

2022

# Elementary Integration Guide

FIRST GRADE



MISSISSIPPI STATE UNIVERSITY™  
CENTER FOR CYBER EDUCATION

## **Acknowledgements**

The following people assisted in the development of this integration guide:

### **Elementary Task Force**

Melissa Atkins, Instruction Technologist, Lamar County School District  
Kacy Baggett, Teacher, Rankin County School District  
Heather Barry-Fenster, Teacher, Lowndes County School District  
Brittany Boatman, Teacher, Houston School District  
Kimberly Brammer, Teacher, Pascagoula-Gautier School District  
Michelle Carter, Teacher, Picayune School District  
Jana Chao, Teacher, Clinton Public School District  
Lerenda Dixon, Teacher, McComb School District  
Samantha Elizondo, Teacher, Tupelo Public School District  
Angie Frazier, Teacher, Rankin County School District  
Tammy Hale, Teacher, Tate County School District  
Kayla Hathcock, Teacher, Amory School District  
Ashley Hawkins, Teacher, Biloxi Public School District  
Vicky Johnson, Teacher, Franklin County School District  
Dinah Lachney, Teacher, Madison County School District  
Ashley Matthews, Administrator, Lowndes County School District  
Kayla Moore, Teacher, Enterprise School District  
Olivia Moore, Teacher, DeSoto County School District  
Beth Neese, Teacher, Western Line School District  
Hannah Padgett, Teacher, Tate County School District  
Dr. Lee Pambianchi, Administrator, Rankin County School District  
Kristen Phillips, Teacher, Oxford School District  
Brittney Price, Teacher, South Panola School District  
Melissa Sundberg, Teacher, Ocean Springs School District  
Melissa Tingle, Teacher, Lauderdale School District  
Susie Williams, Curriculum Coordinator, Leland School District

### **Mississippi Department of Education Team**

Wendy Clemons, executive director, Office of Secondary Education  
Louella Webster, supervisor for computer science/STEM

### **Center for Cyber Education Team**

Shelly Hollis, Director  
Lizzie Brandon, Project Manager  
Amanda Taylor, Project Manager

### **Research and Curriculum Unit Team**

Brock Turnipseed, Marketing and Communications Manager  
Chris McMillen, Communications Coordinator  
Heather Craig, Editor  
Will Graves, Project Coordinator

Funding for the development of this guide was provided by:



## Introduction

In March 2021, The Mississippi Computer Science and Cyber Education Equality Act ([House Bill 633](#)) was passed requiring all districts to offer computer science content and courses by the 2024-2025 school year. The bill allows for a phased-in approach as listed below:

- 2022-2023: All middle schools offer at least one (1) course in computer science, and 50% of elementary schools offer a minimum of one (1) hour of instruction in computer science each week at each grade level.
- 2023-2024: All elementary schools offer a minimum of one (1) hour of instruction in computer science each week at each grade level, and 50% of high schools offer at least one (1) course in computer science.
- 2024-2025: All schools will offer instruction in computer science.

To make the integration of computer science content as seamless as possible for elementary teachers, a task force of elementary teachers, principals, the Mississippi Department of Education, and the Mississippi State University Center for Cyber Education was formed to write an integration guide for each grade level, kindergarten through fifth grade. These guides provide plans for a minimum of 40, 60-minute lessons covering six computer science topics: coding, robotics, digital literacy, digital citizenship, keyboarding, and unplugged activities.

Each guide contains a breakdown of content by integrated subjects, content by computer science topics, and a calendar/pacing guide. Teachers may choose to start at the beginning and teach each lesson once a week in chronological order or teach the lesson that integrates with another core subject topic at a more relevant time. In addition to a lesson overview and links to required resources, each lesson plan maps to a Mississippi Computer Science Standard and a core subject area standard. A suggestion on how to break the lesson into smaller segments to be covered throughout the week is also provided in the "Time needed" section.

There are several resources available in each integration guide. Some may require the creation of accounts, but all resources referenced are free. The pacing guide notes lessons requiring account creation so teachers can plan ahead. A list of sites used is provided for technology departments to whitelist or unblock. All resources may be used on any internet-capable device, including Chromebooks and tablets.

<b>Resources</b>	
Computing resources	<ul style="list-style-type: none"> <li>● <a href="#">Code.org</a> CS Fundamentals               <ul style="list-style-type: none"> <li>○ <a href="#">1st Grade: Course B</a></li> </ul> </li> <li>● <a href="#">Common Sense Digital Media</a></li> <li>● <a href="#">Kodable</a></li> <li>● <a href="#">Scratch, Jr.</a></li> </ul>
CS4MS website materials	<ul style="list-style-type: none"> <li>● <a href="#">2018 Mississippi Computer Science Standards</a></li> <li>● <a href="#">CS4MS Website</a></li> </ul>
Mouse practice	<ul style="list-style-type: none"> <li>● <a href="#">Alphabetical Order</a></li> <li>● <a href="#">Mouse Practice</a> <ul style="list-style-type: none"> <li>○ Apple Catch</li> <li>○ Coyote Concentration (card matching game)</li> <li>○ Desert Dive</li> <li>○ Frost Bite</li> <li>○ Helipopper</li> <li>○ Penguin Drop</li> <li>○ Pickle Pop</li> <li>○ Pig Pile</li> <li>○ Simon Sees</li> </ul> </li> </ul>
Keyboard practice	<p>Online:</p> <ul style="list-style-type: none"> <li>● <a href="#">Astro Bubbles Keyboard Practice</a></li> <li>● <a href="#">Big Brown Bear</a></li> <li>● <a href="#">Read Today</a></li> </ul> <p>Unplugged:</p> <ul style="list-style-type: none"> <li>● <a href="#">Keyboard Callout:</a> <ul style="list-style-type: none"> <li>○ <b>Paper keyboard:</b> Using a paper keyboard, the teacher will call out letters, numbers, symbols, and/or words for students to "type" on their keyboard.</li> <li>○ <b>Computer with no internet:</b> The teacher will call out letters, numbers, symbols, and/or words. Students will use their keyboard to type into a blank document on their computer/tablet.</li> </ul> </li> <li>● <a href="#">Keyboard Bingo</a> <ul style="list-style-type: none"> <li>○ Preparation: The teacher will print squares with letters, numbers, and symbols (4-5 of each letter, 1-2 of each number/symbol). The teacher will cut out and laminate each square, then use a piece of tape or glue to adhere a magnet to each square.</li> <li>○ The teacher will project a keyboard onto a smartboard.</li> <li>○ The teacher will call out letters, numbers, symbols, or words for students to find using their preprinted squares.</li> <li>○ Students will raise their hands if they have the key that the teacher calls out. The teacher will choose a student to place their key on the board.</li> </ul> </li> </ul>
Teacher / student accounts	<ul style="list-style-type: none"> <li>● <a href="#">Code.org</a></li> <li>● <a href="#">Common Sense Digital Media</a></li> <li>● <a href="#">Kodable</a></li> <li>● <a href="#">Scratch, Jr.</a></li> </ul>
For help with this guide	<ul style="list-style-type: none"> <li>● Contact Mississippi State University's Center for Cyber Education: <a href="http://www.tinyurl.com/ccehelpdesk">www.tinyurl.com/ccehelpdesk</a></li> </ul>



# Contents by Integrated Subjects

## English

- Week 1: RF.1.1—Features of print
- Week 2: W.1.8, SL.1.1, SL.1.2, SL.1.3—Recall, Collaborative conversations, Oral questioning, Clarification
- Week 3: SL.1.1, SL.1.2, SL.1.3, SL.1.5—Collaborative conversations, Oral questioning, Clarification, Add visual displays
- Week 4: RF.1.2—Understand words, symbols, and sounds
- Week 5: SL.1.1, SL.1.5—Collaborative conversations, Add visual displays
- Week 6: SL.1.1, SL.1.2, SL.1.3, SL.1.5—Collaborative conversations, Oral questioning, Clarification, Add visual displays
- Week 7: SL.1.1, SL.1.2, SL.1.3, SL.1.5—Collaborative conversations, Oral questioning, Clarification, Add visual displays
- Week 9: SL.1.1, SL.1.2, SL.1.3, SL.1.5—Collaborative conversations, Oral questioning, Clarification, Add visual displays
- Week 11: SL.1.1, SL.1.2, SL.1.3, SL.1.5—Collaborative conversations, Oral questioning, Clarification, Add visual displays
- Week 12: SL.1.1, SL.1.2, SL.1.3, SL.1.5—Collaborative conversations, Oral questioning, Clarification, Add visual displays
- Week 13: SL.1.1—Collaborative conversations
- Week 15: L.1.5—Word categorization
- Week 16: SL.1.1—Collaborative conversations
- Week 17: SL.1.1, SL.1.5, SL.1.1—Collaborative Conversations, Add visual displays
- Week 18: W.1.8, SL.1.1, SL.1.2, SL.1.3, SL.1.5—Recall, Collaborative conversations, Oral questioning, Clarification, Add visual displays
- Week 19: SL.1.1, SL.1.2, SL.1.3—Collaborative conversations, Oral questioning, Clarification
- Week 20: SL.1.1, SL.1.4—Collaborative conversations, Description
- Week 21: SL.1.1, SL.1.2, SL.1.3—Collaborative conversations, Oral questioning, Clarification
- Week 22: RL.1.3, SL.1.1, SL.1.2, SL.1.3—Connections, Collaborative conversations, Oral questioning, Clarification
- Week 23: R.L.1.4—Ask/answer questions for clarification
- Week 24: SL.1.1, SL.1.2, SL.1.3—Collaborative conversations, Oral questioning, Clarification
- Week 25: SL.1.1, SL.1.2—Collaborative Conversations, Oral questioning
- Week 26: SL.1.1, SL.1.6, L.1.6—Collaborative conversations, Complete sentences, Words/phrases in conversation
- Week 28: RF.1.1, SL.1.1, SL.1.2, SL.1.3, SL.1.5—Features of print, Collaborative Conversations, Oral questioning, Clarification, Add visual displays
- Week 29: SL.1.1, SL.1.2, SL.1.3, SL.1.5—Collaborative Conversations, Oral questioning, Clarification, Add visual displays
- Week 30: SL.1.1, SL.1.2, SL.1.3, SL.1.5—Collaborative Conversations, Oral questioning, Clarification, Add visual displays
- Week 31: SL.1.1, SL.1.2, SL.1.3, SL.1.5—Collaborative Conversations, Oral questioning, Clarification, Add visual displays
- Week 32: L.1.1—Simple/compound sentences
- Week 33: L.1.1, RF.1.3, SL.1.1, SL.1.2, SL.1.3—Simple/compound sentences, Phonics, Collaborative Conversations, Oral questioning, Clarification
- Week 34: SL.1.1, SL.1.2, SL.1.2—Collaborative Conversations, Oral questioning, Clarification
- Week 36: RF.1.1—Features of print
- Week 37: L.1.1, SL.1.1, SL.1.2, SL.1.3—Simple/compound sentences, Collaborative conversation, Oral questioning, Clarification
- Week 38: SL.1.1, SL.1.2, SL.1.3—Collaborative conversation, Oral questioning, Clarification
- Week 39: SL.1.1, SL.1.2, SL.1.3—Collaborative conversation, Oral questioning, Clarification
- Week 40: RF.1.3, SL.1.1, SL.1.2, SL.1.3, SL.1.5—Phonics, Collaborative conversation, Oral questioning, Clarification, Add visual displays

## **Math**

- Week 4: 1.NBT.1—Count to 120
- Week 7: 1.MD.4—Organize, represent, and interpret data up to 3 categories
- Week 8: 1.MD.4—Organize, represent, and interpret data up to 3 categories
- Week 10: 1.MD.4—Organize, represent, and interpret data up to 3 categories
- Week 12: 1.NBT.1—Count to 120
- Week 14: 1.NBT.1—Count to 120
- Week 15: 1.OA.1, 1.OA.2—Add/subtract within 20, Solve word problems adding 3 whole numbers
- Week 16: 1.G.2—2D/3D shapes
- Week 17: 1.G.2—2D/3D shapes
- Week 20: 1.G.1—Defining/non-defining attributes of shapes
- Week 23: 1.OA.1—Add/subtract within 20
- Week 25: 1.GA.3—Partition circles and rectangles
- Week 27: 1.NBT.1—Count to 120
- Week 29: 1.G.3—Partition circles and rectangles
- Week 30: 1.OA.2—Solve word problems adding 3 whole numbers
- Week 31: 1.OA.1—Add/subtract within 20
- Week 34: 1.MD.4—Organize, represent, and interpret data up to 3 categories
- Week 35: 1.MD.5—U.S. coins
- Week 36: 1.MD.4—Organize, represent, and interpret data up to 3 categories
- Week 38: 1.OA.5—Add/subtract within 20
- Week 39: 1.G.1, 1.G.2, 1.G.3—Defining/non-defining attributes of shapes, 2D/3D shapes, Partition circles and rectangles

## **Science**

- Week 13: L.1.1—Explain observations using drawings, writings, or model
- Week 20: L.1.2—Obtain information from text and other media to describe the function of plant parts.
- Week 27: E.1.9a—Weather
- Week 28: E.1.10—Human impact on water resources

## **Social Studies**

- Week 6: G.1.3—Directions
- Week 7: G.1.3—Directions
- Week 9: G.1.3—Directions
- Week 10: G.1.3—Directions
- Week 21: G.1.3—Directions
- Week 25: G.1.3—Directions
- Week 26: G.1.3—Directions
- Week 35: E.1.3—Needs vs. wants

# Contents by Topics

## Coding

- Week 7
- Week 9
- Week 10
- Week 11
- Week 12
- Week 14
- Week 15
- Week 16
- Week 17
- Week 21
- Week 25
- Week 26
- Week 27
- Week 28
- Week 29
- Week 30
- Week 31
- Week 36
- Week 37
- Week 38

## Digital Citizenship

- Week 3
- Week 5
- Week 32

## Digital Literacy

- Week 1
- Week 8
- Week 13
- Week 18
- Week 23
- Week 34
- Week 35

## Keyboarding

- Week 2
- Week 4

## Robotics

- Week 24
- Week 33
- Week 39
- Week 40

## Unplugged

- Week 6
- Week 8
- Week 18
- Week 19
- Week 20
- Week 22
- Week 24
- Week 30
- Week 32
- Week 33
- Week 34
- Week 35
- Week 36
- Week 37
- Week 38
- Week 39
- Week 40

# Calendar/Pacing per week:

→ Teachers will need to create a FREE **teacher and/or student account** (see notes section of lesson.)

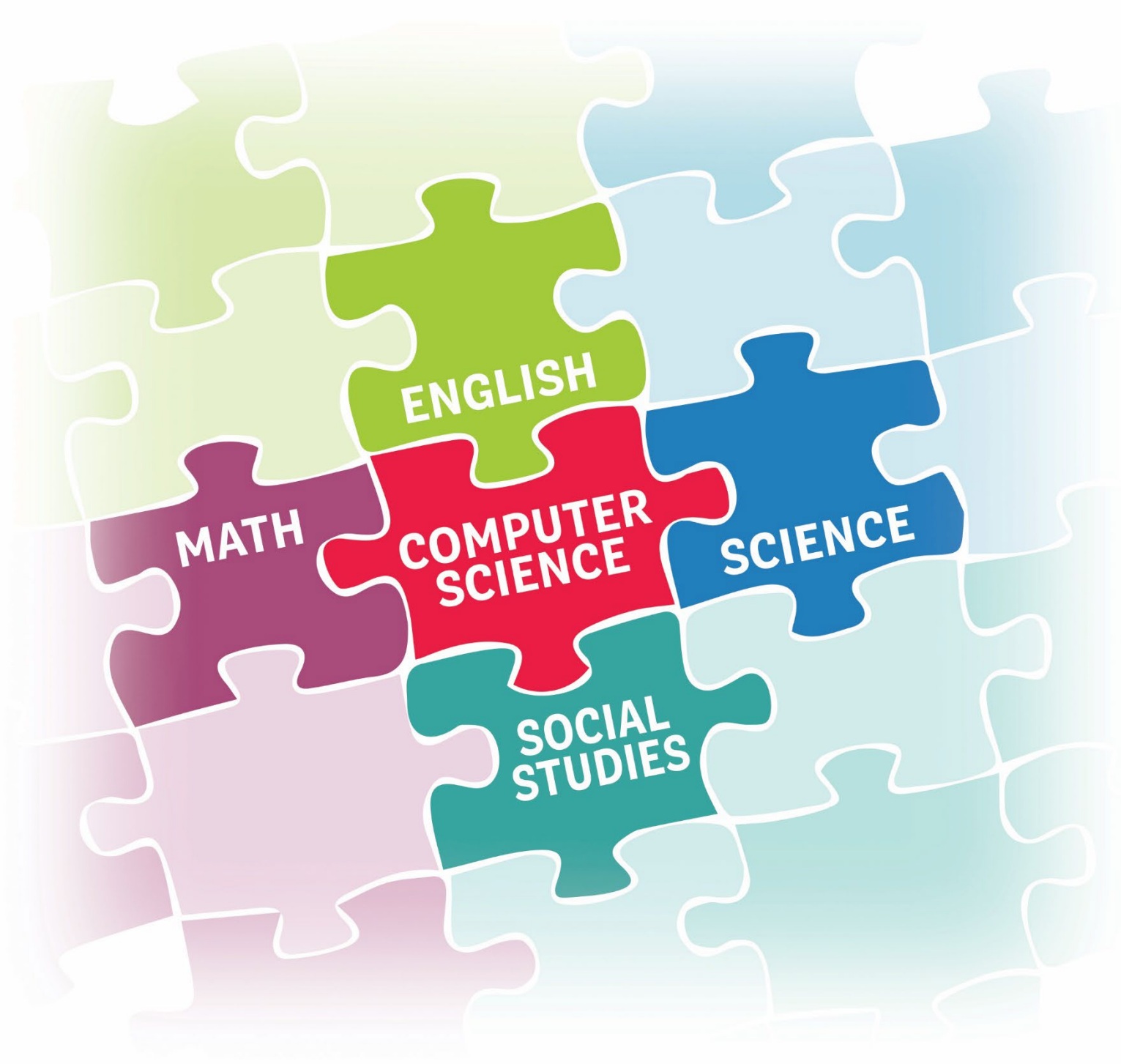
Week	Title	Topics	CS Standard	Subject Integrated	Integrated Standards
1	What is a Computer?	Digital Literacy	CS.1A.2	ELA	RF.1.1
2	Introduction to Mouse and Keyboard	Keyboarding	CS.1B.1	ELA	W.1.8 SL.1.1 SL.1.2 SL.1.3
3	Pause and Think Online → Account creation needed	Digital Citizenship	1C.1A.2 1C.1A.3	ELA	SL.1.1 SL.1.2 SL.1.3 SL.1.5
4	Code.org: Learn to Drag and Drop → Account creation needed	Keyboarding	CS.1A.2 CS.1A.3 AP.1A.4 IC.1A.2 IC.1A.3	ELA Math	RF.1.2 1.NBT.1
5	Code.org: Digital Trails	Digital Citizenship	IC.1A.2 IC.1A.3 NI.1A.1	ELA	SL.1.2 SL.1.5
6	Code.org: Move It, Move It	Unplugged	AP.1A.1 AP.1A.3 AP.1A.4	ELA Social Studies	SL.1.1 SL.1.2 SL.1.3 SL.1.5 G.1.3
7	Code.org: Sequencing With Angry Birds	Coding	AP.1A.4	ELA Math Social Studies	SL.1.1 SL.1.2 SL.1.3 SL.1.5 1.MD.4 G.1.3
8	Code.org: Interpreting Sticky Data	Unplugged Digital Literacy	AP.1A.4	Math	1.MD.4
9	Code.org: Programming With Angry Birds	Coding	AP.1A.2 AP.1A.4	ELA Social Studies	SL.1.1 SL.1.2 SL.1.3 SL.1.5 G.1.3
10	Code.org: Programming With Harvester	Coding	AP.1A.2 AP.1A.4	Math Social Studies	1.MD.4 G.1.3
11	Code.org: Getting Loopy	Coding	AP.1A.2 AP.1A.3	ELA	SL.1.1 SL.1.2

			AP.1A.4 AP.1A.7		SL.1.3 SL.1.5
12	Code.org: Loops With Harvester	Coding	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.7	ELA Math	SL.1.1 SL.1.2 SL.1.3 SL.1.5 1.NBT.1
13	Code.org: Corn Crazy Choice Board	Digital Literacy	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.7	ELA Science	SL.1.1 L.1.1
14	Code.org: Loops With Laurel	Coding	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.7	Math	1.NBT.1
15	Code.org: Treasure Tracker	Coding	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.7	ELA Math	L.1.1.5a 1.OA.1 1.OA.2
16	Code.org: Drawing Gardens With Loops	Coding	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.7	ELA Math	SL.1.1 SL.1.5 1.G.2
17	Code.org: Loopy Shapes	Coding	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.7	ELA Math	SL.1.1 SL.1.5 1.G.2
18	Code.org: The Right App	Unplugged Digital Literacy	CS.1A.1 IC.1A.1	ELA	W.1.8 SL.1.1 SL.1.2 SL.1.3 SL.1.5
19	Code.org: The Big Event Jr.	Unplugged	AP.1A.2 AP.1A.4	ELA	SL.1.1 SL.1.2 SL.1.3
20	Code.org: Zookeeper Survival Events	Unplugged	AP.1A.2 AP.1A.4	ELA Math Science	SL.1.1 SL.1.4 1.G.1 L.1.2
21	Scratch Jr.: Can I Make My Car Drive Across the City?	Coding	AP.1A.3 DA.1A.1	ELA Social Studies	SL.1.1 SL.1.2 SL.1.3 G.1.3
22	Kodable: Coding Basics Unplugged → Account creation needed	Unplugged	AP.1A.1 AP.1A.3 AP.1A.4 AP.1A.7	ELA	RL.1.3 SL.1.1 SL.1.2 SL.1.3

23	Mystery Island Coding Quest → Account creation needed	Digital Literacy	AP.1A.1 AP.1A.3 AP.1A.4 AP.1A.7	ELA Math	RI.1.4 SL.1.1 SL.1.2 SL.1.3 1.OA.1
24	Kodable: Hour of Code: Beginner	Unplugged Robotics	IC.1A.1 AP.1A.8	ELA	SL.1.1 SL.1.2 SL.1.3
25	Kodable: Maze Maker	Coding	AP.1A.5	ELA Social Studies	SL.1.1 SL.1.2 G.1.3
26	Kodable: Show What You Know!	Coding	AP.1A.3 AP.1A.4	ELA Social Studies	SL.1.1 SL.1.6 L.1.6 G.1.3
27	Kodable: If Flash, Then Clap	Coding	AP.1A.4	Math Science	NBT.1 E.1.9a
28	Kodable: Beach Cleanup	Coding	AP.1A.3 AP.1A.4 AP.1A.7	ELA Science	RF. 1.1 SL.1.1 SL.1.2 SL.1.3 SL.1.5 E.1.10
29	Kodable: Sequence 1— Introduction	Coding	AP.1A.1	ELA Math	SL.1.1 SL.1.2 SL.1.3 SL.1.5 1.G.3
30	Kodable: Pizza Party!	Unplugged Coding	AP.1A.1 AP.1A.4 AP.1A.5	ELA Math	SL.1.1 SL.1.2 SL.1.3 SL.1.5 1.OA.2
31	Kodable: Hour of Code	Coding	AP.1A.3 AP.1A.4 AP.1A.8 AP.1A.7	ELA Math	SL.1.1 SL.1.2 SL.1.3 SL.1.5 1.OA.1
32	How Does Technology Make You Feel?	Unplugged Digital Citizenship	IC.1A.1 IC.1A.2 IC.1A.3	ELA	L.1.1
33	Unplugged Robotic Activity	Unplugged Robotics	AP.1A.1 AP.1A.5 AP.1A.7	ELA	L.1.1 RF.3 SL.1.1 SL.1.2 SL.1.3
34	Candy Data/Unplugged	Unplugged	DA.1A.3	ELA	SL.1.1




		Digital Literacy		Math	SL.1.2 SL.1.3 1.MD.4
35	Design Your Own Computer	Unplugged Digital Literacy	CS.1A.2	Math	1.MD.5
36	The Kodable World	Unplugged Coding	AB.1A.3A	ELA Math	RF.1.1 1.MD.4
37	Debugging ELA/Unplugged	Unplugged Coding	AP.1A.2 AP.1A.4	ELA	L.1.1 SL.1.1 SL.1.2 SL.1.3
38	Debugging Math/Unplugged	Unplugged Coding	AP.1A.2 AP.1A.4	ELA Math	SL.1.1 SL.1.2 SL.1.3 1.OA.5
39	Geometry Robot Review	Unplugged Robotics	AP.1A.4 AP.1A.5	Math	1.G.1 1.G.2 1.G.3
40	Spelling Robot Review	Unplugged Robotics	AP.1A.4 AP.1A.5	ELA	RF.1.3 SL.1.1 SL.1.2 SL.1.3 SL.1.5



# Lessons and Activities


FIRST GRADE

## Week 1: What is a Computer?

<p>Lesson overview:</p> 	<p><b>Purpose:</b> In this lesson, your students will learn the basic parts of computers. This is an introduction to technology that they will use throughout school. Students will become familiar with the input and output devices that are associated with a computer. Students will practice basic writing skills by re-writing the letters of key terms. Students will practice matching by drawing a line from the object to its name.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Warm Up             <ul style="list-style-type: none"> <li>○ Ask students “What is a computer?” and let them give you the answers to what they think a computer is.</li> <li>○ Watch the “<a href="#">What is a Computer for Kids?</a>” video.</li> </ul> </li> <li>● Identifying Computer Parts Activity             <ul style="list-style-type: none"> <li>○ <a href="#">Identifying Computer Parts PowerPoint</a>: As you are going through the PowerPoint, students will be finding and matching the pictures of the computer part on the <a href="#">Identifying Computer Parts Worksheet</a>. Once they find the picture, they will draw a line from the picture to the correct term. (Please do not rush through the slides because it allows students to see the picture and term on the board.)</li> <li>○ Once the PowerPoint and matching on the worksheet have been completed, the students will practice their writing skills by rewriting each term on the worksheet.</li> </ul> </li> <li>● Enrichment             <ul style="list-style-type: none"> <li>○ <a href="#">Computer Coloring Page</a>: Students will color the page to reinforce the parts of the computer that they have just learned. (For additional enrichment, the students can rewrite their terms for each part onto the coloring page.)</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">“What is a Computer for Kids?”</a> video</li> <li>● <a href="#">Identifying Computer Parts PowerPoint</a></li> <li>● <a href="#">Identifying Computer Parts Worksheet</a></li> <li>● <a href="#">Computer Coloring Page</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Identify parts of a computer</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>CS.1A.2</b>—Use appropriate terminology in identifying and describing the function of common physical components of computing systems (hardware).</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 62 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up <b>7 min</b> <ul style="list-style-type: none"> <li>○ <i>What is a Computer for Kids?</i> Video</li> </ul> </li> <li>● Activity: <b>30 min</b> <ul style="list-style-type: none"> <li>○ Identifying Computer Parts worksheet and PowerPoint (matching)</li> <li>○ Identifying Computer Parts (re-writing terms)</li> </ul> </li> <li>● Enrichment: <b>25 min</b> <ul style="list-style-type: none"> <li>○ Computer Coloring Page</li> </ul> </li> </ul>
<p>Materials needed:</p>	<p><b>Teacher:</b></p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> </ul>

	<ul style="list-style-type: none"> <li>● <a href="#">Identifying Computer Parts PowerPoint</a></li> </ul> <b>Students:</b> <ul style="list-style-type: none"> <li>● <a href="#">Identifying Computer Parts Worksheet</a></li> <li>● <a href="#">Computer Coloring Page</a></li> <li>● Pencil</li> <li>● Crayons (or other coloring utensil)</li> </ul>
Subject integrated:	ELA
Other standards addressed:	<b>RF.1.1</b> —Demonstrate understanding of the organization and basic features of print.
Vocabulary:	<p><u>Computer</u>: A device for working with information</p> <p><u>Desktop</u>: Screen that appears if you are not browsing the internet, reading a file or playing a game; your icons are on this screen</p> <p><u>Input</u>: To add information/data into the computer</p> <p><u>Keyboard</u>: Where all the letters, numbers, and other buttons are located; when you type on it, the symbols appear on the monitor</p> <p><u>Laptop</u>: Small portable computer</p> <p><u>Monitor</u>: Screen that shows you what you are doing; a viewer that displays what the computer sends to it</p> <p><u>Mouse</u>: A hand-held device for moving a cursor around the screen</p> <p><u>Output</u>: To send information/data out of the computer</p> <p><u>Tablet</u>: A wireless, portable personal computer with a touchscreen interface</p>
Notes:	

## Week 2: Introduction to Mouse and Keyboard

<p>Lesson overview:</p> 	<p><b>Purpose:</b>  <b>Mouse Practice</b>          The computer mouse activities in this lesson are designed to help beginning computer users learn mouse skills through hand-eye coordination by dragging, dropping, clicking, double-clicking, and scrolling.</p> <p><b>Introduction to Keyboards</b>          The computer keyboard activities in this lesson are designed to help beginning computer users learn keyboarding skills through hand-eye coordination by finding letters/numbers/symbols and using a finger to press the key. Teachers, please understand your students will mainly be using the “find and peck” method of typing. That is okay! In this grade, our main concern is making sure that students use their left pointer finger to select keys on the left side of the keyboard and right pointer finger on the right side of the keyboard. We also want students to become familiar with finding letters/numbers/symbols on the QWERTY keyboard layout.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Refresher</li> <li>● Warm Up             <ul style="list-style-type: none"> <li>○ <a href="#">How Do Computers Work?</a></li> </ul> </li> <li>● Mouse Practice             <ul style="list-style-type: none"> <li>○ <a href="#">Using Your Computer Mouse</a></li> <li>○ <a href="#">Mouse Practice (online)</a></li> <li>○ <a href="#">Mouse Practice (unplugged)</a></li> </ul> </li> <li>● Student Voice</li> <li>● <a href="#">Keyboard L-R Coloring Sheet</a> (unplugged)</li> <li>● Keyboarding Practice             <ul style="list-style-type: none"> <li>○ <a href="#">Keyboard Coloring Page</a> (unplugged)</li> <li>○ <a href="#">Astro Bubbles Keyboard Practice</a> (online)</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">Intro to Mouse and Keyboard Lesson Plan</a></li> <li>● <a href="#">How Do Computers Work?</a></li> <li>● <a href="#">Using Your Computer Mouse</a></li> <li>● <a href="#">Mouse Practice (online)</a></li> <li>● <a href="#">Alphabetical Order</a></li> <li>● <a href="#">Mouse Practice (unplugged)</a></li> <li>● <a href="#">Astro Bubbles Keyboard Practice</a></li> <li>● <a href="#">Keyboard L-R Coloring Sheet</a> (unplugged)</li> <li>● <a href="#">Keyboard Coloring Page</a> (unplugged)</li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Operate a mouse/keypad</li> <li>● Find/select letters on a keyboard</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>CS.1B.1</b>—Describe how internal and external parts of computing devices function to form a system.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Refresher <b>15 min</b> <ul style="list-style-type: none"> <li>○ Review</li> <li>○ <a href="#">How Do Computers Work?</a> (video)</li> </ul> </li> <li>● Warm Up <b>2 min</b> <ul style="list-style-type: none"> <li>○ <a href="#">Using Your Computer Mouse</a> (Video)</li> </ul> </li> <li>● Mouse Practice <b>15 min</b> <ul style="list-style-type: none"> <li>○ Online Option</li> </ul> </li> </ul>

	<ul style="list-style-type: none"> <li>○ Unplugged Option</li> <li>● Student Voice <b>3 min</b> <ul style="list-style-type: none"> <li>○ How does a mouse help a computer?</li> </ul> </li> <li>● Keyboard Coloring <b>10 min</b> <ul style="list-style-type: none"> <li>○ Keyboard L-R Coloring Sheet</li> </ul> </li> <li>● Keyboarding Practice <b>15 min</b> <ul style="list-style-type: none"> <li>○ Online Option</li> <li>○ Unplugged Option</li> </ul> </li> </ul>
Materials needed:	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● Identifying Parts of a Computer PowerPoint presentation (for refresher)</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Laptop or desktop computer, or tablet connected to the internet</li> <li>● Mouse Worksheet</li> <li>● Keyboard L-R Coloring Sheet</li> <li>● Keyboard Coloring Page</li> </ul>
Subject integrated:	ELA
Other standards addressed:	<ul style="list-style-type: none"> <li>● <b>W.1.8</b>—With guidance and support, recall info from experiences or gather info from sources.</li> <li>● <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>● <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>● <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> </ul>
Vocabulary:	<p><u>Computer</u>: A device for working with information</p> <p><u>Desktop</u>: Screen that appears if you are not browsing the internet, reading a file or playing a game; your icons are on this screen</p> <p><u>Input</u>: To add information/data into the computer</p> <p><u>Keyboard</u>: Where all the letters, numbers, and other buttons are located; when you type on it, the symbols appear on the monitor</p> <p><u>Laptop</u>: Small portable computer</p> <p><u>Monitor</u>: Screen that shows you what you are doing; a viewer that displays what the computer sends to it</p> <p><u>Mouse</u>: A hand-held device for moving a cursor around the screen</p> <p><u>Output</u>: To send information/data out of the computer</p> <p><u>Tablet</u>: A wireless, portable personal computer with a touchscreen interface</p>
Notes:	



## Week 3: Pause and Think Online

Lesson overview:



**Purpose:**

How can we be safe, responsible, and respectful online?

**Lesson:**

- Pause and Think Online
  - Watch Pause and Think Online
  - Explore: Head to Toe
  - Pause and Think Moment
- Internet Traffic Light
  - Explore: Go! Caution! Stop!
  - Read Internet Traffic Light
  - Play: Traffic Light Game
  - Pause and Think Moment

Lesson links/resources:

- [Pause and Think Online](#)
- [Internet Traffic Light](#)

CS standards addressed:

Students will be able to:

- Understand the importance of being safe, responsible, and respectful online
- Learn the "Pause and Think Online" song to remember basic digital citizenship concepts
- Understand that being safe online is similar to staying safe in real life
- Learn to identify websites and apps that are "just right" and "not right" for them
- Know how to get help from an adult if they are unsure about a website

Standards

- **1C.1A.2**—Work respectfully online with others.
- **1C.1A.3**—Keep login information private; log off devices properly.

Time needed:

**Total Time: 60 min**

- Pause and Think Online **25 min**
  - Watch Pause and Think Online **5 min**
  - Explore: Head to Toe **15 min**
  - Pause and Think Moment **5 min**
- Internet Traffic Light **35 min**
  - Explore: Go! Caution! Stop! **15 min**
  - Read Internet Traffic Light **5 min**
  - Play: Traffic Light Game **10 min**
  - Pause and Think Moment **5 min**

Materials needed:

Teacher:


- Computer
- Projector/smartboard with sound
- [Pause and Think Online Slides](#)
- [Digital Citizen Coloring Book](#)
- [Student Video](#)
- [Common Sense](#) account

Students:

- Computer/tablet with internet access
- [Common Sense](#) account
- [Internet Traffic Light Online Slides](#)
- [Internet Traffic Light Video](#)


	<ul style="list-style-type: none"> <li>• <a href="#">Internet Traffic Light Handout</a></li> <li>• <a href="#">Poem Poster</a></li> <li>• Pencil</li> <li>• Coloring utensils</li> </ul>
Subject integrated:	ELA
Other standards addressed:	<ul style="list-style-type: none"> <li>• <b>SL.1.1</b>—Participate in collaborative conversation</li> <li>• <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>• <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> <li>• <b>SL.1.5</b>—Add drawing or other visual displays to descriptions when appropriate to clarify ideas, thoughts, and feelings.</li> </ul>
Vocabulary:	<p><u>Online</u>: Using a computer, phone, or tablet to visit a website or app</p> <p><u>Pause</u>: To stop what you're doing or saying</p>
Notes:	<p>→Teachers will need to create FREE teacher and/or student accounts (when applicable) at <a href="https://www.common sense.org/education/">https://www.common sense.org/education/</a>.</p>

## Week 4: Code.org, Course A, Lesson 2—Learn to Drag and Drop

<p>Lesson overview:</p> 	<p><b>Purpose:</b> Review drag and drop skills from Course A. Review hardware and how to use it. This is a skill-building lesson that will give students an idea of what to expect when they head to the computer lab. It begins with a brief discussion introducing them to computer lab manners, then progresses into using a computer to complete online puzzles.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"><li>• Introduction<ul style="list-style-type: none"><li>◦ Behaving in the computer lab</li></ul></li><li>• Learn to Drag and Drop<ul style="list-style-type: none"><li>◦ This will teach students how to use Code.org to complete online puzzles.</li></ul></li><li>• Reflection<ul style="list-style-type: none"><li>◦ Prompts provided</li></ul></li><li>• <a href="#">Rhyme With That</a><ul style="list-style-type: none"><li>◦ Rhyme With That is an optional activity aligned to Common Core ELA and math standards, written by our teacher community. Students will use the computer mouse to drag and drop a letter to form CVC words, match rhyming words, and count.</li></ul></li></ul>
<p>Lesson links/resources:</p>	<p><a href="#">Code.org: Learn to Drag and Drop</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"><li>• Recognize what is expected when they transition into the computer lab.</li><li>• Use appropriate terminology when referring to a computer mouse, trackpad, or touchscreen.</li></ul> <p>Standards:</p> <ul style="list-style-type: none"><li>• <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.</li><li>• <b>CS.1A.2</b>—Use appropriate terminology in identifying and describing the function of common physical components of computing systems (hardware).</li><li>• <b>CS.1A.3</b>—Describe basic hardware and software problems using accurate terminology.</li><li>• <b>IC.1A.2</b>—Work respectfully and responsibly with others online.</li><li>• <b>IC.1A.3</b>—Keep login information private and log off of devices appropriately.</li></ul>
<p>Time needed:</p>	<p><b>Total Time: 65 min</b></p> <ul style="list-style-type: none"><li>• Introduction <b>10 min</b></li><li>• Learn to Drag and Drop <b>20 min</b></li><li>• Reflection <b>5 min</b></li><li>• Rhyme With That <b>30 min</b></li></ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"><li>• Computer</li><li>• Smartboard/projector with sound</li><li>• <a href="#">Code.org</a> account</li><li>• <a href="#">20/20/20 Rule</a> - Resource</li><li>• <a href="#">Getting Started - Creating a Class Section</a> - Video</li><li>• <a href="#">Wiggles-Go Noodle</a> - Video</li></ul>

	<p>Students:</p> <ul style="list-style-type: none"> <li>• Computer/tablet with internet access</li> <li>• <a href="#">Code.org</a> account</li> <li>• <a href="#">Pair Programming</a> - Student Video</li> </ul>
Subject integrated:	<p>ELA Math</p>
Other standards addressed:	<p>ELA</p> <ul style="list-style-type: none"> <li>• <b>RF.1.2</b>—Demonstrate understanding of spoken words, syllables, and sounds (rhyming).</li> </ul> <p>Math</p> <ul style="list-style-type: none"> <li>• <b>1.NBT.1</b>—Count to 120 from any given number within 120.</li> </ul>
Vocabulary:	<p><u>Drag/Drop</u>: The action of selecting and moving an object (dragging) and then placing it (dropping) in a different area  <u>Computer</u>: A device for working with information  <u>Desktop</u>: Screen that appears if you are not browsing the internet, reading a file or playing a game; your icons are on this screen  <u>Input</u>: To add information/data into the computer  <u>Keyboard</u>: Where all the letters, numbers, and other buttons are located; when you type on it, the symbols appear on the monitor  <u>Laptop</u>: Small portable computer  <u>Monitor</u>: Screen that shows you what you are doing; a viewer that displays what the computer sends to it  <u>Mouse</u>: A hand-held device for moving a cursor around the screen  <u>Output</u>: To send information/data out of the computer  <u>Tablet</u>: A wireless, portable personal computer with a touchscreen interface</p>
Notes:	<p>→ Teachers will need to create FREE teacher and/or student accounts at <a href="https://studio.code.org/users/sign_in">https://studio.code.org/users/sign_in</a>.</p>


## Week 5: Code.org, Course B, Lesson 1—Digital Trails

<p>Lesson overview:</p> 	<p><b>Purpose:</b> Common Sense Education created this lesson to teach students about "digital footprints" and what information is okay for them to have on their own.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"><li>• Keyboarding Practice<ul style="list-style-type: none"><li>◦ Choose an online or unplugged keyboard practice game.</li></ul></li><li>• Follow the Digital Trail<ul style="list-style-type: none"><li>◦ Students will learn what is not okay to share online.</li></ul></li><li>• Animal Tracks<ul style="list-style-type: none"><li>◦ Students will follow the trails of Ellie the Elephant and Mervin the Mouse and answer two reflection questions.</li></ul></li><li>• Okay to Share?<ul style="list-style-type: none"><li>◦ Students will record what they've learned about Ellie and Mervin.</li><li>◦ Students will then list what is okay to share and what is not okay to share.</li></ul></li><li>• Wrap Up: Pause and Think Moment<ul style="list-style-type: none"><li>◦ Teachers and students will have a moment to think on the activity and reflect with the class.</li></ul></li></ul>
<p>Lesson links/resources:</p>	<p><a href="#">Code.org: Lesson 1- Digital Trails</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"><li>• Explore what information is okay to be shared online</li><li>• Learn that the information they share online leaves a digital footprint or "trail"</li></ul> <p>Standards:</p> <ul style="list-style-type: none"><li>• <b>IC.1A.2</b>—Work respectfully and responsibly with others online.</li><li>• <b>IC.1A.3</b>—Keep login information private and log off of devices appropriately.</li><li>• <b>NI.1A.1</b>—Explain what passwords are and why we use them.</li></ul>
<p>Time needed:</p>	<p><b>Total Time: 55 min</b></p> <ul style="list-style-type: none"><li>• Keyboarding Practice <b>10 min</b></li><li>• Follow the Digital Trail <b>5 min</b></li><li>• Animal Tracks <b>15 min</b></li><li>• Okay to Share? <b>15 min</b></li><li>• Wrap Up <b>5 min</b></li><li>• Pause and Think Moment <b>5 min</b></li></ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"><li>• Computer</li><li>• Projector/smartboard with sound</li><li>• <a href="#">Code.org</a> account</li><li>• <a href="#">Digital Trails: Follow the Digital Trail</a> - Lesson Video (<a href="#">Download</a>)</li><li>• <a href="#">Digital Trail Squares</a></li><li>• <a href="#">Digital Trails: Lesson Slides</a> - Slide Deck</li></ul> <p>Students:</p> <ul style="list-style-type: none"><li>• Computer/tablet with internet access</li><li>• <a href="#">Code.org</a> account</li><li>• <a href="#">Digital Trails: Animal Tracks</a> - Student Handout</li><li>• <a href="#">Digital Trails: Digital Trail Squares</a> - Student Handout</li></ul>

Subject integrated:	<b>ELA</b>
Other standards addressed:	<ul style="list-style-type: none"><li>● <b>SL.1.2</b>—Ask and answer questions about key details in a text read aloud or information presented orally or through other media.</li><li>● <b>SL.1.5</b>—Add drawing or other visual displays to descriptions when appropriate to clarify ideas, thoughts, and feelings.</li></ul>
Vocabulary:	<p><u>Digital footprint</u>: A record of what you do online, including the sites you visit and the things you share. It is permanent (i.e., lasts forever).</p> <p><u>Private information</u>: Information about you that can be used to identify who you are</p> <p><u>Digital trail</u>: A path or track</p>
Notes:	

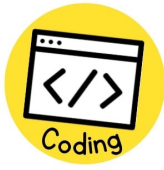


## Week 6: Code.org, Course B, Lesson 2—Move It, Move It

<p>Lesson overview:</p> 	<p><u>Purpose:</u> By using physical movement to program their classmates, students will run into issues and emotions similar to what they will feel when they begin coding on a computer. Encountering those stresses in a playful and open environment will help alleviate intensity and allow students to practice necessary skills before they run into problems on their own.</p> <p><u>Lesson:</u></p> <ul style="list-style-type: none"> <li>● Keyboarding Practice             <ul style="list-style-type: none"> <li>○ Choose an online or unplugged keyboard practice game.</li> </ul> </li> <li>● Where Did I Go Wrong?             <ul style="list-style-type: none"> <li>○ Give the students a warmup to find the bug in the program. (This unplugged activity has an easy path where students use directions to get to the gem.)</li> </ul> </li> <li>● Move It, Move It             <ul style="list-style-type: none"> <li>○ The students will create their own program by giving directions to another student to follow their paper path.</li> </ul> </li> <li>● Reflection             <ul style="list-style-type: none"> <li>○ Prompts provided</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Code.org: Lesson 2- Move It, Move It</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Define a list of steps (algorithm) to get a friend from their starting position to their goal</li> <li>● Identify and fix errors in the execution of an algorithm</li> <li>● Translate a list of steps into a series of physical actions</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1A.1</b>—Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.</li> <li>● <b>AP.1A.3</b>—Develop programs with sequences and simple loops to express ideas or address a problem.</li> <li>● <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem in a precise sequence of instructions.</li> </ul>
<p>Time needed:</p>	<p><b><u>Total Time: 60 min</u></b></p> <ul style="list-style-type: none"> <li>● Keyboard Practice <b>10 min</b></li> <li>● Where Did I Go Wrong? <b>20 min</b></li> <li>● Move It, Move It <b>20 min</b></li> <li>● Reflection <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">Code.org</a> account</li> <li>● <a href="#">Move It, Move It</a> - Worksheet Answer Key</li> <li>● <a href="#">Move it, Move it</a> - Teacher Video</li> </ul> <p>Students</p> <ul style="list-style-type: none"> <li>● Computer/tablet with internet access</li> <li>● <a href="#">Code.org</a> account</li> <li>● <a href="#">Feeling Faces Emotion Image</a> - Resource</li> <li>● <a href="#">Move It, Move It</a> - Map Activity</li> <li>● <a href="#">Move It, Move It</a> - Worksheet</li> </ul>


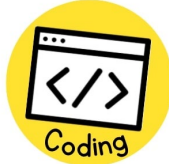
Subject integrated:	ELA Social Studies
Other standards addressed:	<p>ELA</p> <ul style="list-style-type: none"> <li>● <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>● <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>● <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> <li>● <b>SL.1.5</b>—Add drawing or other visual displays to descriptions when appropriate to clarify ideas, thoughts, and feelings.</li> </ul> <p>Social Studies</p> <ul style="list-style-type: none"> <li>● <b>G.1.3</b>—Recognize maps, graphs, and other representations of Earth. Construct a map from a student's home to school, applying cardinal and intermediate directions.</li> </ul>
Vocabulary:	<p><u>Algorithm</u>: A sequence of steps, followed in order, to finish a task that can be performed with or without a computer.</p> <p><u>Bug</u>: Part of a program that does not work correctly</p> <p><u>Debugging</u>: Finding and fixing problems in an algorithm or program</p>
Notes:	<p>Enrichment: <a href="#">Code and Go Mouse</a> can be used to teach cardinal directions.</p>

## Week 7: Code.org, Course B, Lesson 3—Sequencing With Angry Birds

<p>Lesson overview:</p> 	<p><b>Purpose:</b> In this lesson, students will develop programming and debugging skills on a computer platform. The block-based format of these puzzles helps students learn about sequence and concepts without having to worry about perfecting syntax.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Keyboarding Practice             <ul style="list-style-type: none"> <li>○ Choose an online or unplugged keyboard practice game.</li> </ul> </li> <li>● Behaving in the Computer Lab             <ul style="list-style-type: none"> <li>○ Review expectations of how students should behave in a classroom/computer lab.</li> </ul> </li> <li>● Bridging Activity             <ul style="list-style-type: none"> <li>○ <u>Online</u>: Choose a puzzle from the corresponding puzzles on this level. Have students draw arrows to program the angry bird to get to the pig. Rename the arrows north, south, east, and west.</li> <li>○ <u>Unplugged</u>: Select a map from "<a href="#">Move It, Move It</a> - Map Activity" and allow students to use paper blocks to program the robot.</li> </ul> </li> <li>● Sequencing with Angry Birds             <ul style="list-style-type: none"> <li>○ Students will be activating prior knowledge and improving their skills by learning how to sequence with Angry Birds puzzles.</li> </ul> </li> <li>● Reflection:             <ul style="list-style-type: none"> <li>○ Prompts provided</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Code.org: Lesson 3 - Sequencing with Angry Birds</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Experiment with standard block-based programming actions such as clicking, dragging, dropping, etc.</li> <li>● Model proper computer lab behaviors</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem in a precise sequence of instructions.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Keyboarding Practice <b>15 min</b></li> <li>● Behaving in the Computer Lab <b>10 min</b></li> <li>● Bridging Activity <b>10 min</b></li> <li>● Sequencing with Angry Birds <b>20 min</b></li> <li>● Reflection <b>5 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">20/20/20 Rule</a> - Resource</li> <li>● <a href="#">How to Make a Class Section on Code.org</a> - Teacher Video</li> <li>● <a href="#">Code.org</a> account</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Computer/tablet with internet access</li> <li>● <a href="#">Code.org</a> account</li> <li>● <a href="#">Drag and Drop Practice</a></li> </ul>

	<ul style="list-style-type: none"> <li>● <a href="#">Feeling Faces</a> - Emotion Images</li> <li>● <a href="#">Move It, Move It</a> - Map Activity</li> <li>● <a href="#">Pair Programming</a> - Student Video</li> <li>● <a href="#">Unplugged Blockly Blocks (Grades K-1)</a> - Manipulatives</li> <li>● <a href="#">Wiggles - GoNoodle</a> - Video</li> </ul>
Subject integrated:	ELA Math Social Studies
Other standards addressed:	ELA <ul style="list-style-type: none"> <li>● <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>● <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>● <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> <li>● <b>SL.1.5</b>—Add drawing or other visual displays to descriptions when appropriate to clarify ideas, thoughts, and feelings.</li> </ul> Math <ul style="list-style-type: none"> <li>● <b>1.MD.4</b>—Organize, represent, and interpret data with up to three categories; ask and answer questions about the total number of data points, how many in each category, and how many more or less are in one category than in another.</li> </ul> Social Studies <ul style="list-style-type: none"> <li>● <b>G.1.3</b>—Recognize maps, graphs, and other representations of Earth. Construct a map from a student's home to school, applying cardinal and intermediate directions.</li> </ul>
Vocabulary:	<u>Click</u> : Press the mouse button <u>Double-click</u> : Press the mouse button very quickly <u>Drag</u> : Click the mouse button and hold as you move the mouse pointer to a new location <u>Drop</u> : Release the mouse button to "let go" of an item that you are dragging
Notes:	

## Week 8: Code.org, Course B, Lesson 3—Interpreting Sticky Data

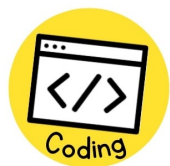
<p>Lesson overview:</p>  	<p><b>Purpose:</b> Students will use paper manipulatives to organize, represent, and interpret data.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Keyboard Practice             <ul style="list-style-type: none"> <li>○ Choose an online or unplugged keyboard practice game.</li> </ul> </li> <li>● BBC Learning: What is Coding?             <ul style="list-style-type: none"> <li>○ Watch the video and ask your students about what they learned in the video.</li> </ul> </li> <li>● Interpreting Sticky Data             <ul style="list-style-type: none"> <li>○ Students will use <a href="#">Blockly Blocks of Code</a> to write the program to get the angry bird to the pig.</li> </ul> </li> <li>● BrainPop Jr. Computer Programming             <ul style="list-style-type: none"> <li>○ Watch the video and ask your students the following question:                 <ul style="list-style-type: none"> <li>■ “What did you learn from the video?” Make sure to give all students an opportunity to speak if they'd like.</li> </ul> </li> </ul> </li> <li>● Reflection             <ul style="list-style-type: none"> <li>○ Ask students to draw a picture of what they learned today.</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">Interpreting Sticky Data Lesson</a></li> <li>● <a href="#">BBC Learning - What is Coding?</a></li> <li>● <a href="#">Computer Programming Introduction</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Experiment with standard block-based programming actions such as clicking, dragging, dropping, etc.</li> <li>● Model proper computer lab behaviors</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem in a precise sequence of instructions.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 55 min</b></p> <ul style="list-style-type: none"> <li>● Keyboarding Practice <b>15 min</b></li> <li>● BBC Learning: What is Coding? <b>5 min</b></li> <li>● Interpreting Sticky Data <b>20 min</b></li> <li>● BrainPop Jr Computer Programming <b>10 min</b></li> <li>● Reflection <b>5 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">Code.org</a> account</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Computer/tablet with internet access</li> <li>● <a href="#">Code.org</a> account</li> <li>● <a href="#">Interpreting Sticky Data Student Handout</a></li> <li>● <a href="#">Blockly Blocks of Code</a> (paper manipulatives worksheet)</li> <li>● Scissors</li> <li>● Glue</li> </ul>
<p>Subject integrated:</p>	<p>Math</p>
<p>Other standards</p>	<p><b>1.MD.4</b>—Organize, represent, and interpret data with up to three categories;</p>

addressed:	ask and answer questions about the total number of data points, how many in each category, and how many more or less are in one category.
Vocabulary:	<u>Program</u> : A list of steps that a computer follows in order to finish a task <u>Computer Coding</u> : Step-by-step instructions for a computer <u>Bug</u> : Part of a program that does not work correctly <u>Debugging</u> : Finding and fixing problems in an algorithm or program
Notes:	Interpreting Sticky Data (Cross-Curricular Opportunity) is located at the end of Lesson 3.



## Week 9: Code.org, Course B, Lesson 4—Programming With Angry Birds

Lesson overview:



**Purpose:**

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

**Lesson:**

- Review Unplugged Activity
  - Show students a picture from the "Move It, Move It."
  - Ask students to recall the symbols used in "Move It, Move It."
  - What would you do when you saw the "North" arrow? How about the "East" arrow?
  - Once you are satisfied that your students remember "Move It, Move It", you can move into the Bridging Activity.
- Keyboarding Review- See resources guide.
- Bridging Activity:
  - You will choose an activity to review concepts of Move It, Move It to prepare for students' online lessons.
- Previewing Online Puzzles as a class
  - Online: Online puzzles using arrows
  - Unplugged: using paper blocks
- Programming with Angry Birds
  - Online programming levels using angry birds.
- Reflection
  - Prompts provided

Lesson links/resources:

[Code.org: Lesson 4- Programming with Angry Birds](#)

CS standards addressed:

- Students will be able to:
- Build a computer program from a set of written instructions.
  - Construct a program by reorganizing sequential movements.
  - Translate movements into a series of commands
- Standards:
- **AP.1A.2**—Model the way programs store and manipulate data by using numbers or other symbols to represent information.
  - **AP.1A.4**—Decompose (break down) the steps needed to solve a problem in a precise sequence of instructions.

Time needed:

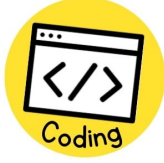
- Total Time: 60 min**
- Review Unplugged Activity: **5 min**
  - Keyboarding Practice: **5min**
  - Bridging Activity: **10 min**
  - Previewing Online Puzzles as a Class: **5 min**
  - Programming with Angry Birds: **30 min**
  - Reflection: **5 min**

Materials needed:

- Teacher:
- Computer
  - Projector/Smartboard with sound
  - [Code.org](#) account
- Students:
- Computer/Tablet with internet access
  - [Code.org](#) account
  - [Compass Rose](#) - Handout
  - [Feeling Faces - Emotion Image](#) - Resource

	<ul style="list-style-type: none"> <li>• <a href="#">Happy Map Game Pieces</a> - Manipulatives</li> <li>• <a href="#">Move It, Move It</a> - Map Activity</li> <li>• <a href="#">Pair Programming</a> - Student Video</li> <li>• <a href="#">Unplugged Blockly Blocks (Grades K-1)</a> - Manipulatives</li> </ul>
Subject integrated:	ELA Social Studies
Other standards addressed:	<p>ELA</p> <ul style="list-style-type: none"> <li>• <b>SL.1.1</b>—Participate in collaborative conversation</li> <li>• <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>• <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> <li>• <b>SL.1.5</b>—Add drawing or other visual displays to descriptions when appropriate to clarify ideas, thoughts, and feelings.</li> </ul> <p>Social Studies</p> <ul style="list-style-type: none"> <li>• <b>G.1.3</b>— Recognize maps, graphs, and other representations of Earth. Construct a map from a student's home to school, applying cardinal and intermediate directions.</li> </ul>
Vocabulary:	<p><u>Algorithm</u>: A sequence of steps, followed in order, to finish a task that can be performed with or without a computer.</p> <p><u>Program</u>: A sequence of instructions given to a computer in code that the computer can understand. The computer follows the instructions and carries out the task</p> <p><u>Programming</u>: The art of creating a program</p>
Notes:	

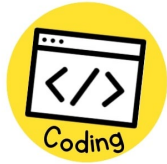
## Week 10: Code.org, Course B, Lesson 5—Programming With Harvester

Lesson overview: <div style="text-align: center; margin-top: 10px;">  </div>	<p><b>Purpose:</b> In this lesson, students will develop debugging skills and continue developing their programming skills.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Review:                         <ul style="list-style-type: none"> <li>○ Extended Learning Lesson 4:                                 <ul style="list-style-type: none"> <li>■ In small groups, let students design their own mazes on paper and challenge other students or groups to write programs to solve them. For added fun, make life-size mazes with students as the pig and bird.</li> </ul> </li> </ul> </li> <li>● Vocabulary Introduction                         <ul style="list-style-type: none"> <li>○ Introduce students to new vocabulary.</li> </ul> </li> <li>● Programming with Harvester</li> <li>● Reflection                         <ul style="list-style-type: none"> <li>○ Prompts provided</li> </ul> </li> </ul>
Lesson links/resources:	<a href="#">Code.org: Lesson 5 - Programming with Harvester</a>
CS standards addressed:	Students will be able to: <ul style="list-style-type: none"> <li>● Identify and locate bugs in a program</li> <li>● Translate movements into a series of commands</li> </ul> Standards: <ul style="list-style-type: none"> <li>● <b>AP.1A.2</b>—Model the way programs store and manipulate data by using numbers or other symbols to represent information.</li> <li>● <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem in a precise sequence of instructions.</li> </ul>
Time needed:	<p><b>Total Time: 55 min</b></p> <ul style="list-style-type: none"> <li>● Extended Learning Session 4: <b>15 min</b></li> <li>● Vocabulary Introduction: <b>5 min</b></li> <li>● Programming with Harvester: <b>30 min</b></li> <li>● Reflection: <b>5 min</b></li> </ul>
Materials needed:	Teacher: <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">Code.org</a> account</li> </ul> Students: <ul style="list-style-type: none"> <li>● Computer/tablet with internet access</li> <li>● <a href="#">Code.org</a> account</li> <li>● <a href="#">Feeling Faces</a> - Emotion Images</li> <li>● <a href="#">Stevie and the Big Project</a> - Resource</li> <li>● <a href="#">Unspotted Bugs</a> - Video</li> </ul>
Subject integrated:	Math Social Studies
Other standards addressed:	Math <ul style="list-style-type: none"> <li>● <b>1.MD.4</b>—Organize, represent, and interpret data with up to three categories; ask and answer questions about the total number of data points, how many in each category, and how many more or less are in one category than in another.</li> </ul> Social Studies

	<ul style="list-style-type: none"><li>● <b>G.1.3</b>—Recognize maps, graphs, and other representations of Earth. Construct a map from a student's home to school, applying cardinal and intermediate directions.</li></ul>
Vocabulary:	<p><u>Algorithm</u>: A sequence of steps, followed in order, to finish a task that can be performed with or without a computer.</p> <p><u>Bug</u>: Part of a program that does not work correctly</p> <p><u>Debugging</u>: Finding and fixing problems in an algorithm or program</p> <p><u>Persistence</u>: Trying again and again, even when something is very hard</p> <p><u>Program</u>: A sequence of instructions given to a computer in code that the computer can understand. The computer follows the instructions and carries out the task</p> <p><u>Programming</u>: The art of creating a program</p>
Notes:	

## Week 11: Code.org, Course B, Lesson 6—Getting Loopy

Lesson overview:



**Purpose:**

At this point in the course, students should have developed comfort with programming a set of linear instructions. The linear set of instructions frequently includes patterns that repeat multiple times, and as students want to write more complex and interesting programs, manually duplicating that code becomes cumbersome and inefficient. To enable students to write more powerful programs, we'll need to rely on structures that break from the single linear list. Loops allow for students to structure their code in a way that repeats. In this lesson, we will focus on identifying patterns in physical movement before moving back onto the computer to look for patterns in our code.

**Lesson:**

- Keyboard Practice
  - Choose an online or unplugged keyboard practice game.
- Repeat After Me
  - Instruct your volunteer to walk around the table (or their chair, or a friend).
  - When they finish, instruct them to do it again, using the exact same words you did before.
- Getting Loopy
  - Show the class what the entire dance looks like done at full-speed. Then run through the dance slowly, asking a different student to call out each line of instructions. Next, have the students perform the dance along with you, saying the instructions aloud as they get to each move.
- Assessment
- Reflection
  - Prompts provided

Lesson links/resources:

[Code.org: Lesson 6 - Getting Loopy](#)

CS standards addressed:

Students will be able to:

- Convert a series of multiple actions into a single loop
- Repeat actions initiated by the instructor
- Translate a picture program into a real-world dance

Standards:

- **AP.1A.2**—Model the way programs store and manipulate data by using numbers or other symbols to represent information.
- **AP.1A.3**—Develop programs with sequences and simple loops to express ideas or address a problem.
- **AP.1A.4**—Decompose (break down) the steps needed to solve a problem in a precise sequence of instructions.
- **AP.1A.7**—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.

Time needed:

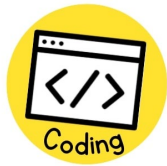
**Total Time: 60 min**

- Keyboard Practice **15 min**
- Repeat After Me **15 min**
- Assessment **10 min**
- Reflection **15 min**

Materials needed:	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">Code.org</a> account</li> <li>● <a href="#">Getting Loopy</a> - Assessment Answer Key</li> </ul> <p>Students</p> <ul style="list-style-type: none"> <li>● Computer/tablet with internet access</li> <li>● <a href="#">Code.org</a> account</li> <li>● <a href="#">Getting Loopy</a> - Unplugged Video (<a href="#">Download</a>)</li> <li>● <a href="#">Getting Loopy</a> - Worksheet</li> <li>● <a href="#">Getting Loopy</a> - Assessment</li> </ul>
Subject integrated:	ELA
Other standards addressed:	<ul style="list-style-type: none"> <li>● <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>● <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>● <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> <li>● <b>SL.1.5</b>—Add drawing or other visual displays to descriptions when appropriate to clarify ideas, thoughts, and feelings.</li> </ul>
Vocabulary:	<p><u>Loop</u>: A command used to repeat a portion of code.  <u>Function</u>: A set of steps given a simple name that a programmer can easily call on and re-use again in a program  <u>Repeat</u>: To do something again  <u>Pattern</u>: A repeated, noticeable sequence.</p>
Notes:	

## Week 12: Code.org, Course B, Lesson 7—Loops With Harvester

Lesson overview:



**Purpose:**

In this lesson, students will learn 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.

**Lesson:**

- Extended Learning Lesson 6
  - So Moving: Give the students pictures of actions or dance moves 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.
- Intro to Loops
  - What are loops?
  - How do we use them?
- Bridging Activity
  - You will choose an activity to review concepts of Getting Loopy to prepare for students' online lessons.
- Loops with Harvester
  - As students work through the puzzles, see if they can figure out how many blocks they use with a loop vs. without a loop.
- Reflection
  - Prompts provided

Lesson links/resources:

[Code.org: Lesson 7 - Loops with Harvester](#)

CS standards addressed:

Students will be able to:

- Break down a long sequence of instructions into the smallest repeatable sequence possible
- Create a program for a given task which loops a sequence of commands
- Employ a combination of sequential and looped commands to reach the end of a maze
- Identify the benefits of using a loop structure instead of manual repetition

Standards:

- **AP.1A.2**—Model the way programs store and manipulate data by using numbers or other symbols to represent information.
- **AP.1A.3**—Develop programs with sequences and simple loops to express ideas or address a problem.
- **AP.1A.4**—Decompose (break down) the steps needed to solve a problem in a precise sequence of instructions.
- **AP.1A.7**—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.

Time needed:

**Total Time: 60 min**

- Warm Up **15 min**
- Main Activity **15 min**
- Independent Practice **15 min**

	<ul style="list-style-type: none"> <li>• Wrap Up <b>15 min</b></li> </ul>
Materials needed:	<p>Teacher:</p> <ul style="list-style-type: none"> <li>• Computer</li> <li>• Projector/smartboard with sound</li> <li>• <a href="#">Code.org</a> account</li> </ul> <p>Students</p> <ul style="list-style-type: none"> <li>• Computer/tablet with internet access</li> <li>• <a href="#">Code.org</a> account</li> <li>• <a href="#">CS Fundamentals Main Activity Tips</a> - Lesson Recommendations</li> <li>• <a href="#">Feeling Faces Emotion Image</a> - Resource</li> <li>• <a href="#">Unplugged Blockly Blocks (Grades K-1)</a> - Manipulatives</li> </ul>
Subject integrated:	<p>ELA Math</p>
Other standards addressed:	<p>ELA</p> <ul style="list-style-type: none"> <li>• <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>• <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>• <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> <li>• <b>SL.1.5</b>—Add drawing or other visual displays to descriptions when appropriate to clarify ideas, thoughts, and feelings</li> </ul> <p>Math</p> <ul style="list-style-type: none"> <li>• <b>1.NBT.1</b>—Read and write numerals to 120</li> </ul>
Vocabulary:	<p><u>Loop</u>: A command used to repeat a portion of code. <u>Function</u>: A set of steps given a simple name that a programmer can easily call on and re-use again in a program <u>Repeat</u>: To do something again</p>
Notes:	



## Week 13: Code.org, Course B, Lesson 7—Corn Crazy Choice Board

Lesson overview:



**Purpose:**

[Corn Crazy Choice Board](#) will allow students to practice their speaking and listening skills as they determine which properties of Harvester's corn can be used to solve a human problem through biomimicry.

**Lesson:**

- Place students in groups of two to complete [Loops with Harvester, Levels 1-13](#).
  - **Teacher Note: If students are not familiar with pair programming, it is recommended that you show the video found in [Programming with Angry Birds, Level 1](#).**
- Once students have completed Loops with Harvester, they should go to the [Corn Crazy Choice Board Slides](#) to select one part of the corn plant to design a solution to a human problem. Your design should mimic how one part of the corn plant could be used to help humans survive, grow, or meet their needs.
  - The following instructions are also listed in the speaker notes section of the [Corn Crazy Choice Board Slides](#).
- **Lesson Introduction (Slide 1)**
  - Ask the class "What is a human problem?"
    - The slide deck shows an image of a helmet. Discuss with your students how a helmet is a solution to the human problem of needing protection.
  - Now ask the class "How does a helmet mimic something we see in plants or animals?"
    - The slide deck shows an image of a turtle shell. Discuss with your students how a turtle shell protects a turtle like a helmet protects a human.
- **Choice Board Instructions (Slide 2)**
  - Select one part of the corn plant that you wish to use in order to design a solution to a human problem.
  - Click on the box labeled with the particular part of the corn plant you have selected. This will take you to a new slide.
  - On your next slide, describe or draw your solution in the provided text box. Students can draw by going to the "Insert" menu, then click on "Line" and "Scribble."
- **Optional Wrap-Up Questions (Slides 3-8)**
  - How would you explain your design to someone else?
  - What do other students think about your design?
  - How would you describe one of your classmate's designs?

Lesson links/resources:

[Corn Crazy Choice Board](#)

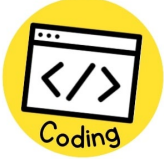
CS standards addressed:

Students will be able to:

- Break down a long sequence of instructions into the smallest repeatable sequence possible
- Create a program for a given task which loops a sequence of commands

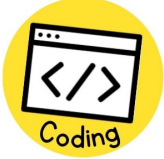
	<ul style="list-style-type: none"> <li>• Employ a combination of sequential and looped commands to reach the end of a maze</li> <li>• Identify the benefits of using a loop structure instead of manual repetition</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>• <b>AP.1A.2</b>—Model the way programs store and manipulate data by using numbers or other symbols to represent information.</li> <li>• <b>AP.1A.3</b>—Develop programs with sequences and simple loops to express ideas or address a problem.</li> <li>• <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem in a precise sequence of instructions.</li> <li>• <b>AP.1A.7</b>—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.</li> </ul>
Time needed:	<p><b>Total Time: 60min</b></p> <ul style="list-style-type: none"> <li>• Lesson Introduction <b>10 min</b></li> <li>• Choice Board <b>40 min</b></li> <li>• Wrap-Up Questions <b>10 min</b></li> </ul>
Materials needed:	<p>Teacher:</p> <ul style="list-style-type: none"> <li>• Computer and projector/smartboard with sound</li> <li>• <a href="#">Code.org</a> account</li> <li>• <a href="#">Corn Crazy Choice Board Slides</a> (Google Slides)</li> </ul> <p>Students</p> <ul style="list-style-type: none"> <li>• Computer/tablet with internet access</li> <li>• <a href="#">Code.org</a> account</li> </ul>
Subject integrated:	<p>ELA Science</p>
Other standards addressed:	<p>ELA</p> <ul style="list-style-type: none"> <li>• <b>SL.1.1a</b>—Follow agreed-upon rules for discussions (e.g., listening to others with care, speaking one at a time about the topics and texts under discussion).</li> <li>• <b>SL.1.1b</b>—Build on others' talk in conversations by responding to the comments of others through multiple exchanges.</li> <li>• <b>SL.1.1c</b>—Ask questions to clear up any confusion about the topics and texts under discussion.</li> </ul> <p>Science</p> <p><b>L.1.1</b>—Students will demonstrate an understanding of basic needs and structures of plants.</p>
Vocabulary:	<p><u>Loop</u>: A command used to repeat a portion of code  <u>Function</u>: A set of steps given a simple name that a programmer can easily call on and re-use again in a program  <u>Repeat</u>: To do something again</p>
Notes:	<ul style="list-style-type: none"> <li>• Teachers can determine if this activity is best used as whole class instruction, small group, or individual work. Students can get their own copy of the <a href="#">Corn Crazy Choice Board Slides</a> by clicking "File" and "Make a Copy" in order to record their answers.</li> <li>• Teachers may also choose to have students complete one option/slide of the <a href="#">Corn Crazy Choice Board Slides</a> or multiple options/slides.</li> </ul>

## Week 14: Code.org, Course B, Lesson 8—Loops With Laurel

<p>Lesson overview:</p> 	<p><b>Purpose:</b> This lesson gives students more practice with loops and encourages them to put multiple blocks inside of a repeat as they try to collect as much treasure as possible.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Keyboard Practice             <ul style="list-style-type: none"> <li>○ Choose an online or unplugged keyboard practice game.</li> </ul> </li> <li>● Introduction to Loops             <ul style="list-style-type: none"> <li>○ What are loops?</li> <li>○ Why do we use them?</li> </ul> </li> <li>● Teacher Demo             <ul style="list-style-type: none"> <li>○ Preview Loops in Collector with the class.</li> </ul> </li> <li>● Loops in Collector</li> <li>● Reflection             <ul style="list-style-type: none"> <li>○ Prompts provided</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● Code.org: <a href="#">Lesson 8 - Loops with Laurel</a></li> <li>● <a href="#">How Technology Makes You Feel</a> (video 1:07)</li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Break down a long sequence of instructions into the smallest repeatable sequence possible</li> <li>● Identify the benefits of using a loop structure instead of manual repetition</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1A.2</b>—Model the way programs store and manipulate data by using numbers or other symbols to represent information.</li> <li>● <b>AP.1A.3</b>—Develop programs with sequences and simple loops to express ideas or address a problem.</li> <li>● <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem in a precise sequence of instructions.</li> <li>● <b>AP.1A.7</b>—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Keyboarding Practice <b>15 min</b></li> <li>● Introduction to Loops <b>10 min</b></li> <li>● Main Activity <b>30 min</b></li> <li>● Wrap Up <b>5 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">Code.org</a> account</li> </ul> <p>Students</p> <ul style="list-style-type: none"> <li>● Computer/tablet with internet access</li> <li>● <a href="#">Code.org</a> account</li> <li>● <a href="#">CS Fundamentals Main Activity Tips</a> - Lesson Recommendations</li> <li>● <a href="#">Feeling Faces Emotion Image</a> - Resource</li> </ul>
<p>Subject integrated:</p>	<p>Math</p>

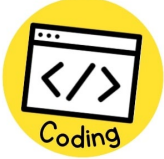
Other standards addressed:	1.NBT.1—Read and count to 120 starting at any number less than 120.
Vocabulary:	<u>Loop</u> : A command used to repeat a portion of code <u>Function</u> : A set of steps given a simple name that a programmer can easily call on and re-use again in a program <u>Repeat</u> : To do something again
Notes:	

## Week 15: Code.org, Course B, Lesson 8—Treasure Tracker

<p>Lesson overview:</p> 	<p><b>Purpose:</b> Like all good adventurers, Laurel keeps a close eye on her treasure to prevent any “sticky fingers” from taking her stuff. Help Laurel track her treasure as she counts and categorizes her loot.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>• Students will pair program to complete <a href="#">Loops with Laurel, Levels 1-13</a>.</li> <li>• Students will then complete the <a href="#">Treasure Tracker Student Handout</a>.</li> <li>• To wrap up, students will complete the “Treasure Tracker Wrap-Up” on the <a href="#">Treasure Tracker Student Handout</a>. Students should discuss their answers with a shoulder partner.</li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Treasure Tracker Lesson</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>• Break down a long sequence of instructions into the smallest repeatable sequence possible</li> <li>• Identify the benefits of using a loop structure instead of manual repetition</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>• <b>AP.1A.2</b>—Model the way programs store and manipulate data by using numbers or other symbols to represent information.</li> <li>• <b>AP.1A.3</b>—Develop programs with sequences and simple loops to express ideas or address a problem.</li> <li>• <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem in a precise sequence of instructions.</li> <li>• <b>AP.1A.7</b>—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>• Intro: <b>5 min</b></li> <li>• <a href="#">Treasure Tracker</a>: <b>20 min</b> <ul style="list-style-type: none"> <li>◦ Loops with Laurel</li> </ul> </li> <li>• Handout: <b>20-30 min</b></li> <li>• Wrap Up/ Equations: <b>5-10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>• Computer</li> <li>• Projector/smartboard with sound</li> <li>• <a href="#">Code.org</a> account</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>• Computer/tablet with internet access</li> <li>• <a href="#">Code.org</a> account</li> <li>• <a href="#">Treasure Tracker Student Handout</a></li> </ul>
<p>Subject integrated:</p>	<p>ELA Math</p>
<p>Other standards addressed:</p>	<p>ELA</p> <ul style="list-style-type: none"> <li>• <b>L.1.1.5a</b>—Sort words into categories (e.g., colors, clothing) to gain a sense of concepts the categories represent.</li> </ul> <p>Math</p>

	<ul style="list-style-type: none"> <li>● <b>1.OA.1</b>—Use addition and subtraction within 20 to solve word problems involving situations of adding to, taking from, putting together, taking apart, and comparing, with unknowns in all positions, e.g., by using objects, drawings, and equations with a symbol for the unknown number to represent the problem.</li> <li>● <b>1.OA.2</b>—Solve word problems that call for addition of three whole numbers whose sum is less than or equal to 20, e.g., by using objects, drawings, and equations with a symbol for the unknown number to represent the problem.</li> </ul>
Vocabulary:	<p><u>Loop</u>: A command used to repeat a portion of code  <u>Function</u>: A set of steps given a simple name that a programmer can easily call on and re-use again in a program  <u>Repeat</u>: To do something again  <u>Algorithm</u>: A sequence of steps, followed in order, to finish a task that can be performed with or without a computer.  <u>Sequence</u>: Instructions given to the computer to be followed in the exact order they are written and written in code using commands from programmers  <u>Block</u>: To stop from happening</p>
Notes:	Pre-teach the meaning of “sticky fingers” and loot.

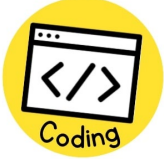
## Week 16: Code.org, Course B, Lesson 9—Drawing Gardens With Loops

<p>Lesson overview:</p> 	<p><b>Purpose:</b> This lesson gives a different perspective on how loops can create things in programming. Students will test their critical thinking skills by evaluating the given code and determining what needs to be added to solve the puzzle. Students can also reflect on the inefficiency of programming without loops here because of how many blocks the program would require without the help of repeat loops.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Keyboard Practice             <ul style="list-style-type: none"> <li>○ Choose an online or unplugged keyboard practice game.</li> </ul> </li> <li>● Introduction to Loops             <ul style="list-style-type: none"> <li>○ Quickly review the definition of a loop—the action of doing something over and over.</li> <li>○ Discuss different patterns like zigzags and stairsteps.                 <ul style="list-style-type: none"> <li>■ How would you explain to someone how to draw that pattern?</li> <li>■ How could you draw this using a loop?</li> </ul> </li> </ul> </li> <li>● Loops in Artiste             <ul style="list-style-type: none"> <li>○ Teacher demonstration</li> <li>○ Skill-building levels</li> </ul> </li> <li>● Reflection             <ul style="list-style-type: none"> <li>○ Prompts provided</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">Code.org: Lesson 9 - Drawing Gardens with Loops</a></li> <li>● <a href="#">How Technology Makes You Feel</a> (Video 1:07)</li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Count the number of times an action should be repeated and represent it as a loop</li> <li>● Create a program that draws complex shapes by repeating simple sequences</li> <li>● Decompose a shape into its largest repeatable sequence</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1A.2</b>—Model the way programs store and manipulate data by using numbers or other symbols to represent information.</li> <li>● <b>AP.1A.3</b>—Develop programs with sequences and simple loops to express ideas or address a problem.</li> <li>● <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem in a precise sequence of instructions.</li> <li>● <b>AP.1A.7</b>—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Keyboarding Practice <b>15 min</b></li> <li>● Introduction to Loops <b>10 min</b></li> <li>● Main Activity <b>30 min</b></li> <li>● Reflection <b>5 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">Code.org</a> account</li> <li>● <a href="#">CS Fundamentals Main Activity Tips</a> - Lesson Recommendations</li> <li>● <a href="#">Pause and Think Online</a> - Video</li> </ul>

	<p>Students:</p> <ul style="list-style-type: none"> <li>• Computer/tablet with internet access</li> <li>• <a href="#">Code.org</a> account</li> <li>• <a href="#">Feeling Faces Emotion Image</a> - Resource</li> </ul>
Subject integrated:	<p>ELA Math</p>
Other standards addressed:	<p>ELA</p> <ul style="list-style-type: none"> <li>• <b>SL.1.1</b>—Participate in collaborative conversations with diverse partners about Grade 1 topics and texts with peers and adults in small and larger groups.</li> <li>• <b>SL.1.1c</b>—Ask questions to clear up any confusion about the topics and texts under discussion (consider using this in more lessons.)</li> <li>• <b>SL.1.5</b>—Add drawings or other visual displays to descriptions when appropriate to clarify ideas, thoughts, and feelings.</li> </ul> <p>Math</p> <ul style="list-style-type: none"> <li>• <b>1.G.2</b>—Compose two-dimensional shapes (rectangles, squares, trapezoids, triangles, half-circles, and quarter-circles) or three-dimensional shapes (cubes, right rectangular prisms, right circular cones, and right circular cylinders) to create a composite shape and compose new shapes from the composite shape.</li> </ul>
Vocabulary:	<p><u>Loop</u>: A command used to repeat a portion of code <u>Function</u>: A set of steps given a simple name that a programmer can easily call on and re-use again in a program</p>
Notes:	





## Week 17: Code.org, Course B: Lesson 9—Loopy Shapes

<p>Lesson overview:</p> 	<p><b>Purpose:</b>  <a href="#">Loopy Shapes</a> is an activity aligned to Common Core ELA and math standards, written by our teacher community. Working with a partner, students will use loops to program two-dimensional shapes while using collaborative conversations to ask clarifying questions.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>• Keyboard Practice             <ul style="list-style-type: none"> <li>◦ Choose an online or unplugged keyboard practice game.</li> </ul> </li> <li>• Working with a partner, students will use loops to program two-dimensional shapes. Students should use collaborative conversations to ask clarifying questions.</li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>• <a href="#">Loopy Shapes</a></li> <li>• <a href="#">Loopy Shapes Reference Guide</a></li> <li>• <a href="#">Loopy Shapes Reference Guide Teacher Answer Key</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>• Count the number of times an action should be repeated and represent it as a loop</li> <li>• Create a program that draws complex shapes by repeating simple sequences</li> <li>• Decompose a shape into its largest repeatable sequence</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>• <b>AP.1A.2</b>—Model the way programs store and manipulate data by using numbers or other symbols to represent information.</li> <li>• <b>AP.1A.3</b>—Develop programs with sequences and simple loops to express ideas or address a problem.</li> <li>• <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem in a precise sequence of instructions.</li> <li>• <b>AP.1A.7</b>—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>• Keyboard Practice <b>15 min</b></li> <li>• <a href="#">Loopy Shapes</a> <b>45 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>• <a href="#">Projector/smartboard</a></li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>• Computer</li> <li>• <a href="#">Loopy Shapes Reference Guide</a></li> <li>• <a href="#">Loopy Shapes Reference Guide Teacher Answer Key</a></li> </ul>
<p>Subject integrated:</p>	<p>ELA Math</p>
<p>Other standards addressed:</p>	<p><b>ELA</b></p> <ul style="list-style-type: none"> <li>• <b>SL.1.1</b>—Participate in collaborative conversations with diverse partners about Grade 1 topics and texts with peers and adults in small and larger groups.</li> <li>• <b>SL.1.1c</b> Ask questions to clear up any confusion about the topics and texts under discussion.</li> <li>• <b>SL.1.5</b>—Add drawings or other visual displays to descriptions when appropriate to clarify ideas, thoughts, and feelings.</li> </ul> <p>Math</p>


	<ul style="list-style-type: none"><li>● <b>1.G.2</b>—Compose two-dimensional shapes (rectangles, squares, trapezoids, triangles, half-circles, and quarter-circles) or three-dimensional shapes (cubes, right rectangular prisms, right circular cones, and right circular cylinders) to create a composite shape and compose new shapes from the composite shape.</li></ul>
Vocabulary:	<u>Loop</u> : A command used to repeat a portion of code <u>Function</u> : A set of steps given a simple name that a programmer can easily call on and re-use again in a program
Notes:	

## Week 18: Code.org, Course B, Lesson 10—The Right App

<p>Lesson overview:</p>  	<p><b>Purpose:</b> Today, computing is more accessible than ever before. People from all walks of life use software on many different devices—particularly smartphones—for information, communication, and entertainment. Because people have such diverse experiences and needs, it is important for budding computer scientists to empathize with people and identify solutions with them in mind.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● What's an app?             <ul style="list-style-type: none"> <li>○ Discuss apps and examples.</li> </ul> </li> <li>● Unplugged Activity: The Right App scenarios             <ul style="list-style-type: none"> <li>○ Display each scenario from the slide deck. While displaying a scenario image, read its accompanying script to the class (see below, or the notes beneath each slide in the deck). After each scenario, ask students to vote on which app they would choose for their friend, then discuss their reasons why.</li> <li>○ The Right App Design: The final scenario allows students to sketch their own app design.</li> </ul> </li> <li>● Reflection             <ul style="list-style-type: none"> <li>○ Prompts provided</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Code.org: Lesson 10 - The Right App</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Apply empathy and creativity to design technology for others</li> <li>● List several different examples of smartphone apps</li> <li>● Recommend technology to others based on their unique needs</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>CS.1A.1</b>—Select and operate appropriate software to perform a variety of tasks and recognize that users have different needs and preferences for the technology they use.</li> <li>● <b>IC.1A.1</b>—Compare how people live and work before and after the implementation or adoption of new computing technology.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 65 min</b></p> <ul style="list-style-type: none"> <li>● What's an app? <b>5 min</b></li> <li>● The Right App Scenarios <b>15 min</b></li> <li>● App Design Activity <b>20 min</b></li> <li>● Reflection <b>5 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">Code.org</a> account</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Computer/tablet with internet access</li> <li>● <a href="#">Code.org</a> account</li> <li>● <a href="#">The Right App Scenarios</a> - Slide Deck</li> </ul>
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards</p>	<ul style="list-style-type: none"> <li>● <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>● <b>SL.1.2</b>—Ask and answer questions about key details presented</li> </ul>


addressed:	through text or orally. <ul style="list-style-type: none"><li>● <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li><li>● <b>SL.1.5</b>—Add drawing or other visual displays to descriptions when appropriate to clarify ideas, thoughts, and feelings.</li><li>● <b>W.1.8</b>—Recall information and gather information from provided sources to answer a question.</li></ul>
Vocabulary:	<u>App</u> : A computer program that performs a special function
Notes:	

## Week 19: Code.org, Course B, Lesson 11—The Big Event Jr.

<p>Lesson overview:</p> 	<p><b>Purpose:</b> Today, students will learn to distinguish events from actions. The students will see activities interrupted by having a "button" pressed on a paper remote. When seeing this event, the class will react with a unique action. Events are widely used in programming and should be easily recognizable after this lesson.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Intro to Events             <ul style="list-style-type: none"> <li>○ Introduction to vocabulary and discussion about a series of events.</li> </ul> </li> <li>● The Big Event             <ul style="list-style-type: none"> <li>○ The Big Event Unplugged Activity</li> </ul> </li> <li>● Flash Chart and Reflection             <ul style="list-style-type: none"> <li>○ Prompts provided</li> </ul> </li> <li>● Assessment</li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Code.org: Lesson 11- The Big Event Jr.</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Practice differentiating between predefined actions and event-driven ones</li> <li>● Recognize actions of the teacher as signals to initiate commands</li> <li>● Repeat commands given by an instructor</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1A.2</b>—Model the way programs store and manipulate data by using numbers or other symbols to represent information.</li> <li>● <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem in a precise sequence of instructions.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up <b>15 min</b></li> <li>● Main Activity <b>15 min</b></li> <li>● Wrap Up <b>15 min</b></li> <li>● Assessment <b>15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">Code.org</a> account</li> <li>● <a href="#">The Big Event</a> - Assessment Answer Key</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Computer/tablet with internet access</li> <li>● <a href="#">Code.org</a> account</li> <li>● <a href="#">Feeling Faces</a> - Emotion Images</li> <li>● <a href="#">The Big Event</a> - Unplugged Video (Download)</li> <li>● <a href="#">The Big Event</a> - Assessment</li> <li>● <a href="#">The Big Event</a> (Courses A, B) - Controller Image</li> </ul>
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards addressed:</p>	<ul style="list-style-type: none"> <li>● <b>SL.1.1</b>—Participate in collaborative conversations.</li> <li>● <b>SL.1.2</b>—Ask and answer questions about info presented orally. (Can be used in other lessons.)</li> </ul>

	<ul style="list-style-type: none"><li>• <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li></ul>
Vocabulary:	<u>Event</u> : An action that causes something to happen
Notes:	

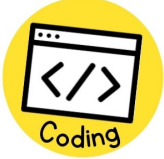
## Week 20: Code.org, Course B, Lesson 11—Zookeeper Survival Events

Lesson overview: <div style="text-align: center; margin-top: 10px;">  </div>	Purpose: <a href="#">Zookeeper Survival Events</a> is an activity where students examine how cause and effect can be seen in animals as you view the behavioral patterns of offspring (the cause) to determine how to increase the offspring's survival (the effect). Lesson: <ul style="list-style-type: none"> <li>• Keyboard Practice                         <ul style="list-style-type: none"> <li>◦ Choose an online or unplugged keyboard practice game.</li> </ul> </li> <li>• As the zookeeper at an animal rescue park, it is your job to act as the animals' parent and provide for their needs. You will quickly find that just like there is a "cause and effect" component of programming with events, the same is true for taking care of animals. In this activity, you will examine how cause and effect can be seen in animals as you view the behavioral patterns of offspring (the cause) to determine how to increase the offspring's survival (the effect).</li> </ul>
Lesson links/resources:	<a href="#">Zookeeper Survival Events</a>
CS standards addressed:	Students will be able to: <ul style="list-style-type: none"> <li>• Practice differentiating between predefined actions and event-driven ones</li> <li>• Recognize actions of the teacher as signals to initiate commands</li> <li>• Repeat commands given by an instructor</li> </ul> Standards: <ul style="list-style-type: none"> <li>• <b>AP.1A.2</b>—Model the way programs store and manipulate data by using numbers or other symbols to represent information.</li> <li>• <b>AP.1A.4</b>— Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.</li> </ul>
Time needed:	<b>Total Time: 55 min</b> <ul style="list-style-type: none"> <li>• Keyboarding Practice <b>15 min</b></li> <li>• <a href="#">Zookeeper Survival Events</a> <b>45 min</b></li> </ul>
Materials needed:	Teacher: <ul style="list-style-type: none"> <li>• Computer</li> <li>• Projector/smartboard with sound</li> <li>• <a href="#">Code.org</a> account</li> <li>• <a href="#">The Big Event - Unplugged Video</a></li> </ul> Students: <ul style="list-style-type: none"> <li>• Computer/tablet with internet access</li> <li>• <a href="#">Code.org</a> account</li> <li>• <a href="#">The Big Event Controller</a></li> <li>• <a href="#">Survival Events Worksheet</a></li> </ul>
Subject integrated:	ELA Math Science

Other standards addressed:	<p>ELA</p> <ul style="list-style-type: none"><li>● <b>SL.1.1</b>—Participate in collaborative conversations with diverse partners about Grade 1 topics and texts with peers and adults in small and larger groups.</li><li>● <b>SL.1.4</b>—Describe people, places, things, and events with relevant details, expressing ideas and feelings clearly.</li></ul> <p>Math</p> <ul style="list-style-type: none"><li>● <b>1.G.1</b>—Distinguish between defining attributes (e.g., triangles are closed and three-sided) versus non-defining attributes (e.g., color, orientation, overall size); build and draw shapes to possess defining attributes.</li></ul> <p>Science</p> <ul style="list-style-type: none"><li>● <b>L.1.2</b>—Students will demonstrate an understanding of how living things change in form as they go through a life cycle.</li></ul>
Vocabulary:	<u>Event</u> : An action that causes something to happen
Notes:	



## Week 21: Scratch Jr.: Can I Make My Car Drive Across the City?

<p>Lesson overview:</p> 	<p><b>Purpose:</b> Students will pick a background and then select a character to resize and move across the screen. They are utilizing motion blocks and cardinal directions.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>• Keyboard Practice             <ul style="list-style-type: none"> <li>◦ Choose an online or unplugged keyboard practice game.</li> </ul> </li> <li>• Scratch Jr.: Can I make my car drive across the city?</li> </ul> <p>Consider:</p> <ul style="list-style-type: none"> <li>• How would you make the car go only halfway across the screen?</li> <li>• What would happen if a wizard, a dragon, or an elephant appeared on the sidewalk? (If these are teacher prompts, move to overview or time.)</li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Scratch Jr. Drive Across the City</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>• Practice differentiating predefined actions and event-driven ones</li> <li>• Recognize actions of the teacher as signals to initiate commands</li> <li>• Repeat commands given by an instructor</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>• <b>AP.1A.3</b>—Develop programs with sequences and simple loops to express ideas or address a problem.</li> <li>• <b>DA.1A.1</b>—Store, copy, search, retrieve, modify, and delete information using a computing device and define the information stored as data.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>• Keyboard Practice <b>15 min</b></li> <li>• <a href="#">Scratch Jr. Drive Across the City</a> <b>45 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>• Computer</li> <li>• Projector/smartboard with sound</li> <li>• <a href="#">Scratch Jr.</a> account</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>• Computer/tablet with internet access</li> <li>• <a href="#">Scratch Jr.</a> account</li> </ul>
<p>Subject integrated:</p>	<p>ELA Social Studies</p>
<p>Other standards addressed:</p>	<p>ELA</p> <ul style="list-style-type: none"> <li>• <b>SL.1.1</b>—Participate in collaborative conversation</li> <li>• <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>• <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> </ul> <p>Social Studies</p> <ul style="list-style-type: none"> <li>• <b>G.1.3</b>—Recognize maps, graphs, and other representations of Earth. Construct a map from a student's home to school, applying cardinal and intermediate directions.</li> </ul>

Vocabulary:	<p><u>Background</u>: The image on the back screen of a computer or tablet <u>Character</u>: A person, animal, or thing that the topic is about <u>Shrink</u>: To make smaller <u>Grid</u>: Frame with bars running across it that is used to cover an opening</p>
Notes:	<p><a href="#">Scratch, Jr. - Chromebook</a></p>

## Week 22: Coding Basics Unplugged

Lesson overview:



**Purpose:**

Students will review basic programming skills and practice using core coding commands.

**Lesson:**

- Unplugged Activity:
  - Write an algorithm for brushing teeth one day, practice the sequence to look for bugs, debug the next day, then compare the strategies and talk about successes.
- Keyboard Practice
  - Choose an online or unplugged keyboard practice game.
- Introduce the definition of code (No. 4 under the Introduction to Hour of Code section of the lesson)
  - [What Are Computer Programs?](#) video
- Unplugged Activity
  - [My First Code Worksheet Packet](#)
- Wrap Up
  - Debrief and reflect
  - **Debrief:** “Earlier I asked, ‘What does a programmer or a computer scientist do?’ Now, after watching the video and working on our packets, what do you think? What does a programmer do?”
  - **Reflect:** “What was difficult about this packet? What was your favorite worksheet in the packet? What is something you did, or learned, from today’s hour of code that you want to share with your family?” (Move to overview or time.)

Lesson links/resources:

[Kodable: Coding Basics - Unplugged](#)  
[My First Code Worksheet Packet](#)

CS standards addressed:

Students will be able to:

- Create and model algorithms
- Express ideas or address problems by developing programs with a sequence
- Break down the steps needed to solve a problem into a precise sequence of instructions
- Use various strategies to fix problems in algorithms

Standards:

- **AP.1A.1**—Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.
- **AP.1A.3**—Develop programs with sequences and simple loops to express ideas or address a problem.
- **AP.1A.4**—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.
- **AP.1A.7**—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.

Time needed:

**Total Time: 50 min**

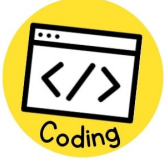
- Keyboard Practice **15 min**
- Kodable Lesson **35 min**
  - Introduction to Coding **5 min**
  - Unplugged Activity **20 min**
  - Wrap Up **10 min**

Materials needed:

Teacher:

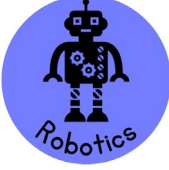

	<ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">Kodable</a> account</li> <li>● <a href="#">Kodable: Coding Basics - Unplugged</a></li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Computer/tablet with internet access</li> <li>● <a href="#">Kodable</a> account</li> <li>● <a href="#">Kodable Unplugged Worksheet Packet</a></li> <li>● <a href="#">Intro to Coding for Kids 4-6 Video</a></li> <li>● Pen/pencil</li> </ul>
Subject integrated:	ELA
Other standards addressed:	<ul style="list-style-type: none"> <li>● <b>RL.1.3</b>—Describe characters, setting, and major events in the story.</li> <li>● <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>● <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>● <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> </ul>
Vocabulary:	<p><u>Code</u>: The programming language that humans create and use to tell computers what to do</p> <p><u>Programmer</u>: A person who writes the code (i.e., language) that tells the computer what to do</p> <p><u>Sequence</u>: Instructions given to the computer to be followed in the exact order they are written and written in code using commands from programmers</p> <p><u>Condition</u>: Allows the program to perform different actions, depending on the condition being true or false</p> <p><u>Loop</u>: A command used to repeat a portion of code</p> <p><u>Function</u>: A set of steps given a simple name that a programmer can easily call on and re-use again in a program</p> <p><u>Function</u>: A set of steps given a simple name that a programmer can easily call on and reuse again in a program</p> <p><u>Variable</u>: A container that stores a value that can change</p>
Notes:	<p>→Teachers will need to create FREE teacher and/or student accounts (when applicable) at <a href="https://www.kodable.com/">https://www.kodable.com/</a>.</p>

## Week 23: Mystery Island Coding Quest

<p>Lesson overview:</p> 	<p><b>Purpose:</b> The Mystery Island Coding Quest by Monster Coding offers a fun-filled self-guided adventure that teaches several key programming concepts to children. Each block-based activity builds on the previous, introducing kids to functions, Boolean values, loops, if/else statements, and arrays using colorful animated graphics and audio instructions.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction             <ul style="list-style-type: none"> <li>○ Explain the difference between true and false.                     <ul style="list-style-type: none"> <li>■ “Today, we will use true and false statements to code.”</li> </ul> </li> <li>○ Quiz the students by asking them true or false statements.</li> <li>○ Show the <a href="#">What are Booleans in Programming?</a> Video.</li> </ul> </li> <li>● Main Activity             <ul style="list-style-type: none"> <li>○ Students will complete the <a href="#">Mystery Island Coding Quest</a>, where they will create a monster, program it to find treasure, and leave the island.</li> </ul> </li> <li>● Exit Ticket             <ul style="list-style-type: none"> <li>○ What did you learn by doing the <a href="#">Mystery Island Coding Quest?</a></li> <li>○ What was the hardest part about this activity?</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">What are Booleans in Programming?</a></li> <li>● <a href="#">Mystery Island Coding Quest</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Use Boolean when coding</li> <li>● Use loops in their code</li> <li>● Use if/else statements in their code</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1A.1</b>—Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.</li> <li>● <b>AP.1A.3</b>—Develop programs with sequences and simple loops to express ideas or address a problem.</li> <li>● <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.</li> <li>● <b>AP.1A.7</b>—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 50 min</b></p> <ul style="list-style-type: none"> <li>● Introduction <b>10 min</b></li> <li>● Main Activity <b>30 min</b></li> <li>● Exit Ticket <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">What are Booleans in Programming?</a></li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Computer/tablet with internet access</li> <li>● <a href="#">Mystery Island Coding Quest</a></li> </ul>
<p>Subject integrated:</p>	<p>ELA Math</p>

Other standards addressed:	<p>ELA</p> <ul style="list-style-type: none"><li>● <b>RI.1.4</b>—Ask and answer questions to help determine or clarify the meaning of words and phrases.</li><li>● <b>SL.1.1</b>—Participate in collaborative conversation.</li><li>● <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li><li>● <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li></ul> <p>Math</p> <ul style="list-style-type: none"><li>● <b>1.OA.1</b>—Use addition and subtraction within 20 to solve word problems involving situations of adding to, taking from, putting together, taking apart, and comparing, with unknowns in all positions, e.g., by using objects, drawings, and equations with a symbol for the unknown number to represent the problem.</li></ul>
Vocabulary:	<p><u>Boolean</u>: A type of data that has only two possible values: true or false</p>
Notes:	<p>Students may get stuck and frustrated. It is important that you talk them through the activity without giving them the answer.</p>

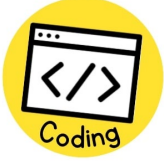
## Week 24: Kodable Hour of Code: Beginner

<p>Lesson overview:</p>  	<p><b>Purpose:</b> Students will create and navigate mazes that involve geometrical aspects.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Direct instruction             <ul style="list-style-type: none"> <li>○ Instructional guidance on lesson plan</li> </ul> </li> <li>● Guided Practice Activity</li> <li>● Independent Practice</li> <li>● Check for Understanding             <ul style="list-style-type: none"> <li>○ What is a programmer?</li> <li>○ What is an example of a program?</li> <li>○ How do computers or computer programs work?</li> <li>○ What happens if we give the computer directions in the wrong order?</li> <li>○ Who is smarter, computers or people?</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">Kodable - Hour of Code: Beginner Lesson Plan</a></li> <li>● <a href="#">Vocabulary Slides</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Compare how people worked and lived before technology.</li> <li>● Talk and write about goals and expected outcomes.</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>IC.1A.1</b>—Compare how people live and work before and after the implementation or adoption of new computing technology.</li> <li>● <b>AP.1A.8</b>—Using correct terminology, describe steps taken and choices made during the iterative process of program development.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Direct Instruction <b>15 min</b></li> <li>● Guided Practice Activity <b>25 min</b></li> <li>● Independent Practice <b>10 min</b></li> <li>● Check for Understanding <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">Kodable</a> account</li> <li>● <a href="#">Kodable - Hour of Code: Beginner Lesson Plan</a></li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Computer/tablet with internet access</li> <li>● <a href="#">Kodable</a> account</li> <li>● <a href="#">Vocabulary Slides</a></li> <li>● Chalkboard, whiteboard, etc.</li> <li>● Markers, chalk, etc.</li> <li>● Vocabulary cards</li> <li>● Floor space</li> <li>● Kodable (web, desktop, iPad, or Android) Available at <a href="http://Kodable.com/download">Kodable.com/download</a></li> </ul>
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards addressed:</p>	<ul style="list-style-type: none"> <li>● <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>● <b>SL.1.2</b>—Ask and answer questions about key details presented</li> </ul>

	<p>through text or orally.</p> <ul style="list-style-type: none"><li>● <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li></ul>
Vocabulary:	<p><u>Code</u>: The programming language that humans create and use to tell computers what to do</p> <p><u>Sequence</u>: Instructions given to the computer to be followed in the exact order they are written and written in code using commands from programmers</p> <p><u>Bug</u>: Part of a program that does not work correctly</p> <p><u>Program</u>: A sequence of instructions given to a computer in code that the computer can understand. The computer follows the instructions and carries out the task</p> <p><u>Command</u>: A specific instruction given to a computer in written code from a programmer</p> <p><u>Debug</u>: Finding and fixing a mistake in the code to allow a program to run as expected</p>
Notes:	

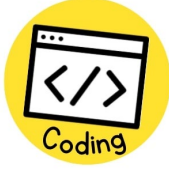


## Week 25: Kodable.com—Maze Maker

<p>Lesson overview:</p> 	<p><b>Purpose:</b> Students will create and navigate mazes that involve geometrical aspects.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Keyboard Practice             <ul style="list-style-type: none"> <li>○ Choose an online or unplugged keyboard practice game.</li> </ul> </li> <li>● Maze Maker Challenge             <ul style="list-style-type: none"> <li>○ Introduction to Hour of Code</li> <li>○ Maze Maker Challenge                 <ul style="list-style-type: none"> <li>■ Student will complete five levels.</li> </ul> </li> <li>○ Make Your Own Maze                 <ul style="list-style-type: none"> <li>■ Once the students complete the levels, they will be prompted to create their own maze. Let them create their own maze and test it as many times as they'd like.</li> </ul> </li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">Kodable: Maze Maker</a></li> <li>● <a href="#">Kodable: Maze Maker Game</a></li> <li>● <a href="#">How to Make a Maze Video</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Develop and visually illustrate the plan of what a program will do</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1A.5</b>—Develop plans that describe a program's sequence of events, goals, and expected outcomes.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 55 min</b></p> <ul style="list-style-type: none"> <li>● Keyboard Practice <b>15 min</b></li> <li>● Maze Maker Challenge             <ul style="list-style-type: none"> <li>○ Introduction to Hour of Code <b>10 min</b></li> <li>○ Maze Maker Challenge (levels) <b>10 min</b></li> <li>○ Make your own maze <b>20 min</b></li> </ul> </li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">Kodable</a> account</li> <li>● <a href="#">Kodable: Maze Maker</a></li> <li>● <a href="#">How to Make a Maze Video</a></li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Computer/tablet with internet access</li> <li>● <a href="#">Kodable</a> account</li> <li>● <a href="#">Kodable: Maze Maker</a></li> <li>● <a href="#">Maze Maker Challenge handout</a></li> </ul>
<p>Subject integrated:</p>	<p>ELA Math</p>
<p>Other standards addressed:</p>	<p>ELA</p> <ul style="list-style-type: none"> <li>● <b>SL.1.1</b>—Participate in collaborative conversations.</li> <li>● <b>SL.1.2</b>—Ask and answer questions about key ideas and details.</li> </ul> <p>Math</p> <ul style="list-style-type: none"> <li>● <b>1.G.3</b>—Partition circles and rectangles into two and four equal shares; describe the shares using the words halves, fourths, and quarters; and use the phrases half of, fourth of, and quarter of.</li> </ul>

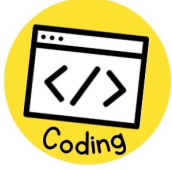
	<p>Describe the whole as two of, or four of the shares. Understand for these examples that decomposing into more equal shares creates smaller shares.</p>
<p>Vocabulary:</p>	<p><u>Code</u>: The programming language that humans create and use to tell computers what to do  <u>Sequence</u>: Instructions given to the computer to be followed in the exact order they are written and written in code using commands from programmers  <u>Bug</u>: Part of a program that does not work correctly  <u>Program</u>: A sequence of instructions given to a computer in code that the computer can understand. The computer follows the instructions and carries out the task  <u>Command</u>: A specific instruction given to a computer in written code from a programmer  <u>Debug</u>: Finding and fixing a mistake in the code to allow a program to run as expected</p>
<p>Notes:</p>	<p><b>How to Access the Maze Maker Activity</b>  <u>On the iOS app:</u></p> <ul style="list-style-type: none"> <li>● Open the Kodable app.</li> <li>● If learners have a class code, follow the usual steps to log in by clicking "School Profile" and entering the class code. If learners do NOT have a class code, click the Hour of Code button.</li> <li>● Click the Hour of Code beach hut on Fuzztopia.</li> <li>● Click the "Make Levels" button and choose the grade level.</li> </ul> <p><u>On the web:</u></p> <ul style="list-style-type: none"> <li>● Go to <a href="https://kodable.com/hour-of-code">kodable.com/hour-of-code</a>.</li> <li>● Scroll to "Make your Own Mazes" and click the button "Play Online."</li> <li>● If learners have a class code, follow the usual steps to log in by clicking "School Profile" and entering the class code. If learners do not have a class code, click the "Play Without Saving" button.</li> <li>● Select the "Make Levels" button and choose the grade level for the first grade.</li> </ul>

## Week 26: Kodable.com—Show What You Know!

<p>Lesson overview:</p> 	<p><b>Purpose:</b> Students will design, draw, and solve their own programming problems.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>• Keyboard Practice             <ul style="list-style-type: none"> <li>◦ Choose an online or unplugged keyboard practice game.</li> </ul> </li> <li>• Vocabulary Review</li> <li>• Show What You Know!             <ul style="list-style-type: none"> <li>◦ Show What You Know Maze Maker Activity!</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">Show What You Know! lesson plan</a></p>
<p>CS standards addressed:</p>	<p>Students be able to:</p> <ul style="list-style-type: none"> <li>• Express ideas and address problems by developing programs with sequence and loops</li> <li>• Break down the steps needed to solve a problem</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>• <b>AP.1A.3</b>—Develop programs with sequences and simple loops to express ideas or address a problem.</li> <li>• <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 55 min</b></p> <ul style="list-style-type: none"> <li>• Keyboarding Practice <b>15 min</b></li> <li>• Show What You Know! Activity <b>40 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>• Computer</li> <li>• Projector/smartboard with sound</li> <li>• <a href="#">Kodable</a> account</li> <li>• <a href="#">Show what you know! lesson plan</a></li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>• Computer/tablet with internet access</li> <li>• <a href="#">Kodable</a> account</li> <li>• Pencil</li> <li>• Scissors</li> <li>• Glue stick</li> <li>• fuzzFamily Maze Maker handout</li> <li>• Coloring supplies</li> </ul>
<p>Subject integrated:</p>	<p>ELA Social Studies</p>
<p>Other standards addressed:</p>	<p>ELA</p> <ul style="list-style-type: none"> <li>• <b>SL.1.1</b>—Participate in collaborative conversations.</li> <li>• <b>SL.1.6</b>—Produce complete sentences.</li> <li>• <b>L.1.6</b>—Use words and phrases acquired through conversations.</li> </ul> <p>Social Studies</p> <ul style="list-style-type: none"> <li>• <b>G.1.3</b>— Recognize maps, graphs, and other representations of Earth. Construct a map from a student's home to school, applying cardinal and intermediate directions.</li> </ul>
<p>Vocabulary:</p>	<p><u>Code</u>: Language that programmers create and use to tell a computer what</p>

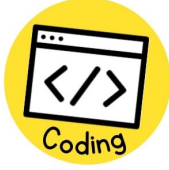
	<p>to do</p> <p><u>Sequence</u>: Instructions given to the computer to be followed in the exact order they are written and written in code using commands from programmers</p> <p><u>Bug</u>: A mistake in the code that prevents a program from running as expected</p> <p><u>Program</u>: A sequence of instructions given to a computer in code that the computer can understand. The computer follows the instructions and carries out the task</p> <p><u>Command</u>: A specific instruction given to a computer in written code from a programmer</p> <p><u>Debug</u>: Finding and fixing a mistake in the code to allow a program to run as expected</p>
Notes:	<p>If your students have unlocked the Maze Maker in Kodable, encourage them to bring their maze to life.</p> <ul style="list-style-type: none"><li>● Students log in to their Kodable accounts.</li><li>● Click on the orange "Create" button to access the Maze Maker.</li><li>● Follow the draft completed with the maze design to build their maze.</li><li>● Students decorate, add stars, and test their mazes.</li></ul> <p>Save completed mazes to the class and have students try each other's mazes.</p>

## Week 27: Kodable.com—If Flash, Then Clap

<p>Lesson overview:</p> 	<p><b>Purpose:</b> Students will explore conditional statements using critical-thinking skills and incorporating science.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>• Direct Instruction             <ul style="list-style-type: none"> <li>◦ Guidance provided on lesson plan</li> </ul> </li> <li>• If Lightning, Then Thunder!             <ul style="list-style-type: none"> <li>◦ A storm is coming! Students will examine the logic behind thunder and lightning and relate this condition to programming.</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>• <a href="#">If Flash, Then Clap! Lesson Plan</a></li> <li>• <a href="#">Lesson Resources</a></li> <li>• <a href="#">IF Statements Video</a></li> <li>• <a href="#">What Causes Thunder and Lightning? video</a></li> </ul>
<p>CS standards addressed:</p>	<p>Student will be able to:</p> <ul style="list-style-type: none"> <li>• Determine the effect of a condition being true</li> <li>• Connect real-world conditions with “if” statements in programming</li> <li>• Create “if” statements to describe real-world cause and effect.</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>• <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 50 min</b></p> <ul style="list-style-type: none"> <li>• Direct Instruction <b>15 min</b></li> <li>• If Lightning, Then Thunder! <b>35 min</b></li> <li>• Optional On-Screen Practice <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>• Computer</li> <li>• Projector/smartboard with sound</li> <li>• <a href="#">Kodable</a> account</li> <li>• <a href="#">If Flash, Then Clap! Lesson Plan</a></li> <li>• <a href="#">Lesson Resources</a></li> <li>• <a href="#">IF Statements Video</a></li> <li>• <a href="#">What Causes Thunder and Lightning? video</a></li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>• Computer/tablet with internet access</li> <li>• <a href="#">Kodable</a> account</li> <li>• Code.org video: <a href="#">Conditional Statements</a></li> <li>• <a href="#">What is Thunder and Lightning? video</a></li> <li>• Decision tree graphic organizer</li> <li>• Vocabulary cards</li> <li>• 1 fluorescent light bulb</li> <li>• 1 rubber balloon</li> <li>• Brown paper bags for each student</li> <li>• Exit Ticket: Weather “if” statements</li> </ul>
<p>Subject integrated:</p>	<p>Math Science</p>
<p>Other standards addressed:</p>	<p>Math</p> <ul style="list-style-type: none"> <li>• <b>1.NBT.A.1</b>—Count to 120 starting at any number.</li> </ul>

	<p>Science</p> <ul style="list-style-type: none"><li>● <b>E.1.9a</b>—Students will demonstrate an understanding of the patterns of weather by describing, recording, and analyzing weather data to answer questions about daily and seasonal weather patterns.</li></ul>
Vocabulary:	<p><u>Sequence</u>: Instructions given to the computer to be followed in the exact order they are written and written in code using commands from programmers</p> <p><u>Condition</u>: A statement that tells a program to run in a certain way, only if certain conditions are met. Conditional statements are "if, then" statements: If a condition is true, then something happens.</p> <p><u>Program</u>: A sequence of instructions given to a computer in code that the computer can understand. The computer follows the instructions and carries out the task</p> <p><u>Programmer</u>: A person who writes the code (i.e., language) that tells the computer what to do</p> <p><u>Code</u>: The programming language that humans create and use to tell computers what to do</p> <p><u>Command</u>: A specific instruction given to a computer in written code from a programmer</p> <p><u>If statement</u>: A logic statement used in programming that allows a computer program to act differently each time it is executed, depending on whether an input is evaluated to be true or false</p>
Notes:	

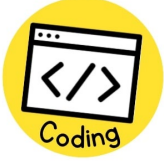

## Week 28: Kodable.com—Beach Cleanup

<p>Lesson overview:</p> 	<p><b>Purpose:</b> Students will be able to design and create mazes based on preexisting obstacles. Students will then write simple programs to solve mazes using basic coding concepts. Next, the students will examine ways technology can be used to solve real-world problems.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction             <ul style="list-style-type: none"> <li>○ Students will watch the <a href="#">Tommy the SudBudz Turtle - Listen Up Yo! Keep the Oceans Clean :)</a> video.</li> </ul> </li> <li>● Brainstorm             <ul style="list-style-type: none"> <li>○ The teacher will talk about types of pollution, and students will fill out the “Brainstorm” section of the <a href="#">Lesson Worksheet</a>.</li> </ul> </li> <li>● Beach Cleanup             <ul style="list-style-type: none"> <li>○ The students will complete the <a href="#">Beach Cleanup</a> coding activity.</li> <li>○ Note: Students will need to create the path all the way across the screen in order to be able to play their game.</li> </ul> </li> <li>● Wrap Up             <ul style="list-style-type: none"> <li>○ Students will complete the wrap-up portion of the <a href="#">Lesson Worksheet</a>.</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">Kodable Lesson Frame</a></li> <li>● <a href="#">Beach Cleanup</a></li> <li>● <a href="#">Lesson Worksheet</a></li> <li>● <a href="#">Tommy the SudBudz Turtle - Listen Up Yo! Keep the Oceans Clean :)</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Design and create mazes based on preexisting obstacles</li> <li>● Write simple programs to solve mazes using basic coding concepts</li> <li>● Examine ways technology can be used to solve real-world problems</li> <li>● Collaborate and communicate effectively with peers</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1A.3</b>—Develop programs with sequences and simple loops to express ideas or address a problem.</li> <li>● <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.</li> <li>● <b>AP.1A.7</b>—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 46-60 min</b></p> <ul style="list-style-type: none"> <li>● Intro Video <b>6 min</b></li> <li>● Group Brainstorm <b>15 min</b></li> <li>● Coding Activity <b>15 min</b></li> <li>● Wrap Up <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">Kodable</a> account</li> <li>● <a href="#">Kodable Lesson Frame</a></li> <li>● <a href="#">Tommy the SudBudz Turtle - Listen Up Yo! Keep the Oceans Clean :)</a></li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Computer/tablet with internet access</li> <li>● <a href="#">Kodable</a> account</li> </ul>

	<ul style="list-style-type: none"> <li>• <a href="#">Beach Cleanup</a></li> <li>• <a href="#">Lesson Worksheet</a></li> <li>• Pens/pencils</li> </ul>
Subject integrated:	ELA Science
Other standards addressed:	ELA <ul style="list-style-type: none"> <li>• <b>RF.1.1</b>—Demonstrate understanding of the organization and basic features of print.</li> <li>• <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>• <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>• <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> <li>• <b>SL.1.5</b>—Add drawing or other visual displays to descriptions when appropriate to clarify ideas, thoughts, and feelings.</li> </ul> Science <ul style="list-style-type: none"> <li>• <b>E.1.10</b>—Students will demonstrate an understanding of human dependence on clean and renewable water resources.</li> </ul>
Vocabulary:	<p><u>Program</u>: A sequence of instructions given to a computer in code that the computer can understand. The computer follows the instructions and carries out the task.</p> <p><u>Sequence</u>: Instructions given to the computer to be followed in the exact order they are written and written in code using commands from programmers</p> <p><u>Condition</u>: Allows the program to perform different actions, depending on the condition being true or false</p> <p><u>Loop</u>: A command used to repeat a portion of code</p> <p><u>Function</u>: A set of steps given a simple name that a programmer can easily call on and re-use again in a program</p> <p><u>Debug</u>: Finding and fixing a mistake in the code to allow a program to run as expected</p>
Notes:	

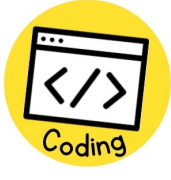



## Week 29: Kodable.com, Sequence 1—Introduction

<p>Lesson overview:</p>  	<p><b>Purpose:</b> Students will be able to explain what a programmer does. Students will be able to make a robot move forward, spin, and jump using basic programming language.</p> <p>Lesson:</p> <ul style="list-style-type: none"> <li>• Direct Instruction:             <ul style="list-style-type: none"> <li>◦ Introduce vocabulary.</li> </ul> </li> <li>• Guided Practice:             <ul style="list-style-type: none"> <li>◦ Students will act as programmers.</li> </ul> </li> <li>• Independent Practice</li> <li>• Check for Understanding</li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>• <a href="#">Kodable Lesson Frame</a></li> <li>• <a href="#">Lesson Resources</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>• Create and follow algorithms</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>• <b>AP.1A.1</b>—Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>• Direct Instruction <b>15 min</b></li> <li>• Guided Practice <b>25 min</b></li> <li>• Independent Practice <b>10 min</b></li> <li>• Wrap Up <b>10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>• Computer</li> <li>• Projector/smartboard with sound</li> <li>• <a href="#">Kodable</a> account</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>• Markers, chalk, etc.</li> <li>• <a href="#">Lesson Worksheet</a></li> </ul>
<p>Subject integrated:</p>	<p>ELA Social Studies</p>
<p>Other standards addressed:</p>	<p>ELA:</p> <ul style="list-style-type: none"> <li>• <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>• <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>• <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> <li>• <b>SL.1.5</b>—Add drawing or other visual displays to descriptions when appropriate to clarify ideas, thoughts, and feelings.</li> </ul> <p>Math</p> <ul style="list-style-type: none"> <li>• <b>1.G.3</b>—Partition circles and rectangles into two and four equal shares, describe the shares using the words halves, fourths, and quarters, and use the phrases half of, fourth of, and quarter of. Describe the whole as two of or four of the shares. Understand for these examples that decomposing into more equal shares creates</li> </ul>

	smaller shares.
Vocabulary:	<p><u>Programmer</u>: A person who writes the code (i.e., language) that tells the computer what to do</p> <p><u>Code</u>: The programming language that humans create and use to tell computers what to do</p> <p><u>Sequence</u>: The order of events. Computers must be given instruction in the correct order and will carry out commands exactly as they are written.</p> <p><u>Computer</u>: A device that stores data, processes information, and performs tasks, under the control of a set of instructions called a program.</p>
Notes:	

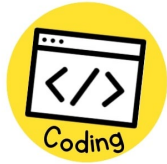
## Week 30: Kodable—Pizza Party!

<p>Lesson overview:</p>  	<p><b>Purpose:</b> Students will practice creating a sequence. Students will be able to write simple numerical expressions and evaluate them in proper sequence.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>• The students will build a sequence in code to create their slices of pizza. Each day consists of a 20-25 min session.             <ul style="list-style-type: none"> <li>○ Day 1: Vocabulary, I follow a sequence when I..., Exit Ticket</li> <li>○ Day 2: Brush Teeth Algorithm, Make a Pizza, Exit Ticket</li> <li>○ Day 3: Inquiry Sheet, Exit Ticket</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>• <a href="#">Kodable Lesson Frame</a></li> <li>• <a href="#">Pizza Party Worksheets</a></li> <li>• <a href="#">Dominic's Pizza Party</a></li> <li>• Sequence Sector Lesson 1: "1, 2, 3 Roll" 1.1-1.5 <a href="#">Optional On-Screen Practice</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>• Create and follow algorithms</li> <li>• Break down the steps needed to solve a problem</li> <li>• Develop a plan to illustrate what a program will do</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>• <b>AP.1A.1</b>—Model daily processes by creating and following algorithms.</li> <li>• <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.</li> <li>• <b>AP.1A.5</b>—Develop plans that describe a program's sequence of events, goals, and expected outcomes.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 55-65 min</b> This lesson can be broken down into three days as indicated in the Kodable lesson framework.</p> <p><u>Day 1 (20-25 min)</u></p> <ul style="list-style-type: none"> <li>• Vocabulary cards with visuals and definitions <b>5-10 min</b></li> <li>• Sequence sentence frame graphic organizer <b>10 min</b></li> <li>• Exit Ticket <b>5 min</b></li> </ul> <p><u>Day 2 (20 min)</u></p> <ul style="list-style-type: none"> <li>• Dominic's Pizza video <b>2 min</b></li> <li>• "Brush teeth" algorithm pseudocode visual <b>3 min</b></li> <li>• Pizza ingredients (Build your own!) <b>10 min</b></li> <li>• Exit Ticket: School algorithm <b>5 min</b></li> </ul> <p><u>Day 3 (15-20 min)</u></p> <ul style="list-style-type: none"> <li>• Inquiry sheet <b>5 min</b></li> <li>• Exit Ticket: K-W-L <b>5 min</b></li> <li>• Kodable on-screen practice <b>5-10 min</b></li> </ul>
<p>Materials needed:</p>	<p>Students:</p> <ul style="list-style-type: none"> <li>• <a href="#">Pizza Party Worksheets</a></li> <li>• Vocabulary cards with visuals and definitions</li> <li>• Sequence sentence frame graphic organizer</li> <li>• Exit Ticket: Emoji</li> </ul>

	<ul style="list-style-type: none"> <li>● Dominic's Pizza video</li> <li>● "Brush teeth" algorithm pseudocode visual</li> <li>● Pizza algorithm pseudocode graphic organizer</li> <li>● Pizza ingredients (Build your own!)</li> <li>● Exit Ticket: School algorithm</li> <li>● Inquiry sheet</li> <li>● Exit Ticket: K-W-L</li> <li>● Optional: Kodable on-screen practice</li> </ul>
Subject integrated:	ELA Math
Other standards addressed:	<p>ELA:</p> <ul style="list-style-type: none"> <li>● <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>● <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>● <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> <li>● <b>SL.1.5</b>—Add drawing or other visual displays to descriptions when appropriate to clarify ideas, thoughts, and feelings.</li> </ul> <p>Math:</p> <ul style="list-style-type: none"> <li>● <b>1.OA.2</b>—Solve word problems that call for addition of three whole numbers whose sum is less than or equal to 20, e.g., by using objects, drawings, and equations with a symbol for the unknown number to represent the problem.</li> </ul>
Vocabulary:	<p><u>Programmer</u>: A person who writes the code (i.e., language) that tells the computer what to do.  <u>Program</u>: A sequence of instructions given to a computer in code that the computer can understand. The computer follows the instructions and carries out the task.  <u>Language</u>: A way of communicating ideas or feelings through sounds, symbols, signs, or words. There are thousands of languages in the world.  <u>Communication</u>: The act of using words, signs, sounds, or symbols to exchange information to someone else  Code: The language written by humans that gives instructions to a computer  <u>Command</u>: A specific instruction given to a computer in written code from a programmer  <u>Sequence</u>: The order of events. Computers must be given instruction in the correct order and will carry out commands exactly as they are written.  <u>Algorithm</u>: A sequence of steps, followed in order, to finish a task that can be performed with or without a computer.  <u>Bug</u>: Part of a program that does not work correctly  <u>Debugging</u>: Finding and fixing problems in an algorithm or program</p>
Notes:	This lesson is developed to take place over the course of three days (20-25 min lesson each). You can modify it to take place on one day if needed.

## Week 31: Kodable—Hour of Code

Lesson overview:



### **Purpose:**

What's your favorite dance move? Help the fuzzFamily choreograph a dance using loops to repeat favorite moves. Keep the directions short.

### **Lesson:**

- Introduction
  - Students will participate in a think-pair-share activity to activate prior knowledge.
- Guided Practice
  - Teacher (say): "The fuzzFamily is having a party for all their fuzzy friends. Help the fuzzFamily choreograph and perform a dance at their party!"
  - Using the provided sheet, choreograph a dance to complete as a class. To get started, come up with three dance moves for your routine, and write them in the blank spaces on the provided sheet. Practice these moves with your students beforehand to ensure they are familiar with them before you attempt the full routine.
  - Now it is time to run through the first iteration of your dance! Run through your choreographed dance once and explain to your students that this is called an iteration, or a single pass through a loop. In computer programming, simple loops contain a few iterations, and more complex loops can contain multiple iterations. If there are any problems with the choreography, debug your dance iteration so it is ready to loop.
  - After you have completed the first iteration of your dance, perform the entire choreographed dance by repeating the iteration three times as a class. Following the successful completion of your dance, explain to your students that the dance repeated three times because it was part of a loop.
- Independent Practice
  - Students complete Smeeborg Loopy Lagoon, "Loopy Lessons" 3.1-3.5 independently on their devices.
- Exit Ticket
  - **What** did we learn today?
  - So **What**? (Explain the relevance or usefulness.)
  - Now **What**? (How does this build on what we already know and fit into what we're learning?)

Lesson links/resources:

- [Kodable Lesson Frame](#)
- [Kodable Lesson Resources](#)
- [Smeeborg Loopy Lagoon, "Loopy Lessons" 3.1-3.5](#)

CS standards addressed:

Students will be able to:



- Define a loop
- Define and exercise an iteration
- Apply loops off-screen
- Identify where code repeats and appropriately apply a loop to their code to run a program

Standards:

- **AP.1A.3**—Develop programs with sequences and simple loops to express ideas or address a problem.
- **AP.1A.4**—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.

	<ul style="list-style-type: none"> <li>● <b>AP.1A.8</b>—Using correct terminology, describe steps taken and choices made during the iterative process of program development.</li> <li>● <b>AP.1A.7</b>—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.</li> </ul>
Time needed:	<p><b>Total Time: 45-60 min</b></p> <ul style="list-style-type: none"> <li>● Think-Pair-Share <b>15 min</b></li> <li>● Guided Practice <b>15 min</b></li> <li>● Independent Practice <b>15 min</b></li> <li>● Exit Ticket <b>15 min</b></li> </ul>
Materials needed:	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">Kodable</a> account</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Computer/tablet with internet access</li> <li>● <a href="#">Kodable</a> account</li> <li>● <a href="#">Vocab and image cards</a></li> <li>● <a href="#">fuzzFamily Dance Party handout</a></li> <li>● Music</li> <li>● Floor space</li> <li>● Kodable on-screen: web, iPad, Android, desktop</li> <li>● Exit Ticket handout</li> </ul>
Subject integrated:	<p>ELA Math</p>
Other standards addressed:	<p>ELA:</p> <ul style="list-style-type: none"> <li>● <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>● <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>● <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> <li>● <b>SL.1.5</b>—Add drawing or other visual displays to descriptions when appropriate to clarify ideas, thoughts, and feelings.</li> </ul> <p>Math:</p> <ul style="list-style-type: none"> <li>● <b>1.OA.1</b>—Use addition and subtraction to solve word problems.</li> </ul>
Vocabulary:	<p><u>Loop</u>: A programming element that repeats a portion of code a set number of times until the desired process is complete</p> <p><u>Iteration</u>: The act of repeating a process. Each pass through a loop is an iteration.</p> <p><u>Program</u>: A sequence of instructions given to a computer in code that the computer can understand. The computer follows the instructions and carries out the task</p> <p><u>Code</u>: The programming language that humans create and use to tell computers what to do</p> <p><u>Command</u>: A specific instruction given to a computer in written code from a programmer</p> <p><u>Bug</u>: A mistake in the code that prevents a program from running as expected</p> <p><u>Debugging</u>: Finding and fixing problems in an algorithm or program</p>
Notes:	

## Week 32: How Does Technology Make You Feel?

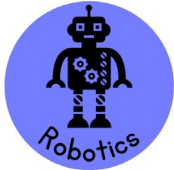
<p>Lesson overview:</p>  	<p><b>Purpose:</b> This foundational digital citizenship lesson challenges kids to pay attention to their feelings while using tech. With an engaging emoji game, students learn practical strategies for managing their feelings—good, bad, and everything in between.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Explore: How Do You Feel?</li> <li>● Play: The Emoji Game</li> <li>● Read</li> <li>● Pause and Think</li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">CommonSense.org Lesson Framework</a></li> <li>● <a href="#">Lesson Slides</a></li> <li>● <a href="#">How Technology Makes You Feel video</a></li> <li>● <a href="#">Emoji Handout</a></li> <li>● <a href="#">Pause and Think Moment Handout</a></li> <li>● <a href="#">Poem Poster</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Compare how people lived and worked before technology</li> <li>● Demonstrate how to work online with others respectfully</li> <li>● Understand that some login details and personally identifying information should be kept private</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>IC.1A.1</b>—Compare how people live and work before and after the adaptation of new computing technology.</li> <li>● <b>IC.1A.2</b>—Work respectfully and responsibly with others online.</li> <li>● <b>IC.1A.3</b>—Keep login information private and log off devices appropriately.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Explore <b>5 min</b></li> <li>● Emoji Game <b>15 min</b></li> <li>● Read <b>5 min</b></li> <li>● Pause and Think <b>5 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">Common Sense</a> account</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Computer/tablet with internet access</li> <li>● <a href="#">Common Sense</a> account</li> <li>● Popsicle sticks</li> <li>● Tape</li> <li>● Scissors</li> <li>● Crayons or markers</li> </ul>
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards</p>	<p><b>L.1.1</b>—Produce and expand simple and compound sentences.</p>

addressed:	
Vocabulary:	<u>Pause</u> : To stop what you're doing or saying <u>Uncomfortable</u> : Causing a feeling of hurt or worry
Notes:	



## Week 33: Unplugged Robotic Activity

Lesson overview:



### **Purpose:**

In this lesson, students will learn the basics of using robotics. This is an unplugged activity to practice the basics of coding a robot with simple directions: left, right, up, and down.

### **Lesson:**

- Setup
  - Divide the students into groups of 4-5. Each player will select a role.
    - Roles: programmer, robot, maze setup, etc.
  - Select an area of the classroom/space to use as a maze.
  - Select an area that will serve as your grid.
    - You can use the tiles on your floor, or you can tape off an area for students.
    - Try to make your grid a 10x10 at least.
  - Distribute the card decks for the groups.
  - Have the students put the question in the top left box of the grid. They will place index cards with potential answers on the grid.
- Activity
  - Each group will create a robot maze where the start box of the grid will be the question, and the group will place the answer somewhere in the grid.
  - The students will then create a maze where other groups will write a code to get to the correct answer.
    - Example: The (start) question: What color is a banana?
    - A code that would reach the correct answer would be:




Start				Red
		Orange		Purple
		Yellow	Blue	

- Allow the groups to write a program for each maze and use a codable robot (Some examples include: [Code and Go Mouse](#), [Botley](#), [Dash](#)).
- If you do not have access to codable robots, the students can take turns acting as the robot.
- The students will use [coding cards](#) to map out their program before running it.
- Card Decks
  - Index cards with simple questions the students can independently read
    - These questions can be vocabulary, math problems, social studies, science, reading, etc.
- Roles
  - Designer: Will designate a starting and stopping point, like a rectangle or square area, and create an established route to the endpoint

	<ul style="list-style-type: none"> <li>○ <u>Robot</u>: Will be the student moving</li> <li>○ <u>Programmer</u>: Draws a card, discusses the question with the group, and answers the question, then decides the directions of how to navigate the maze to reach the ending point</li> <li>○ <u>Debugger</u>: Will help check the answers for each question and tell the robot when to move</li> <li>○ <u>Timer</u>: Will be over the overall time for the lesson: 5 min, 10 min, etc.</li> </ul>
Lesson links/resources:	<ul style="list-style-type: none"> <li>● Robot (Some examples include: <a href="#">Code and Go Mouse</a>, <a href="#">Botley</a>, <a href="#">Dash</a>)</li> <li>● <a href="#">Coding cards</a></li> </ul>
CS standards addressed:	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Create and follow algorithms</li> <li>● Develop and illustrate the plan for what a program will do</li> <li>● Use various strategies and steps following an algorithm</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1A.1</b>—Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.</li> <li>● <b>AP.1A.5</b>—Develop plans that describe a program’s sequence of events, goals, and expected outcomes.</li> <li>● <b>AP.1A.7</b>—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.</li> </ul>
Time needed:	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up <b>15 min</b></li> <li>● Main Activity <b>15 min</b></li> <li>● Independent Practice <b>15 min</b></li> <li>● Wrap Up <b>15 min</b></li> </ul>
Materials needed:	<p>Teachers</p> <ul style="list-style-type: none"> <li>● Question cards (teacher generated)</li> <li>● Robot (Some examples include: <a href="#">Code and Go Mouse</a>, <a href="#">Botley</a>, <a href="#">Dash</a>)</li> </ul> <p>Students</p> <ul style="list-style-type: none"> <li>● Answer key</li> <li>● Notecards</li> <li>● Grid area for “Maze”</li> <li>● <a href="#">Coding cards</a></li> </ul>
Subject integrated:	ELA
Other standards addressed:	<ul style="list-style-type: none"> <li>● <b>L.1.1</b>—Produce and expand simple and compound sentences.</li> <li>● <b>RF.3</b>—Know and apply grade-level phonics to decode unknown words.</li> <li>● <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>● <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>● <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> </ul>
Vocabulary:	

Notes:	
--------	--



## Week 34: Unplugged Candy Data Activity

<p>Lesson overview:</p> 	<p><b>Purpose:</b> In the lesson, your students will analyze and interpret data.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction <ul style="list-style-type: none"> <li>○ Students will learn about data by watching <a href="#">Data!</a></li> <li>○ Teachers will discuss pictographs and how they can use the same information for a bar graph.</li> </ul> </li> <li>● Candy Data Activity <ul style="list-style-type: none"> <li>○ Students will sort and analyze the colors in a bag of M&amp;M's or Skittles.</li> <li>○ Place the students in small groups or complete the activity individually. Sort the candy into the individual colors and count the total.</li> <li>○ Stop at the midpoint and discuss the following: <ul style="list-style-type: none"> <li>■ What is the most common color?</li> <li>■ What color do you have the least amount of?</li> <li>■ If you opened a new bag, what color would most likely be the most popular?</li> </ul> </li> <li>○ Continue the activity and create a bar graph for each group/student. Then create a class bar graph.</li> <li>○ Once your class completes the bar graph, ask the following questions: <ul style="list-style-type: none"> <li>■ Do you think a computer would be able to sort the candy by color?</li> <li>■ Do you think a computer could notice a pattern to figure out the most popular color?</li> <li>■ How do you think computers notice patterns?</li> </ul> </li> </ul> </li> <li>● Machine Learning <ul style="list-style-type: none"> <li>○ Students will watch <a href="#">What is Machine Learning?</a></li> <li>○ Talk with students about what they learned from the video.</li> </ul> </li> <li>● Exit Ticket <ul style="list-style-type: none"> <li>○ How does a computer learn? (Students will write their answers on a piece of paper or in a journal.)</li> </ul> </li> </ul>
Lesson links/resources:	<ul style="list-style-type: none"> <li>● <a href="#">Data!</a> (Video)</li> <li>● <a href="#">What is Machine Learning?</a> (Video)</li> </ul>
CS standards addressed:	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Analyze data in visual formats</li> <li>● Identify patterns and make predictions based on the patterns</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>DA.1A.3</b>—Identify and describe patterns in data visualizations, such as charts or graphs, to make predictions.</li> </ul>
Time needed:	<p><b>Total Time: 50 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up <b>5 min</b></li> <li>● Candy Data Activity <b>20 min</b></li> </ul>

	<ul style="list-style-type: none"> <li>• Independent Practice <b>15 min</b></li> <li>• Machine Learning Video and Discussion <b>5 min</b></li> <li>• Exit Ticket <b>5 min</b></li> </ul>
Materials needed:	<p>Students:</p> <ul style="list-style-type: none"> <li>• Individual candies or bag of M&amp;M's</li> <li>• Recording paper</li> <li>• Pencils</li> <li>• Bar graph template</li> </ul>
Subject integrated:	<p>ELA Math</p>
Other standards addressed:	<p>ELA:</p> <ul style="list-style-type: none"> <li>• <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>• <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>• <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> </ul> <p>Math:</p> <ul style="list-style-type: none"> <li>• <b>1.MD.4</b>—Organize and represent data with up to three categories.</li> </ul>
Vocabulary:	<p><u>Sequencing</u>: Putting commands in the correct order so the computer can read the commands</p>

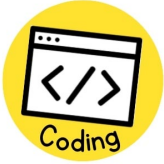

Notes:

## Week 35: Design Your Own Computer

<p>Lesson overview:</p>  	<p><b>Purpose:</b> In this lesson, students will use a variety of materials, including printouts, to create their own laptop. Students will be given a budget and must purchase their materials to make their computers. By adding a budget, students will explore the use of money to make computers and use their math skills to keep up with their own budget.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Introduction             <ul style="list-style-type: none"> <li>○ Review parts of the computer.                     <ul style="list-style-type: none"> <li>■ Mouse/trackpad, keyboard, monitor, speakers, charging/USB ports, on/off switch, etc.</li> </ul> </li> <li>○ <a href="#">Parts of a Computer</a> video</li> </ul> </li> <li>● Main Activity             <ul style="list-style-type: none"> <li>○ Explain to students they have a \$2 budget and have to buy the parts needed for their laptop. (Teachers will need to print <a href="#">Build Your Own Computer- Images</a> and cut them out for students to buy.)                     <ul style="list-style-type: none"> <li>■ Laptop Frame: Manila file folder, construction paper, or poster board</li> <li>■ Keyboard</li> <li>■ Monitor</li> <li>■ Speaker</li> <li>■ Trackpad</li> <li>■ Power button</li> <li>■ Background image</li> </ul> </li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>● <a href="#">Build Your Own Computer Lesson Plan</a></li> <li>● <a href="#">Parts of a Computer</a></li> <li>● <a href="#">Budget Sheet</a></li> <li>● <a href="#">Build Your Own Computer- Images</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Identify parts of a computer</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>CS.1A.2</b>—Use appropriate terminology in identifying and describing the function of common physical components of computing systems (hardware).</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 50-60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up <b>10 min</b></li> <li>● Main Activity <b>30-45 min</b></li> <li>● Wrap Up <b>10-15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Computer</li> <li>● Projector/smartboard with sound</li> <li>● <a href="#">Build Your Own Computer Lesson Plan</a></li> <li>● <a href="#">Parts of a Computer</a></li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Manila file folder, construction paper, or poster board</li> <li>● Scissors</li> <li>● <a href="#">Build Your Own Computer- Images</a></li> <li>● Crayons, colored pencils, markers, pencils, pens</li> </ul>

	<ul style="list-style-type: none"> <li>• Glue or tape</li> <li>• <a href="#">Budget Sheet</a> (Use this budget sheet or create your own according to your students' math knowledge.)</li> </ul>
Subject integrated:	Math
Other standards addressed:	<b>1.MD.5a</b> —Identify the value of all U.S. coins (penny, nickel, dime, quarter, half-dollar, and dollar coins). Use appropriate cent and dollar notation (e.g., 25¢, \$1).
Vocabulary:	<p><u>Computer</u>: A device for working with information</p> <p><u>Desktop</u>: Screen that appears if you are not browsing the internet, reading a file or playing a game; your icons are on this screen</p> <p><u>Input</u>: To add information/data into the computer</p> <p><u>Keyboard</u>: Where all the letters, numbers, and other buttons are located; when you type on it, the symbols appear on the monitor</p> <p><u>Laptop</u>: Small portable computer</p> <p><u>Monitor</u>: Screen that shows you what you are doing; a viewer that displays what the computer sends to it</p> <p><u>Mouse</u>: A hand-held device for moving a cursor around the screen <u>Output</u>: To send information/data out of the computer</p> <p><u>Tablet</u>: A wireless, portable personal computer with a touchscreen interface</p>
Notes:	

## Week 36: The Kodable World

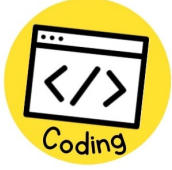

<p>Lesson overview:</p>  	<p><b>Purpose:</b> We follow sequences all the time. We are going to use the sentence puzzle pieces to put words in the correct order to build a sentence that makes sense.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>● Warm Up             <ul style="list-style-type: none"> <li>○ Read “The Kodable World.” Model the skills students are working on: following words from left to right, page to page. Show students what readers do in their head as they read.</li> </ul> </li> <li>● Main Activity             <ul style="list-style-type: none"> <li>○ You will practice sentence organization together as a class. Provide words and have students put them together in the correct order.</li> <li>○ Provide students with a sentence puzzle and have them follow on their own graphic organizers. Hand each student a copy of the Kodable sentence puzzle graphic organizer maze to fill out. Call students up to put the words in the correct sequence and share their completed sentences at the end.</li> </ul> </li> <li>● Wrap Up             <ul style="list-style-type: none"> <li>○ Students will apply what they learned from the lesson to complete Kodable’s sequence sector lessons 1.1-1.5.</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<p><a href="#">The Kodable World</a></p>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>● Express ideas of addressing problems by developing programs with loops</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1A.3A</b>—Students should be able to express ideas or address problems by developing programs with sequences and simple loops.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Warm Up <b>15 min</b></li> <li>● Main Activity <b>15 min</b></li> <li>● Independent Practice <b>15 min</b></li> <li>● Wrap Up <b>15 min</b></li> </ul>
<p>Materials needed:</p>	<p>Students:</p> <ul style="list-style-type: none"> <li>● Individual candies or bag of M&amp;M's</li> <li>● Recording paper</li> <li>● Pencils</li> <li>● Bar graphs template</li> </ul>
<p>Subject integrated:</p>	<p>ELA Math</p>
<p>Other standards addressed:</p>	<p>ELA</p> <ul style="list-style-type: none"> <li>● <b>RF.1.1</b>—Demonstrate understanding of the organization and basic features of print.</li> </ul> <p>Math</p> <ul style="list-style-type: none"> <li>● <b>1.MD.4</b>—Organize and represent data with up to three categories.</li> </ul>
<p>Vocabulary:</p>	<p><u>Sequence</u>: Instructions given to the computer to be followed in the exact</p>



order they are written and written in code using commands from programmers  
Programming: The art of creating a program

Notes:

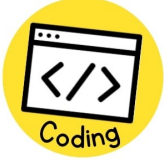

## Week 37: Debugging ELA/Unplugged

<p>Lesson overview:</p>  	<p><b>Purpose:</b> Whether fixing your code, correcting a math problem, or proofreading your writing, debugging is an important part of life. Learning how to spot errors and knowing how to correct them will help prevent bugs.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>• The class will review the definitions of code and debugging.</li> <li>• Students will be given scrambled sentences either in bags or on one sheet of paper.</li> <li>• The students will debug the sentence and rewrite it as correct code.</li> <li>• Self-Reflection: Did you feel frustrated while debugging your code? Explain why debugging was or was not frustrating to you.</li> <li>• To wrap up, students will turn to a shoulder partner and discuss the following:             <ul style="list-style-type: none"> <li>◦ Today you practiced debugging code in sentences. How do you use debugging in other areas of your life?</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>• “Bugged” Sentences—Pick one area to focus on: capitalization, punctuation, etc. (The teacher will create.)</li> <li>• <a href="#">Debugging Video</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>• Identify and locate bugs in a program</li> <li>• Translate movements into a series of commands</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>• <b>AP.1A.2</b>—Model the way programs store and manipulate data by using numbers or other symbols to represent information.</li> <li>• <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 45-60 min</b></p> <ul style="list-style-type: none"> <li>• Review <b>5-10 min</b></li> <li>• Debugging Activity <b>15-20 min</b></li> <li>• Self-Reflection <b>10-15 min</b></li> <li>• Wrap Up <b>10-20 min</b></li> </ul>
<p>Materials needed:</p>	<p>Students:</p> <ul style="list-style-type: none"> <li>• Pencil</li> <li>• Paper</li> <li>• “Bugged” Sentences</li> </ul>
<p>Subject integrated:</p>	<p>ELA</p>
<p>Other standards addressed:</p>	<ul style="list-style-type: none"> <li>• <b>L.1.1</b>—Demonstrate the command and conventions of standard English grammar and usage.</li> <li>• <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>• <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>• <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> </ul>
<p>Vocabulary:</p>	<p><u>Algorithm</u>: A sequence of steps, followed in order, to finish a task that can be performed with or without a computer.</p> <p><u>Bug</u>: Part of a program that does not work correctly</p> <p><u>Debugging</u>: Finding and fixing problems in an algorithm or program</p>

Sequencing: Putting commands in the correct order so the computer can read the commands

Notes:

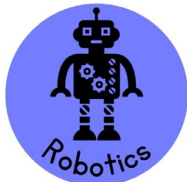
## Week 38: Debugging Math/Unplugged

<p>Lesson overview:</p>  	<p><b>Purpose:</b> Whether fixing your code, correcting a math problem, or proofreading your writing, debugging is an important part of life. Learning how to spot errors and knowing how to correct them will help prevent bugs.</p> <p><b>Lesson:</b></p> <ul style="list-style-type: none"> <li>• The class will review the definitions of code and debugging.</li> <li>• Students will be given scrambled math sentences either in bags or on one sheet of paper.</li> <li>• The students will debug the sentence and rewrite them as correct. Code.</li> <li>• Self-Reflection: Did you feel frustrated while debugging your code? Explain why debugging was or was not frustrating to you.</li> <li>• To wrap up, students will turn to a shoulder partner and discuss the following:             <ul style="list-style-type: none"> <li>◦ Today you practiced debugging code in math sentences. How do you use debugging in other areas of your life?</li> </ul> </li> </ul>
<p>Lesson links/resources:</p>	<ul style="list-style-type: none"> <li>• "Bugged" Sentences—Pick one area to focus on: addition and subtraction (The teacher will create.)</li> <li>• <a href="#">Debugging Video</a></li> </ul>
<p>CS standards addressed:</p>	<p>Students will be able to:</p> <ul style="list-style-type: none"> <li>• Identify and locate bugs in a program</li> <li>• Translate movements into a series of commands</li> </ul> <p>Standards:</p> <ul style="list-style-type: none"> <li>• <b>AP.1A.2</b>—Model the way programs store and manipulate data by using numbers or other symbols to represent information.</li> <li>• <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.</li> </ul>
<p>Time needed:</p>	<p><b>Total Time: 45-60 min</b></p> <ul style="list-style-type: none"> <li>• Review <b>5-10 min</b></li> <li>• Debugging Activity <b>15-20 min</b></li> <li>• Self-Reflection <b>10-15 min</b></li> <li>• Wrap Up <b>10-20 min</b></li> </ul>
<p>Materials needed:</p>	<p>Students:</p> <ul style="list-style-type: none"> <li>• Pencil</li> <li>• Paper</li> <li>• "Bugged" Sentences</li> </ul>
<p>Subject integrated:</p>	<p>ELA Math</p>
<p>Other standards addressed:</p>	<p>ELA</p> <ul style="list-style-type: none"> <li>• <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>• <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>• <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> </ul> <p>Math</p> <ul style="list-style-type: none"> <li>• <b>1.OA.5</b>—Add and subtract within 20.</li> </ul>

Vocabulary:	<p><u>Algorithm</u>: A sequence of steps, followed in order, to finish a task that can be performed with or without a computer.</p> <p><u>Bug</u>: Part of a program that does not work correctly</p> <p><u>Debugging</u>: Finding and fixing problems in an algorithm or program</p> <p><u>Sequencing</u>: Putting commands in the correct order so the computer can read the commands</p>
Notes:	

## Week 39: Geometry Robot Review

Lesson overview:



**Purpose:**




In this lesson, students will answer geometry questions correctly to code the robot to reach the correct destination.

**Lesson:**

- Introduction
  - Review geometry concepts.
- Setup
  - Teachers will create a grid for the robot to travel.
    - The grid could be floor tiles or a taped off area.
  - Teachers will create review questions written on index cards that show defining/non-defining attributes, 2D and 3D shapes, and partition circles/rectangles.
  - Provide answers on separate cards.
  - Place the review question in the start square.
  - Place the correct answer, along with incorrect answers, on the grid.
- Main Activity
  - Students can participate individually or in groups.
  - Students will use [coding cards](#) to map out the code they will use to program the robot to reach the correct answer.
  - Students will need to code the robot (Some examples include: [Code and Go Mouse](#), [Botley](#), [Dash](#)) to reach the correct answer.
  - If you do not have a coding robot in your class, the students can act as the robot.

Example: Which of these is a triangle?



Start			
			
			

- Exit Ticket
  - What was your favorite part of today's activity?
  - Can you draw the shapes we talked about today?

Lesson links/resources:

Robot (Some examples include: [Code and Go Mouse](#), [Botley](#), [Dash](#))  
Materials to create grid.

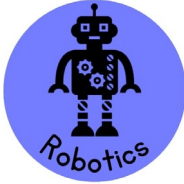
CS standards addressed:

- Students will be able to:
- Solve a math problem correctly and code a mouse robot to gather cheese.

	<p>Standards:</p> <ul style="list-style-type: none"> <li>● <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.</li> <li>● <b>AP.1A.5</b>—Develop plans that describe a program's sequence of events, goals, and expected outcomes.</li> </ul>
Time needed:	<p><b><u>Total Time:</u> 60 min</b></p> <ul style="list-style-type: none"> <li>● Review <b>10 min</b></li> <li>● Main Activity <b>40 min</b></li> <li>● Exit Ticket <b>10 min</b></li> </ul>
Materials needed:	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Review question index card (The teacher will create.)</li> <li>● Answer question index cards (The teacher will create.)</li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Robot (Some examples include: <a href="#">Code and Go Mouse</a>, <a href="#">Botley</a>, <a href="#">Dash</a>)</li> <li>● <a href="#">Coding cards</a></li> </ul>
Subject integrated:	Math
Other standards addressed:	<ul style="list-style-type: none"> <li>● <b>1.G.1</b>—Distinguish between definition attributes (e.g., triangles are closed and three-sided) versus non-defining attributes (e.g., color, orientation, overall size); build and draw shapes to possess defining attributes.</li> <li>● <b>1.G.2</b>—Compose two-dimensional shapes (rectangles, squares, trapezoids, triangles, half-circles, and quarter-circles) or three-dimensional shapes (cubes, right rectangular prisms, right circular cones, and right circular cylinders) to create a composite shape and compose new shapes from the composite shape.</li> <li>● <b>1.G.3</b>—Partition circles and rectangles into two and four equal shares, describe the shares using the words halves, fourths, and quarters, and use the phrases half of, fourth of, and quarter of. Describe the whole as two of or four of the shares. Understand for these examples that decomposing into more equal shares creates smaller shares.</li> </ul>
Vocabulary:	<p><u>Algorithm</u>: A sequence of steps, followed in order, to finish a task that can be performed with or without a computer.</p> <p><u>Bug</u>: Part of a program that does not work correctly</p> <p><u>Debugging</u>: Finding and fixing problems in an algorithm or program</p> <p><u>Sequencing</u>: Putting commands in the correct order so the computer can read the commands</p>
Notes:	

## Week 40: Spelling Robot Review

Lesson overview:



**Purpose:**

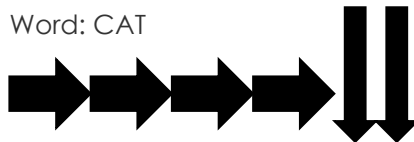
In this lesson, students will spell a high frequency word correctly in order to gather cheese.

**Lesson:**

- Introduction
  - Review high frequency words.
- Setup
  - Students can participate individually or in groups.
  - Teachers will write individual letters of the alphabet on index cards.
  - Teachers will create a grid for the robot to travel.
    - The grid could be floor tiles or a taped off area.
  - Designate the top right square of the grid as the starting block.
  - Place the letters of high frequency words in the tiles.
- Main Activity
  - Teachers will call out the word and have students start by spelling the word correctly.
    - Students can use a whiteboard or pencil/paper to spell the word correctly.
  - Next, students will use [coding cards](#) to map out the code they will use to program the robot to travel through the blocks and collect the correct letters (in order).
  - Students will need to code the robot (Some examples include: [Code and Go Mouse](#), [Botley](#), [Dash](#)) to travel through the letters to spell the high frequency word.
  - If you do not have a coding robot in your class, the students can act as the robot.

Example:

Word: CAT



START		C		A
	K		O	
I				T

- Exit Ticket
  - What was your favorite part of today's activity?
  - What words did you spell correctly?
  - What are some words you spelled incorrectly?

Lesson links/resources:

[Sight Word Flash Cards](#)

CS standards addressed:

Students will be able to:

- Create a code to gather correctly spelled words

Standards:



	<ul style="list-style-type: none"> <li>● <b>AP.1A.4</b>—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.</li> <li>● <b>AP.1A.5</b>—Develop plans that describe a program's sequence of events, goals, and expected outcomes.</li> </ul>
Time needed:	<p><b>Total Time: 60 min</b></p> <ul style="list-style-type: none"> <li>● Review <b>10 min</b></li> <li>● Main Activity <b>40 min</b></li> <li>● Exit Ticket <b>10 min</b></li> </ul>
Materials needed:	<p>Teacher:</p> <ul style="list-style-type: none"> <li>● Alphabet cards (Each card should have individual letters. The teacher will create.)</li> <li>● <a href="#">Sight Word Flash Cards</a></li> </ul> <p>Students:</p> <ul style="list-style-type: none"> <li>● Robot (Some examples include: <a href="#">Code and Go Mouse</a>, <a href="#">Botley</a>, <a href="#">Dash</a>)</li> <li>● Materials to create grid.</li> <li>● <a href="#">Coding cards</a></li> </ul>
Subject integrated:	ELA
Other standards addressed:	<ul style="list-style-type: none"> <li>● <b>RF.1.3</b>—Know and apply grade-level phonics and word analysis skills.</li> <li>● <b>SL.1.1</b>—Participate in collaborative conversation.</li> <li>● <b>SL.1.2</b>—Ask and answer questions about key details presented through text or orally.</li> <li>● <b>SL.1.3</b>—Ask and answer questions about what a speaker says to clarify something that is not understood.</li> <li>● <b>SL.1.5</b>—Add drawing or other visual displays to descriptions when appropriate to clarify ideas, thoughts, and feelings.</li> </ul>
Vocabulary:	<p><u>Algorithm</u>: A sequence of steps, followed in order, to finish a task that can be performed with or without a computer.</p> <p><u>Bug</u>: Part of a program that does not work correctly</p> <p><u>Debugging</u>: Finding and fixing problems in an algorithm or program</p> <p><u>Sequencing</u>: Putting commands in the correct order so the computer can read the commands</p>
Notes:	

## Appendix A: Code.org

**I'd like to start using Code.org in my classroom. How should I start?**

<https://support.code.org/hc/en-us/articles/228116468-I-d-like-to-start-using-Code-org-in-my-classroom-How-should-I-start->

**How to create a teacher account:**

<https://support.code.org/hc/en-us/articles/228116468-I-d-like-to-start-using-Code-org-in-my-classroom-How-should-I-start->

**How to create a classroom section:**

<https://support.code.org/hc/en-us/articles/115000488132-Creating-a-classroom-section>

**Finding curriculum and lesson plans:**

<https://support.code.org/hc/en-us/articles/115001595051-Finding-curriculum-and-lesson-plans>

**Code.org Support**

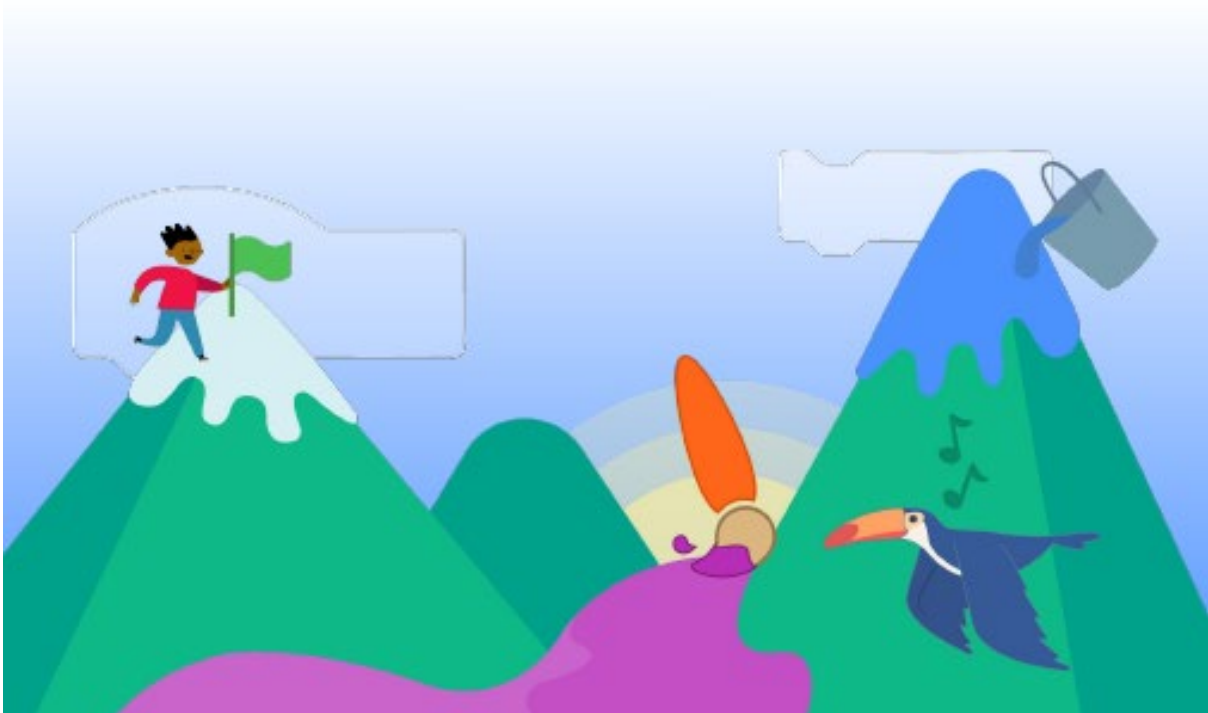
<https://support.code.org/hc/en-us>

## Appendix B: Scratch

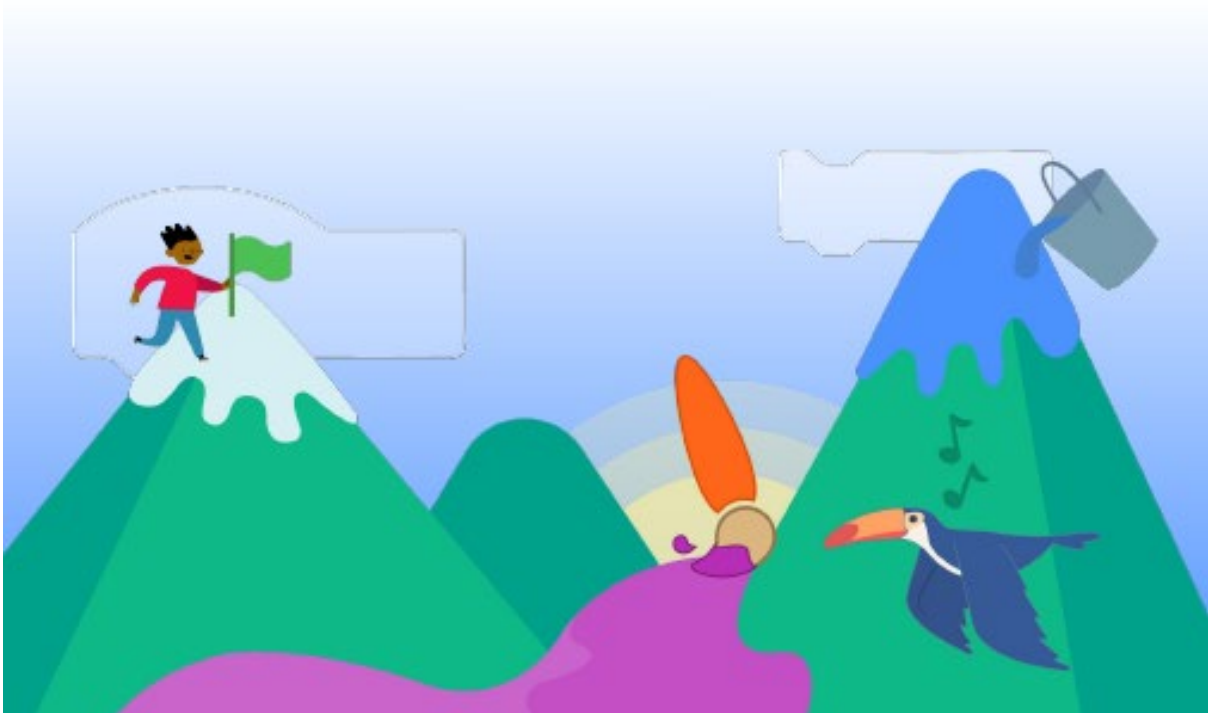
# SCRATCH

## Educator's Guide

- Teacher Accounts
- Beginner's Guide
- Lesson Guides



- Teacher Accounts
- Beginner's Guide
- Lesson Guides





# Teacher Accounts

As an educator, you can request a Scratch Teacher Account. A Scratch Teacher Account provides educators with additional features to manage student participation on Scratch, including the ability to create student accounts, organize student projects into studios, and monitor student comments. This guide will walk you through creating an account, creating a class, adding and managing your students, and creating class studios. You can also see our [Scratch for Educators](#) page and our [Teacher Account FAQ](#) page for additional information.

## Create Your Teacher Account

Visit this link to get started: <https://scratch.mit.edu/educators/register>

You'll be prompted to create a username and password. **Make sure that your username does not contain your name or personal information**, like your school, location, or email address.

Within the Scratch community, all users are asked to refrain from sharing personal information through their usernames. **It's important that both you and your students follow these guidelines. Accounts that do not adhere to these guidelines will be deleted.**

### Creating your teacher account

Create a username

QuirkyArtTeacher

Password

.....

Show password

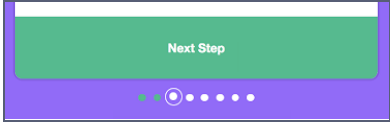


Next Step

#### Tips for making your username

- **Incorporate the name of the subject you teach**
  - ex: QuirkyArtTeacher
- **Use a tool or term from the subject you teach**
  - ex: MetamorphicRocks
- **Add an important date, be unique**
  - ex: Bibliophile1440
- **Make it memorable with a pun or an alliteration!**
  - ex: TyranoTeacher

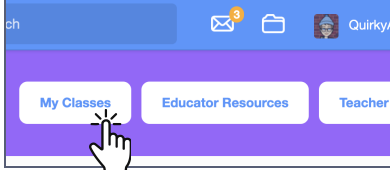

**Be sure to make a note of your username and password.**



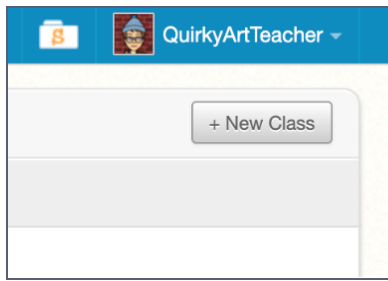
	<p>Click through each step to <b>complete registration</b>.</p>
	<p>Log into your email and confirm your email address.</p> <p>Check your spam folder if you do not see the email.</p> <p>Once you have <b>confirmed your email address</b>, we'll review your account.</p>
	<p>Once your account has been reviewed and approved, <b>you will receive a welcome email</b>. Then, you can <b>log into your teacher account at <a href="https://scratch.mit.edu">scratch.mit.edu</a></b>!</p>

## Create a Class

Creating classes allows you to manage groups of students, and create studios where your students can add their projects.

<h3>Creating your class</h3>	
	<p>Once you have successfully logged into your Teacher Account, if you are looking at the homepage, there will be a bar at the top of the screen with three options. Select <b>“My Classes.”</b></p>
	<p>You can also access your classes from the dropdown under your username.</p>



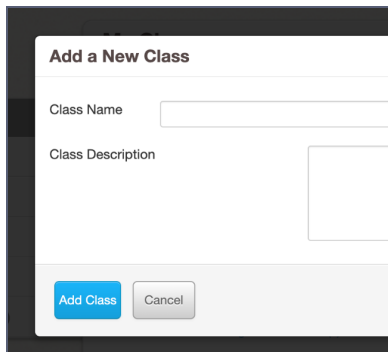


To create a class, click the “+ **New Class**” button at the top right of the page.

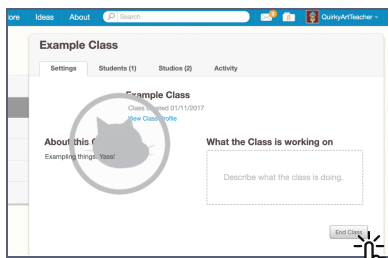
Enter the class name and description.

**Warning:** Do not include real names and locations, like the name of your school or city/town.

Once you have created a class, you can add students.



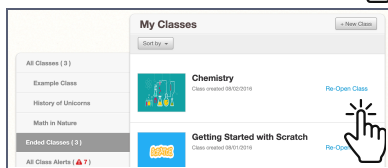
## Ending your class



To end a class, under “My Classes,” choose your class and on the Settings tab, click the “End Class” button.

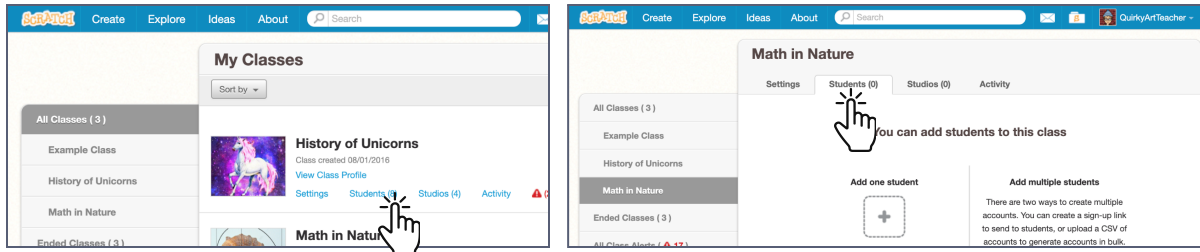
When you end a class, your class profile page will be hidden and your students will no longer be able to log in (but their projects and the class studios will still be visible on the site).

You may re-open the class at any time. By going to the “Ended Classes” tab and clicking the “Re-Open Class” link near the class you want to reopen.



# Add Students to Your Class

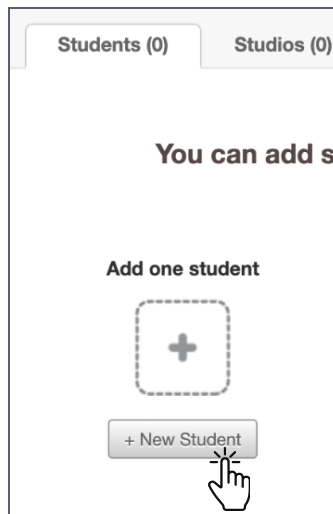
While on “My Classes,” select the class and then click on “Students” (either the link under the class name or the Students tab). Once created, your student accounts will appear here.



There are three ways to add students to your class. The first method allows you to add an individual student to a class. Methods 2 and 3 allow you to add multiple students to a class.

**Tip:** Create a naming convention as a guideline for generating usernames. For example, you may want each name to include an abbreviation for the course name, the class section, and the student’s number on your roster (ex: VisArts-02-17). Use the [Student Username List](#) we have created to record the usernames and passwords your students have created.

## Method 1: Add Individual Students



Click the “+ New Student” button to add students individually.

Confirm the correct class is showing in the “Add to Class” dropdown menu.

You will be prompted to create a username for this student.

**Warning:** Make sure that the usernames you create do not contain identifying information about yourself, your students, or your school. Accounts that do not adhere to these guidelines will be deleted.

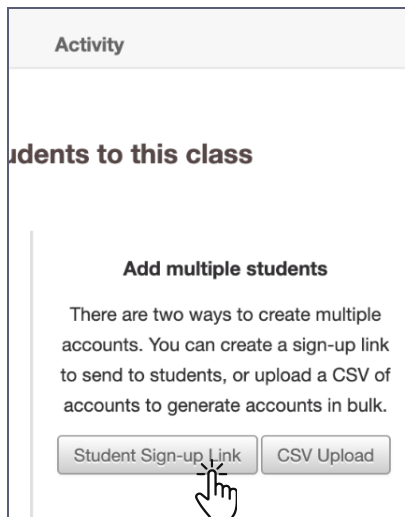
The password for this student username will automatically be set as the username of your teacher account.



Have students log into their accounts and change their passwords as soon as possible.

**Tip:** It is not possible to add an existing Scratch account to a classroom. You will need to create a new Student Account for them using your Teacher Account. A student can only be a part of one class, and it is not possible to transfer students from one class or teacher to another.

## Method 2: Student Sign-up Link



Clicking the “Student Sign-Up Link” button brings you to another window and clicking the “Get Link” button will generate a link that will allow your students to join the class you have just created. The link will start with “http://scratch.mit.edu/signup...”

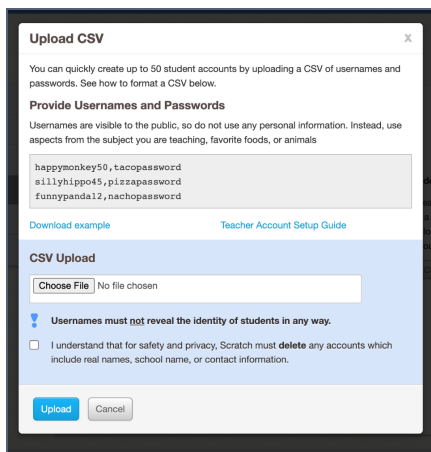
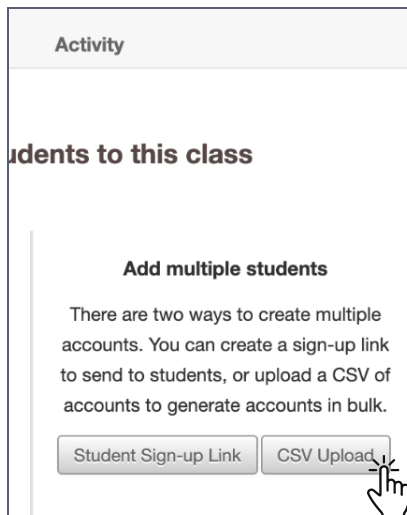
Students can then create their own usernames and passwords.

**Warning:** Remind your students that, when making their usernames, the username should not contain identifying information about themselves, their teacher, or their school. Accounts that do not adhere to these guidelines will be deleted.





## Method 3: CSV Upload



The template spreadsheet has two columns, A and B, and two rows of student data.

	A	B
1	student1	password1
2	student2	password2

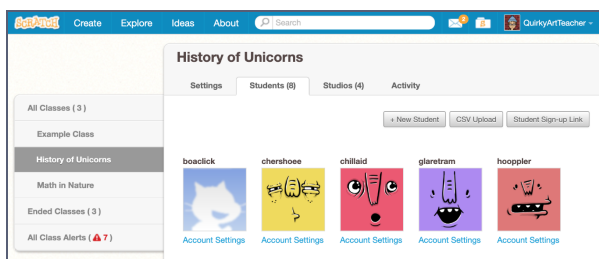
Click the “CSV Upload” button on the class page.

Using the template provided by clicking the “Download example” link, create a username and password for each of your students. You can use the template provided or create your own spreadsheet with student usernames in column A and passwords in column B. To upload your own template, you’ll need to save the file as a CSV file.

Once you’ve created usernames and passwords for each student and saved the file, click the “Choose file” button to locate the file, then click the “Upload” button.

It is not possible to add more than 250 students to a single class. You can, however, create a new class and add another 250 student accounts to each new class.

**Warning:** Make sure that the usernames you create do not contain identifying information about yourself, your students, or your school. Accounts that do not adhere to these guidelines will be deleted.



You can add students via any of these methods at any time under the “Students” tab.



# Creating Studios for Student Work

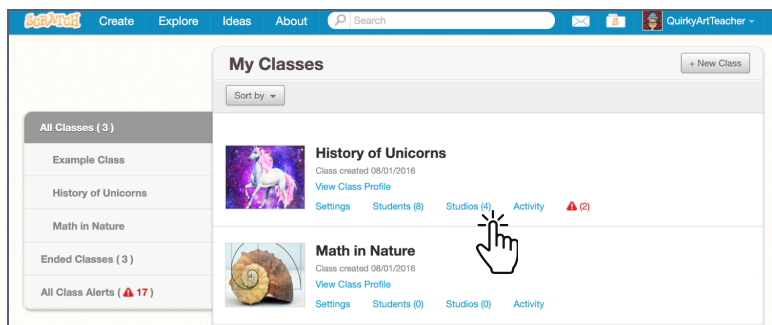
Studios allow you to create collections of student projects for specific classes or assignments. This makes it easier for you to view their projects throughout their creative process. It also makes it easier for students to collaborate and be inspired by each other's work.

Scratcher status is required in order to create a studio, and the person who created the studio is automatically assigned the role of "host." There is only one host per studio, and only studio hosts can edit the title, thumbnail, and description.

Studios are immediately public, even those created in the context of a class. Unlike Scratch projects, there is no share/unshare option for studios. Everyone can follow a studio, see studio comments and projects, and leave a comment or add a project (unless commenting or the ability to add projects is turned off).

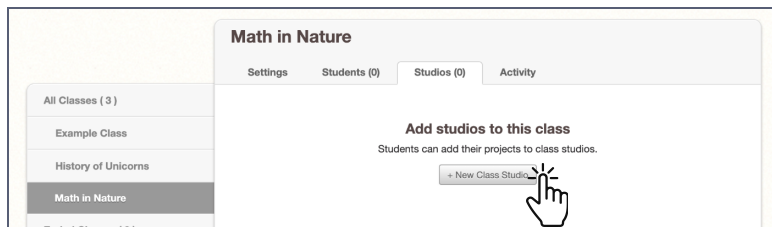
There are two ways to create a studio on a teacher account. Method one creates studios that automatically add all students in a class as curators. Method two creates studios without automatically adding students as curators, and students or any Scratcher can be individually added as curators.

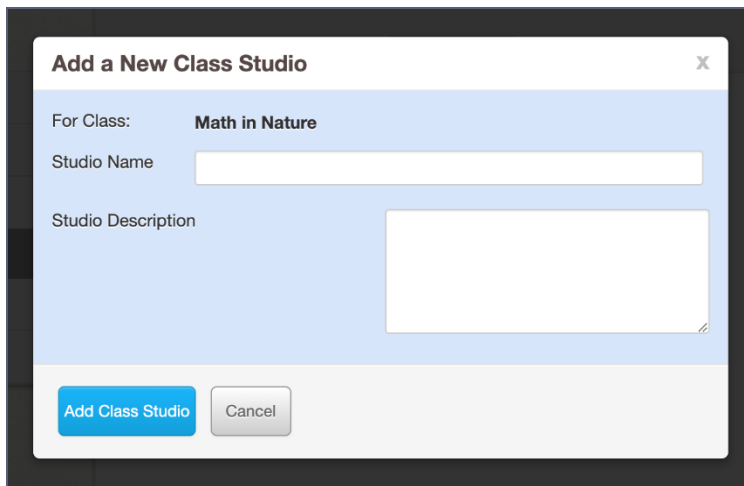
## Method 1: Create a studio that automatically adds all students in a class as curators



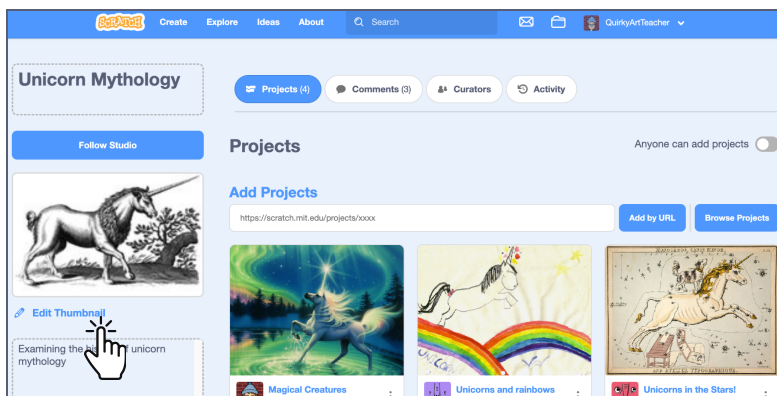
Once logged into your Scratch account, go to "My Classes."

Choose the class to assign the studio to, then click on "Studios" (either the link under the class name or the Studios tab). Then click the "+ New Class Studio" button.





On the window that appears, you will be asked to **give the studio a name and description**. (These can always be adjusted in the studio later.) In the description, you can share the theme of the studio, what kinds of projects you are looking to include... Just be sure your title and description don't reveal any personal information (like school name or first and last name).

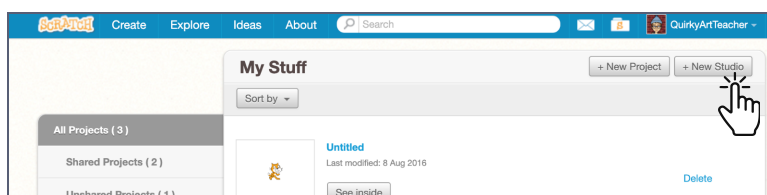


Then, click the “Add Class Studio” button.

Once in the studio, click the “Edit Thumbnail” button to change the default gray cat image in the upper left-hand corner. **Upload your own studio thumbnail image**. The maximum file size for a thumbnail is 512 KB and your image must be less than 500x500 pixels.

When you click on the “Curators” tab, you should see all the class students have been set as studio curators.

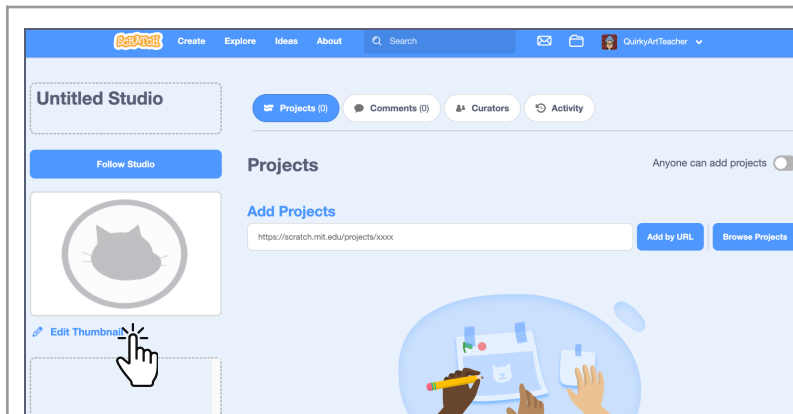
## Method 2: Create a studio without automatically adding students as curators



Once logged into your Scratch account, go to “My Stuff.”

Choose the “+ **New Studio**” button at the top right.





Click on “Untitled Studio” to **give your studio a name and description**. In the description, you can share the theme of the studio, what kinds of projects you are looking to include... Just be sure your title and description don't reveal any personal information (like school name or first and last name).

Click the “Edit Thumbnail” button to change the default gray cat image in the upper left-hand corner. **Upload your own studio thumbnail image**. The maximum file size for a thumbnail is 512 KB and your image must be less than 500x500 pixels.

When you click on the “Curators” tab, you should see no curators have been assigned yet.

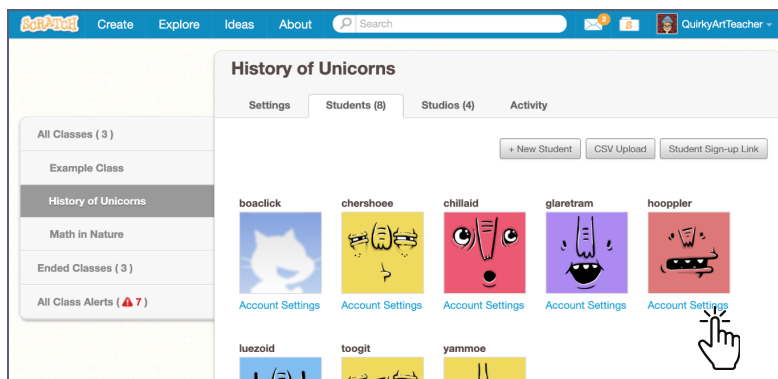
See our [Studio Guide](#) for detailed information on:

- Studio Definitions
- How to Manage a Studio
- How to Add Projects to a Studio

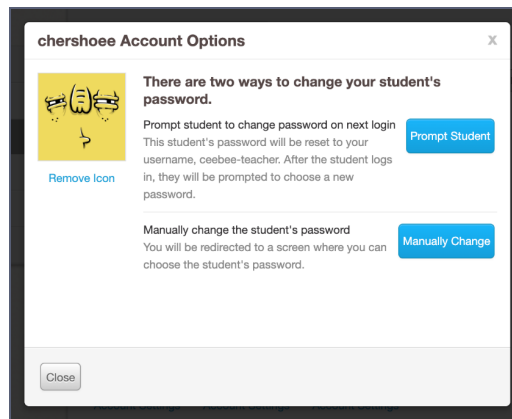


# Managing Your Students

## Managing a student



You can manually **reset a student password** from within your Scratch Teacher Account. First, navigate to “My Classes” and choose the class and go to the “Students” tab. Then click on the “Account Settings” link below the student’s account.



You cannot delete a student’s account by using a Teacher Account, but students can delete their own account.



You can see alerts about notifications your students receive on the “Activity” tab of a class or the “All Class Alerts” tab.

**Tip:** If you’d like to translate this guide, [click here to make a copy](#) of this Google doc.



# Getting Started with

# SCRATCH

## Beginner's Guide

Create your own games, animations,  
interactive stories, and more.



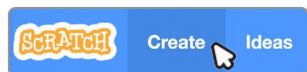


# GETTING STARTED

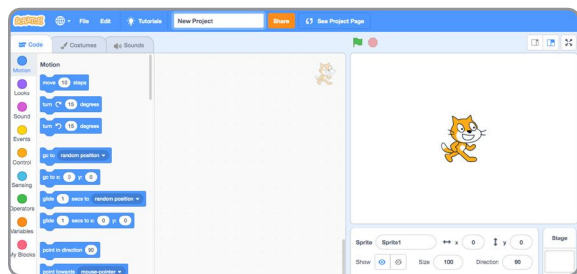
You can use Scratch online at: [scratch.mit.edu](https://scratch.mit.edu)

[scratch.mit.edu](https://scratch.mit.edu)

Once you've navigated to [scratch.mit.edu](https://scratch.mit.edu), click **Create**.



This will bring you to the **Scratch Editor**, where you can start creating projects.



If your computer uses an older operating system, or your internet connection is unreliable, you can download Scratch and use it offline.



Visit: <https://scratch.mit.edu/download>  
for information on downloading and installing the Scratch app.



# THE SCRATCH EDITOR

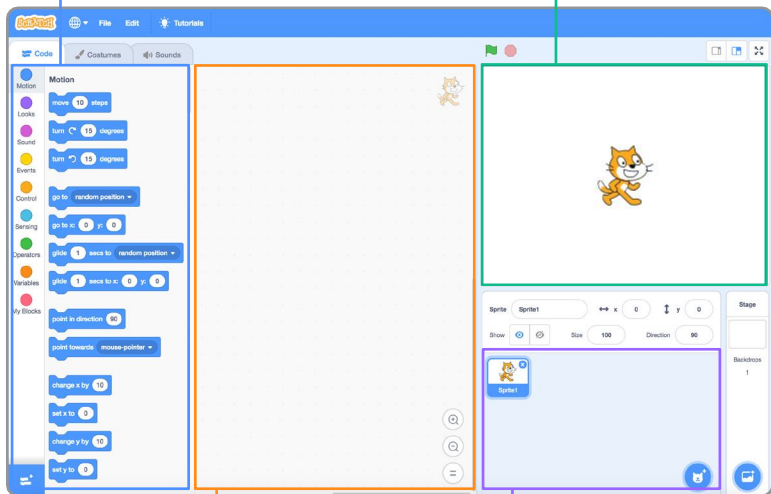
The Scratch Editor is where you create projects in Scratch. Here are its main parts:

## Blocks Palette

Blocks for coding your projects

## The Stage

Where your creations come to life



## Coding Area

Drag in blocks and snap them together to code your sprites

## Sprite List

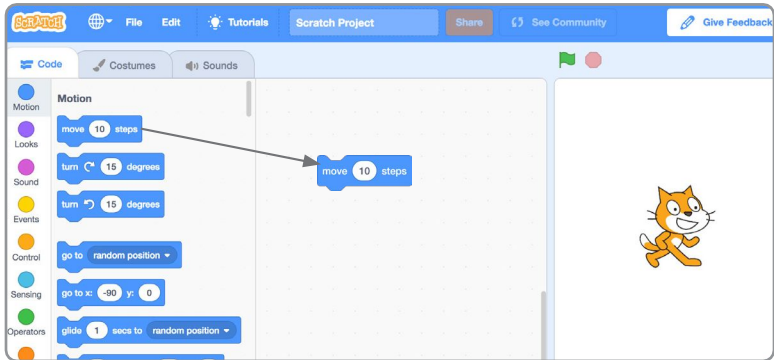
Click the thumbnail of a sprite to select it



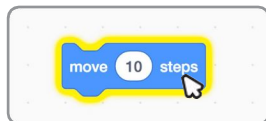


# LET'S CODE!

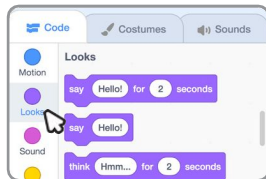
To code projects in Scratch, you snap together blocks. Start by dragging out a **move** block.



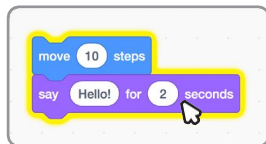
Click the block to try it.  
Does your cat move?



Now say something!  
Click the **Looks** category.



Drag out a **say** block.  
Snap it onto the **move** block. Click on your blocks to try them.





## WHAT IS A SPRITE?

In Scratch, any character or object is called a sprite. Every new project in Scratch starts with the Cat sprite.



Want to choose a different sprite?



Click the New Sprite icon.

Or, hover over the **New Sprite** icon to see more options.

Upload an image from your computer.

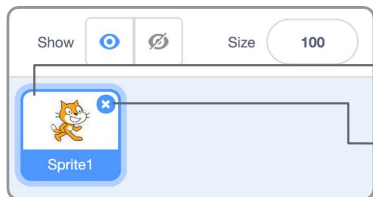
Draw your own sprite.



Click for a surprise sprite!

Choose a sprite from the library.

Want to **delete a sprite** from your project?



First, select the sprite by clicking on its thumbnail in the Sprite List.

Then, click here to delete the sprite.



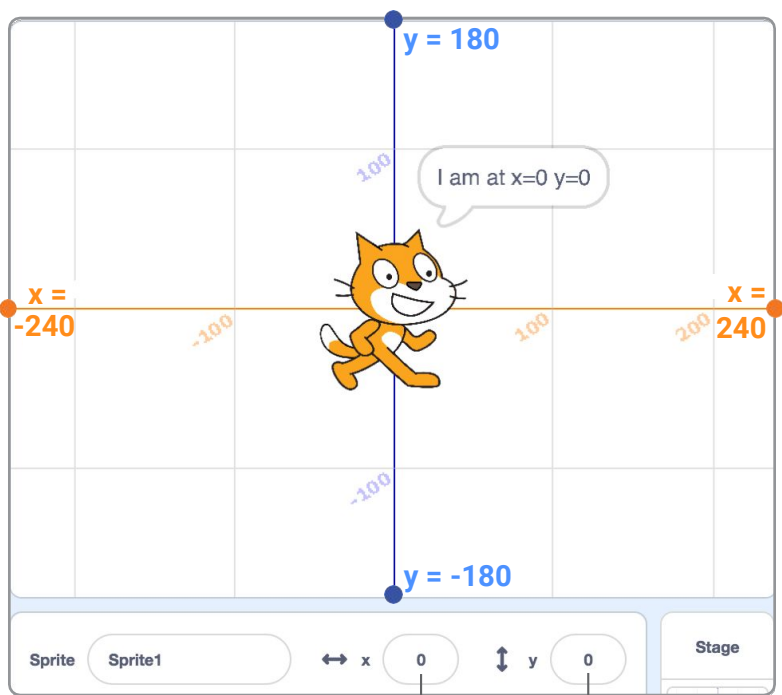
## WHERE IS YOUR SPRITE?

Every sprite has an **x** and **y** position on the Stage.

**x** is the position of the sprite from left-to-right.

**y** is the position from top-to-bottom.

At the very center of the stage, **x** is 0 and **y** is 0.



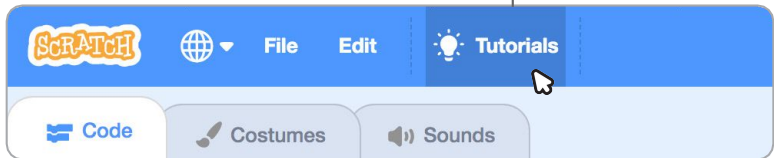
When you move your sprite, you can see its **x** and **y** position change.



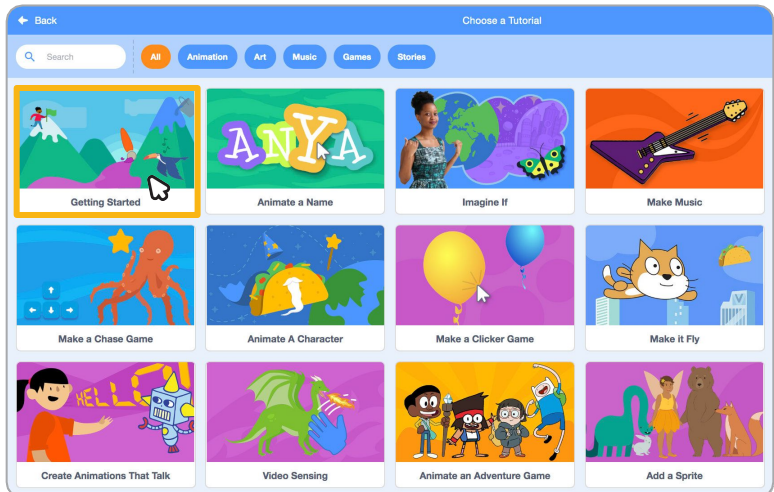
# TUTORIALS

There are a range of tutorials available in the Scratch **Tutorials Library**, which guide learners in creating projects with Scratch. Students can get started making their own stories, animations, and games.

You can get to the Tutorials Library from the Scratch Editor by clicking the **Tutorials** button.



The **Getting Started** tutorial will walk you through the basics.

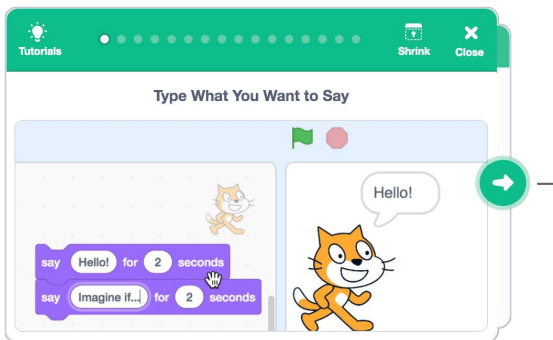




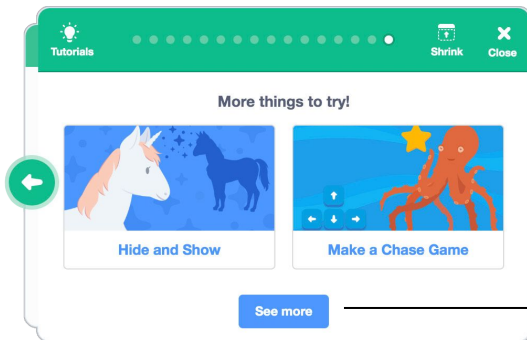
# TUTORIALS

Once you've selected the tutorial, it will open in the Scratch Editor.

Click the green arrow to see each step.



When you've reached the end of a tutorial you can select another tutorial, and keep adding to your project.



Click here to see all the Tutorials.

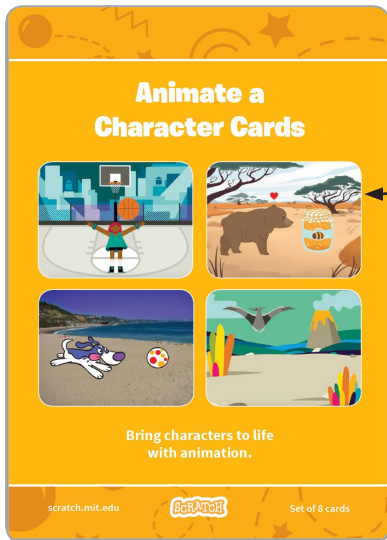


# CODING CARDS

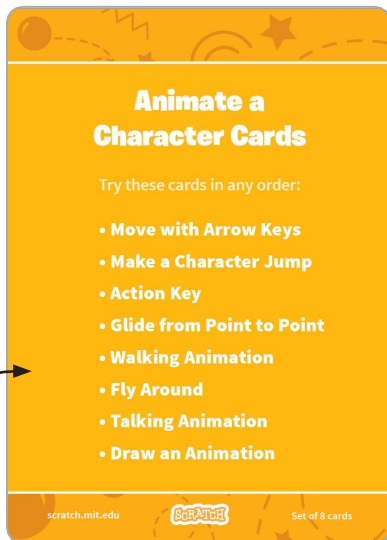
The Scratch **Coding Cards** provide another way to learn to create projects with Scratch. Download the cards at [scratch.mit.edu/ideas](https://scratch.mit.edu/ideas).

Each set of cards starts with a title card, which shows you what you can create.

The **Animate a Character** cards are a great set to start with.



Examples of what you can create



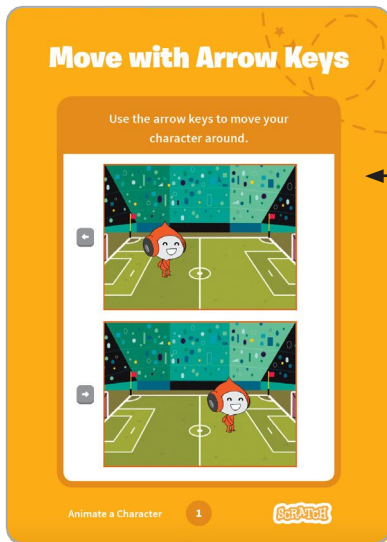
A list of all the cards in this set



# USING THE CODING CARDS

After each title card is a series of cards walking you through each step of creating a project.

Add your own sprites, backdrops and more!



The front of each card shows you what you can create.



The back shows you how to do it.



## GET CREATIVE!

Encourage students to use their imagination as you create projects. There are many different ways they can make their Scratch projects unique.

You can choose or draw your own characters.



Choose a sound or record your own.



Try changing numbers or adding blocks to your code to see what happens.



**Experiment and customize your project however you want!**

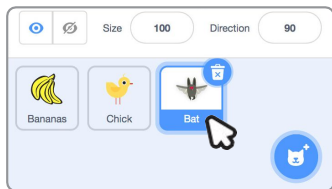




## GET CREATIVE WITH SPRITES!

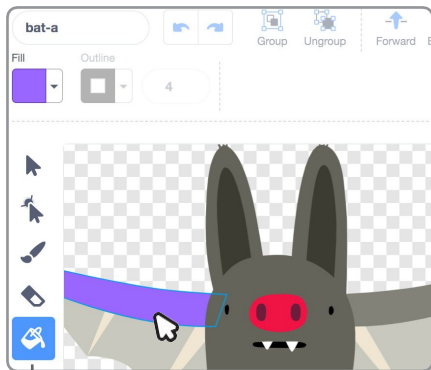
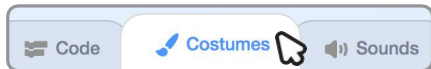
Scratch has its own paint tools, which allow you to customize sprites from the library, or even create sprites of your own.

Let's start by editing a sprite from the library.



Select a sprite to edit by clicking on it in the Sprite list.

Click the **Costumes** tab at the top left to see the paint tools.



The paint tools allow you to recolor sprites, add to them with a paint brush, and change them in a variety of ways.

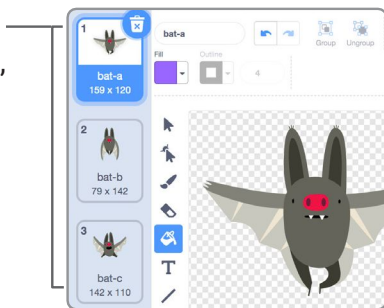
You can use the **paint bucket** tool to recolor different parts of a sprite.



## GET CREATIVE WITH SPRITES!

Some sprites, like the Bat sprite have multiple costumes, or poses.

You can see a sprite's costumes by clicking the **Costumes** tab.

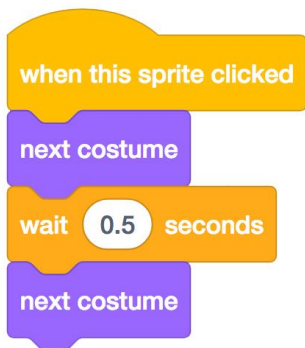


If your sprite only has one costume, right click on the costume to duplicate it (On Mac control + click).



Now you can modify the second costume using the paint tools, so your sprite has two different poses or facial expressions.

Click the **Code** tab, then try adding these blocks.





# ADD YOUR OWN PHOTOS

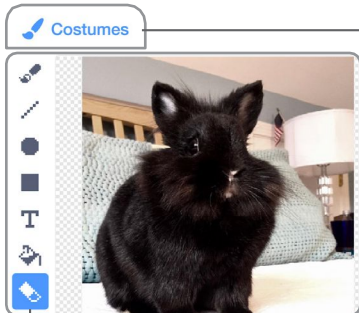
There are many ways to create your own sprites and artwork using the Scratch paint tools.

You can create your own sprites by uploading photos or images and erasing the background.



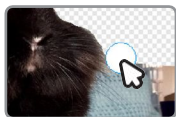
Hover over the **New Sprite** button, then select **Upload Sprite**.

Upload Sprite



Next click the **Costumes** tab. You will see bitmap tools for editing your image.

Click the **eraser** icon and use the eraser tool to remove the background from your photo.



**Tip:** to adjust the size of the eraser, type a larger or smaller number.

There are two modes for drawing in Scratch:

1. **Bitmap Mode** allows you to edit photos and paint with pixels.
2. **Vector Mode** allows you to create and edit shapes.

**Tip:** If you'd like to remix and customize this guide, [click here to make your own copy](#) of the Google Slides template.



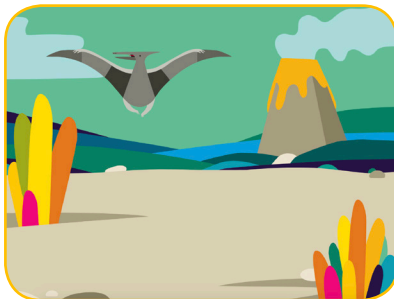
**Created by the Scratch Team** ([scratch.mit.edu](https://scratch.mit.edu)) and shared under the Creative Commons Attribution-ShareAlike 4.0 International Public License (CCbySA 4.0).

||| ||| ||| ||| ||| COLOR SCHEME

## EDUCATOR GUIDE

# Animate a Character

With this guide, you can plan and lead a 55-minute lesson using Scratch. Students will gain experience with coding as they bring characters to life with animation.



## Lesson Outline

Objective: Students will become familiar with the Scratch environment by animating a character.



**IMAGINE**  
*10 minutes*

First, gather as a group to introduce the theme and spark ideas.



**CREATE**  
*40 minutes*

Next, help students as they animate characters, working at their own pace through the tutorial.



**SHARE**  
*5 minutes*

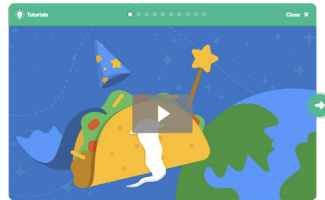
At the end of the session, gather together to share and reflect.

## Get Ready for the Lesson

Use this checklist to prepare for the lesson.

### Preview the Tutorial

The *Animate a Character* tutorial shows students how to create their own projects. Preview the tutorial before your lesson and try the first few steps: [scratch.mit.edu/tutorials](https://scratch.mit.edu/tutorials)



### Print the Activity Cards (optional)

Print a few sets of *Animate a Character* cards to have available for students during the lesson. [scratch.mit.edu/ideas](https://scratch.mit.edu/ideas)



### Make sure students sign into their Scratch accounts

Have students sign into their own Scratch accounts at [scratch.mit.edu](https://scratch.mit.edu).

### Set up computers or laptops

Arrange computers so that students can work individually or in pairs.

### Set up a computer with projector or large monitor

You can use a projector to show examples and demonstrate how to get started.

## Imagine



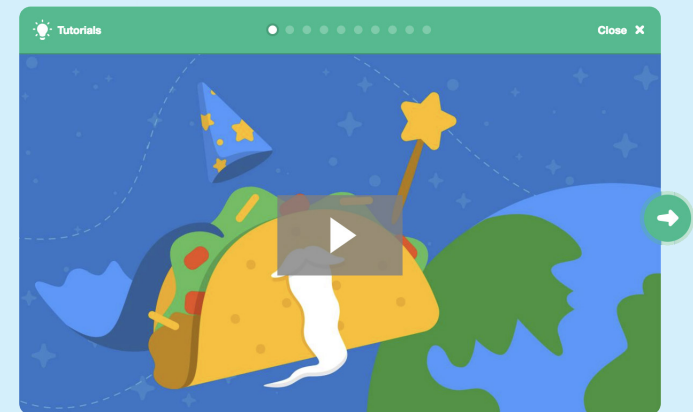
Begin by gathering the students to introduce the theme and spark ideas for projects.

### Warm-up Activity: Favorite Characters

Gather the group in a circle. Ask each student to say their name, then share a favorite character from a book, movie, or TV show, and one or two of their favorite things about that character.

### Provide Ideas and Inspiration

To spark ideas, watch the *Animate a Character* video at the start of the tutorial. The video shows a variety of projects to spark ideas and inspiration.



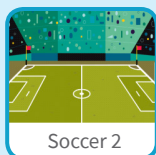
View the [scratch.mit.edu/ideas](https://scratch.mit.edu/ideas)

## Demonstrate the First Steps



Demonstrate the first few steps of the tutorial so students can see how to get started.

### Choose a backdrop.



### Choose a character to animate.



### Make your sprite move right and left with arrow keys:

when right arrow key pressed  
change x by 10

Choose right arrow from the menu.

when left arrow key pressed  
change x by -10

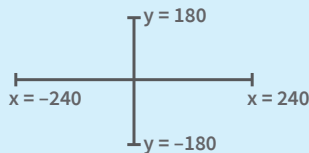
Choose left arrow from the menu.

Type a minus sign to move left.

Press the left arrow and right arrow keys on your keyboard to move.



**Helpful Hint:** Understanding x y coordinates will help students figure out how to move sprites around the stage.



y is the position on the Stage from top to bottom.

x is the position on the Stage from right to left.

## Create



Support students as they create animated Scratch projects.

### Start with Prompts

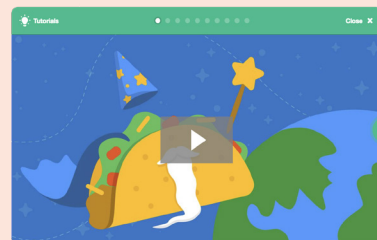
Ask students questions to get started

*Which character would you like to animate?*

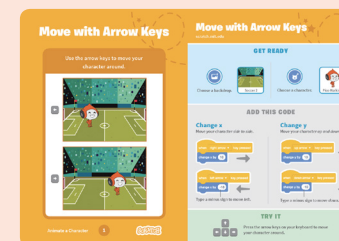
*What do you want your character to do?*

### Provide Resources

Offer options for getting started



Some students may want to follow the online tutorial: [scratch.mit.edu/animate](https://scratch.mit.edu/animate)



Others may want to explore using the activity cards: [scratch.mit.edu/ideas](https://scratch.mit.edu/ideas)

### Suggest Ideas for Starting

- Choose a character to animate.
- Animate your character: make it jump, fly, glide or talk!
- Choose a backdrop.



### More Things to Try

- Try combining more than one kind of animation.
- If you're not sure what to do, pick a card and try something new.
- Add a second character or object to animate.



### Support collaboration

- When someone gets stuck, connect them to another participant who can help.
- See a cool idea? Ask the creator to share with others.



### Encourage experimenting

The Animate a Character activity can be done in any order, with a range of different character and object sprites.

Encourage students to try new things:

*What will your character do next?*

*How can you make your animation interactive?*



## Share

Have students share their project with their neighbors.

### Ask questions they can discuss:

*What do you like best about the project you made?*

*What was the hardest part?*

*If you had more time, what would you add or change?*

## What's Next?

Students can use the ideas and concepts from this lesson to create a wide variety of projects. Encourage them to continue developing their projects into games, stories or interactive art with the resource listed below.



### Video Sensing

Interact with characters and objects in Scratch with video sensing.

Find this project and more in the Tutorials library: [scratch.mit.edu/ideas](https://scratch.mit.edu/ideas)

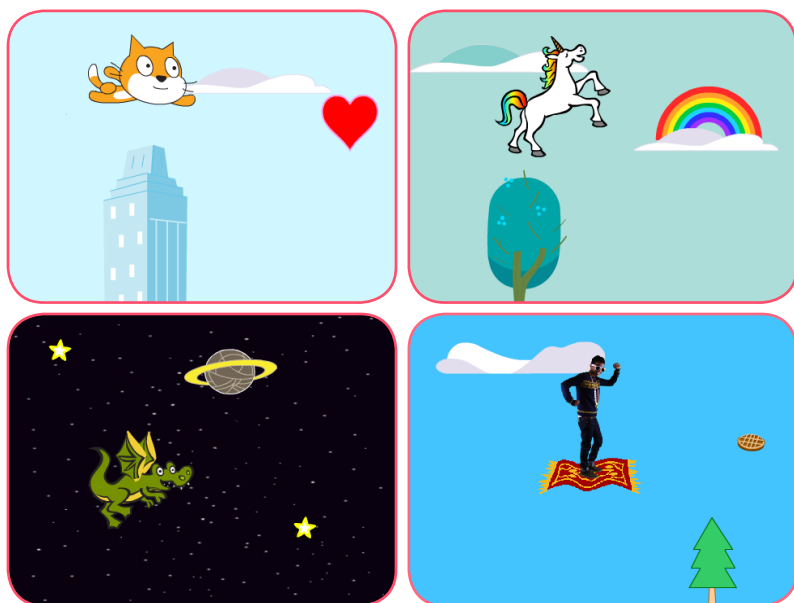
Scratch is a project of the Lifelong Kindergarten Group at the MIT Media Lab.



## EDUCATOR GUIDE

# Make It Fly

With this guide, you can plan and lead a 55-minute lesson using Scratch. Students will choose a character and program it to fly.



## Lesson Outline

Objective: Students will create an animation with the illusion of a flying character.



**IMAGINE**  
*10 minutes*

First, gather as a group to introduce the theme and spark ideas.



**CREATE**  
*40 minutes*

Next, help students as they create a flying animation, working at their own pace through the tutorial.



**SHARE**  
*5 minutes*

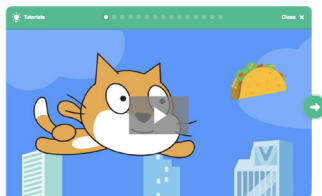
At the end of the session, gather together to share and reflect.

## Get Ready for the Lesson

Use this checklist to prepare for the lesson.

### Preview the Tutorial

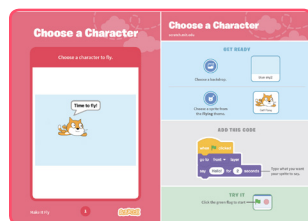
The *Make It Fly* tutorial shows students how to create their own projects. Preview the tutorial before your lesson and try the first few steps: [scratch.mit.edu/fly](https://scratch.mit.edu/fly)



### Print the Activity Cards (optional)

Print a few sets of *Make It Fly* cards to have available for students during the lesson.

[scratch.mit.edu/fly/cards](https://scratch.mit.edu/fly/cards)



### Make sure students sign into their Scratch accounts

Have students sign into their own Scratch accounts at [scratch.mit.edu](https://scratch.mit.edu).

### Set up computers or laptops

Arrange computers so that students can work individually or in pairs.

### Set up a computer with projector or large monitor

You can use a projector to show examples and demonstrate how to get started.

## Imagine



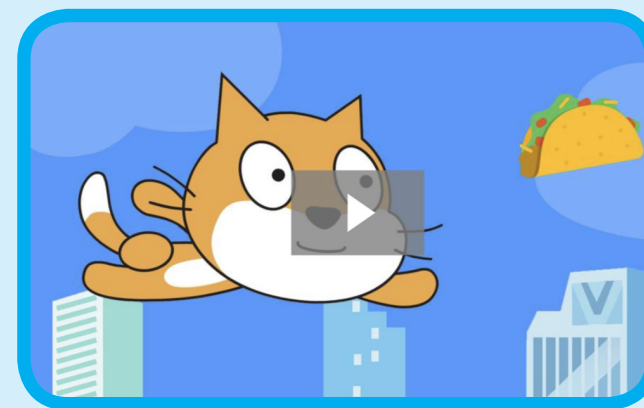
Begin by gathering the students to introduce the theme and spark ideas for projects.

### Warm-up Activity: If I Could Fly...

Gather the group in a circle and ask, “If you could fly, where would you want to go?” Suggest that they close their eyes and imagine flying through their favorite place. Ask, “Where are you? What kinds of things do you see below you?” If there’s time, have each person say where they imagined flying or something they saw on their flight.

### Provide Ideas and Inspiration

Show the introductory video for the *Make It Fly* tutorial. The video shows a variety of projects for ideas and inspiration.



View at [scratch.mit.edu/fly](https://scratch.mit.edu/fly) or [vimeo.com/llk/fly](https://vimeo.com/llk/fly)

## Demonstrate the First Steps



Demonstrate the first few steps of the tutorial so students can see how to get started.

In Scratch, click Create.  
Choose a flying sprite from the library:



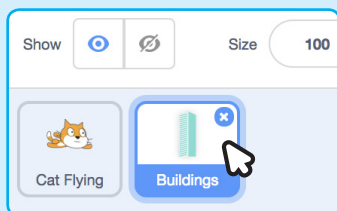
Cat Flying

Choose a new sprite for your character to fly past:



Buildings

Make the building move across the stage to make your character look like it's flying:



## Create



Support students as they make a flying animation.

### Start with Prompts

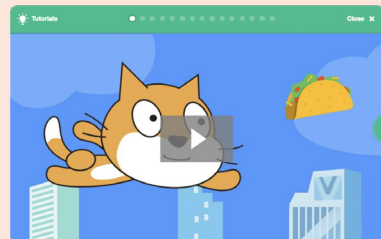
Ask students questions to get started

*What character would you like to make fly?*

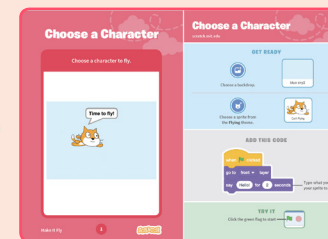
*Where will your character go flying?*

### Provide Resources

Offer options for getting started



Some students may want to follow the online tutorial:  
[scratch.mit.edu/fly](https://scratch.mit.edu/fly)



Others may want to explore using the activity cards:  
[scratch.mit.edu/fly/cards](https://scratch.mit.edu/fly/cards)

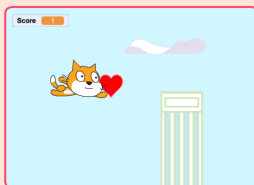
### Suggest Ideas for Starting

- Choose a character
- Choose buildings or other scenery
- Make the character say something
- Make the scenery move



### More Things to Try

- Switch costumes to change the scenery.
- Make your character move when you press a key.
- Add clouds and other floating objects.
- Score points when touching an object.



### Encourage Debugging

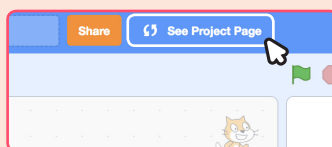
Here are some strategies to suggest to help students fix any bugs or difficulties they encounter:

- When stuck, talk out what you're working on with someone.
- Try out small bits of code at a time to figure out what's happening at each step.
- Look closely at the blocks on the tutorial or activity cards to see if they are the same or different from the blocks you're using.
- Remember that bugs always arise when creating a computer program. Debugging is a helpful skill to know not just in coding, but throughout life.

### Prepare to Share

To add instructions and credits to a project, click the button: "See project page".

Give your project a title, add instructions and credits, then click Share.



Share



## Share

Share projects with others in the room. Organize a flying character showcase. Ask half the room show their projects, while the others view them. Then switch.

Suggest that they ask each other questions, such as:

*What do you like best about the project you made?*

*What might you like to change or make next?*

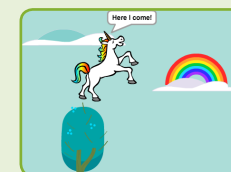
## What's Next?

Students can use the ideas and concepts from this lesson to create other projects. Here are a couple of variations on the flying character project you could suggest.



### Flying Game

Make a game where you avoid some objects and try to catch others. Add or subtract points based on what your character touches.



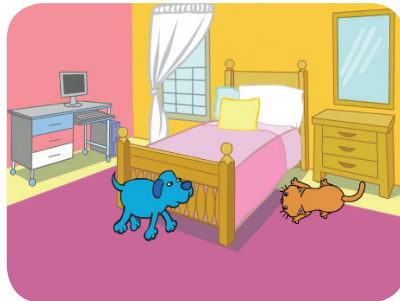
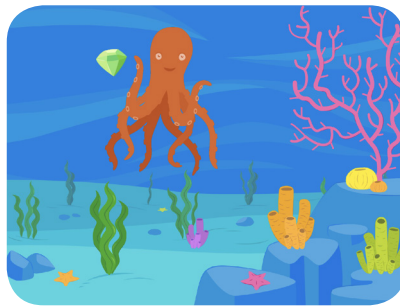
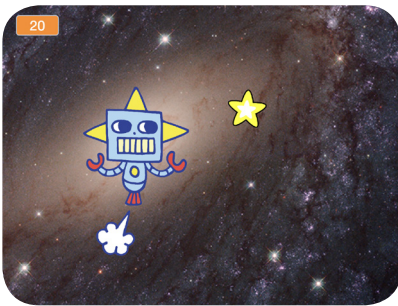
### Flying Stories

Tell a story about your flying characters. You can record your voice and play sound clips. Or, use say blocks to make voice bubbles.

## EDUCATOR GUIDE

# Make a Chase Game

With this guide, you can plan and lead a 55-minute lesson using Scratch. Students will make a game that includes a variable to keep score.



## Lesson Outline

Objective: Students will create a game using sensing.



**IMAGINE**  
*10 minutes*

First, gather as a group to introduce the theme and spark ideas.



**CREATE**  
*40 minutes*

Next, help students as they make chase games, working at their own pace through the tutorial.



**SHARE**  
*5 minutes*

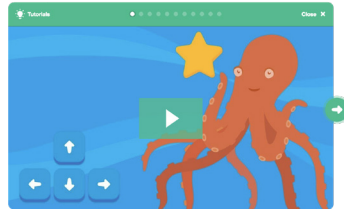
At the end of the session, gather together to share and reflect.

## Get Ready for the Lesson

Use this checklist to prepare for the lesson.

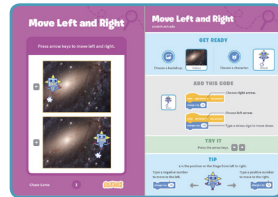
### Preview the Tutorial

The *Make a Chase Game* tutorial shows students how to create their own projects. Preview the tutorial before your lesson and try the first few steps,



### Print the Activity Cards (optional)

Print a few sets of *Chase Game* cards to have available for students during the lesson. You can download the cards at: [scratch.mit.edu/ideas](https://scratch.mit.edu/ideas)



### Make sure students sign into their Scratch accounts

Have students sign into their own Scratch accounts at [scratch.mit.edu](https://scratch.mit.edu).

### Set up computers or laptops

Arrange computers so that students can work individually or in pairs.

### Set up a computer with projector or large monitor

You can use a projector to show examples and demonstrate how to get started.

## Imagine



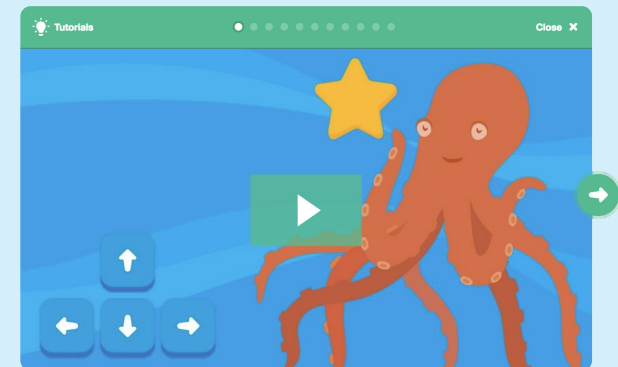
Begin by gathering the students to introduce the theme and spark ideas for projects.

### Warm-up Activity: Imaginary Chase

Gather the students in a circle. Start by giving an example of one thing chasing another, such as “The dog is chasing the dinosaur.” The next person adds on, such as, “The dinosaur is chasing a donut.” The following person adds on by saying, “The donut is chasing a duck.” or whatever creature or object they choose. Continue until each person has added on to this imaginary game of chase.

### Provide Ideas and Inspiration

To spark ideas, watch the *Make a Chase Game* video at the start of the tutorial.



View the video at [scratch.mit.edu/chase](https://scratch.mit.edu/chase)

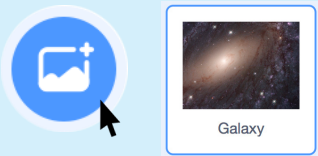


## Demonstrate the First Steps

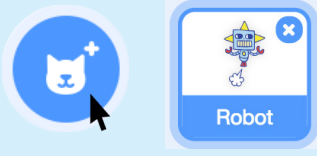


Demonstrate the first few steps of the tutorial so students can see how to get started.

**Choose a backdrop.**



**Choose a Sprite, like Robot.**



**Make your sprite move right and left with arrow keys.**

when **right arrow** key pressed

change x by **10**

Choose right arrow from the menu.



when **left arrow** key pressed

change x by **-10**





Choose left arrow from the menu.

Type a minus sign to move left.

Press the left arrow and right arrow keys on your keyboard to move.

**Discuss next steps they can try, such as coding the sprite to move up and down and adding a sprite to chase.**

## Create



Support students as they create catch games. Suggest working in pairs.

### Start with Prompts

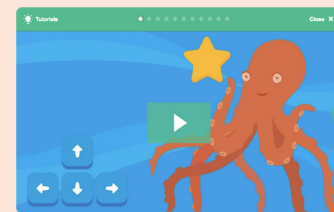
Ask students questions to get started

*Which backdrop would you like to choose for your game?*

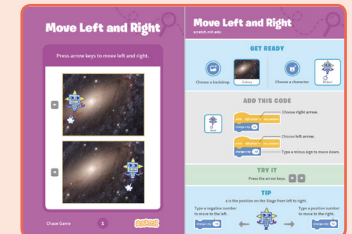
*Who do you want as the main character in your game? What will it chase?*

### Provide Resources

Offer options for getting started



Some students may want to follow the online tutorial: [scratch.mit.edu/chase](https://scratch.mit.edu/chase)



Others may want to explore using the printed cards: [scratch.mit.edu/ideas](https://scratch.mit.edu/ideas)

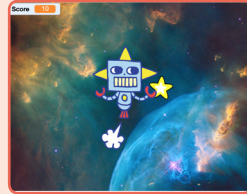
### Suggest Ideas for Starting

- Choose a backdrop
- Choose or draw a main character
- Make it move with arrow keys.
- Select an object to chase.



### More Things to Try

- Code the star or other sprite to chase
- Add a variable to keep score
- Add sounds
- Add a level
- Show a message when reaching the new level



### Encourage Tinkering

- Encourage students to feel comfortable trying combinations of blocks and seeing what happens.
- Suggest students look inside other chase games to see the code.
- If they find code they like, they can drag the scripts or sprites into the backpack to reuse in their own project.

### Prepare to Share

To add instructions and credits to a project, click the button: “*See project page*”.



# Share

Have students share their projects with their neighbors.

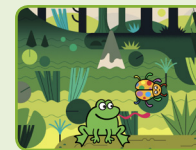
### Ask questions that encourage reflection:

*What do you like best about your game?*

*If you had more time, what would you add or change?*

## What's Next?

*Chase Game* projects provide an introduction to creating interactive games in Scratch. Here are a few ways that learners can build on the concepts they learned from this project.



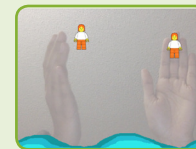
### Add Obstacles

For a more complex game, add obstacles to avoid. Subtract points when you hit the obstacles.



### Make a Two-Player Game

For an extra challenge, make a version of the game that allows two players to play.



### Video Sensing

If the computers have a web camera attached or built-in, learners can make a game that they interact by moving their bodies. See the Video Sensing tutorial and educator guide for support.

Created by the Scratch Team



## EDUCATOR GUIDE

# Pong Game

With this guide, you can plan and lead a 55-minute lesson using Scratch. Students will gain experience with coding as they design a bouncing ball game.



## Lesson Outline

Objective: Students will develop an interactive game using variables to keep score.



**IMAGINE**  
*10 minutes*

First, gather as a group to introduce the theme and spark ideas.



**CREATE**  
*40 minutes*

Next, help students as they make games, working at their own pace through the tutorial.



**SHARE**  
*5 minutes*

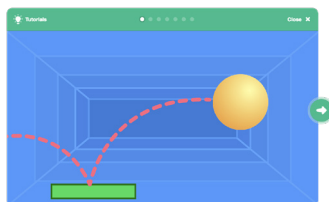
At the end of the session, gather together to share and reflect.

## Get Ready for the Lesson

Use this checklist to prepare for the lesson.

### Preview the Tutorial

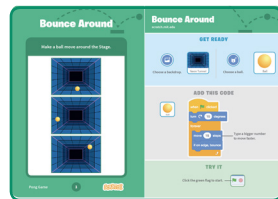
The *Pong Game* tutorial shows students how to create their own projects. Preview the tutorial before your lesson and try the first few steps: [scratch.mit.edu/pong](https://scratch.mit.edu/pong)



### Print the Activity Cards (optional)

Print a few sets of *Pong Game* cards to have available for students during the lesson.

[scratch.mit.edu/ideas](https://scratch.mit.edu/ideas)



### Make sure students sign into their Scratch accounts

Have students sign into their own Scratch accounts at [scratch.mit.edu](https://scratch.mit.edu).

### Set up computers or laptops

Arrange computers so that students can work individually or in pairs.

### Set up a computer with projector or large monitor

You can use a projector to show examples and demonstrate how to get started.

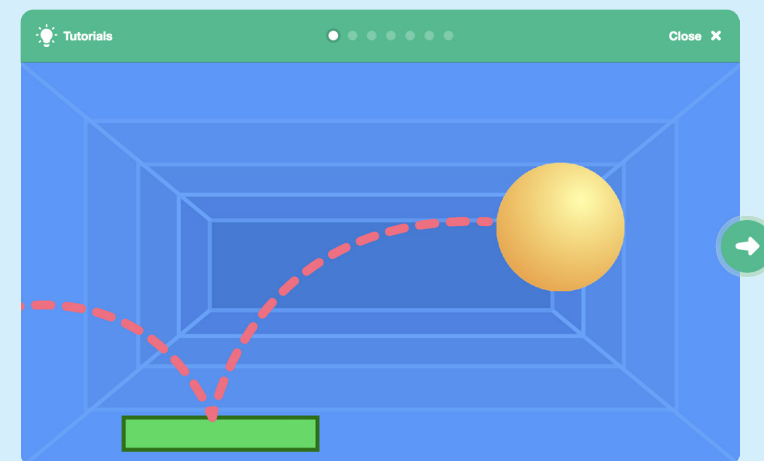
## Imagine



Begin by gathering the students to introduce the theme and spark ideas for projects.

### Provide Ideas and Inspiration

Show the introductory video for the *Pong Game* tutorial. The video shows pong games with a variety of themes, including everything from soccer to a magic potion-themed Pong game.



View at [scratch.mit.edu/pong](https://scratch.mit.edu/pong)

### Warm-up Activity: Bouncing Ideas

To get students thinking about a theme for their game, take turns calling out a theme, such as pizza pong or flower pong and brainstorming ideas for the type of images they could use to represent the theme.

## Demonstrate the First Steps



Demonstrate the first few steps of the tutorial so students can see how to get started.

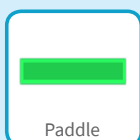
Go to the Scratch website. Click Create. Choose a new backdrop:



Choose a ball sprite and make it bounce around:



Add a paddle sprite and control it with the mouse:



## Create



Support students as they create pong games, on their own or in pairs.

### Start with Prompts

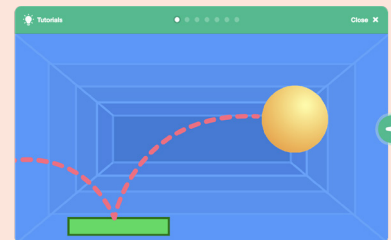
Ask students questions to get started

*What background do you want for your game?*

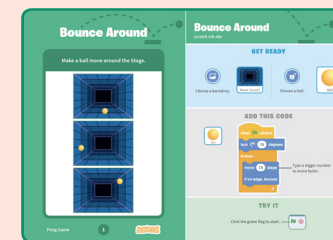
*What color or type of ball?*

### Provide Resources

Offer options for getting started



Some participants may want to follow the online tutorial:  
[scratch.mit.edu/pong](https://scratch.mit.edu/pong)



Others may want to use the printed activity cards:  
[scratch.mit.edu/ideas](https://scratch.mit.edu/ideas)

### Suggest Ideas for Starting

- Choose a backdrop
- Choose or draw a ball sprite and make it bounce around
- Add a paddle sprite that you can control
- Make the ball bounce off the paddle



### More Things to Try

- Add sounds and color effects
- Keep score by adding a variable
- Add a way to win or lose the game
- Change the backdrop when you reach a certain number of points
- Duplicate the ball for an added challenge



### Offer strategies for problem solving

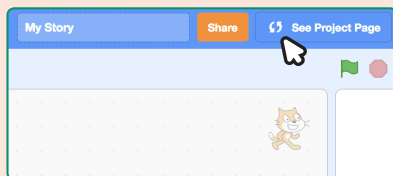
- Talk out what you're working on with someone
- Try out small bits of code at a time to figure out what's happening at each step
- Look closely at the blocks on the tutorial or activity cards to see if they are the same or different from the blocks you're using
- Look at the code for other pong games on the Scratch site



### Prepare to Share

To add instructions and credits to a project, click the button: "See project page".

Then click the Share button if you want the project visible to others online.



## Share

Have participants share their projects with others in the room.

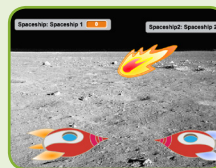
### Ask questions to encourage reflection:

*What did you notice about the games you tried?*

*What ideas might you add to your game?*

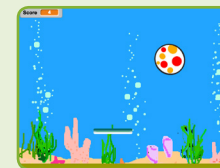
## What's Next?

Here are a couple of other directions you could suggest:



### Two-Player Game

For a more advanced project, try making a two-player game. To make a new version of your own project, click **File > Save as a Copy**.



### Remix a Game

A different way to make a pong game is to remix someone else's project, adding images and ideas. Find a project to remix in the **Pong Game Studio**: [scratch.mit.edu/studios/644508/](https://scratch.mit.edu/studios/644508/) Click '**See inside**', then click the '**Remix**' button.

Scratch is a project of the Lifelong Kindergarten Group at the MIT Media Lab.

## EDUCATOR GUIDE

# Create a Story

With this guide, you can plan and lead a 55-minute lesson using Scratch. Students will create a story with settings, characters, and dialogue.



## Lesson Outline

Objective: Students will create an animated story between at least two characters.



**IMAGINE**  
*10 minutes*

First, gather as a group to introduce the theme and spark ideas.



**CREATE**  
*40 minutes*

Next, help students as they create story projects, working at their own pace through the tutorial.



**SHARE**  
*5 minutes*

At the end of the session, gather together to share and reflect.



## Get Ready for the Lesson

Use this checklist to prepare for the lesson.

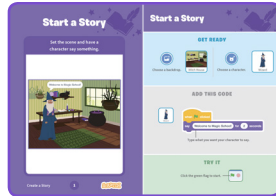
### Preview the Tutorial

The *Create a Story* tutorial shows students how to create their own projects. Preview the tutorial before your lesson and try the first few steps: [scratch.mit.edu/story](https://scratch.mit.edu/story)



### Print the Coding Cards (optional)

Print a few sets of *Create a Story* cards to have available for students during the lesson. You can download from this page: [scratch.mit.edu/ideas](https://scratch.mit.edu/ideas)



### Make sure students sign into their Scratch accounts

Have students sign into their own Scratch accounts at [scratch.mit.edu](https://scratch.mit.edu).

### Set up a studio for project sharing on Scratch

Set up a studio so students will be able to add their projects. Go to your *My Stuff* page, then click the **+New Studio** button. Type in a name for the studio.

### Set up computers or laptops

Arrange computers so that students can work individually or in pairs.

## Imagine



Begin by gathering the students to introduce the theme and spark ideas for projects.

### Warm-up Activity: Story Starters in a Bag

Have students make up a brief story by giving them a bag with three objects in it, and asking them to include all of the items in the story. In each bag, you could include small objects, pictures of animals or characters, and/or words (people, places, or things). Divide students into groups of two or three, and have each pick a bag. Give them a few minutes to come up with a quick story.

### Provide Ideas and Inspiration

You can show the *Create a Story* tutorial video to show students how they can start making stories in Scratch.



View the video at: [scratch.mit.edu/story](https://scratch.mit.edu/story)

## Demonstrate the First Steps



Demonstrate the first few steps of the tutorial so students can see how to get started.

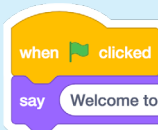
In Scratch, click Create.  
Choose a backdrop.



Choose any character (in Scratch called a *sprite*).



Code your character to say something.



Type what you want  
your character to say.

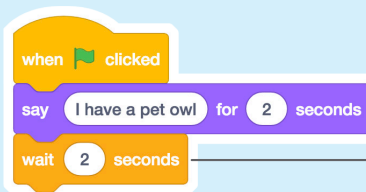
Click the green flag to start.



Add another character.



Add code to the new character.



Use this block to have the  
second character wait  
before they say something.

## Create



Support students as they create Story projects, on their own or in pairs.

### Start with Prompts

Ask students questions to get started

*Where will your  
story take place?*

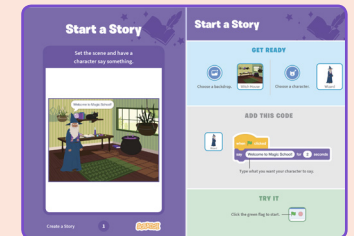
*What will  
happen first?*

### Provide Resources

Offer options for getting started



Some students may want to follow the online tutorial:  
[scratch.mit.edu/story](https://scratch.mit.edu/story)



Others may want to explore using the coding cards:  
[scratch.mit.edu/ideas](https://scratch.mit.edu/ideas)

### Suggest Ideas for Starting

- Choose a backdrop.
- Choose a character.
- Make a character say something
- Make a character hide and show.



### More Things to Try

- Switch backdrops.
- Make your characters have a conversation.
- Move your characters.
- Change something when you click on it.



### Support Tinkering

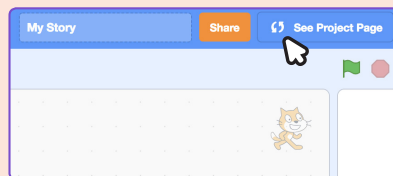
Scratch is designed to support creating by experimenting and tinkering. So, your students may want to start their stories without planning beforehand. As they create, one idea can spark another. Celebrate their sparks of creativity and the unexpected turns their stories may take.



### Prepare to Share

To add instructions and credits to a project, click the button: “See project page”.

Then click the Share button if you want the project visible to others online.



# Share

Help the students add their projects to a shared studio in Scratch. Give them a link to the studio. Then they can click ‘Add Projects’ at the bottom of the page.

Ask for volunteers to show their project to the group.

## What’s Next?

Students can use these ideas and concepts to create a variety of projects. Here are some variations on the story project you could suggest:



### Retell a story

Start with a story you know and make it in Scratch. Imagine a new ending or a different setting.



### Neighbourhood story

Take photos of your classroom, school, or neighborhood and use them as backdrops in your story.



### Round-robin story

Give everyone 5 minutes to start a story. Then, have them switch to the next computer to add to the story. Repeat.

Created by the Scratch Team