- **Previewing Online Puzzles as a Class**

Main Activity (30 min)

- **Online Puzzes**

Wrap Up (5 - 10 min)

- **Journaling**

Extended Learning

View on Code Studio

Objectives

Students will be able to:

- Identify the benefits of using a loop structure instead of manual repetition.
- Break down a long sequence of instructions into the largest repeatable sequence.
- Employ a combination of sequential and looped commands to reach the end of a maze.

Preparation

- [ ] Play through the lesson to determine if there will be any problem areas for your class.
- [ ] Make sure every student has a journal.

Links

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

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

Bridging Activity - Loops (10 min)

Previewing Online Puzzles as a Class

Pull up the online puzzles and choose a puzzle to do in front of the class. We recommend puzzle 8 for its staircase pattern. Ask the students to write a program to solve the puzzle on paper.

Content Corner

Teaching this course as a class?

Our grade-aligned CS Fundamentals courses use unplugged lessons to build community and introduce tricky computer science concepts, including **loops**. Check out the lesson **My Loopy Robotic Friends** from **Course C**!

Main Activity (30 min)

Online Puzzes

As students work through the puzzles, see if they can figure out how many blocks they use with a loop vs. not using a loop. **Pair Programming - Student Video** works really well with this set of puzzles because there are a few ways to fill the loops. Push for friendly discussion between pairs in instances of disagreement on how to solve the puzzle. Have the students ask each other questions like:

- How did you come up with that solution?
- What are some benefits of solving the puzzle that way?

We also recommend having paper on hand for students to write out their code and find any repetition to use in loops.

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?
- How did loops make your program easier to write?
- Think of something that repeats over and over again. What might the program for that look like?

Extended Learning

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

So Moving

- Give the students pictures of actions or dance moves that they can do.
- Have students arrange moves and add loops to choreograph their own dance.
- Share the dances with the rest of the class.

Connect It Back

- Find some YouTube videos of popular dances that repeat themselves.
- Can your class find the loops?
- Try the same thing with songs!

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

- **AP** - Algorithms & Programming

Lesson 7: Sticker Art with Loops

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)
- **Online Puzzes**

Wrap Up (15 min)
- **Journaling**

View on Code Studio

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 lesson to find any potential problem areas for your class.
- [ ] Make sure every student has a journal.

Links

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

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:

- As a class, playing through a puzzle from the last lesson, "Loops with Rey and BB-8"
- Reviewing how to use Artist by playing through a puzzle from "Programming in Artist"
- 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)

Online Puzzes

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 (2017)

- ▸ **AP** - Algorithms & Programming

Lesson 8: Nested Loops in 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)
- **Online Puzzles**

Wrap Up (15 min)
- **Journaling**

View on Code Studio

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 the lesson to find any potential problem areas for your class.
- [ ] Make sure every student has a journal.

Links

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

For the Students

- **Think Spot Journal** - Reflection Journal

Vocabulary

- **Command** - An instruction for the computer. Many commands put together make up algorithms and computer programs.
- **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)

Online Puzzles

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 (2017)

▸ **AP** - Algorithms & Programming

Lesson 9: Snowflakes with Anna and Elsa

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)
- **Introduction**

Main Activity (30)
- **Online Puzzes**

Wrap Up (15)
- **Journaling**

View on Code Studio

Objectives

Students will be able to:

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

Preparation

- [ ] Play through the lesson to find and potential problem areas for your class.
- [ ] Make sure every student has a journal.

Links

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

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)

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)

Online Puzzes

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)

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 (2017)

▸ **AP** - Algorithms & Programming

Lesson 10: Looking Ahead with Minecraft

Overview

This lesson was originally created for the Hour of Code, alongside the Minecraft team. Students will get the chance to practice ideas that they have learned up to this point, as well as getting a sneak peek at conditionals!

Purpose

This set of puzzles will work to solidify and build on the knowledge of loops, and introduce conditionals. By pairing these two concepts together, students will be able to explore the potential for creating fun and innovative programs in a new and exciting environment.

Agenda

Warm Up (15 min)
- **Introduction**

Main Activity (30 min)
- **Online Puzzles**

Wrap Up (15 min)
- **Journaling**

Extended Learning

View on Code Studio

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 the lesson to find any potential problem areas for your class.
- [ ] Make sure every student has a journal.

Links

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

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.

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** during this lesson. 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)

Online Puzzles

Students are in for a real treat with this lesson. It's likely most of your students have heard of Minecraft, but give a brief introduction for those that may not know.

Minecraft is a game of cubes. You can play as Alex or Steve as you work through mazes. You'll need to avoid lava, pick up items, and explore in a world made up of cubes of things.

Demonstrate one of the puzzles to the class (we recommend puzzle 11.) Once all questions have been addressed, transition students to computers and let them start pair programming.

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:

- Draw a feeling face to show how you felt during today’s lesson.
- Draw something else you could have built in this Minecraft world.
- Can you draw a scene where someone is using a conditional?

Extended Learning

More Minecraft

If you find that your class really enjoys the Minecraft environment,**here are some links to other Minecraft games they can play online**. These games will also teach basic coding skills.

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

- ▶ **AP** - Algorithms & Programming

Lesson 11: If/Else with Bee

Overview

Up until this point students have been writing code that executes exactly the same way each time it is run - reliable, but not very flexible. In this lesson, your class will begin to code with conditionals, allowing them to write code that functions differently depending on the specific conditions the program encounters.

Purpose

After being introduced to conditionals in "Conditionals with Cards," students will now practice using them in their programs. The `if / else` blocks will allow for a more flexible program. The bee will only collect nectar **if** there is a flower or make honey **if** there is a honeycomb. Students will also practice and recognize a connection between `if / else` blocks and `while` loops in this set of puzzles.

Agenda

Preview of Online Puzzles (15 min)
Main Activity (30 min)
    Online Puzzles
Wrap Up (15 min)
    Journaling
Extended Learning

View on Code Studio

Objectives

Students will be able to:

- Translate spoken language conditional statements into a program.
- Solve puzzles using a combination of looped sequences and conditionals.

Preparation

- [ ] Play through the lesson 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 Students

- **Think Spot Journal** - Reflection Journal

Vocabulary

- **Conditionals** - Statements that only run under certain conditions.

Teaching Guide

Preview of Online Puzzles (15 min)

Pull up a puzzle from the esson, we recommend puzzle 9.

- Ask the class what the bee should do when it gets to the cloud.
  - The bee should use a conditional to check for a flower or a honeycomb.
- Use the `if at flower / else` block. Ask the class what the bee should do if there's a flower. If there's not a flower, there will be a honeycomb. What should the bee do then?
  - The bee should `get nectar` if there is a flower and `make honey` if there is a honeycomb.

Fill in the rest of the code and press `Run` . Discuss with the class why this worked.

Main Activity (30 min)

Online Puzzles

These puzzles might sprout some questions, so have the students work in pairs or implement the "Ask three before you ask me" rule (have the students ask three other peers for help before they go to the teacher.) This will spark discussions that will develop each student's understanding.

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 today's lesson make you feel?
- What conditionals did you use in your code today?
- What are some other conditionals a bee might use? Examples include:
  - if there is a tree in front of me, buzz out of the way
  - if my wing is hurt, rest on the ground
  - if I see another bee, say "Hello!"

Extended Learning

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

True/False Tag

- Line students up as if to play **Red Light / Green Light** .
- Select one person to stand in front as the Caller.
- The Caller chooses a condition and asks everyone who meets that condition to take a step forward.
  - If you have a red belt, step forward.
  - If you are wearing sandals, take a step forward.
- Try switching it up by saying things like "If you are not blonde, step forward."

Nesting

- Break students up into pairs or small groups.
- Have them write if statements for playing cards on strips of paper, such as:
  - If the suit is clubs
  - If the color is red
- Have students create similar strips for outcomes.
  - Add one point
  - Subtract one point
- Once that's done, have students choose three of each type of strip and three playing cards, paying attention to the order selected.
- Using three pieces of paper, have students write three different programs using only the sets of strips that they selected, in any order.
  - Encourage students to put some if statements inside other if statements.
- Now, students should run through all three programs using the cards that they drew, in the same order for each program.
  - Did any two programs return the same answer?
  - Did any return something different?

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

▸ **AP** - Algorithms & Programming

Lesson 12: While Loops with the Farmer

Overview

By the time students reach this lesson, they should already have plenty of practice using `repeat` loops, so now it's time to mix things up.

While loops are loops that continue to repeat commands while a condition is met. `While` loops are used when the programmer doesn't know the exact number of times commands need to be repeated, but does know what condition needs to be true in order for the loop to continue repeating. For example, students will be working to fill holes and dig dirt in Farmer. They will not know the size of the holes or the height of the mountains of dirt, but the students will know they need to keep filling the holes and digging the dirt as long as the ground is not flat.

Purpose

As your students continue to deepen their knowledge of loops, they will come across problems where a command needs to be repeated, but it is unknown how many times it needs to be repeated. This is where `while` loops come in. In today's lesson, students will develop a beginner's understanding of condition-based loops and also expand their knowledge of loops in general.

Agenda

Warm Up (10 min)
- **Introduction**

Main Activity (30 min)
- **Online Puzzles**

Wrap Up (15 min)
- **Journaling**

Bridging Activity (15 min)
- **Preview of Online Puzzles**

View on Code Studio

Objectives

Students will be able to:

- Distinguish between loops that repeat a fixed number of times and loops that repeat as long as a condition is true.
- Use a while loop to create programs that can solve problems with unknown values.

Preparation

- [ ] Play through the lesson 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 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.
- **Loop** - The action of doing something over and over again.
- **Repeat** - Do something again
- **While Loop** - A loop that continues to repeat while a condition is true.

Teaching Guide

Warm Up (10 min)

Introduction

Use "while" in a sentence in front of the students. Ask the students what the word "while" means. If you were to say "while there is a hole, fill it with dirt" what would they do? How long would they do that?

When you use a word like “while”, you are relying on a condition to tell the computer how long the loop should run. A condition is a statement that is tested and found to be true or false. In the case above, the condition is if there is a hole. It’s only possible for there to be a hole or for there not to be a hole, thus the statement is only ever true or false.

Tell the students they will be learning about a new kind of loop. Previously, students only used loops to repeat a command a certain number of times. Here, they won't always know how many times to repeat the command, however, they will know when to stop or when to keep going. `While` loops allow the programmer to repeat a command as long as a condition is still true. In the previous example, the condition is the existence of a hole.

If there's time, have the students discuss other times using a `while` loop would be useful. Examples include:

- Running toward a ball **while** it is in front of you.
- Filling a glass **while** it has space for more liquid.
- Walk forward **while** there is a path ahead.

Main Activity (30 min)

Online Puzzles

`While` loops are not always a difficult concept for students to understand, but if you think your class might struggle with these puzzles, we recommend **Pair Programming - Student Video**. This will allow students to bounce ideas of each other before implementing the code. Pair programming works to increase confidence and understanding with topics like `while` loops.

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 the difference between a `while` loop and a normal `repeat` loop?
- Give an example of a puzzle where you would use a `while` loop, but not use a `repeat` loop. Can you give an example of a puzzle where you would use a `repeat` loop, but not a `while` loop?

Bridging Activity (15 min)

Preview of Online Puzzles

Pull up a puzzle from the lesson, we recommend Puzzle 6.

- Ask the class what the farmer should do when she gets to the pile of dirt.
  - The farmer should use a conditional in a `while` loop to check if there is a pile.
- Use the `while there is a pile / do` block. Ask the class what the farmer should do if there is no pile. When should the farmer stop?
  - The farmer should `remove 1` if there is a pile. The while loop will continue until there is no pile, so then your code is finished!

Fill in the rest of the code and press `Run`. Discuss with the class why this worked.

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

▶ **AP** - Algorithms & Programming

Lesson 13: Conditionals in Minecraft: Voyage Aquatic

Overview

This lesson was originally created for the Hour of Code, alongside the Minecraft team. Students will get the chance to practice ideas that they have learned up to this point, as well as getting a sneak peek at conditionals!

Purpose

This set of puzzles will work to solidify and build on the knowledge of loops, and introduce conditionals. By pairing these two concepts together, students will be able to explore the potential for creating fun and innovative programs in a new and exciting environment.

Agenda

Warm Up (15 min)
- **Introduction**

Main Activity (30 min)
- **Online Puzzles**

Wrap Up (15 min)
- **Journaling**

Extended Learning

View on Code Studio

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 the puzzles associated with this lesson to find any potential problem areas for your class.
- [ ] Make sure every student has a journal.

Links

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

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.

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** during this lesson. 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)

Online Puzzles

Students are in for a real treat with this lesson. It's likely most of your students have heard of Minecraft, but give a brief introduction for those that may not know.

Minecraft is a game of cubes. You can play as Alex or Steve as you work through mazes. You'll need to apick up items, and explore in a world made up of cubes of things.

Demonstrate one of the puzzles to the class (we recommend puzzle 11.) Once all questions have been addressed, transition students to computers and let them start pair programming.

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:

- Draw a feeling face to show how you felt during today’s lesson.
- Draw something else you could have built in this minecraft world.
- Can you draw a scene where someone is using a conditional?

Extended Learning

More Minecraft

If you find that your class really enjoys the Minecraft environment,**here are some links to other Minecraft games they can play online**. These games will also teach basic coding skills.

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

- ▶ **AP** - Algorithms & Programming

Lesson 14: Until Loops in Maze

Overview

In this lesson, students will learn about `until` loops. Students will build programs that have the main character repeat actions `until` they reach their desired stopping point.

Purpose

This set of puzzles will work to solidify and build on the knowledge of loops by adding the `until` conditional. By pairing these concepts together, students will be able to explore the potential for creating complex and innovative programs.

Agenda

Warm Up (10 min)
- **Introduction**

Bridging Activity (15 min)
- **Preview of Online Puzzles**

Main Activity (30 min)
- **Online Puzzles**

Wrap Up (15 min)
- **Journaling**

View on Code Studio

Objectives

Students will be able to:

- Build programs with the understanding of multiple strategies to implement conditionals.
- Translate spoken language conditional statements and loops into a program.

Preparation

- [ ] Play through the lesson 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 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.
- **Loop** - The action of doing something over and over again.
- **Repeat** - Do something again
- **Until** - A command that tells you to do something only up to the point that something becomes true.

Teaching Guide

Warm Up (10 min)

Introduction

In this lesson, students will be creating loops that only run until a condition is true. Help the students understand how this works by leading them in group activities and having them do an action until some condition is true. **For example: Have students touch their nose** until **you tell them to stop.**

Bridging Activity (15 min)

Preview of Online Puzzles

Pull up a puzzle. We recommend Puzzle 4.

- Ask the class what the bird should repeat to get to the pig.
  - The bird should repeat move forward , turn right , move forward , and then turn left .
- Ask the class what they can use to repeat this code.
  - The bird should repeat this pattern **until** it reaches the pig.

Fill in the rest of the code using the repeat until loop and press Run . Discuss with the class why this worked.

Main Activity (30 min)

Online Puzzles

Bringing together concepts is not easy, but this set of lessons is meant to help students see the endless possibilities of coding when using conditions. If students struggle at all with understanding the similarities or differences between while loops and until loops, have them try to think of how they would use similar statements in their real lives.

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's the difference between an until loop and a while loop?

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

▸ **AP** - Algorithms & Programming

Lesson 15: Harvesting with Conditionals

Overview

Students will practice `while` loops, `until` loops, and `if / else` statements. All of these blocks use conditionals. By practicing all three, students will learn to write complex and flexible code.

Purpose

Practicing the use of conditionals in different scenarios helps to develop a student's understanding of what conditionals can do. In the previous lesson, students only used conditionals to move around a maze. In this lesson, students will use conditionals to help the farmer know when to harvest crops. New patterns will emerge and students will use creativity and logical thinking to determine the conditions where code should be run and repeated.

Agenda

Warm Up (5 min)
- **Introduction**

Main Activity (30 min)
- **Online Puzzles**

Wrap Up (15 min)
- **Journaling**

View on Code Studio

Objectives

Students will be able to:

- Nest conditionals to analyze multiple value conditions using if, else if, else logic.
- Pair a loop and conditional statement together.

Preparation

- [ ] Play through the lesson 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 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.
- **Loop** - The action of doing something over and over again.
- **Repeat** - Do something again
- **While Loop** - A loop that continues to repeat while a condition is true.

Teaching Guide

Warm Up (5 min)

Introduction

Students shouldn't need as much of an introduction to concepts today because they have had practice with them in the previous lesson. Instead, you can share the story of the harvester.

The harvester is trying to pick crops like pumpkins, lettuce, and corn. However, the farmer has forgotten where she planted these crops, so she needs to check each plant before harvesting.

Main Activity (30 min)

Online Puzzles

Students will continue to work with `if / else` statements, `while` loops, and `until` loops. These puzzles are a bit more challenging, though, so encourage students to stick with them until they can describe what needs to happen for each program.

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?
- How can you see conditionals being useful in programs?
- What if people only spoke in if/else statements? What would be some advantages and disadvantages of this?

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

▸ **AP** - Algorithms & Programming

Lesson 16: Functions in Minecraft

Overview

Students will begin to understand how functions can be helpful in this fun and interactive Minecraft adventure!

Purpose

Students will discover the versatility of programming by practicing functions in different environments. Here, students will recognize reusable patterns and be able to incorporate named blocks to call pre-defined functions.

Agenda

Warm Up (10 min)
- **Introduction**
- **Teaching this course as a class?**

Bridging Activity - Functions (15 min)
- **Preview of Online Puzzles**

Main Activity (30 min)
- **Online Puzzles**

Wrap Up (15 min)
- **Journaling**

View on Code Studio

Objectives

Students will be able to:

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

Preparation

- [ ] Play through the lesson 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 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

🎓 Help the class understand that functions are simply a chunk of code that has a name. Once defined, you can use that name over and over in your program to tell the computer to run the chunk of code that you assigned to it.

Content Corner

Teaching this course as a class?

Our grade-aligned CS Fundamentals courses use unplugged lessons to build community and introduce tricky computer science concepts, including **functions**. Check out the lesson **Songwriting** from **Course E**!

Bridging Activity - Functions (15 min)

Preview of Online Puzzles

Pull up a puzzle from the lesson. We recommend puzzle 9 of this lesson. 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. Why can you not just use a loop?

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)

Online Puzzles

We recommend providing paper and pencils for students to write (or draw) out 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 (2017)

▸ **AP** - Algorithms & Programming

Lesson 17: Functions with 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)
- **Online Puzzles**

Wrap Up (15 min)
- **Journaling**

View on Code Studio

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 the lesson 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 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)

Online Puzzles

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 (2017)

▸ **AP** - Algorithms & Programming

Lesson 18: Functions with 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**

Main Activity (30 min)
- **Online Puzzles**

Wrap Up (15 min)
- **Journaling**

Extended Learning

View on Code Studio

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 the lesson to find any potential problem areas for your class.
- [ ] Make sure every student has a journal.

Links

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

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 (15 min)

Introduction

\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.

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.

Main Activity (30 min)

Online Puzzles

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 (2017)

- **AP** - Algorithms & Programming

Lesson 19: Variables with Artist

Overview

In this lesson, students will explore the creation of repetitive designs using variables in the Artist environment. Students will learn how variables can be used to make code easier to write and easier to read, even when the values don't change at runtime.

Purpose

This stage teaches the most basic use for variables, as a constant that reoccurs frequently in a program.

Agenda

Warm Up (15 min)

- **Introduction**
- **Teaching this course as a class?**

Bridging Activity - Variables (15 min)

- **Preview of Online Puzzles as a Class**

Main Activity (30 min)

- **Online Puzzles**

Wrap Up (15 min)

- **Journaling**

View on Code Studio

Objectives

Students will be able to:

- Assign values to existing variables.
- Utilize variables in place of repetitive values inside of a program.

Preparation

- [ ] Play through the lesson to find any potential problem areas for your class.
- [ ] Make sure every student has a journal.

Links

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

For the Students

- **Think Spot Journal** - Reflection Journal

Vocabulary

- **Constant** - A variable used throughout a program that never changes value
- **Variable** - A placeholder for a piece of information that can change.

Teaching Guide

Warm Up (15 min)

Introduction

🎓 **Discussion:**

- What is a variable? (A placeholder for a piece of information that can change.)
- When can a variable be helpful? (When you don't know what information is going to be used in a certain place until runtime, or when you have lots of places that one piece of information will be used, but that information might change someday.)

Ask the class when they could see a variable being helpful in programming. When would they NOT want to use a variable?

Content Corner

Teaching this course as a class?

Our grade-aligned CS Fundamentals courses use unplugged lessons to build community and introduce tricky computer science concepts, including **variables**. Check out the lesson **Envelope Variables** from **Course F**!

Bridging Activity - Variables (15 min)

Preview of Online Puzzles as a Class

Demo: Display a puzzle for the class. We recommend the 6th puzzle. Go over the code with the students to make sure they understand what's happening before they help you convert the code to use variables. Can they think of something that might happen that would make them really glad that they used variables instead of hardcoded numbers?

Transition: Now it's time for your students to move to their own machines and get started!

Main Activity (30 min)

Online Puzzles

Notice that this stage first covers the idea of a variable as a constant (a variable that you use in many places, but it does not change.) This might be something that students find helpful as they're creating their own projects.

Watch out for puzzle #6. It is the first time that students will be expected to set a variable on their own. This can be tricky if they don't have a true grasp on the concept. If they're having trouble, send them back to the prediction level (#5) and have them explain to their partners why the answer ended up what it was. Once both partners are convinced, let them continue back to puzzle #6.

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 is a variable? Why is it helpful in programming?
- How well do you think you understand variables? (Answer on a scale from 1-5 or with an emoticon.) If you're having

troubles, can you put into words what you don't understand?

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

- **AP** - Algorithms & Programming

Lesson 20: Changing Variables with Bee

Overview

This lesson will help illustrate how variables can make programs more powerful by allowing values to change while the code is running.

Purpose

You don't always know what a value is going to be before you begin your program. Sometimes, values change while your code is running. This lesson will illustrate how code with changing values can be helpful.

Agenda

Warm Up (15 min)
- **Introduction**

Main Activity (30 min)
- **Online Puzzles**

Wrap Up (15 min)
- **Journaling**

View on Code Studio

Objectives

Students will be able to:

- Identify areas where they can use variables to modify quantities during runtime.
- Examine code to find places where variables can be substituted for specific values.

Preparation

- [ ] Play through the lesson to find any potential problem areas for your class.
- [ ] Make sure every student has a journal.

Links

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

For the Students

- **Think Spot Journal** - Reflection Journal

Vocabulary

- **Variable** - A placeholder for a piece of information that can change.

Teaching Guide

Warm Up (15 min)

Introduction

This series is a little different than what students have done in the past. Now, instead of simply assigning a value to a variable and running your code, you'll need to help students see how a variable can be modified during program runtime.

Display Show students the play area from one of the later puzzles.

There are several things to unpack here, so you might need to give your students a chance to look at it critically before you expect them to do anything with it.

Think/Pair: Suppose all of the flowers/honeycomb in this picture had the same amount of nectar/honey. How would you solve this puzzle?

Share: Let students share their ideas until you decide on one that the entire class is good with.

Discuss: Now, imagine that we didn't want to have to write a separate chunk of code for each set of flowers and honeycomb. How could we use a variable to have our loop do this for us?

(Eventually, you'll want to get to the place where you initialize a variable to the original value, then change it each time through the loop so that it's ready for the next time.)

Main Activity (30 min)

Online Puzzles

This set of puzzles takes some serious computational thinking skills. If you find that students are getting stuck, help them break down the puzzles into the individual pieces:

- What would it look like if the flowers/honeycomb all had the same amount of nectar/honey?
- What would it look like without the functions?
- Now how can you use a variable to get the quantities the way you want them?
- Now can you build it back up to use a function?

Hint: Puzzle 7 becomes much easier if students utilize the `while path ahead` loop instead of a variable.

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 are some ways you have used variables so far?
- What else do you think you can do with variables?

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

▸ **AP** - Algorithms & Programming

Lesson 21: Changing Variables with Artist

Overview

In this lesson, students will explore the creation of repetitive designs using variables in the Artist environment. Students will learn how variables can be used to make code easier to write and easier to read. After guided puzzles, students will end in a freeplay level to show what they have learned and create their own designs.

Purpose

Variables are essentially placeholders for values that might be unknown at the time that you run your program or for values that can change during the execution of a program. These are vital to creating dynamic code because they allow your program to change and grow based on any number of potential modifications. This stage reinforces the use of variables, using the most basic capabilities of setting and using them.

Agenda

Warm Up (5 min)
- **Introduction**

Main Activity (20 min)
- **Online Puzzles**

Wrap Up (15 min)
- **Journaling**

View on Code Studio

Objectives

Students will be able to:

- Assign values to existing variables.
- Utilize variables in place of repetitive values inside of a program.
- Use variables to change values inside of a loop.

Preparation

- [ ] Play through the lesson to find any potential problem areas for your class.
- [ ] Make sure every student has a journal.

Links

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

For the Students

- **Variables** - Student Video (**download**)
- **Think Spot Journal** - Reflection Journal

Vocabulary

- **Variable** - A placeholder for a piece of information that can change.

Teaching Guide

Warm Up (5 min)

Introduction

It might be helpful to remind students of "Variables that Change in Bee," since variables will be used in a similar way here.

- How can we change the value of a variable inside of a loop?
- Do we have to change the value of a variable only by one each time?

Main Activity (20 min)

Online Puzzles

The latter half of this series is made up of freeplay puzzles. Students will have the opportunity to play with variables, shape, and color to create something unique.

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?
- Have you tried mixing multiple variables into one program? What might that look like? When would it be helpful?

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

▸ **AP** - Algorithms & Programming

Lesson 22: For Loops with Bee

Overview

Featuring Bee, this lesson focuses on `for` loops and using an incrementing variable to solve more complicated puzzles. Students will begin by reviewing loops from previous lessons, then they'll walk through an introduction to `for` loops so they can more effectively solve complicated problems.

Purpose

Today's concept, `for` loops, are a very important topic in computer science. Not only are they widely used, the process of learning `for` loops enhances the learning of other important concepts (such as variables and parameters.) Students will have plenty of practice critically thinking through problems by determining the starting, ending, and stepping values for each `for` loop. This concept uses plenty of math as well, so feel free to pair it with a math lesson for an even deeper learning experience.

Agenda

Bridging Activity - For Loops (15 min)

Previewing Online Puzzles as a Class

Main Activity (30 min)

Online Puzzles

Wrap Up (15 min)

Journaling

View on Code Studio

Objectives

Students will be able to:

- Determine starting value, stopping value, and stepping value for a `for` loop.
- Recognize when to use a `for` loop and when to use other loops such as `repeat` and `while` loops.

Preparation

- [ ] Play through the lesson to find any potential problem areas for your class.
- [ ] Make sure every student has a journal.

Links

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

For the Students

- **Think Spot Journal** - Reflection Journal

Vocabulary

- **For Loop** - Loops that have a predetermined beginning, end, and increment (step interval).

Teaching Guide

Bridging Activity - For Loops (15 min)

Previewing Online Puzzles as a Class

Display a puzzle from the lesson. We recommend puzzle #4 because it displays a potential solution and asks the user to evaluate it.

Using a number line, mark the start and ending values of the given `for` loop (if you aren't using puzzle #4, you will need to come up with a potential solution first). With the class's help, circle the values between the start and end that the `for` loop will touch. If you are working on puzzle #4, ask the class what they think the answer is to the question, given what they found with the number line.

Main Activity (30 min)

Online Puzzles

Some students may have a hard time differentiating between `repeat` loops and `for` loops. We recommend having scratch paper out for students to make guesses on values like the start, stop, and step. Implementing pair programming amongst the class might also be helpful for your students.

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?
- How is a `for` loop different from a `repeat` loop?
- Why do you think `for` loops could be useful?

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

- ▸ **AP** - Algorithms & Programming

Lesson 23: For Loops with Artist

Overview

In this lesson, students continue to practice for loops, but this time with Artist. Students will complete puzzles combining the ideas of variables, loops, and for loops to create complex designs. At the end, they will have a chance to create their own art in a freeplay level.

Purpose

Creativity and critical thinking come together beautifully in this lesson. Students will continue their practice with for loops and variables while they create jaw-dropping images. This lesson inspires a creative mind while teaching core concepts to computer science.

Agenda

Warm Up (15 min)
- **Introduction**

Main Activity (30 min)
- **Online Puzzles**

Wrap Up (15 min)
- **Journaling**

View on Code Studio

Objectives

Students will be able to:

- Use `for` loops to change loop several times with different values.
- Recognize when to use a `for` loop and when to use other loops such as `repeat` and `while` loops.

Preparation

- [ ] Play through the lesson to find and potential problem areas for your class.
- [ ] Make sure every student has a journal.

Links

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

For the Students

- **Think Spot Journal** - Reflection Journal

Vocabulary

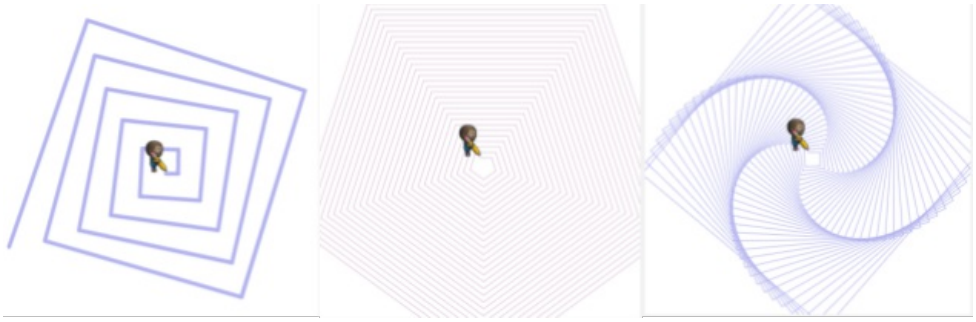
- **For Loop** - Loops that have a predetermined beginning, end, and increment (step interval).

Teaching Guide

Warm Up (15 min)

Introduction

On a board displayed to the entire class, draw (or display via projector) one of the final projects from the lesson. We recommend one of the following:



Ask the class how a computer might draw the drawing you displayed.

After a few predictions have been said, reply with `for` loops of course!

Tell the students they will soon be learning how to create these fine drawings using `for` loops and variables.

Main Activity (30 min)

Online Puzzles

These puzzles are super fun, but it may be helpful for students to have protractors and scratch paper to see these designs made in the physical form. If that isn't an option in your class, try to get the students to trace on the computer screen with their fingers.

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?
- Draw one of the designs you made today. What was the code needed to create it?
- What are some designs you would like to create? How do you think `for` loops or variables could help create those?

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

- ▸ **AP** - Algorithms & Programming

Lesson 24: Swimming Fish in Sprite Lab

Overview

In this lesson, students will learn about the two concepts at the heart of Sprite Lab: sprites and behaviors. Sprites are characters or objects on the screen that students can move, change, and manipulate. Behaviors are actions that sprites will take continuously until they are stopped.

Purpose

This lesson is designed to introduce students to the core vocabulary of Sprite Lab, and allow them to apply concepts they learned in other environments to this tool. By creating a fish tank, students will begin to form an understanding of the programming model of this tool, and explore ways they can use it to express themselves.

Agenda

Warm Up (10 min)

Introduction
Teaching this course as a class?

Main Activity (20 min)

Online Puzzles

Wrap Up (15 min)

Journaling

View on Code Studio

Objectives

Students will be able to:

- Define “sprite” as a character or object on the screen that can be moved and changed.
- Create a new sprite and choose its appearance.

Preparation

- [ ] Play through the lesson to find any potential problem areas for your class.
- [ ] Make sure ever student has a journal.

Links

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

For the Students

- **Think Spot Journal** - Reflection Journal

Vocabulary

- **Behavior** - An action that a sprite performs continuously until it’s told to stop.
- **Sprite** - A graphic character on the screen with properties that describe its location, movement, and look.

Teaching Guide

Warm Up (10 min)

Introduction

Today students will learn how to work with sprites in Sprite Lab.

Display: Pull up a previous puzzle from Code.org, ideally one containing a "main character" like Scrat from Ice Age or one of the Angry Birds.

Discuss: Let the students know that this character on the screen is a "sprite." It is a graphic that is controlled by a program. In this lesson, students will have the opportunity to choose their own sprites to control.

Content Corner

Teaching this course as a class?

Our grade-aligned CS Fundamentals courses use unplugged lessons to build community and introduce tricky computer science concepts, including **sprite behaviors**. Check out the lesson **Simon Says** from **Course E**!

Display: Begin by showing Puzzle 1 to your students.

Think/Pair: Ask them to predict what will happen when the code is run, and to discuss with their neighbors. Run the code, and discuss the outcome.

Now is a great time to introduce the lesson vocabulary.

Main Activity (20 min)

Goal: Today, students will be creating their own Fish Tank. They'll begin by learning how to put some sprites on the screen, then they will make them move. Finally, they'll customize their fish tank to add whatever creatures and objects they want.

Online Puzzles

Transition: Move students to their machines. Encourage students to follow the instructions for each puzzle. Help them realize that this is a creative activity, intended to help them learn Sprite Lab. It is not an assessment activity of any sort.

Reminder: If puzzles are sharable, 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**.

Teaching Tip

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. Have students describe the problem that they're seeing:

- What is it supposed to do?
- What does it do?
- What does that tell you?

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?
- How did it feel to make a scene that was more creative?

- Was it difficult to finish a lesson where there was no clear "right" and "wrong"?

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

- ▸ **AP** - Algorithms & Programming

Lesson 25: Alien Dance Party

Overview

This lesson features Sprite Lab, a platform where students can create their own alien dance party with interactions between characters and user input. Students will work with events to create game controls.

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**

Main Activity (30 min)
- **Online Puzzles**

Wrap Up (15 min)
- **Journaling**

View on Code Studio

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 lesson to find any potential problem areas for your class.
- [ ] Make sure ever student has a journal.

Links

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

For the Students

- **Think Spot Journal** - Reflection Journal

Vocabulary

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

Teaching Guide

Warm Up (15 min)

Introduction

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

Demo: 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 something like "when I say 'raise your hands,' you raise your hands".

You can also think of cities as declaring events. There are laws that say "when there is a green light, cars move through the intersection".

Discuss: 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 clicking the screen or two characters running into each other.

Display: Begin by showing Puzzle 1 to your students.

Think/Pair: Ask them to predict what will happen when the code is run, and to discuss with their neighbors. Run the code, and discuss the outcome.

Main Activity (30 min)

Goal: Today, students will be creating their own alien dance party! They'll begin by reviewing how to put sprites on the screen, then they will assign them behaviors and learn to change those behaviors when an event is initiated.

Online Puzzles

Transition: Move students to their machines. Encourage students to follow the instructions for each puzzle. Help them realize that this is a creative activity, intended to help them learn Sprite Lab. It is not an assessment activity of any sort.

Reminder: If puzzles are sharable, 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**.

Teaching Tip

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. Have students describe the problem that they're seeing:

- What is it supposed to do?
- What does it do?
- What does that tell you?

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?
- How did it feel to have control over what your characters were able to do?
- Did you change the program in any way to make it feel more like your own?

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

- ▶ **AP** - Algorithms & Programming

Lesson 26: Behaviors in Sprite Lab

Overview

Here, students will use Sprite Lab to create their own customized behaviors.

Purpose

Students will use events to make characters move around the screen, change size, and change colors based on user input. This lesson offers a great introduction to events in programming and even gives a chance to show creativity!

Agenda

Warm Up (15 min)
- **Introduction**

Main Activity (30 min)
- **Online Puzzles**

Wrap Up (15 min)
- **Journaling**

View on Code Studio

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 puzzles to find any potential problem areas for your class.
- [ ] Make sure every student has a journal.

Links

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

For the Students

- **Think Spot Journal** - Reflection Journal

Teaching Guide

Warm Up (15 min)

Introduction

Today students will revisit **sprite behaviors**, this time learning how to edit behaviors directly and even make new ones.

Review: Ask students questions about the Swimming Fish and Alien Dance Party lessons.

- What are some of the behaviors we can assign to our sprites?
- What do you imagine the code **inside** a behavior might look like?

Display: Begin by showing Puzzle 1 to your students.

Think/Pair: Ask them to predict what will happen when the code is run, and to discuss with their neighbors. Be sure to open the behavior editor by clicking "edit" on the mystery behavior block. Run the code, and discuss the outcome.

Discuss Ask students questions about how they might change this behavior's code to create a different effect.

- What would happen if you change the -1 to another number? -5? Positive 1? 0?
- What other properties might we be able to change about a sprite besides its size?

Main Activity (30 min)

Goal: Today, students will be editing and creating their own behaviors! They'll begin by making small changes to some familiar but new behaviors and gradually move towards writing their own behaviors from scratch.

Online Puzzles

Transition: Move students to their machines. Encourage students to follow the instructions for each puzzle. Help them realize that this is a creative activity, intended to help them learn Sprite Lab. It is not an assessment activity of any sort.

Reminder: If puzzles are sharable, 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**.

Teaching Tip

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. Have students describe the problem that they're seeing:

- What is it supposed to do?
- What does it do?
- What does that tell you?

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 other options would you like to be able to have your pet do?

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

- **AP** - Algorithms & Programming

Lesson 27: Virtual Pet with Sprite Lab

Overview

In this lesson, students will create an interactive Virtual Pet that looks and behaves how they wish. Students will use Sprite Lab's "Costumes" tool to customize their pet's appearance. They will then use events, behaviors, and other concepts they have learned to give their pet a life of its own!

Purpose

This lesson allows students to apply programming concepts from prior lessons in a more creative context. For example, students will use a variable to store their pet's "happiness" as an integer, which should help them understand how variables can be used in other applications. Last, completing this lesson should prepare students to engage with the open-ended final project.

Agenda

Warm Up (15 min)
- **Introduction**

Main Activity (30 min)
- **Online Puzzles**

Wrap Up (15 min)
- **Journaling**

View on Code Studio

Objectives

Students will be able to:

- Create an interactive virtual pet using events, behaviors, variables, and custom art.
- Program solutions to problems that arise when designing a virtual pet, like feeding it or monitoring its happiness.

Preparation

- [ ] Play through the puzzles to find any potential problem areas for your class.
- [ ] Make sure every student has a journal.

Links

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

For the Students

- **Think Spot Journal** - Reflection Journal

Teaching Guide

Warm Up (15 min)

Introduction

Revisit **events** and **behaviors** in programming. Additionally, introduce the Sprite Lab "Costumes" tool that allows students to draw their own costumes.

Review: Ask students questions about events and behaviors. Show them the Alien Dance Party mini-project (from the **Events with Sprite Lab** lesson) as an example.

- Do you remember what an event is?
- Can you name any of the events that you used to make the aliens dance? What do they do?
  - when clicked
  - when blue alien touches pink alien
  - when arrow pressed
- Do you remember what a behavior is?
- Can you remember some of the behaviors you used to make the aliens dance? What do they do?
  - patrolling
  - jittering
  - spinning right/left

Display: Begin by showing Puzzle 1 of today's lesson to your students.

Think/Pair: Ask students to predict what will happen when the code is run, and to discuss with their neighbors. Run the code, and discuss the outcome.

Display: Show Puzzle 2. Briefly demonstrate how to do the following:

- Navigate between the **Code** and **Costumes** tabs.
- Draw a costume.
- Choose a costume from the costume library.
- Change the virtual pet's sprite's costume to a custom one.

Main Activity (30 min)

Goal: Today, students will be creating their own virtual pet! They will begin by drawing or selecting a new costume for a sprite. Then they will create events that cause actions and behaviors upon interaction.

Online Puzzles

Transition: Move students to their machines. Encourage students to follow the instructions for each puzzle. Help them realize that this is a creative activity, intended to help them learn Sprite Lab. It is not an assessment activity of any sort.

Reminder: If puzzles are sharable, 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**.

Teaching Tip

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. Have students describe the problem that they're seeing:

- What is it supposed to do?
- What does it do?
- What does that tell you?

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 other options would you like to be able to have your pet do?

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

▶ **AP** - Algorithms & Programming

Lesson 28: End of Course Project

Overview

The next five 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. This is intended to be a multi-lesson or multi-week project where students spend time brainstorming, learning about the design process, building, and then presenting their final work.

In the "Explore" stage, students will play around with pre-built Artist and Sprite Lab programs 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, Sprite Lab, or any other interface that you are comfortable providing. (This is likely the longest stage of the project.) Students will then revise their code after testing and peer review. Finally, students will be able to present their finished work to their classmates.

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. Provide 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

Explore Project Ideas (45 min)

Example Projects

Build Your Project (45 min)

Extension Activity

View on Code Studio

Objectives

Students will be able to:

- Learn to plan in advance for an ongoing assignment.
- Be able to explain how system limitations can affect project design.
- Describe how compromise can help keep a project on track and inspire creativity.
- Draft and implement plans to resolve any issues in their code.
- Articulate the design process and how it helped shape the finished culminating project.

Preparation

- [ ] Spend time making your own project with both the Artist and Sprite Lab. Familiarize yourself with the capabilities and limitations of each tool.
- [ ] Modify the rubric to fit your class goals and print out a copy for each student.
- [ ] Modify the 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 Teachers

- **Design Process** - Teacher Prep Guide **Make a Copy**
- **Final Project Design** - Worksheet **Make a Copy**
- **CS Fundamentals Final Project** - Rubric

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

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 F 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.

Build Your Project (45 min)

Goal: Students will build a project.

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 their original ideas (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.

Extension Activity

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

Standards Alignment

CSTA K-12 Computer Science Standards (2017)

- **AP** - Algorithms & Programming