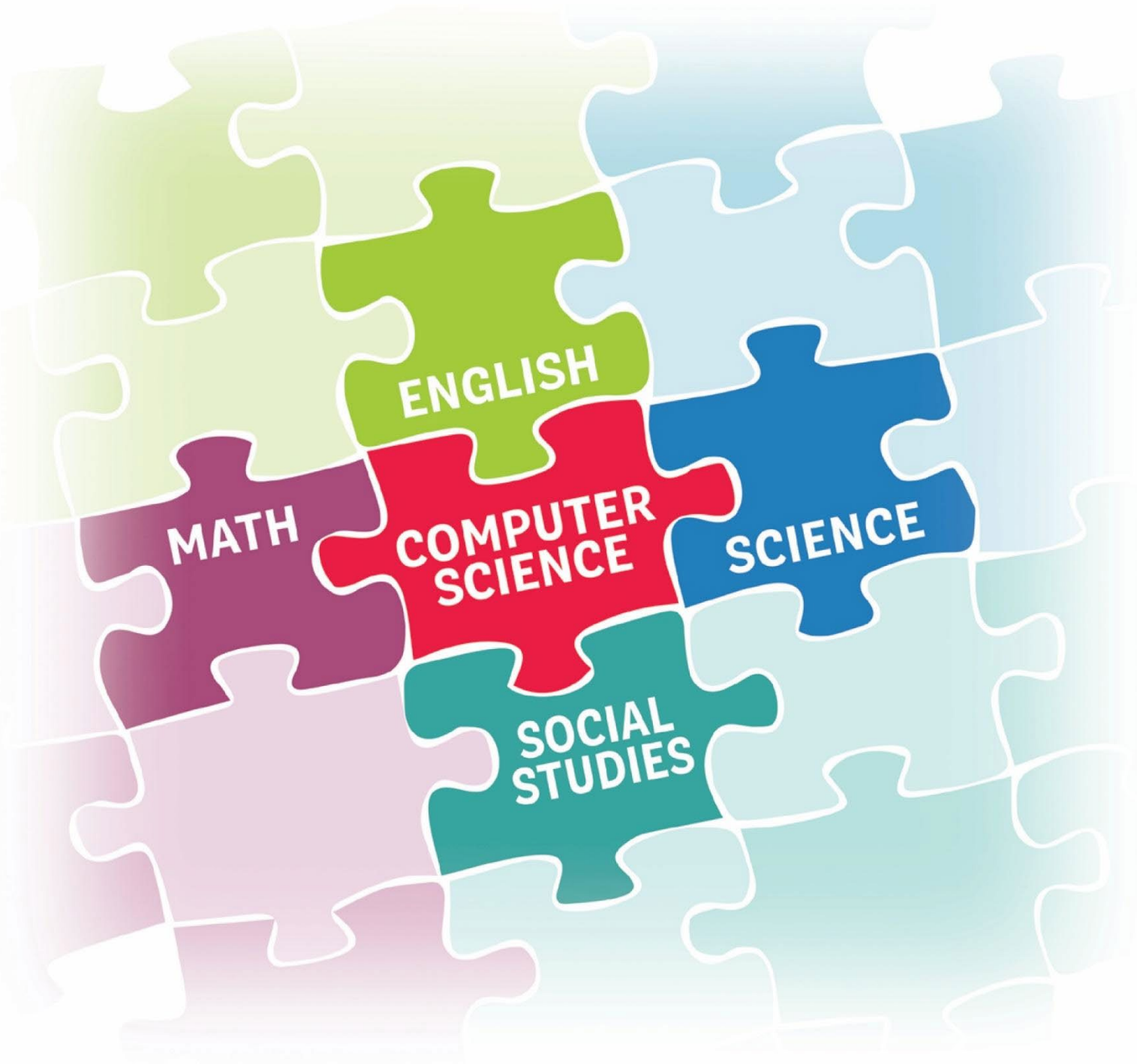


2022

Elementary Integration Guide

SECOND 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, Teach
Brittany Boatman, Teacher, H
Kimberly Brammer, Teacher, P
Michelle Carter, Teacher, Pica
Jana Chao, Teacher, Clinton P
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.

Resources	
Computing resources	• Code.org CS Fundamentals ○ 2nd Grade: Course C • Common Sense Digital Media • Kodable

	• Scratch, Jr.
CS4MS website materials	• 2018 Mississippi Computer Science Standards • CS4MS Website

Mouse practice	• Alphabetical Order
	• Mouse Practice
	  ○ Apple Catch
	  ○ Coyote Concentration (card matching game)
	  ○ Desert Dive
	  ○ Frost Bite
	  ○ Helipopper
	  ○ Penguin Drop
	  ○ Pickle Pop
	  ○ Pig Pile
	  ○ Simon Sees

Keyboard practice	• Astro Bubbles Keyboard Practice
	• Big Brown Bear
	• Read Today
	Unplugged:
	• Keyboard Callout
	  ○ **Paper keyboard**: Using a paper keyboard, the teacher will call out letters, numbers, symbols, and/or words for students to "type" on their keyboard.
	  ○ **Computer with no internet:** 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.
	• Keyboard Bingo
	  ○ 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. The teacher will distribute an equal number of random squares to students.
	  ○ The teacher will project a keyboard onto a smartboard.
	  ○ The teacher will call out letters, numbers, symbols, or words for students to find using their preprinted squares.
	  ○ 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.

Teacher / student accounts	• Code.org • Common Sense Digital Media • Kodable • Scratch, Jr.
For help with this guide	• Contact Mississippi State University's Center for Cyber Education: www.tinyurl.com/ccehelpdesk

Contents by Integrated Subjects

English

Math

Science

- **Week 17:** E.2.10.1, E.2.10.4, E.2.10.5—Renewable/nonrenewable resources, Soil erosion, Prevent/repair soil erosion
- **Week 38:** E.2.10.1—Renewable/nonrenewable resources

Social Studies

- **Week 4:** 2.CI.1.2—Citizenship in school and community
- **Week 5:** 2.CI.1.2—Citizenship in school and community
- **Week 6:** 2.CI.1.2—Citizenship in school and community
- **Week 7:** 2.CI.1.2—Citizenship in school and community
- **Week 8:** 2.CI.1.2, 2.CR.1.2—Citizenship in school and community, Civil rights
- **Week 9:** 2.CI.1.2—Citizenship in school and community
- **Week 10:** 2.CI.1.2—Citizenship in school and community
- **Week 15:** G.2.1.1—Map skills
- **Week 20:** E.2.1.1—Identify consumers/producers

Contents by Topics

Calendar/ Pacing Per Week:

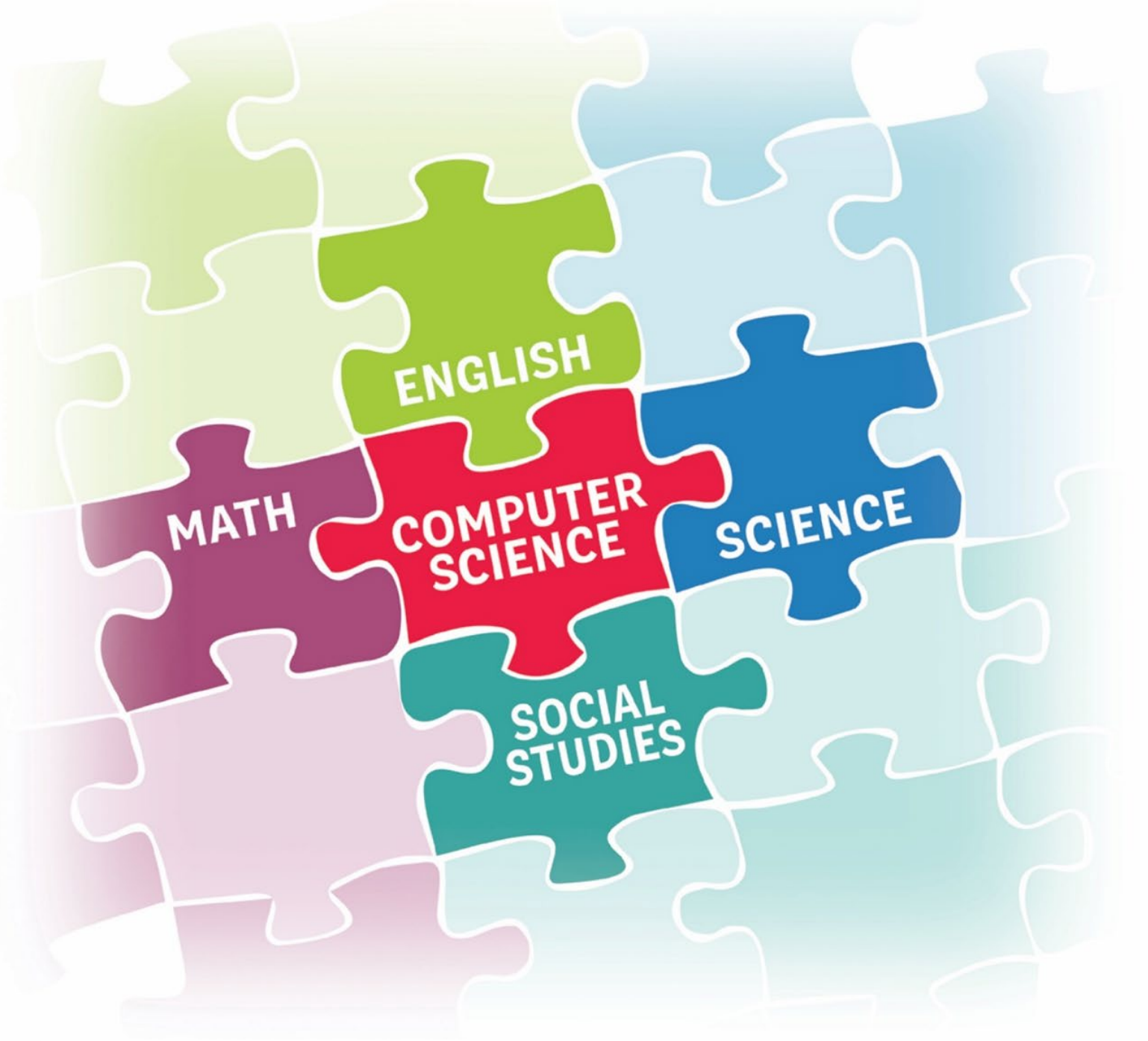
→Teachers will need to create a FREE **teacher and/or student account** (see notes section of lesson.)

Week	Title	Topics	CS Standard	Subject Integrated	Core Standard
1	What is a Computer?	Digital Literacy	CS.1A.2	ELA	L.2.5

2	Mouse Practice	Digital Literacy	CS.1A.2	ELA	L.2.5
3	Introduction to Keyboarding	Keyboarding Unplugged	CS.1A.2	ELA	L.2.5
4	We, the Digital Citizens → Account creation needed	Digital Citizenship	IC.1A.1 IC.1A.2	Social Studies	2.CI.1.2
5	Device-Free Moments	Digital Citizenship	IC.1A.1 IC.1A.2	Social Studies	2.CI.1.2
6	That's Private!	Digital Citizenship	IC.1A.3	Social Studies	2.CI.1.2
7	Digital Trails	Digital Citizenship	IC.1A.3	Social Studies	2.CI.1.2
8	Who is in Your Online Community?	Digital Citizenship	IC.1A.1 IC.1A.2	Social Studies	2.CI.1.2 2.CR.1.2
9	Let's Give Credit	Digital Citizenship	IC.1A.2	Social Studies ELA	2.CI.1.2 RI.2.1
10	Code.org: Putting a STOP to Online Meanness → Account creation needed	Digital Citizenship	IC.1A.2	Social Studies	2.CI.1.1 2.CI.1.2
11	Code.org: My Robotic Friends Jr.	Robotics Unplugged	AP.1A.2	ELA Math	SL.2.1 2.MD.9
12	Code.org: Programming With Angry Birds	Coding	AP.1A.2 AP.1A.4	ELA	L.2.1 L.2.2
13	Code.org: Bugs, Bugs, and More Bugs	Unplugged	AP.1A.2 AP.1A.4	ELA Math	L.2.1 L.2.2 2.OA.2 2.NBT.4
14	Code.org: Debugging in Maze	Coding	AP.1A.2 AP.1A.4 AP.1A.7	ELA	SL.2.1
15	Code.org: Collecting Treasure With Laurel	Coding	AP.1A.2 AP.1A.4	Social Studies	G.2.1.1
16	Code.org: Creating Art With Code	Coding	AP.1A.2 AP.1A.4 AP.1A.7	Math	2.G.1
17	Code.org: Shapes and Landscapes	Coding	AP.1A.2 AP.1A.4 AP.1A.7	Math Science	2.MD.3 E.2.10.1 E.2.10.4 E.2.10.5
18	Code.org: My Loopy Robotic Friends Jr.	Coding Unplugged	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.7	ELA Math	W.2.2 OA.2
19	Code.org: Loops With Rey and BB-8	Coding	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.7	ELA Math	W.2.1 2.OA.2

20	Code.org: Harvesting Crops With Loops	Coding	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.7	Social Studies	E.2.1
21	Code.org: Harvester's Farm	Coding Unplugged	AP.1A.5 AP.1A.8	ELA Math	W.2.2 W.2.5 2.OA.2
22	Code.org: Mini-Project—Sticker Art	Coding	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.7	ELA Math	W.2.1 2.G.A.1
23	Code.org: Loopy Forms and Their Functions	Coding	AP.1A.2 AP.1A.3 AP.1A.4 AP.1A.5	ELA Math	W.2.1 2.G.A.1
24	Code.org: The Big Event	Unplugged	AP.1A.2 AP.1A.4	ELA	SL.2.1
25	Code.org: Build a Flappy Game	Coding	AP.1A.2 AP.1A.4	ELA	SL.2.1
26	Code.org: Become a Game Designer	Coding	AP.1A.2 AP.1A.4 DA.1A.2	Math	2.MD.10
27	Code.org: Mini-Project: Chase Game	Coding	AP.1A.2 AP.1A.4	ELA	SL.2.1
28	Code.org: Picturing Data	Coding	AP.1A.09 DA.1A.1 DA.1A.2 DA.1A.3	Math	2.MD.10
29	Code.org: Binary Bracelets	Coding Unplugged	AP.1A.09 AP.1A.11	ELA	SL.2.6
30	Code.org: End of Course Project	Coding	AP.1A.10 AP.1A.12 AP.1A.13 AP.1A.15	ELA	SL.2.3
31	Gingerbread Coding	Coding Unplugged	AP.1A.1 AP.1A.3	ELA	W.2.2
32	Kodable.com: Pizza Party → Account creation needed	Coding Unplugged	AP.1A.1 AP.1A.4 AP.1A.5 2.MD.8a	ELA	SL.2.1 SL.2.2 SL.2.3
33	Kodable.com: Magnificent Maze Maker	Coding	AP. 1A.1	Math	G.A.2
34	Kodable.com: Magnificent Maze Maker Extended and Unplugged	Unplugged	AP. 1A.1	Math	2.G.2
35	Kodable.com: Introduction to Coding and Coding Basics Unplugged	Coding Unplugged	AP.1A.1 AP.1A.3 AP.1A.4 AP.1A.7	ELA	RL.2.1 SL.2.1

36	Kodable.com: Build Your Own Fuzz!	Coding	CS.1A.1	ELA	RL.2.7
37	Kodable.com: If Flash, Then Clap	Unplugged Coding	AP.1A.4	Math	2.OA.2
38	Kodable.com: Beach Cleanup	Coding	AP.1A.3 AP.1A.4 AP.1A.7	ELA Science	SL.2.1 SL.2.2 SL.2.3 E.2.10.1
39	Spelling Robotics	Robotics Unplugged	AP.1A.4 AP.1A.5	ELA	RF.2.3
40	Marching Orders	Robotics Unplugged	AP.1A.4 AP.1A.3 AP.1A.5	Math	2.G.1



Lessons and Activities

SECOND GRADE

Week 1: What is a Computer?

Lesson overview: 	**Purpose:** In this lesson, your students will learn the basic parts of computers including computer, monitor, desktop tower, keyboard, mouse, laptop, and tablet. This is an introduction to technology that they will use throughout school. Students will become familiar with the terms, complete a matching worksheet, and write definitions for each computer part while the teacher navigates through a PowerPoint presentation. **Lesson:** ● Warm Up ○ (Teacher Prompt): "What is a Computer?" - Let students give you the answers to what they think a computer is. ○ Watch the What is a Computer for Kids? video. ● Identifying Computer Parts Activity ○ Identifying Computer Parts PowerPoint: As you are going through the PowerPoint, students will find and match the pictures of the computer part on the Identifying Computer Parts Worksheet. Once they find the picture, students should draw a line from the picture to the correct term. (Please do not rush through the slides because moving slowly allows students to see both the picture and term on the board.) ● Using the "Find the Technology" game, have students start with the "review" feature to learn about different devices; then have them click on the "home" symbol to return to the main page and choose "search" to locate devices in the room displayed on the screen.
Lesson links/resources:	● What is a Computer for Kids? ● Identifying Computer Parts PowerPoint ● Identifying Computer Parts Worksheet ● "Find the Technology" Game ● Keyboarding Practice
CS standards addressed:	Students will be able to: ● Identify parts of the computer Standards: ● **CS.1A.2**—The student will use appropriate terminology in identifying and describing the function of common physical components of computing systems (hardware).
Time needed:	**Total Time: 60 min** ● Warm Up **10 min** ○ What is a Computer for Kids? Video ○ Discussion of video ● Activity 1 **20 min** ○ Identifying Computer Parts worksheet and PowerPoint (matching) ○ Identifying Computer Parts (re-writing terms) ● Activity 2 **20 min** ○ "Find the Technology" game ● Keyboarding **10 min**
Materials needed:	Teacher: ● Computer ● Projector/smartboard with sound ● Identifying Computer Parts PowerPoint Students:

	• Computer/tablet with internet access • Identifying Computer Parts Worksheet • "Find the Technology" game • Pencil • Computer
Subject integrated:	ELA
Other standards addressed:	**L.2.5**—Demonstrate understanding of word relationships and nuances in word meanings.
Vocabulary:	Computer: A device for working with information: data, music, videos, sounds Desktop: Screen that appears if you are not browsing the internet, reading a file, or playing a game; icons are on this screen Keyboard: Where all the letters, numbers and other buttons are located; when you type on it, the symbols appear on the monitor Laptop: A small, portable computer Monitor: Screen that shows you what you are doing; a viewer that displays what the computer sends to it Mouse: A little device you move with your hand, which then moves the cursor on the screen Tablet: A small, handheld computer with a touch screen
Notes:	As your students are completing the worksheet, make sure that you are going through the PowerPoint to show the various images/terms. This will help matching.

Week 2: Mouse Practice	
Lesson overview: 	**Purpose:** Computer Mouse Practice games help beginning computer users learn mouse skills through hand-eye coordination by dragging, dropping, clicking, double-clicking, and scrolling. • Apple Catch • Coyote Concentration (card matching game) • Desert Dive • Frostbite • Helipopper • Penguin Drop

	• Pickle Pop
	• Pig Pile
	• Simon Sees
	Lesson:
	• Refresher: Identifying Computer Parts PowerPoint
	• Review the parts of the computer with your students. Once you have gone back over the parts of the computer, show the video How Do Computers Work?
	• Warm Up
	  ○ Show the Using Your Computer Mouse video.
	  ○ Optional: Using the Trackpad for Chromebooks for kids video
	• Mouse Practice
	  ○ **Online:** Go to the mouse practice websites (links below). Let the students choose a game to play to practice using the mouse. (If your students are using a laptop or tablet, that's okay! They can still use the practice to operate a trackpad or touchscreen.)
	  ○ **Unplugged:** Mouse Practice Worksheet. Provide students with coloring utensils and the Mouse Worksheet. Guide students through the completion of the worksheet.
	• Student Voice
	  ○ Have students turn and talk to answer the questions: "How does a mouse help a computer?"
	  ○ Give students the opportunity to explain, in their own words, how they think a mouse is helpful.

Lesson links/resources:	• How Do Computers Work? • Using Your Computer Mouse • Alphabetical Order • Mouse Practice • Mouse Practice (unplugged)
CS standards addressed:	Students will be able to: • Operate a mouse and keypad Standards: • **CS.1A.2**—Use appropriate terminology in identifying and describing the function of common physical components of computing systems (hardware).
Time needed:	**Total Time: 60 min** • Refresher and Warm Up **15 min** • Mouse Practice **45 min** (can be divided into 15 min sessions across three days this week)
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Identifying Computer Parts PowerPoint(for refresher) Students: • Computer/tablet with internet access • Alphabetical Order • Mouse Practice • Mouse Practice (unplugged) • Pencil • Markers, crayons, color pencils, etc.
Subject integrated:	ELA

Other standards addressed:	**L.2.5**—Demonstrate understanding of word relationships and nuances in word meanings.
Vocabulary:	Computer: A device for working with information: data, music, videos, sounds Desktop: Screen that appears if you are not browsing the internet, reading a file, or playing a game; icons are on this screen Keyboard: Where all the letters, numbers, and other buttons are located; when you type on it, the symbols appear on the monitor Laptop: A small, portable computer Monitor- Screen that shows you what you are doing; a viewer that displays what the computer sends to it Mouse- A little device you move with your hand, which then moves the cursor on the screen Tablet- A small, handheld computer with a touch screen
Notes:	Even though this practice is predominantly for practice using a mouse, your class can use a tablet. This will still allow students to practice using a touchscreen device.

Week 3: Introduction to Keyboarding	

Lesson overview:
 
 
	Purpose:
	The computer keyboard activities in this lesson are designed to help beginning computer users learn keyboard skills through hand-eye coordination by finding letters/numbers/symbols and using a finger to press the key. Teachers, please understand that your students will mainly be using the "find & 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.
	Lesson:
	• Keyboard L-R Coloring Sheet
	  ○ Students will identify the left/right side of the keyboard. Students will color the left side of the keyboard to coordinate with the left hand. They will color the right side of the keyboard to coordinate with the right hand.
	  ○ The teacher will call out various letters/numbers to help students practice finding letters/numbers on a keyboard.
	• Keyboarding
	  ○ Online Option: Students can spend 15 minutes playing Astro Bubbles or another of the Keyboarding games.
	  ○ Unplugged Option: Using the Keyboard L-R Coloring Sheet, call out random letters and numbers and have your students find them and place their finger on them.

Lesson links/resources:	• Keyboarding games • Astro Bubbles Keyboard Practice • Keyboard L-R Coloring Sheet (unplugged)
CS standards addressed:	Students will be able to: • Label the letter for each key on their keyboard worksheet • Identify the left and right side of the keyboard, and they will learn which hands to use • Locate various letters and numbers on a keyboard Standards: • **CS.1A.2**—Use appropriate terminology in identifying and describing the function of common physical components of computing systems (hardware).
Time needed:	**Total Time: 60 min** • Keyboard Introduction **15 min** • Keyboarding Practice **45 min** (can be divided into 15 min sessions across three days this week)
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Identifying Parts of a Computer PowerPoint presentation (for refresher) Students: • Markers, crayons, color pencils, etc. • Keyboard coloring sheet • Laptop computer, desktop computer, or tablet that is connected to the internet
Subject integrated:	ELA
Other standards addressed:	**L.2.5**—Demonstrate understanding of word relationships and nuances in word meanings.
Vocabulary:	Computer: A device for working with information: data, music, videos, sounds Desktop: Screen that appears if you are not browsing the internet, reading a file, or playing a game; your icons are on this screen Input: To add information/data into the computer Keyboard: Where all the letters, numbers, and other buttons are located; when you type on it, the symbols appear on the monitor Laptop: A small, portable computer Monitor: Screen that shows you what you are doing; a viewer that displays what the computer sends to it Mouse: A little device you move with your hand, which then moves the cursor on the screen Output: To send information/data out of the computer Tablet: A small, handheld computer with a touch screen
Notes:	

Week 4: We, the Digital Citizens	
Lesson overview: 	**Purpose:** Students will understand that being a good digital citizen means being safe and responsible online. Students will take a pledge to be a good digital citizen. **Lesson:** ● We, The Digital Citizens: Lesson Plan– Read through this before the lesson. This contains prompts for the lesson. ● Warm Up ○ Digital Citizens Coloring Book: Choose different pages from this coloring book for students to color so everyone has something to share during the review in the slides. ○ Begin the We, The Digital Citizens Slides ● Play Student Video: We, The Digital Citizens Video ○ Follow prompts in the lesson plan. ● Explore the Digital Citizenship Pledge ● Pause and Think ○ We, The Digital Citizens Hand-out Any remaining time after the Commonsense Education lesson allows the students to practice keyboarding using the links from previous lessons.
Lesson links/resources:	● We, The Digital Citizens: Lesson Plan ● We, The Digital Citizens Video ● We, The Digital Citizens Slides ● Digital Citizens Coloring Book ● We, The Digital Citizens Handout
CS standards addressed:	Students will be able to: ● Understand digital citizenship ● Understand how to be responsible online Standards: ● **IC.1A.1**—Compare how people live and work before and after the implementation or adoption of new computing technology. ● **IC.1A.2**—Work respectfully and responsibly with others online.
Time needed:	**Total Time: 50 min** ● Warm Up **5 min** ● Play Student Video **2 min** ● Explore **10 min**

	• Pause and Think **5 min** • Keyboarding of your choice ~**30 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • We, The Digital Citizens- Lesson Plan • We, The Digital Citizens Video • Common Sense Account Students: • Computer/tablet with internet access • We, The Digital Citizens Coloring Book • We, The Digital Citizens Hand-out
Subject integrated:	Social Studies
Other standards addressed:	**2.CI.1.2**—Discuss how common civic virtues among citizens help create peaceful and orderly communities.
Vocabulary:	Digital citizen: Someone who uses technology safely and responsibly Pledge: A promise that one makes
Notes:	→Teachers will need to create FREE teacher and/or student accounts (when applicable) at https://www.commonsense.org/education/

Week 5: Device-Free Moments	
Lesson overview: 	**Purpose:** The students will recognize the ways in which digital devices can be distracting, identify how they feel when others are distracted by their devices, and identify ideal device-free moments for themselves and others. **Lesson:** ● Device-Free Moments: Lesson plan - Read through this before the lesson. This contains prompts for the lesson. ● Warm Up ○ Follow prompts and show either Sesame Street Device-Free Dinner Video or PBS Kids "Arthur - No Internet?!" Video. ● Explore: When and where devices should be used ○ Talk About: safety, respect, concentration, and sleep ● Create Family Device-free Rules ○ Device-Free Family Handout ● Reflect on finding a good balance in our lives ○ Pause & Think Handout Any remaining time after the Commonsense Education lesson allows the students to practice keyboarding using the links from previous lessons.
Lesson links/resources:	● Device-Free Moments: Lesson plan ● Device-Free Moments Slides ● Sesame Street Device-Free Dinner Video ● PBS Kids "Arthur - No Internet?!" Video ● Device-Free Family Handout ● Pause & Think Handout
CS standards addressed:	Students will be able to: ● Recognize how digital devices can be distracting and create rules for device-free time Standards: ● **IC.1A.1**—Compare how people live and work before and after the implementation or adoption of new computing technology. ● **IC.1A.2**—Work respectfully and responsibly with others online.
Time needed:	**Total Time: 60 min** ● Warm Up **5 min** ● Explore **10 min** ● Family Device-free Rules **20 min** ● Pause and Think **5 min** ● Keyboarding **20 min**
Materials needed:	Teacher: ● Computer ● Projector/smartboard with sound ● Device-Free Moments- Lesson Plan ● Device-Free Moments Slides ● Sesame Street Device-Free Dinner Video ● PBS Kids "Arthur - No Internet?!" Video ● Common Sense Account Students: ● Computer/tablet with internet access ● Device-Free Family Handout

	• Pause & Think Handout
Subject integrated:	Social Studies
Other standards addressed:	**2.CI.1.2**—Discuss how common civic virtues among citizens help create peaceful and orderly communities.
Vocabulary:	Attention: Noticing someone or something as important Concentration: Giving your full attention to a specific activity Distraction: Something that keeps you from giving your full attention
Notes:	

Week 6: That's Private!

Lesson overview:
 
	Purpose:
	The students will recognize the kind of information that is private and understand that they should never give out private information online.
	Lesson:
	• Learn
	  ○ Follow prompts That's Private! Lesson Plan
	  ○ Define and discuss the term "private."
	  ○ Begin That's Private Slides
	• Explore Online Forms
	  ○ Distribute and complete page 1 of Keep It Private Student Handout.
	• Pause and Think
	  ○ Complete page 2 of Keep It Private Student Handout.
	• Any remaining time after the Commonsense Education lesson allows the students time to practice keyboarding and using the mouse by revisiting the links from previous lessons.

Lesson links/resources:	• That's Private! Lesson Plan • That's Private Slides • Keep It Private Student Handout
CS standards addressed:	Students will be able to: • Determine information that should stay private • Know ways to keep information private Standards: • **IC.1A.3**—Keep login information private and log off devices appropriately.
Time needed:	**Total Time: 60 min** • Learn: What is Private? **10 min** • Explore Online Forms **15 min** • Pause and Think **5 min** • Keyboarding **15 min** • Mouse Practice **15 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • That's Private! Lesson Plan • That's Private Slides • Common Sense Account Students: • Computer/tablet with internet access • Keep It Private Student Handout
Subject integrated:	Social Studies
Other standards addressed:	**2.CI.1.2**—Discuss how common civic virtues among citizens help create peaceful and orderly communities.
Vocabulary:	Online: Using a computer, phone, or tablet to visit a website or app Private: Something that you should keep to yourself
Notes:	→Teachers will need to create FREE teacher and/or student accounts (when applicable) at https://www.commonsense.org/education/

Week 7: Digital Trails	
Lesson overview:  	**Purpose:** The students will learn that the information they share online leaves a digital footprint or "trail." Explore what information is okay to be shared online. **Lesson:** ● Before the Lesson ○ Print Digital Trail Squares and follow instructions on Digital Trails-Lesson Plan. ● Watch Following the Digital Trail video. ● Play "Digital Tracks" game. ○ Follow the prompts for Digital Trail Squares in the lesson plan. ● Review what is okay to share. ○ Ask students to complete the chart on slide 10 (also located on the Digital Trails Student Handout).

	• Wrap Up: Pause and think moment
	  ○ Project slide 12 and ask students to summarize which information is okay to share and which is not okay to share.
	  ○ Reflection on Digital Trails Student Handout
	Any remaining time after the Commonsense Education lesson allows the students time to practice keyboarding using the links from previous lessons.

Lesson links/resources:	• Digital Trails: Lesson plan • Digital Trail Squares • Following the Digital Trail: Video • Digital Trails Slides • Digital Trails Student Handout
CS standards addressed:	Students will be able to: • Identify what information is okay to share and what information should stay private Standards: • **IC.1A.3**—Keep login information private and log off of devices appropriately.
Time needed:	**Total Time: 60 min** • Before the Lesson **5 min** • Digital Tracks **15 min** • Okay to Share? **15 min** • Wrap Up: Pause and think moment **5 min** • Keyboarding **20 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Digital Trails- Lesson Plan • Digital Trails Slides • Following the Digital Trail: Video • Common Sense Account Students: • Computer/tablet with internet access • Digital Trail Squares • Digital Trails Student Handout • Common Sense Account
Subject integrated:	Social Studies
Other standards addressed:	**2. CI.1.2**—Distinguish behaviors of different individuals in the community that exhibit good citizenship.
Vocabulary:	Digital footprint: A record of what you do online, including the sites you visit and the things you share Permanent: Something that lasts forever Private information: Information about you that can be used to identify who you are Trail: A path or track
Notes:	→Teachers will need to create FREE teacher and/or student accounts (when applicable) at https://www.commonsense.org/education/

Week 8: Who is in Your Online Community?	
Lesson overview: 	**Purpose:** The student will compare and contrast how they are connected to different people and places in person and on the internet, then demonstrate an understanding of how people can connect on the internet. **Lesson:** ● Warm Up: What is the internet? ○ Follow prompts on Who Is in Your Online Community: Lesson plan ○ Begin Who Is In Your Online Community Slides ● Learn: How are we connected? ○ Holding a large ball or globe, discuss the prompts provided in the Who Is In Your Online Community: Lesson Plan. ● Explore: Online connections ○ Discuss the My Online Connections Handout and have students work independently on the first page to list three people in their community and world. ○ Follow the prompts on the lesson plan. ● Reflect: Pause and think moment ○ Discuss prompt and have students complete page 2 on the My Online Connections Handout. Any remaining time after the Common Sense Education lesson allows the students time to practice keyboarding and using the mouse by revisiting the links from previous lessons.
Lesson links/resources:	● Who Is in Your Online Community: Lesson plan ● Who Is in Your Online Community Slides ● My Online Connections Handout

CS standards addressed:	Students will be able to: • Compare and contrast how they are connected to different people and places around the world through the internet Standards: • **IC.1A.1**—Compare how people live and work before and after the implementation or adoption of new computing technology. • **IC.1A.2**—Work respectfully and responsibly with others online.
Time needed:	**Total Time: 60 min** • Warm Up: What is the internet? **5 min** • Learn: How are we connected? **10 min** • Explore: Online connections **10 min** • Reflect: Pause and think moment" **5 min** • Keyboarding practice twice more during this week using the links provided in previous lessons **15 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Large ball or globe • Who Is in Your Online Community Lesson Plan • Who Is in Your Online Community Slides • Common Sense Account Students: • Computer/tablet with internet access • My Online Connections Handout
Subject integrated:	Social Studies
Other standards addressed:	• **2.CI.1.2**—Distinguish behaviors of different individuals in the community that exhibit good citizenship. • **2.CR.1.2**—Evaluate the qualities that build unity among diverse populations.
Vocabulary:	Community: People who share a common neighborhood, background, or interests Internet: A worldwide network that connects people using computers, phones, or other devices
Notes:	→Teachers will need to create FREE teacher and/or student accounts (when applicable) at https://www.commonsense.org/education/

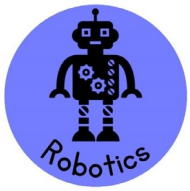
Week 9: Let's Give Credit	
Lesson overview: 	**Purpose:** The student will explain how giving credit signals respect for people's work and learn how to give credit in their schoolwork for content they use from the internet. **Lesson:** ● Warm Up: That's Mine! ○ Follow prompts on Let's Give Credit Lesson Plan ○ Begin Let's Give Credit Slides ● Explore: How do we give credit? ○ Follow slides and prompts ● Create: Research report ○ Let's Give Credit Student Report ● Reflect: Pause and think moment ○ Pause and Think Handout Any remaining time after the Common Sense Education lesson allows the students time to practice keyboarding and mouse techniques using the links from previous lessons.
Lesson links/resources:	● Let's Give Credit Lesson Plan ● Let's Give Credit Slides ● Let's Give Credit Teacher Report ● Let's Give Credit Student Report ● Pause and Think Handout
CS standards addressed:	Students will be able to: ● Show how to give credit to other people's work Standards: ● **IC.1A.2**—Work respectfully and responsibly with others online.
Time needed:	**Total Time: 60 min** ● Warm Up: That's Mine! **5 min** ● Explore: How Do We Give Credit? **10 min** ● Create Research Report **10 min** ● Reflect: Pause and think moment **5 min** ● Keyboarding **30 min** (can be divided into 15 min sessions across two days this week)
Materials needed:	Teacher: ● Computer ● Projector/smartboard with sound ● Let's Give Credit Lesson Plan ● Let's Give Credit Slides ● Let's Give Credit Teacher Report ● Common Sense Account

	Students: • Computer/tablet with internet access • Let's Give Credit Student Report • Pause and Think Handout • Common Sense Account
Subject integrated:	Social Studies ELA
Other standards addressed:	**2.CI.1.2**—Distinguish behaviors of different individuals in the community that exhibit good citizenship. **RI.2.1**— Ask and answer such questions as who, what, where, when, why, and how to demonstrate understanding of key details in a text.
Vocabulary:	Credit: Giving recognition to a person that created something Respect: Showing that you appreciate someone
Notes:	→Teachers will need to create FREE teacher and/or student accounts (when applicable) at https://www.commonsense.org/education/

Week 10: Code.org, Course C, Lesson 1—Putting a STOP to Online Meanness

Lesson overview:
 
	Purpose:
	Common Sense Education created this lesson to teach students about meanness and what they should do if they encounter it online.
	Lesson:
	• Learn: What is meanness?
	  ○ Follow prompts on Code.org, Lesson 1: Putting a STOP to Online Meanness Lesson Plan
	  ○ Begin Putting a STOP to ONLINE Meanness Slides
	  ○ Review key vocabulary
	• Perform: Online vs. face-to-face
	  ○ Follow prompts
	• Explore: Stop meanness
	  ○ STOP Online Meanness Student Handout
	  ○ Follow prompts
	• Wrap Up: Pause and think
	  ○ Follow prompts and complete handout

Lesson links/resources:	• Code.org, Lesson 1: Putting a STOP to Online Meanness Lesson Plan • Putting a STOP to Online Meanness Slides • STOP Online Meanness Student Handout
CS standards addressed:	Students will be able to: • Identify ways to respond to mean words online, using S-T-O-P • Understand what online meanness can look like and how it can make people feel Standards: • **IC.1A.2**—Work respectfully and responsibly with others online.
Time needed:	**Total Time: 60 min** • Learn: What is meanness? **10 min** • Perform: Online vs. face-to-face **5 min** • Explore: Stop meanness **15 min** • Wrap Up: Pause and think **5 min** • Keyboarding games **25 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Code.org account • Code.org Lesson 1: Putting a STOP to Online Meanness Lesson Plan • Putting a STOP to Online Meanness Slides Students: • Computer/Tablet with internet access • STOP Online Meanness Student Handout
Subject integrated:	Social Studies
Other standards addressed:	• **2.CI.1.1**—Identify civic virtues and civic responsibilities. • **2.CI.1.2**—Discuss how common civic virtues among citizens help create peaceful and orderly communities
Vocabulary:	Meanness: The state or quality of being mean Respect: To consider worthy of high regard
Notes:	→ Teachers will need to create FREE teacher and/or student accounts https://studio.code.org/users/sign_in

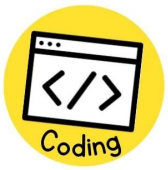
Week 11: Code.org, Course C, Lesson 2—My Robotic Friends Jr.	
Lesson overview:  	**Purpose:** This unplugged lesson brings the class together as a team with a simple task of getting a "robot" to stack cups in a specific design. This activity lays the groundwork for the programming that students will do throughout the course as they learn the importance of defining a clearly communicated algorithm. **Lesson:** ● Before the Lesson ○ Follow prompts in Code.org, Lesson 2: My Robotic Friends Jr. ○ Watch My Robotic Friends-Unplugged Video (Download) ● Warm Up ○ Talking to robots: Watch one of the following videos to demonstrate what robots can do. ■ Asimo by Honda (3:58) ■ Egg Drawing Robot (3:15) ■ Dancing Lego Robot (1:35) ● Activity ○ My Robotic Friends

	• Follow prompts in Code.org, Lesson 2: My Robotic Friends Jr.
	• Wrap Up
	  ○ Reflection
	  ■ Draw your own stack of cups that you would like to see a robot build.
	  ■ Can you create a program for that cup stack?

Lesson links/resources:	• Code.org, Lesson 2: My Robotic Friends Jr. • My Robotic Friends: Cup Spacing • My Robotic Friends: Symbol Key • My Robotic Friends: Unplugged Video (Download) • My Robotic Friends: Cup Stacking Ideas • My Robotic Friends: Paper Trapezoid Template
CS standards addressed:	Students will be able to: • Attend to precision when creating instructions • Identify and address bugs or errors in sequenced instructions Standards: • **AP.1A.2**—Model the way programs store and manipulate data by using numbers or other symbols to represent information.
Time needed:	**Total Time: 60 min** • Warm Up: Talking to robots **5 min** • Activity: My Robotic Friends **30 min** • Wrap Up **10 min** • Keyboarding **15 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Code.org account • My Robotic Friends- Unplugged Video (Download) Students: • Computer/tablet with internet access • My Robotic Friends: Cup spacing • My Robotic Friends: Symbol key • My Robotic Friends: Cup stacking ideas • My Robotic Friends: Paper trapezoid template
Subject integrated:	ELA Math
Other standards addressed:	ELA • **SL.2.1**—Participate in collaborative conversations with diverse partners about Grade 2 topics and texts with peers and adults in small and larger groups. Math • **2.MD.9**—Generate measurement data by measuring lengths of several objects to the nearest whole unit, or by making repeated measurements of the same object. Show the measurements by making a line plot, where the horizontal scale is marked off in whole-number units.
Vocabulary:	Algorithm: A list of steps to finish a task Bug: Part of a program that does not work correctly Debugging: Finding and fixing problems in an algorithm or program Program: An algorithm that has been coded into something that can be run by a machine

Notes:	→ Teachers will need to create FREE teacher and/or student accounts https://studio.code.org/users/sign_in

Week 12: Code.org, Course C, Lesson 3—Programming With Angry Birds

Lesson overview:  	**Purpose:** 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. **Lesson:** ● Warm Up: Introduction ○ Review My Robotic Friends activity from Week 11 ○ Follow prompts in Code.org, Lesson 3: Programming with Angry Birds ● Bridging Activity: Programming ○ Display Maze Bridging Page for students to see ○ Follow prompts ● Previewing Online: Puzzles as a class ○ Pull up puzzle 5 to do in front of the class ○ Follow prompts ● Main Activity: Programming with Angry Birds ○ Students will watch Maze Intro-Programming with Blocks ○ Students will only complete all levels of Programming with Angry Birds ● Wrap Up: Reflection ○ What was today's lesson about? ○ How did you feel during today's lesson? ○ Draw an activity to do that you struggled with the first time. Draw or describe how you got better.
Lesson links/resources:	● Code.org, Lesson 3: Programming with Angry Birds ● Maze Intro: Programming with Blocks video ● Maze Bridging Page ● Unplugged Maze Blocks: Manipulatives
CS standards addressed:	Students will be able to: ● Identify and locate bugs in a program

	• 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 into a precise sequence of instructions.
Time needed:	**Total Time: 55 min** • Warm Up: Introduction **5 min** • Bridging Activity: Programming **10 min** • Previewing Online: Puzzles as a class **5 min** • Main Activity: Programming with Angry Birds **30 min** • Wrap Up **5 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Maze Bridging Page • Maze Intro: Programming with Blocks: Video • Code.org account Students: • Computer/tablet with internet access • Unplugged Maze Blocks: Manipulatives • Code.org account
Subject integrated:	ELA
Other standards addressed:	ELA • **L.2.1**—Demonstrate command of the conventions of standard English grammar and usage when writing (printing, cursive, or keyboarding) or speaking. • **L.2.2**—Demonstrate command of the conventions of standard English capitalization, punctuation, and spelling when writing.
Vocabulary:	Algorithm: A list of steps to finish a task Bug: Part of a program that does not work correctly Debugging: Finding and fixing problems in an algorithm or program Sequencing: Putting commands in correct order so computers can read the commands
Notes:	

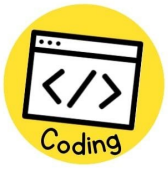
Week 13: Code.org, Course C, Lesson 3 (Extension): Bugs, Bugs, and More Bugs	

Lesson overview:
 
	Purpose:
	Whether fixing code, correcting a math problem or proofreading writing, debugging is an important part of life. Learning how to spot errors and knowing how to correct them will help prevent bugs, bugs, and more bugs!
	Lesson:
	• Note
	  ○ Students completed Programming with Angry Bird, Levels 1-11 in Week 12.
	  ○ Follow prompts in Bugs, Bugs, and More Bugs: Lesson plan
	• Prompt
	  ○ Did you feel frustrated while debugging your code? Explain why or why not debugging was frustrating to you.
	• Video
	  ○ Watch the How to Debug video.
	• Bugs, Bugs, and More Bugs: Student Handout
	  ○ Debugging code
	  ○ Debugging Math Problems
	  ○ Debugging Sentences
	• Wrap Up
	  ○ Have students pair up to discuss the following prompt:
	  ■ Today you had practice debugging code, math problems, and sentences. How do you use debugging in other areas of your life?

Lesson links/resources:	• Bugs, Bugs, and More Bugs: Lesson plan • Bugs, Bugs, and More Bugs: Student handout
CS standards addressed:	Students will be able to: • Identify and locate bugs in a program • 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 into a precise sequence of instructions.
Time needed:	**Total Time: 48 min** • Discuss Prompt **5 min** • Video **3 min** • Student Handout **30 min** • Wrap Up **10 min**
Materials needed:	Teacher: • Computer

	• Projector/smartboard with sound • Bugs, Bugs & More Bugs Lesson Plan • Code.org account Students: • Computer/tablet with internet access • Paper • Bugs, Bugs, and More Bugs Student Handout
Subject integrated:	ELA Math
Other standards addressed:	ELA • **L.2.1**—Demonstrate command of the conventions of standard English grammar and usage when writing (printing, cursive, or keyboarding) or speaking. • **L.2.2**—Demonstrate command of the conventions of standard English capitalization, punctuation, and spelling when writing. Math • **2.OA.2**—Fluently add and subtract within 20 using mental strategies. By the end of Grade 2, know from memory all sums of two one-digit numbers. • **2.NBT.4**—Compare two three-digit numbers based on meanings of the hundreds, tens, and ones digits, using >, =, and < symbols to record the results of comparisons.
Vocabulary:	Algorithm: A list of steps to finish a task Bug: Part of a program that does not work correctly Debugging: Finding and fixing problems in an algorithm or program Sequencing: Putting commands in correct order so computers can read the commands
Notes:	

Week 14: Code.org, Lesson 4—Debugging in Maze	
Lesson overview:	**Purpose:**

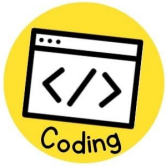
	Students in your class might become frustrated with this lesson because of the essence of debugging. Debugging is a concept that is very important to computer programming. Computer scientists have to get really good at facing the bugs in their own programs. Debugging forces the students to recognize problems and overcome them while building critical thinking and problem-solving skills.
	Lesson:
	• Warm Up
	  ○ Introduction: Debugging
	  ○ Follow prompts in Code.org, Lesson 4 Debugging in Maze Lesson Plan
	• Main Activity
	  ○ Debugging in Maze
	  ■ Students will watch the Debugging with Step Button video and work through the levels in this lesson.
	• Wrap Up: Reflection
	  ○ What was today's lesson about?
	  ○ How did you feel during today's lesson?
	  ○ What kind of bugs did you find today?
	  ○ Draw a bug you encountered in one of the puzzles today. What did you do to debug the program?

Lesson links/resources:	• Code.org, Lesson 4 Debugging in Maze Lesson Plan • Pair Programming Video • Debugging with Step Button • Main Activity Notes • Debugging Guide
CS standards addressed:	Students will be able to: • Modify an existing program to solve errors • Predict where a program will fail • Reflect on the debugging process in an age-appropriate way 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 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: 60 min** • Warm Up: Introduction to debugging **15 min** • Debugging in Maze **30 min** • Wrap Up: Reflection (prompts provided) **10 min** • Keyboarding **5 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Code.org, Lesson 4: Debugging in Maze Lesson Plan • Pair programming video • Debugging with step button • Main activity notes • Code.org account Students: • Computer/tablet with internet access • Debugging guide • Code.org account

Subject integrated:	ELA
Other standards addressed:	**SL.2.1**—Participate in collaborative conversations with diverse partners about Grade 2 topics and texts with peers and adults in small and larger groups.
Vocabulary:	Bug: Part of a program that does not work correctly Debugging: Finding and fixing problems in an algorithm or program Perseverance: The ability to keep doing something in spite of obstacle
Notes:	

Week 15: Code.org, Course C, Lesson 5—Collecting Treasure With Laurel

Lesson overview:



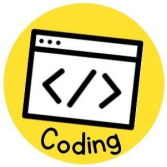
Purpose:

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

Lesson:

- Warm Up: Introduction to collecting
  - Bridging activity: Preview online puzzles
  - Pull a puzzle from the corresponding online stage. We recommend puzzle 7. Have students discuss a pattern that they think will get Laurel the Adventurer to collect all the treasure. Ask the students to share. See how many other students had the same answer!

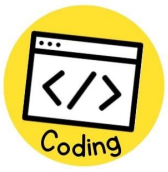
	○ Follow prompts on Code.org, Lesson 5: Collecting Treasure with Laurel Lesson Plan ● Main Activity ○ Collecting Treasure with Laurel ● Wrap Up: ○ What was today's lesson about? ○ How did you feel during today's lesson? ○ Draw a maze that you might solve with the blocks you used today.
Lesson links/resources:	● Code.org, Lesson 5: Collecting Treasure with Laurel Lesson Plan ● Unplugged Blocks: Manipulatives
CS standards addressed:	Students will be able to: ● Develop problem solving and critical thinking skills by reviewing debugging practices ● Order movement commands as sequential steps in a program ● Represent an algorithm as a computer program 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 into a precise sequence of instructions.
Time needed:	**Total Time: 60 min** ● Warm Up: Introduction to collecting **5 min** ● Preview Online Puzzles **10 min** ● Collecting Treasure with Laurel **30 min** ● Wrap Up **5 min** ● Keyboarding **10 min**
Materials needed:	Teacher: ● Computer ● Projector/smartboard with sound ● Code.org, Lesson 5: Collecting Treasure with Laurel Lesson Plan ● Code.org account Students: ● Computer/tablet with internet access ● Unplugged Blocks: Manipulatives ● Code.org account
Subject integrated:	Social Studies
Other standards addressed:	**G.2.1.1**—Differentiate between different types of maps
Vocabulary:	Algorithm: A list of steps to finish a task Program: An algorithm that has been coded into something that can be run by a machine Programming: The art of creating a program
Notes:	

Week 16: Code.org, Course C, Lesson 6—Creating Art With Code	
Lesson overview:  	**Purpose:** Building off of the students' previous experience with sequencing, this lesson will work to inspire more creativity with coding. The purpose of this lesson is to solidify knowledge on sequencing by introducing new blocks and goals. In this case, students learn more about pixels and angles using the new blocks, while still practicing their sequencing skills. Also, students will be able to visualize new goals such as coding the artist to draw a square. **Lesson:** This is Part 1 of a two-part lesson. Part 2 is Shapes and Landscapes (Week 17) ● Warm Up: Introduction to angles ○ Follow prompts on Code.org, Lesson 6: Creating Art with Code ○ Artist Introduction: Student video ○ Turns and Angles: Student video (pause at the 1:54 minute mark for student reference) ● Main Activity ○ Creating art with code ● Wrap Up ○ What was today's lesson about? ○ How did you feel during today's lesson? ○ What are the interior angles that make up a square? What about a triangle? ○ Sketch a simple shape on your paper and imagine the code used to draw it. Can you write that code out next to the shape?
Lesson links/resources:	● Code.org, Lesson 6: Creating Art with Code ● Artist Introduction: Student video ● Turns & Angles: Student video ● Turns & Angles: Student handout

CS standards addressed:	Students will be able to: ● Break complex shapes into simple parts ● Create a program to complete an image using sequential steps 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 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: 60 min** ● Warm Up: Introduction to angles **10 min** ● Activity: Creating art with code **30 min** ● Wrap Up: Reflection (prompts provided) **10 min** ● Keyboarding **10 min**
Materials needed:	Teacher: ● Computer ● Projector/smartboard with sound ● Code.org, Lesson 6: Creating Art with Code ● Code.org account Students: ● Computer/tablet with internet access ● Artist Introduction: Student video ● Turns & Angles: Student video ● Turns & Angles: Student handout ● Code.org account
Subject integrated:	Math
Other standards addressed:	Math **2.G.1**—Recognize and draw shapes having specified attributes, such as a given number of angles or a given number of equal faces. Identify triangles, quadrilaterals, pentagons, hexagons, and cubes.
Vocabulary:	Trapezoid: A quadrilateral with at least one pair of parallel sides
Notes:	

Week 17: Code.org, Course C, Lesson 6 (Cross-Curricular)—Shapes and Landscapes

Lesson overview:

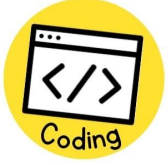


Purpose:
Students are acting as an engineer at the USA Dam Construction Company. A local community is having issues with flooding. Every time it floods, soil washes away and makes the roads unsafe. Students have been asked to design a dam to prevent future flooding. Using code, you will create a blueprint to show the local town council how your dam will look. Building off of the students' previous experience with sequencing, this lesson will work to inspire more creativity with coding. The purpose of this lesson is to solidify knowledge on sequencing by introducing new blocks and goals. In this case, students learn more about pixels and angles using the new blocks while still practicing their sequencing skills. Also, students will be able to visualize new goals such as coding the artist to draw a square.

Lesson: This is Part 2 of a two-part lesson. Part 1 is Creating Art with Code (Week 16)

- Warm Up
  - Review students on dams and other water structures that may shape the land and change the direction of water.
  - Watch one of the following videos:
    - What is a dam?: Student Video
    - Why are there dams?: Student Video
    - Why do we need dams?: Student Video
  - Complete the "K – Write what you know" and "W- Write what you want to know" boxes on the Shapes and Landscapes Student Handout.
- Main Activity:
  - Students completed Creating Art with Code Levels 2-7 in Week 16.
  - Each student will complete the **Coding Your Dam Blueprint** section of the Shapes and Landscapes Student Handout. Students will use Creating Art with Code, Level 2 to complete their dam blueprint.
  - Students will complete the **Math Connection** section of the Shapes and Landscapes Student Handout.
- Wrap Up:
  - Students will complete the "L - Write what you learned" box on the KWL Chart located on the Shapes and Landscapes Student Handout.
- Wrap Up (Optional Extension):
  - Shapes & Landscapes Student Handout: Besides building a dam, what other solutions could you use to slow or prevent wind or water from changing the shape of the land?
    - Examples of solutions could include different designs of dikes and windbreaks to hold back wind and

	water, and different designs for using shrubs, grass, and trees to hold back the land.
Lesson links/resources:	• Shapes & Landscapes Student Handout • What is a dam?: Student Video • Why are there dams?: Student Video • Why do we need dams?: Student Video
CS standards addressed:	Students will be able to: • Break complex shapes into simple parts • Create a program to complete an image using sequential steps 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 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: 60 min** • Warm Up **10 min** • Main Activity: Shapes and Landscapes. **30 min** • Wrap Up: Reflection **10 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Code.org, Lesson 6: Shapes and Landscapes • Code.org account Students: • Computer/tablet with internet access • Shapes & Landscapes Student Handout • What is a dam?: Student Video • Why are there dams?: Student Video • Why do we need dams?: Student Video • Code.org account
Subject integrated:	Math Science
Other standards addressed:	Math • **2.MD.3**—Estimate lengths using units of inches, feet, centimeters, and meters. Science • **E.2.10.1**—Using informational text, other media, and first-hand observations to investigate, analyze and compare the properties of Earth materials (including rocks, soils, sand, and water). • **E.2.10.4**—Using informational text, other media, and first-hand observations to investigate and communicate the process and consequences of soil erosion. • **E.2.10.5**—With teacher guidance, investigate possible solutions to prevent or repair soil erosion.
Vocabulary:	Decompose: To break down or break apart
Notes:	

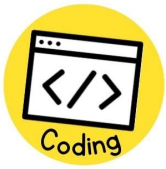
Week 18: Code.org, Course C, Lesson 7: My Loopy Robotic Friends Jr.	
Lesson overview:  	**Purpose:** This lesson serves as a reintroduction to loops, using the now familiar set of "robot" programming instructions. Students will develop critical thinking skills by looking for patterns of repetition in the movements of classmates and determining how to simplify those repeated patterns using loops. **Lesson:** ● Warm Up: My Robotic Friends review ○ Follow prompts in Code.org- Lesson 7: My Loopy Robotic Friends, Jr. Lesson Plan ○ Display My Robotic Friends: Symbol key ○ Pull a puzzle from My Robotic Friends: Cup stacking ideas

	• Introduce My Loopy Friend by showing Unplugged Activity: My Loopy Robotic Friends video
	• Activity: My Loopy Robotic Friends
	  ○ Follow prompts in lesson plan
	  ○ Think, pair, share, model
	  ○ Looping your robots: Follow prompts
	• Wrap Up: Reflection
	  ○ Have the students write or draw something in their journal, or on a piece of paper, that will remind them what loops are. This can come from a prompt like:
	  ■ What does "repeat" mean to you?
	  ■ Draw a picture of you repeating something.

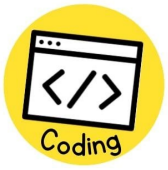
Lesson links/resources:	• Code.org, Lesson 7: My Loopy Robotic Friends Jr. Lesson Plan • My Robotic Friends: Symbol Key • My Robotic Friends: Cup Stacking Ideas • My Robotic Friends: Cup Spacing • My Robotic Friends: Paper Trapezoid Template • Unplugged Activity: My Loopy Robotic Friends video • Feeling Faces Emotion Image- Resource
CS standards addressed:	Students will be able to: • Identify repeated patterns in code that could be replaced with a loop • Write instructions that use loops to repeat patterns 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 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: 60 min** • Warm Up: My Robotic Friends review **10 min** • Activity: My Loopy Robotic Friends **30 min** • Wrap Up: Reflection (prompts provided) **5 min** • Keyboarding **15 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Code.org, Lesson 7: My Loopy Robotic Friends Jr. Lesson Plan • My Robotic Friends: Symbol key • My Robotic Friends: Cup stacking ideas • Code.org account Students: • Journal • Stacking cups (20 per each group of 4 students) • My Robotic Friends: Symbol key • My Robotic Friends: Cup stacking ideas • My Robotic Friends: Cup spacing • My Robotic Friends: Paper trapezoid template
Subject integrated:	ELA Math

Other standards addressed:	ELA • **W.2.2**—Write informative/explanatory texts in which they introduce a topic, use facts and definitions to develop points, and provide a concluding statement or section. Math • **2.OA.2**—Fluently add and subtract within 20 using mental strategies. By the end of Grade 2, know from memory all sums of two one-digit numbers.
Vocabulary:	Loop: The action of doing something over and over again Repeat: To do something again
Notes:	

Week 19: Code.org, Course C, Lesson 8: Loops With Rey and BB-8	

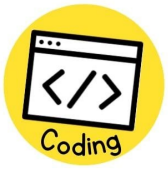
Lesson overview:
 
	Purpose:
	In this lesson, students will be learning more about loops and how to implement them in Blockly code. Using loops is an important skill in programming because manually repeating commands is tedious and inefficient. With the Code.org puzzles, students will learn to add instructions to existing loops, gather repeated code into loops, and recognize patterns that need to be looped. It should be noted that students will face puzzles with many different solutions. This will open up discussions on the various ways to solve puzzles with advantages and disadvantages to each approach.
	Lesson:
	• Warm Up: Introduction—My Loopy Robotic Friends
	  ○ Follow prompts in Code.org, Lesson 8: Loops with Rey and BB-8 Lesson Plan
	• Bridging Activity
	  ○ Choose 1—Unplugged Blocks (Courses C-F)or previewing online puzzles
	• Loops with BB-8
	• Wrap Up: Reflection
	  ○ What was today's lesson about?
	  ○ How did you feel during today's lesson?
	  ○ How did loops make your program easier to write?
	  ○ Think of something that repeats over and over again. What might the program for that look like?

Lesson links/resources:	• Code.org, Lesson 8: Loops with Rey and BB-8 Lesson Plan • Unplugged Blocks (Courses C-F): Manipulatives • Pair Programming: Video
CS standards addressed:	Students will be able to: • Break down a long sequence of instructions into the largest repeatable sequence • Employ a combination of sequential and looped commands to reach the end of a maze • 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 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: 60 min** • Warm Up **5 min** • Bridging Activity **10 min** • Activity: Loops with BB-8 **30 min** • Wrap Up: Reflection **5 min** • Keyboarding **10 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Code.org, Lesson 8: Loops with Rey and BB-8 Lesson Plan • Code.org account Students: • Computer/tablet with internet access • Unplugged Blocks (Courses C-F): Manipulatives • Pair Programming: Video • Code.org account
Subject integrated:	Math
Other standards addressed:	Math • **2.OA.2**—Fluently add and subtract within 20 using mental strategies. By end of Grade 2; know from memory all sums of two one-digit numbers.
Vocabulary:	Loop: The action of doing something over and over again Repeat: To do something again per the code.org lesson plan
Notes:	

Week 20: Code.org, Course C, Lesson 9: Harvesting Crops With Loops	
Lesson overview:  	**Purpose:** In this lesson, students will use loops to repeat actions like harvesting pumpkins. New patterns will emerge, and students will use creativity and logical thinking to determine what code needs to be repeated and how many times. **Lesson:** This is Part 1 of a two-part lesson. Part 2 is Harvester's Farm (Week 21) ● Warm Up: Introduction ○ Watch Goods and Services - student video. ○ Have students discuss what vegetables they like best and why some vegetables are produced more than others. ○ Follow prompts in Code.org, Lesson 9: Harvesting Crops with Loops Lesson Plan ● Activity ○ Harvesting Crops with Loops ● Wrap Up ○ What was today's lesson about? ○ How did you feel during today's lesson? ○ Give two examples of when you used loops in your code. ○ What else could a farmer harvest? Draw the code block that you would need to harvest that item.
Lesson links/resources:	Code.org, Lesson 9: Harvesting Crops with Loops Lesson Plan Goods and Services - student video
CS standards addressed:	Students will be able to: ● Employ a combination of sequential and looped commands to move and perform actions ● Identify when a loop can be used to simplify a repetitive action ● Write a program for a given task which loops a single command 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 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: 60 min**

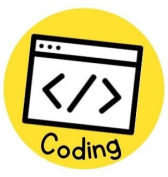
	• Warm Up: Introduction **5 min** • Harvesting Crops with Loops **30 min** • Wrap Up: Reflection (prompts provided) **10 min** • Keyboarding **15 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Code.org, Lesson 9: Harvesting Crops with Loops Lesson Plan • Code.org account Students: • Computer/tablet with internet access • Goods and Services - student video • Code.org account
Subject integrated:	Social Studies
Other standards addressed:	**E.2.1**—Explain how individual wants and needs impact the production of goods and services.
Vocabulary:	Loop: The action of doing something over and over again Repeat: To do something again
Notes:	

Week 21: Code.org, Course C, Lesson 9—Harvester's Farm

Lesson overview:  	**Purpose:** Harvester needs help describing and classifying items on her farm. Students will use the provided graphic organizers to classify and describe these items. The graphic organizers will serve as a pre-writing component as students then write an informative/explanatory text to inform readers about what the Harvester does on her farm. **Lesson:** ● Main Activity - This is Part 2 of a two-part lesson. Part 1 is Harvesting Crops with Loops (Week 20) ○ Students completed Harvesting Crops with Loops, Levels 1-12 in week 20. ○ Follow the prompts on the Harvester's Farm Lesson Plan ○ Students will open the Google Slide file titled Harvester's Farm Graphic Organizers. Each student will want to "Make a Copy" of this file so they can add their own answers to the slides. ○ On the Harvester's Farm Graphic Organizers file, students will follow the instructions provided on each slide in the "speaker notes" section. ■ **Bubble Map instructions:** Help Harvester describe the crop located in the center bubble/circle. You will write 1-2 word descriptions in each of the surrounding bubbles/circles. Remember that your descriptions could include characteristics such as color, texture, shape, etc. ■ **Tree Map instructions:** Harvester has many different things on her farm. Classify each item by its observable properties as either an animal, plant or tool. Then use your mouse to drag the picture to the correct location on the tree map. ○ Each student will write an informative/explanatory text about Harvester's farm. This can be done with paper and pencil OR digital tools such as Google docs. Their writing should include the following: ■ Introduction of the topic: Describe what the Harvester does on her farm. ■ Use facts and definitions to develop points ■ Provide a concluding statement ● Wrap Up ○ Group 3-4 students together and have them read their informative/explanatory text to each other. Students will then provide suggestions for revisions and edits which could focus and strengthen each other's writing.
Lesson links/resources:	● Harvester's Farm Lesson Plan ● Harvesting Crops with Loops, Levels 1-12 ● Harvester's Farm Graphic Organizers
CS standards addressed:	Students will be able to: ● Employ a combination of sequential and looped commands to move and perform actions ● Identify when a loop can be used to simplify a repetitive action ● Write a program for a given task which loops a single command Standards: ● **AP.1A.5**—Develop plans that describe a program's sequence of events, goals, and expected outcomes. ● **AP.1A.8**—Using correct terminology, describe steps taken and

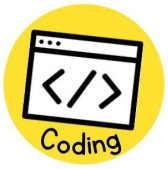
	choices made during the iterative process of program development.
Time needed:	**Total Time: 55 min** • Main Activity **30 min** • Wrap Up **10 min** • Keyboarding **15 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Harvester's Farm Lesson Plan • Code.org account Students: • Harvester's Farm Graphic Organizers (Bubble Maps and Tree Map on Google Slides) • Paper and pencil (optional)
Subject integrated:	ELA Math
Other standards addressed:	ELA • **W.2.2**—Write informative/explanatory texts in which they introduce a topic, use facts and definitions to develop points, and provide a concluding statement or section. • **W.2.5**—With guidance and support from adults and peers, focus on a topic and strengthen writing as needed by revising and editing. Math • **2.OA.2**—Fluently add and subtract within 20 using mental strategies. By the end of Grade 2, know from memory all sums of two-digit numbers.
Vocabulary:	Loop: Doing something over and over again Repeat: To do something again
Notes:	

Week 22: Code.org, Course C, Lesson 10—Mini-Project: Sticker Art	

Lesson overview:
 
 	
	Purpose:
	This series highlights the power of loops with creative and personal designs. Offered as a project-backed sequence, this progression will allow students to build on top of their own work and create amazing artifacts.
	Lesson: This is Part 1 of a two-part lesson. Part 2 is Loopy Forms and Their Functions (Week 23)
	• Warm Up
	  ○ Students should have had plenty of introduction to loops at this point. Based on what you think your class could benefit from, we recommend:
	  ■ Create a new dance with loops just like in "Getting Loopy."

	■ As a class, play through a puzzle from Lesson 9, "Loops with Rey and BB-8." ■ Review how to use Artist by playing through a puzzle from "Programming in Artist." ■ Preview a puzzle from this lesson. ○ All of these options will either review loops or the artist, which will help prepare your class for fun with the online puzzles! ○ Follow the prompts on Code.org, Lesson 10: Mini-Project: Sticker Art Lesson Plan ● Main Activity ○ Sticker Art with Loops ● Wrap Up ○ What was today's lesson about? ○ How did you feel during today's lesson? ○ What was the coolest shape or figure you programmed today? Write a paragraph stating your opinion and reasoning. Then, Draw it out! ○ What is another shape or figure you would like to program? Can you come up with the code to create it?
Lesson links/resources:	Code.org, Lesson 10: Mini-Project: Sticker Art Lesson Plan
CS standards addressed:	**Students will be able to:** ● Differentiate between commands that need to be repeated in loops and commands that should be used on their own ● 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 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: 60 min** ● Warm Up/Introduction **15 min** ● Main Activity **30 min** ● Wrap Up/Reflection **15 min**
Materials needed:	Teacher: ● Computer ● Projector/smartboard with sound ● Code.org, Lesson 10: Mini-Project: Sticker Art Lesson Plan ● Code.org account Students: ● Computer/tablet with internet access
Subject integrated:	ELA Math
Other standards addressed:	ELA ● **W.2.1**—Write opinion pieces in which they introduce the topic or book they are writing about, state an opinion, supply reasons that

	support the opinion, use linking words (e.g., because, and, also) to connect opinion and reasons, and provide a concluding statement or section. Math ● **2.G.A.1**—Recognize and draw shapes having specified attributes, such as a given number of angles or a given number of equal faces.1 Identify triangles, quadrilaterals, pentagons, hexagons, and cubes.
Vocabulary:	Loop: The action of doing something over and over again Repeat: To do something again
Notes:	

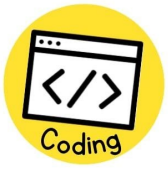
Week 23: Code.org, Course C, Lesson 10—Loopy Forms and Their Functions	
Lesson overview: 	**Purpose:** Loopy Forms and Their Functions: Artist needs to help his friends design their house by drawing blueprints. It is important that students understand basic shapes and how each shape serves different functions in the house blueprints. They will then write opinion pieces, supply reasons, and use linking words to justify their design choices. **Lesson:** This is Part 2 of a two-part lesson. Part 1 is Mini-Project: Sticker Art (Week 22) ● Introduction ○ Students completed Sticker Art with Loops, levels 1-8 in Week 22. ○ Students will draw three KWL Charts. One chart will have the topic "Triangles." The second chart will have the topic "Quadrilaterals." The third chart will have the topic "Pentagons." Students will then complete the "K - Write what you know" and "W - Write what you want to know" columns of each chart. ● Main Activity ○ Then students will complete the Loopy Shapes & Their Funtions_Student Handout, using Sticker Art with Loops, levels 9, to help the Artist design house blueprints for his friends. ● Wrap Up ○ Have students complete the "L - Write what you learned" box on the KWL Chart.
Lesson	● Loopy Forms & Their Functions Lesson Plan

links/resources:	• Sticker Art with Loops, levels 1-8 • Loopy Shapes & Their Funtions_Student Handout
CS standards addressed:	Students will be able to: • Differentiate between commands that need to be repeated in loops and commands that should be used on their own • 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 into a precise sequence of instructions. • **AP.1A.5**—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.
Time needed:	**Total Time: 60 min** • Introduction **20 min** • Main Activity **30 min** • Wrap Up **10 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Code.org, Lesson 10: Mini-Project: Sticker Art Lesson Plan • Loopy Forms & Their Functions Lesson Plan • Code.org account Students: • Computer/tablet with internet access • Loopy Shapes and Their Funtions_Student Handout
Subject integrated:	ELA Math
Other standards addressed:	ELA • **W.2.1**—Write opinion pieces in which they introduce the topic or book they are writing about, state an opinion, supply reasons that support the opinion, use linking words (e.g., because, and, also) to connect opinion and reasons, and provide a concluding statement or section. Math • **2.G.A.1**—Recognize and draw shapes having specified attributes, such as a given number of angles or a given number of equal faces. Identify triangles, quadrilaterals, pentagons, hexagons, and cubes.
Vocabulary:	Loop: The action of doing something over and over again Repeat: To do something again
Notes:	

Week 24: Code.org, Course C, Lesson 11—The Big Event	
Lesson overview: 	**Purpose:** Today, students will learn to distinguish events and 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. **Lesson:** ● Warm Up ○ A series of events ■ Follow the prompts in Code.org, Lesson 11: The Big Event Lesson Plan ● Main Activity ○ The Big Event ● Reflection ○ What did we learn? ○ What are some examples of events? ○ Assessment
Lesson links/resources:	● Code.org- Lesson 11: The Big Event Lesson Plan ● The Big Event: Assessment Answer Key ● The Big Event: Unplugged Video (Download) ● The Big Event: Assessment ● The Big Event (Course C): Event Controller
CS standards addressed:	Students will be able to: ● Practice differentiating pre-defined actions and event-driven ones ● Recognize movements of the teacher as signals to initiate commands ● Repeat commands given by an instructor 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 into a precise sequence of instructions.
Time needed:	**Total Time: 60 min** ● Warm Up **15 min** ● Main Activity **15 min** ● Wrap Up **15 min** ● Keyboarding Practice **15 min**
Materials needed:	Teacher: ● Computer ● Projector/smartboard with sound

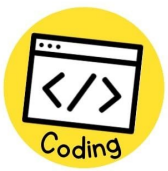
	• Code.org account • The Big Event: Assessment answer key Students: • Computer/tablet with internet access • The Big Event: Unplugged video (Download) • The Big Event : Assessment • The Big Event (Course C): Event controller
Subject integrated:	ELA
Other standards addressed:	**SL.2.1**—Participate in collaborative conversations with diverse partners about grade 2 topics and texts with peers and adults in small and larger groups.
Vocabulary:	Event: An action that causes something to happen
Notes:	

Week 25: Code.org, Course C, Lesson 12—Build a Flappy Game	
Lesson overview:	**Purpose:**

	Events are very common in computer programs. In this lesson, students will further develop their understanding of events by making a Flappy Bird game. Students will learn to make their character move across the screen, make noises, and react to obstacles based on user-initiated events.
	Lesson: This is Part 1 of a two-part lesson. Part 2 is Become a Game Designer (Week 26)
	• Warm Up: Introduction
	  ○ Review activity
	  ○ Follow the prompts on the Code.org, Lesson 12: Build A Flappy Game Lesson Plan
	• Bridging Activity
	  ○ Choose 1 (paper blocks or preview online puzzle)
	• Main Activity
	  ○ Build a flappy game
	• Wrap Up
	  ○ What was today's lesson about?
	  ○ How did you feel about today's lesson?
	  ○ What did you do to make your game unique?
	  ○ Draw out a game you want to make in the future?

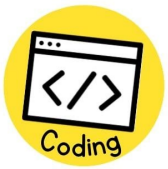
Lesson links/resources:	• Code.org, Lesson 12: Build A Flappy Game Lesson Plan • Open-ended Programming Levels: Answer Key • Unplugged Blocks (Courses C-F): Manipulatives • Pause and Think Online: Video • The Big Event (Course C): Event Controller
CS standards addressed:	Students will be able to: • Create a game using event handlers • Match blocks with the appropriate event handler • Share a creative artifact with other students 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 into a precise sequence of instructions.
Time needed:	**Total Time: 60 min** • Warm Up **10 min** • Bridging Activity **10 min** • Main Activity **30 min** • Wrap Up **10 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Code.org account • Open-ended Programming Levels: Answer Key Students: • Computer/tablet with internet access • Unplugged Blocks (Courses C-F): Manipulatives • Pause and Think Online: Video • The Big Event (Course C): Event controller • Code.org account
Subject integrated:	ELA
Other standards	**SL.2.1**—Participate in collaborative conversations with diverse partners about Grade 2 topics and texts with peers and adults in small and larger

addressed:	groups.
Vocabulary:	Event: An action that causes something to happen
Notes:	

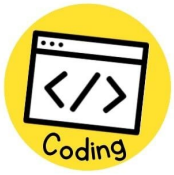
Week 26: Code.org, Course C, Lesson 12—Become a Game Designer	
Lesson overview:  	**Purpose:** Students will become game designers as they create and test a game. Students will also record scores, make observations, create a bar graph, analyze the results, and justify the changes they would make to their game. **Lesson:** This is Part 2 of a two-part lesson. Part 1 is Build a Flappy Game (Week 25) • Follow the prompts on the Code.org- Become a Game Designer Lesson Plan • Vocabulary (Day 1) • Setting the Scene (Day 1) ○ They have already completed the first 4 steps (in Week 25: Build a Flappy Game): identified the problem, brainstormed solutions, selected a design and built a model.

	○ In this activity they will complete the next 2 steps: test and evaluate and optimize the design. • Collecting Data (Day 1) • Switch Roles (Day 2) • Creating a Bar Graph (Day 2) • Analyzing Data (Day 3
Lesson links/resources:	• Code.org- Become a Game Designer Lesson Plan • Engineering Design Poster (display for students) • Data Collection and Analyze Sheet (print double sided) • Bar Graph Video (use this video. The video in the lesson plan does not work) • Create a bar graph
CS standards addressed:	Students will be able to: • Create a game using event handlers • Match blocks with the appropriate event handler • Share a creative artifact with other students 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 into a precise sequence of instructions. • **DA.1A.2**—Collect and present the same data in various visual formats.
Time needed:	**Total Time: 60 min** • Day 1 **20min** • Day 2 **20 min** • Day 3 **20 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Code.org account • Code.org, Become a Game Designer Lesson Plan • Engineering Design Poster : Display for students Students: • Computer/tablet with internet access • Code.org account • Data Collection and Analyze Sheet: Print double sided • Bar Graph Video • Create a bar graph
Subject integrated:	Math
Other standards addressed:	**2.MD.10**—Draw a picture graph and a bar graph (with single-unit scale) to represent a data set with up to four categories. Solve simple put-together, take-apart, and compare problems using information presented in a bar graph.
Vocabulary:	Game designer/programmer: Someone that creates a game Play tester: Someone that plays a game and provides feedback to the game designer Engineering design process: A series of steps that guides engineering teams as they create a solution to a problem Data collection: Gathering information to place in organized graphs or charts for analysis

	Bar graph: A graph using bars to show numbers so they can be easily compared Justify: To prove or show evidence supporting answers or decisions
Notes:	

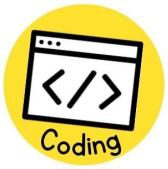
Week 27: Code.org, Course C, Lesson 13—Mini-Project: Chase Game	
Lesson overview: 	**Purpose:** This lesson combines skill-building around events with a mini-project where students get to build their own animated game. Here, students will further develop their understanding of events using Play Lab. Students will use events to make characters move around the screen, make noises, and change backgrounds based on user input. At the end of the puzzle sequence, students will be presented with the opportunity to share their projects. **Lesson:** ● Warm Up ○ Follow prompts on the Code.org Mini-Project Lesson Plan ○ Discuss Flappy Bird game and review events and other actions. ● Main Activity ○ Chase Game with Events ● Wrap Up ○ What was today's lesson about? ○ How did you feel about today's lesson? ○ What is an event your program used today? ○ Is there an event that you would have liked to use in your game that you did not get to use in Play Lab?
Lesson links/resources:	○ Code.org Mini-Project Lesson Plan ○ Open-ended Programming Levels: Answer Key ○ Unplugged Blocks (Courses C-F): Manipulatives ○ Pause and Think Online: Video

CS standards addressed:	Students will be able to: • Create an animated, interactive game using sequence and event-handlers • Identify actions that correlate to input events 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 into a precise sequence of instructions.
Time needed:	**Total Time: 60 min** • Warm Up **10 min** • Main Activity **30 min** • Wrap Up **15 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Code.org account • Open-ended Programming Levels: Answer Key Students: • Computer/tablet with internet access • Code.org account • Unplugged Blocks (Courses C-F): Manipulatives • Pause and Think Online: Video
Subject integrated:	ELA
Other standards addressed:	**SL.2.1**—Participate in collaborative conversations with diverse partners about Grade 2 topics and texts with peers and adults in small and larger groups.
Vocabulary:	Event: An action that causes something to happen
Notes:	

Week 28: Code.org, Course C, Lesson 14—Picturing Data	
Lesson overview: 	**Purpose:** Computers were created to help process data. There is an increasing amount of data in the world, so being able to read and analyze it is important. This lesson is here to make sure students have the basic experience of collecting, visualizing, and analyzing a simple set of data. **Lesson:** ● Warm Up ○ The need for visualization ○ Follow the prompts on the Code.org, Lesson 14: Picturing Data Lesson Plan ● Main Activity ○ Picturing data ● Wrap Up ○ Reflection
Lesson links/resources:	● Code.org, Lesson 14: Picturing Data Lesson Plan ● Graphing Data from Play Lab : Worksheet
CS standards addressed:	Students will be able to: ● Collect and record data about quantities of real objects, or characters on a screen ● Create a bar graph and pie chart to represent simple data ● Make comparisons between data visualizations made by others and use them to make a prediction Standards: ● **AP.1A.09**—Model the way programs store and manipulate data by using numbers or other symbols to represent information. ● **DA.1A.1**—Store, copy, search, retrieve, modify, and delete information using a computing device and define the information stored as data. ● **DA.1A.2**—Collect and present the same data in various visual formats. ● **DA.1A.3**—Identify and describe patterns in data visualizations, such as charts or graphs, to make predictions.
Time needed:	**Total Time: 60 min** ● Warm Up **5 min** ● Main Activity **35 min**

	• Wrap Up **5 min** • Keyboarding **15 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Code.org account • Code.org, Lesson 14: Picturing Data Lesson Plan Students: • Computer/tablet with internet access • Code.org account • Graphing Data from Play Lab: Worksheet
Subject integrated:	Math
Other standards addressed:	**2.MD.10**—Draw a picture graph and a bar graph (with single-unit scale) to represent a data set with up to four categories. Solve simple put-together, take-apart, and compare problems using information presented in a bar graph.
Vocabulary:	Data: A collection of information
Notes:	

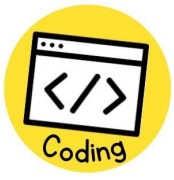
Week 29: Code.org, Course C, Lesson 15—Binary Bracelets

Lesson overview:
 
 
	Purpose:
	In this lesson students will learn how information is represented in such a way that a computer can interpret and store it. When learning binary, students will have the opportunity to write codes and share them with peers as secret messages. This can then be related back to how computers read a program, translate it to binary, use the information in some way, then reply back in a way humans can understand. For example, when we type a sentence into a document then press save, a computer translates the sentence into binary, stores the information, then posts a message indicating the document has been saved.
	Lesson:
	• Warm Up
	  ○ Follow prompts on the Code.org: Binary Bracelets Lesson Plan
	  ○ Introduce vocabulary
	  ○ Binary Bracelets: Lesson in action video
	  ○ Off and on
	  ○ Remarks
	• Main Activity
	  ○ Binary Bracelets: Worksheet
	  ○ Bits Versus Bytes: Student video
	• Wrap Up
	  ○ Flash chat:
	  ■ What else do you think is represented as binary inside of a computer?
	  ■ How else might you represent binary instead of boxes that are filled or not filled?
	  ■ What was your favorite part about that activity?
	  ○ Prompts:
	  ■ What was today's lesson about?
	  ■ How else might you represent binary instead of boxes that are filled or not filled?
	  ■ What was your favorite part about that activity?
	• Assessment
	  ○ Binary Bracelets: Assessment

Lesson links/resources:	• Code.org: Binary Bracelets Lesson Plan • Binary Bracelets: Lesson in Action Video • Binary Bracelets: Assessment Answer Key • Binary Bracelets: Unplugged Video (Download) • Binary Bracelets: Worksheet • Binary Bracelets: Assessment • Bits Versus Bytes: Student Video
CS standards addressed:	Students will be able to: • Decode binary back to letters • Encode letters into binary • Relate the idea of storing letters on paper to the idea of storing information in a computer Standards: • **AP.1A.09**—Model the way programs store and manipulate data by using numbers or other symbols to represent information.

	• **AP.1A.11**—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.
Time needed:	**Total Time: 65 min** • Warm Up **15 min** • Main Activity **20 min** • Wrap Up **5 min** • Assessment **15 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Code.org account • Code.org: Binary Bracelets Lesson Plan • Binary Bracelets: Lesson in Action Video • Binary Bracelets: Assessment answer key Students: • Computer/tablet with internet access • Code.org account • Binary Bracelets: Unplugged video (Download) • Binary Bracelets: Worksheet • Binary Bracelets: Assessment • Bits Versus Bytes: Student video
Subject integrated:	ELA
Other standards addressed:	**SL.2.6**—Produce complete sentences when appropriate to task and situation in order to provide requested detail or clarification.
Vocabulary:	Binary: A way of representing information using only two options
Notes:	

Week 30: Code.org, Course C, Lesson 16—End of Course Project	

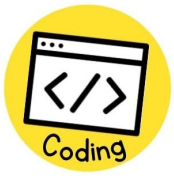
Lesson overview:
 
 	
	Purpose:
	This lesson provides students with space to create a project of their own design, using a step-by-step process that requires planning but also allows for broad creativity.
	Lesson:
	• Warm Up
	  ○ Follow prompts on the Code.org, Lesson 16: End of Course Project
	  ○ Play Lab Project Planning Guide: Worksheet
	• Main Activity

	○ Coding project ● Wrap Up ○ Showcase
Lesson links/resources:	● Code.org, Lesson 16: End of Course Project ● Play Lab Project Planning Guide (Example): Lesson Resource ● Play Lab Project Planning Guide: Worksheet
CS standards addressed:	Students will be able to: ● Overcome obstacles such as time constraints or bugs ● Use a planned design as a blueprint for creation Standards: ● **AP.1A.10**—Develop programs with sequences and simple loops, to express ideas or address a problem. ● **AP.1A.12**—Develop plans that describe a program's sequence of events, goals, and expected outcomes. ● **AP.1A.13** —Give attribution when using the ideas and creations of others while developing programs. ● **AP.1A.15**—Using correct terminology, describe steps taken and choices made during the iterative process of program development.
Time needed:	**Total Time: 60 min** ● Warm Up **10 min** ● Main Activity **25 min** ● Wrap Up **10 min** ● Keyboarding **15 min**
Materials needed:	Teacher: ● Computer ● Projector/smartboard with sound ● Code.org account ● Code.org, Lesson 16: End of Course Project ● Play Lab Project Planning Guide (Exemplar): Lesson Resource Students: ● Computer/tablet with internet access ● Code.org account ● Play Lab Project Planning Guide: Worksheet
Subject integrated:	ELA
Other standards addressed:	**SL.2.3**—Ask and answer questions about what a speaker says in order to clarify comprehension, gather additional information, or deepen understanding of a topic or issue.
Vocabulary:	Actor: A participant in an action or process
Notes:	

Week 31: Gingerbread Coding	
Lesson overview:  	**Purpose:** Gingerbread Coding (Brooke Brown, Teach Outside the Box) is a lesson where students will use block coding to create a path for the gingerbread man to escape. **Lesson:** ● Partner students. Each pair of students will need one Gingerbread Coding Mat (page 3), one set of Map Pieces (Colored on pages 4-5 or black and white on pages 6-7), and 2-4 copies of "Crack the Code!" (Page 8). You may also choose to put copies of page 8 inside clear page protectors so that students can write and wipe codes with dry erase markers multiple times. Page 9 is optional and is provided for you to project or display coding symbols. ● Have pairs of students cut out all the map pieces and color if desired. ● Student 1 arranges the map pieces on the Gingerbread Coding Map, starting with the gingerbread man or woman and ending with the Gingerbread House, with path pieces (colored squares) in between to connect them. Then he or she places two Treats and one Enemy along the path. ● Student 2 then "codes" the path of the gingerbread man or woman on page 8, using the provided symbols to draw the directions that he or she must travel. When the gingerbread man or woman comes to a treat, they will draw the symbol to "eat the treat," and when they reach an enemy, they draw the symbol to "jump over the enemy."

	(Encourage students to use cardinal directions when referring to directions.) ● Student 1 checks the code and coaches Student 2 as needed. ● Map pieces are cleared and students trade places, with Student 2 creating the map and Student 1 coding the action. ● Closing: ○ Students will write a paragraph explaining how to solve the puzzle.
Lesson links/resources:	Gingerbread Coding (you must create a free account to download)
CS standards addressed:	Students will be able to: ● Use basic coding to accomplish tasks ● Debug errors in their codes 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.
Time needed:	**70 min** ● Introduction to the activity **10 min** ● Gingerbread Coding activity **50 min** ● Closing **10 min**
Materials needed:	Teacher: ● Computer ● Projector/smartboard with sound ● Teachers Pay Teachers account Students: ● Gingerbread Coding
Subject integrated:	ELA
Other standards addressed:	**W.2.2**—Write informative/explanatory texts in which they introduce a topic, use facts and definitions to develop points, and provide a concluding statement or section.
Vocabulary:	
Notes:	→Teachers will need to create FREE teacher account for Teachers Pay Teachers

Week 32: Kodable—Pizza Party	

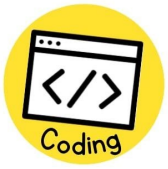
Lesson overview:
 
 	
 
 	
	Purpose:
	Students will practice creating a sequence. Students will be able to write simple numerical expressions and evaluate them in proper sequence.
	Lesson:
	• The students will build a sequence in code to create their slices of pizza. Each day consists of a 20-25 min session.
	  ○ Day 1:
	  ■ Vocabulary,
	  ■ I follow a sequence when I…,
	  ■ Exit Ticket
	  ○ Day 2:
	  ■ Brush Teeth Algorithm,
	  ■ Make a Pizza
	  ■ Exit Ticket
	  ○ Day 3:
	  ■ Inquiry Sheet
	  ■ Exit Ticket

Lesson links/resources:	• Kodable Lesson Frame • Pizza Party Worksheets • Dominic's Pizza Party • Sequence Sector lesson 1: "1, 2, 3 Roll" 1.1-1.5 Optional On-Screen Practice
CS standards addressed:	Students will be able to: • Create and follow algorithms • Break down the steps needed to solve a problem • Develop a plan to illustrate what a program will do Standards: • **AP.1A.1**—Model daily processes by creating and following algorithms. • **AP.1A.4**—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions. • **AP.1A.5**—Develop plans that describe a program's sequence of events, goals, and expected outcomes.
Time needed:	**Total Time: 55-65 min** This lesson can be broken down into 3 days as indicated in the Kodable

	lesson framework: **Day 1 (20-25 min)** ● Vocabulary cards with visuals and definitions **5-10 min** ● Sequence sentence frame graphic organizer **10 min** ● Exit ticket **5 min** **Day 2 (20 min)** ● Dominic's Pizza video **2 min** ● "Brush teeth" algorithm pseudocode visual **3 min** ● Pizza ingredients (build your own!) **10 min** ● Exit ticket: School algorithm **5 min** **Day 3 (15-20 min)** ● Inquiry sheet **5 min** ● Exit ticket: K-W-L **5 min** ● Kodable on-screen practice **5-10 min**
Materials needed:	For the students: ● Pizza Party Worksheets ● Vocabulary cards with visuals and definitions ● Sequence sentence frame graphic organizer ● Exit ticket: Emoji ● Dominic's Pizza video ● "Brush teeth" algorithm pseudocode visual ● Pizza algorithm pseudocode graphic organizer ● Pizza ingredients (build your own!) ● Exit ticket: School algorithm ● Inquiry sheet ● Exit ticket: K-W-L ● Optional: Kodable on-screen practice
Subject integrated:	ELA Math
Other standards addressed:	ELA ● **SL.2.1**—Participate in collaborative conversations with diverse partners about Grade 2 topics and texts with peers and adults in small and larger groups. ● **SL.2.2**—Recount or describe key ideas or details from a text read aloud or information presented orally or through other media. ● **SL.2.3**—Ask and answer questions about what a speaker says in order to clarify comprehension, gather additional information, or deepen understanding of a topic or issue. Math ● **2.MD.8a**— Solve word problems involving dollar bills, quarters, dimes, nickels, and pennies, using $ and ¢ symbols appropriately. *Example: If you have 2 dimes and 3 pennies, how many cents do you have?*
Vocabulary:	Programmer: A person who writes code and communicates instructions to a computer Program: A sequence of instructions given to a computer in code that a computer can understand. The computer follows the instructions and carries out the task. Language: A way of communicating ideas or feelings through sounds, symbols, signs, or words. There are thousands of languages in the world. Communication: 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 Command: A specific instruction given to a computer in written code from a programmer

	Sequence: Instructions given to the computer to be followed in the exact order they are written in code using commands from programmers Algorithm: A sequence of steps followed in order to finish a task (can be performed with or without a computer) Bugs: Mistakes in the program's code that cause the program to function in a way that was not intended Debugging: The process of finding and fixing mistakes in the program so it will run properly
Notes:	This lesson is developed to take place over the course of 3 days (20-25 min lesson each.) You can modify it to take place on one day if needed. →Teachers will need to create a FREE teacher account for Kodable.

Week 33: Kodable—Magnificent Maze Maker	
Lesson overview:	**Purpose:** Can you build a symmetrical maze? A maze with right angles? Get creative and complete maze-building challenges with basic coding concepts.

	Students will be able to create solvable mazes while applying grade-level geometry concepts. Students can create a maze from one place to another, such as mapping from the classroom to the library.
	Lesson:
	• Introduction
	  ○ Follow prompts on the Kodable Magnificent Maze Maker Lesson Plan (see Notes below)
	  ○ Model how to create maze by showing Make Your Own Kodable Maze
	  ○ Review maze maker tips
	• Maze Maker Challenge
	• Creative Challenge (Level 6)
	  ○ Let students spend time on this level being creative! They must create and solve their puzzle before they can swap with another student.
	• Share
	  ○ Let students swap seats (or devices) with other students to attempt each other's puzzles.

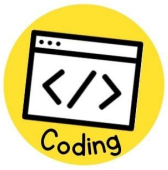
Lesson links/resources:	• Kodable Magnificent Maze Maker Lesson Plan • Reproducible • Make Your Own Kodable Maze • Kodable.com Magnificent Maze maker
CS standards addressed:	Students will be able to: • Students will be able to create solvable mazes while applying grade-level geometry concepts Standards: • **AP.1A.1**—Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.
Time needed:	**Total Time: 50-60 min** • Introduction **10 min** • Maze Maker Challenge **10-20 min** • Creative Challenge **15 min** • Share **15 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Kodable Magnificent Maze Maker Lesson Plan • Make Your Own Kodable Maze Students: • Computer/tablet with internet access • Kodable.com Magnificent Maze maker • Reproducible
Subject integrated:	Math
Other standards addressed:	**2.G.2**—Partition a rectangle into rows and columns of same-size squares and count to find the total number of them.
Vocabulary:	
Notes:	→Without having a Kodable account, students have to work through the first levels in order to get to the maze maker. The maze maker is no longer on the hour of code page.

Week 34: Kodable—Magnificent Maze Maker Extended and Unplugged	

Lesson overview:
 
 	
	Purpose:
	Students will use their knowledge of coding to create a symmetrical marble maze using Legos and Lego base plates
	Students can create a maze from one place to another such as mapping from the classroom to the library.
	Lesson:
	● Introduction
	  ● Review the Kodable Magnificent Maze Maker Lesson Plan from the previous week.
	  ○ Introduce this week's lesson as the unplugged option.
	  ○ Show the How to make a Lego maze video
	● Magnificent Maze Maker Extended (Unplugged)

	○ Students will each need a Lego plate, Legos, and marbles for this activity. ○ Students can work alone, or they can work in pairs. ○ Students should sketch a map of the path they would take from the classroom to a nearby location (ex: restroom, office, gym, lunchroom, library, etc.) ○ Once they sketch the map, students should create that map, using Legos, on the Lego plate. ○ Then, students should write the code a marble would need to take to reach the destination (just like in the Kodable lesson from last week.) ● Wrap Up ○ What did you like most about today's lesson? ○ What did you like least about today's lesson? ○ What did you learn today during this lesson?
Lesson links/resources:	How to make a Lego maze video
CS standards addressed:	Students will be able to: ● Students will be able to create solvable mazes while applying grade-level social studies concepts Standards: ● **AP.1A.1**—Model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.
Time needed:	**Total Time: 60 min** ● Introduction **5 min** ● Magnificent Maze Maker Extended **40 min** ● Wrap Up **5 min**
Materials needed:	Teachers: ● How to make a Lego maze video Students: ● Lego plates ● Legos ● Marbles ● Paper ● Pencil/pen
Subject integrated:	Math
Other standards addressed:	Math ● **2.G.2**—Partition a rectangle into rows and columns of same-size squares and count to find the total number of them.
Vocabulary:	
Notes:	

Week 35: Kodable—Introuduction to Coding and Coding Basics Unplugged

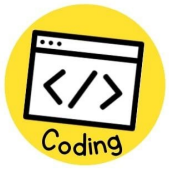
Lesson overview:  	**Purpose:** Students will review basic programming skills and practice using core coding commands. **Lesson:** ● Unplugged Activity ○ Write algorithm for brushing teeth (Day 1). ○ Practice the sequence to look for bugs and debug, then compare the strategies and talk about successes (Day 2). ● Keyboarding ○ Choose an online or unplugged keyboard practice game. ● Introduce the definition of code (#4 under "Introduction to Hour of Code" section of lesson). ○ What Are Computer Programs? video ● Unplugged Activity ○ Kodable: Coding Basics: Unplugged lesson plan ○ Sequence Solver 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?"

Lesson links/resources:	• Kodable: Coding Basics: Unplugged • Sequence Solver Worksheet Packet • What Are Computer Programs? video
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 Student Objectives: • **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**
	• Keyboarding **15 min**
	• Kodable Lesson (**35 min**)
	  ○ Introduction to Coding ***5 min***
	  ○ Unplugged Activity: **20 min**
	  ○ Wrap Up **10 min**

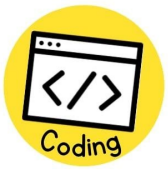
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Kodable account • Kodable, Coding Basics: Unplugged Students: • Computer/Tablet with internet access • Kodable account • Sequence Solver Worksheet Packet • What Are Computer Programs? video • Pen/pencil
Subject integrated:	ELA
Other standards addressed:	• **RL.2.1**—Ask and answer such questions as who, what, where, when, why, and how to demonstrate understanding of key details in a text. • **SL.2.1**—Participate in collaborative conversations with diverse partners about Grade 2 topics and texts with peers and adults in small and larger groups.
Vocabulary:	Code: The programming language that humans create and use to tell computers what to do Programmer: A person who writes the code (language) that tells the computer what to do Sequence: An order of events Condition: An exception to a rule. Conditional statements are "If, then" statements: If a condition is true, then something happens. Loop: A command used to repeat a portion of code Function: A set of steps given a simple name that a programmer can easily

	call on and reuse again in a program Variable: A container that stores a value that can change
Notes:	→Teachers will need to create FREE teacher and/or student accounts (when applicable) at https://www.kodable.com/

Week 36: Kodable—Build Your Own Fuzz!	
Lesson overview:  	**Purpose:** What are some character traits you can identify in books or people you know? Design a fuzz and then modify properties to build it. **Lesson:** ● Introduction ○ Follow prompts on the Kodable: Build Your Own Fuzz Lesson Plan ○ Review characters and character traits. (Give examples from books, movies, games.) ○ Explain that today we will focus on game characters. ○ Show Meet blueFuzz! Video ● Main Activity ○ Build Your Own Fuzz ■ Use a graphic organizer to design fuzz, then complete Build Your Own Fuzz! On Kodable. ● Wrap Up ○ Let students share their fuzz with classmates, allowing students to describe character traits.
Lesson links/resources:	○ Kodable: Build Your Own Fuzz Lesson Plan ○ Meet blueFuzz! Video ○ Build Your Own Fuzz! Student Worksheet
CS standards addressed:	Students will be able to: ● Modify JavaScript properties to customize a fuzz character Standards: ● **CS.1A.1**—Select and operate appropriate software to perform a variety of tasks and recognize that users have different needs and preferences for technology they use.
Time needed:	**Total Time: 50 min** ● Introduction **10 min** ● Main Activity **30 min** ● Wrap Up **10 min**
Materials needed:	Teacher: ● Computer ● Projector/smartboard with sound ● Kodable account ● Kodable: Build Your Own Fuzz Lesson Plan ● Meet blueFuzz! Video Students: ● Computer/tablet with internet access

	• Kodable account • Build Your Own Fuzz! Student Worksheet
Subject integrated:	ELA
Other standards addressed:	**RL.2.7**—Use information gained from the illustrations and words in a print or digital text to demonstrate understanding of its characters, setting, or plot.
Vocabulary:	Variable: A container that stores a value that can change Value: Information stored in a variable. Values can be written as text (strings) or numbers (integers). Properties: Features that describe an object
Notes:	→Many of the Kodable options for customizing the fuzz require a pro membership.

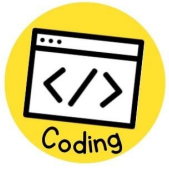
Week 37: Kodable—If Flash, Then Clap!

Lesson overview:
 
 
	Purpose:
	Students will explore conditional statements using critical thinking skills and incorporating science.
	Lesson:
	• Direct Instruction
	  ○ Follow prompts on If Flash, then Clap! Lesson Plan
	• If Lightning, then Thunder!
	  ○ A storm is coming! Students examine the logic behind thunder and lightning and relate this condition to programming.

Lesson links/resources:	• If Flash, then Clap! Lesson Plan • Lesson Resources • IF Statements Video • What Causes Thunder and Lightning Video
CS standards addressed:	Student will be able to: • Students will be able to determine the effect of a condition being true • Students will be able to connect real-world conditions with "if statements" in programming • Students will be able to create "if statements" to describe real world cause and effects Standards: • **AP.1A.4**—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.
Time needed:	**Total Time: 50 min** • Direct Instruction **15 min** • If Lightning, then Thunder! **35 min** • Optional On-screen Practice **10 min**
Materials needed:	Teacher: • Computer • Projector/smartboard with sound • Kodable account • If Flash, then Clap! Lesson Plan • Lesson Resources Students: • Computer/tablet with internet access • Kodable account • If Statements Video • What Causes Thunder and Lightning Video • Decision tree graphic organizer • Vocabulary cards • 1 Fluorescent light bulb • 1 Rubber balloon • Brown paper bags for each student • Exit ticket: Weather If statements
Subject integrated:	Math
Other standards addressed:	**2.OA.2**—Fluently add and subtract within 20 using mental strategies. By the end of Grade 2, know from memory all sums of two one-digit numbers.

Vocabulary:	Sequence: Instructions given to the computer to be followed in the exact order they are written. Written in code using commands from programmers. Condition: Allows the program to perform different actions, depending on the condition being true or false Program: A sequence of instructions given to a computer in code that a computer can understand. The computer follows the instructions and carries out the task. Programmer: A person who writes code and communicates instructions to a computer Code: The language written by humans that gives instructions to a computer Command: A specific instruction given to a computer in written code from a programmer If statement: A logic statement used in programming. Allows a computer program to act differently each time it is executed, depending on if an input is evaluated to be either true or false.
Notes:	→Without having a Kodable account, students have to work through the first levels in order to get to the maze maker. The maze maker is no longer on the hour of code page.

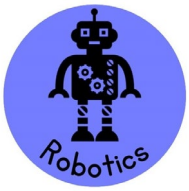
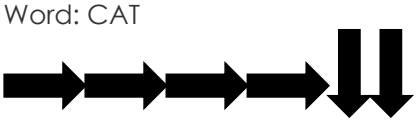
Week 38: Kodable—Beach Clean Up

Lesson overview: 	**Purpose:** 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. **Lesson:** ● Introduction ○ Students will watch the Tommy the SudBudz Turtle - Listen Up Yo! Keep the Oceans Clean :) video. ● Brainstorm ○ The teacher will talk about types of pollution, and students will fill out the "Brainstorm" section of the Lesson Worksheet. ● Beach Cleanup

	○ The students will complete the Beach Cleanup coding activity. ○ Note: Students will need to create the path all the way across the screen in order to be able to play their game. ● Wrap Up ○ Students will complete the wrap up portion of the Lesson Worksheet.
Lesson links/resources:	● Kodable Lesson Frame ● Beach Cleanup ● Lesson Worksheet ● Tommy the SudBudz Turtle - Listen Up Yo! Keep the Oceans Clean :)
CS standards addressed:	Students will be able to: ● Students will be able to design and create mazes based on preexisting obstacles ● Students will be able to write simple programs to solve mazes using basic coding concepts ● Students will be able to examine ways technology can be used to solve real-world problems ● Students will be able to collaborate and communicate effectively with peers 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. ● **AP.1A.7**—Debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.
Time needed:	**Total Time: 46-60 min** ● Intro Video **6 min** ● Group Brainstorm **15 min** ● Coding Activity **15 min** ● Wrap Up **10 min**
Materials needed:	Teacher: ● Computer ● Projector/smartboard with sound ● Kodable account ● Kodable Lesson Frame ● Tommy the SudBudz Turtle - Listen Up Yo! Keep the Oceans Clean :) Students: ● Computer/tablet with internet access ● Kodable account ● Beach Cleanup ● Lesson Worksheet ● Pens/pencils
Subject integrated:	ELA Science
Other standards addressed:	ELA ● **SL.2.1**—Participate in collaborative conversations with diverse partners about Grade 2 topics and texts with peers and adults in small and larger groups. ● **SL.2.2**—Recount or describe key ideas or details from a text read aloud or information presented orally or through other media.

	• **SL.2.3**—Ask and answer questions about what a speaker says in order to clarify comprehension, gather additional information, or deepen understanding of a topic or issue. Science • **E.2.10.1**—Use informational text, other media, and first-hand observations to investigate, analyze and compare the properties of Earth materials (including rocks, soils, sand, and water.)
Vocabulary:	Program: Step-by-step instructions written in programming language (code) that a computer can read and follow Sequence: An order of events executed by a computer exactly as it is written Condition: 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. Loop: A command used to repeat a portion of code until a desired process is complete Function: A set of steps given a simple name that a programmer can easily call on and re-use again in a program Debug: The process of finding and fixing bugs (mistakes) in code so that the computer program will run as expected
Notes:	

Week 39: Spelling Robotics

Lesson overview:
 
 
	Purpose:
	In this lesson, students will spell a word correctly in order to gather cheese. After a student spells the given word correctly, he/she will write a code to gather cheese. The student that gathers three pieces of cheese is the winner.
	Lesson:
	• Introduction
	  ○ Explain to students the function of the codable robot.
	  ○ Discuss the purpose of the lesson and the objective.
	• Setup
	  ○ Students can participate individually or in groups.
	  ○ Teachers will write individual letters of the alphabet on index cards.
	  ○ The teacher will create a grid for the robot to travel.
	  ■ The teacher can use floor tiles as the grid or tape off a particular area.
	  ○ Teacher will designate the top right square of the grid as the start block.
	  ○ Teachers will place the letters of spelling 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 white board 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 in order 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
	 
	
	• 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:	• Codable Robot (Some examples include: Code and Go Mouse, Botley, Dash) • coding cards
CS standards addressed:	The students will be able to: • Code a robot correctly to reach a destination

START		C		A
	K		O	
I				T

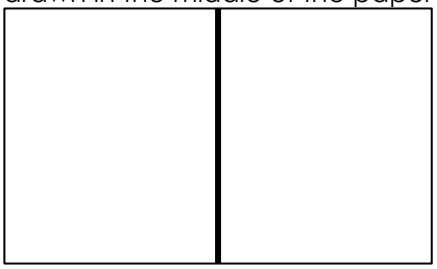
	Standards: • **AP.1A.4**—Decompose (break down) the steps needed to solve a problem into a precise sequence of instructions. • **AP.1A.5**—Develop plans that describe a program's sequence of events, goals, and expected outcomes.
Time needed:	**Total Time: 60 min** • Review **10 min** • Main Activity **40 min** • Exit Ticket **10 min**
Materials needed:	Teachers: • Alphabet cards (each card should have individual letters-teacher will create) • 175 Second Grade Spelling Words For the students: • Robot (Some examples include: Code and Go Mouse, Botley, Dash) • Coding cards
Subject integrated:	ELA
Other standards addressed:	**RF.2.3**—Know and apply grade-level phonics and word analysis skills in decoding words.
Vocabulary:	Algorithm: A list of steps to finish a task Bug: Part of a program that does not work correctly Debugging: Finding and fixing problems in an algorithm or program Sequencing: Putting commands in correct order to understand commands
Notes:	

Week 40: Marching Orders	

Lesson overview:
 
	Purpose:
	Students sit back-to-back. One student gives verbal directions while the other student follows the given directions.
	Lesson:
	• Teacher Preparation
	  ○ Watch Unplugged Coding Activity Video to better understand how the activity will take place.
	• Introduction
	  ○ Provide students with a copy of shapes that your students should know.
	  ■ Ex: Quadrilateral/Non-Quadrilateral Shapes
	  ○ Show the Learning Shapes 2nd Grade Video

<table>
<tr>
<td></td>
<td>

Main Activity

Students will be paired up, and they will sit back-to-back.
One student will have a copy of shapes, while the other has a pen and pencil and a piece of copy paper that has a line drawn in the middle of the paper as pictured below.
 
The student with the shapes should give directions to the student with pencil and paper on how to draw the shape step by step.

The student drawing should start on the left side of the paper for the first attempt.

The students should not identify the shape, tell how many sides it has, give any hints to what the shape is, etc.
Watch the video under teacher preparation to better understand what is expected.

Ex: Triangle

Students giving directions may say, “Start in the middle and draw a line across, then draw a line up to the left, then draw a line down.

The first drawing may look nothing like a triangle, but that is okay!
Have the student giving directions look at the shape that was drawn to determine if it is correct. (More than likely, it will not be correct.)

The student giving directions should think of more specific instructions to provide the artist with more detail and try the activity again on the right side of the paper.

The goal of this activity is for students to learn how to give step by step directions (algorithm), find bugs, and debug a program.
Have students swap roles, and the new artist will attempt to follow the directions given by the other student.
Repeat several times.

Wrap Up:

What was the most difficult part of this activity?
What was your favorite part of this activity?
What did you learn from this activity?

</td>
</tr>
<tr>
<td>Lesson links/resources:</td>
<td>

Unplugged Coding Activity Video
Learning Shapes 2nd Grade Video
Quadrilateral/Non-Quadrilateral Shapes

</td>
</tr>
<tr>
<td>CS standards addressed:</td>
<td>Students will be able to:

Give step by step directions (algorithm), find bugs, and debug a program

Standards:</td>
</tr>
</table>

	• **AP.1A.3**—Develop programs with sequences and simple loops to express ideas or address a problem.
Time needed:	**Total Time: 55 min** • Introduction **5 min** • Main Activity **40 min** • Wrap Up **10 min**
Materials needed:	Teachers: • Computer • Projector/smartboard with sound • Unplugged Coding Activity Video • Learning Shapes 2nd Grade Video Students: • Pen/pencil • Copy paper (with line drawn in middle) • Quadrilateral/Non-Quadrilateral Shapes (or other resource with shapes)
Subject integrated:	Math
Other standards addressed:	**2.G.1**—Recognize and draw shapes having specified attributes, such as a given number of angles or a given number of equal faces. Identify triangles, quadrilaterals, pentagons, hexagons, and cubes.
Vocabulary:	Algorithm: A list of steps to finish a task Sequencing: Putting commands in correct order to understand commands
Notes:	