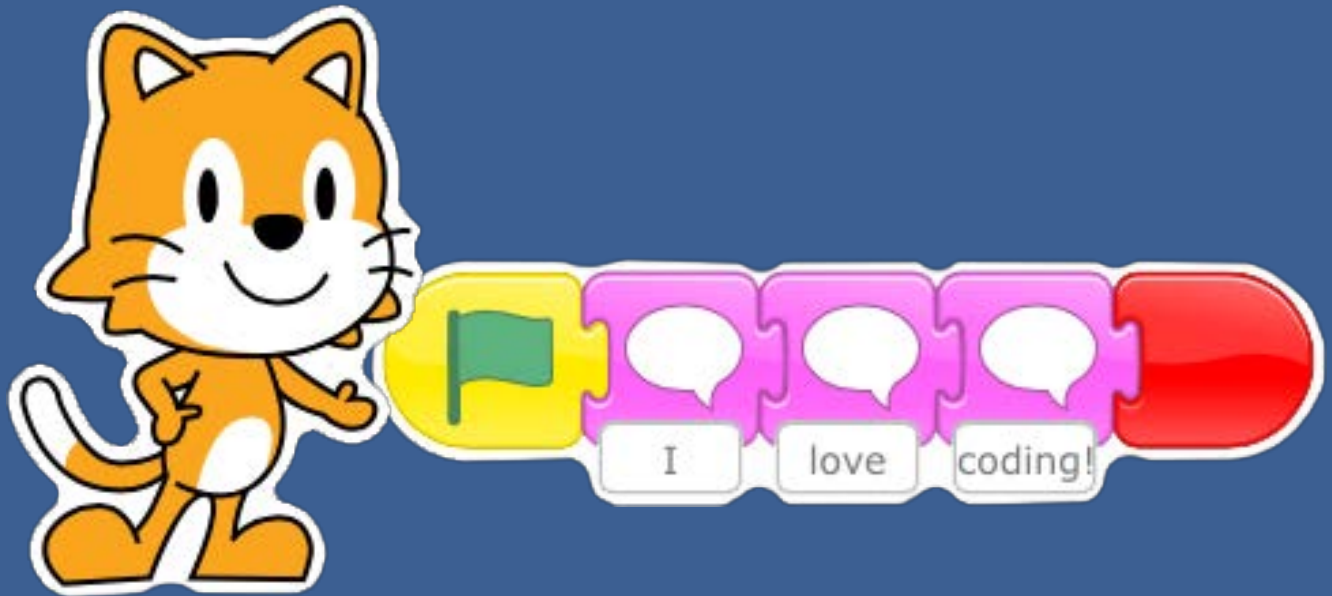


A ScratchJr Coding Curriculum for Emergent Readers

Integrated with Foundational Literacy Topics



Using the ScratchJr application and Coding as Literacy (CAL) approach developed by

DevTech Research Group

Eliot-Pearson Dept. of Child Study and Human Development
Tufts University

by the [DevTech Research Group](#) is licensed under
a [Creative Commons Attribution NonCommercial-ShareAlike 3.0 Unported License](#).

Under this license, you may use and adapt this work but you must attribute the work to the DevTech Research Group.
You **may not** use or adapt this work for commercial purposes.

© 2018, DevTech Research Group, Tufts University.

Tufts
UNIVERSITY



Table of Contents

INTRODUCTION

Coding as Literacy (CAL) Approach

Pacing

Materials

Pedagogical Framework: Positive Technological Development and Dialogic Instruction

Classroom Management

Alignment of Academic Framework

LESSONS

Lesson 1: Foundations

Lesson 2: What is a Program?

Lesson 3: Taking Care of Our Materials

Lesson 4: Sequencing

Lesson 5: Characters

Lesson 6: Programmer and Author

Lesson 7: Programming

Lesson 8: Debugging

Lesson 9: Details

Lesson 10: Repeat Loops

Lesson 11: Final Project - Planning Story Time

Lesson 12: Final Project - Coding Story Time

APPENDICES & REFERENCES

Appendix A: Materials

Appendix B-1: Solve-It Assessment A

Appendix B-2: Solve-It Assessment B

Appendix B-3: Solve-It Assessment C

Appendix B-4: Solve-It Assessment D

Appendix C: Design Journal

References

Introduction

CODING AS LITERACY (CAL) APPROACH

This curriculum introduces powerful ideas from computer science, specifically programming with ScratchJr to kindergarten children in a structured, developmentally appropriate way. The **Coding as Literacy (CAL)** approach, developed by Prof. Marina Umaschi Bers and members of her DevTech Research Group at Tufts University, puts computer science ideas into direct conversation with powerful ideas from literacy. The starting assumption of the CAL curriculum is that both computer science and literacy can enhance one another. Instruction in both can be leveraged in service of the other. Both can support learners in developing new ways of thinking about themselves and the world.

Thinking involves the ability to make sense of, interpret, represent, model, predict, and invent our experiences in the world. Thus, as educators, we must give children one of the most powerful tools for thinking: language. The term **language** refers here to a system of communication, natural or artificial, composed of a formal limited system of signs, governed by syntactic and grammatical combinatory rules, that serves to communicate meaning by encoding and decoding information. Today, we have the opportunity to not only teach children how to think by using natural languages, such as English, but also by learning artificial languages—programming languages such as the one used in the ScratchJr app.

The achievement of literacy in a natural language involves a progression of skills beginning with the ability to understand spoken words, followed by the capacity to code and decode written words, and culminating in the deep understanding, interpretation, and production of text. The ultimate goal of literacy is not only for children to master the syntax and grammar, the orthography and morphology, but also the semantics and pragmatics, the meanings and uses of words, sentences and genres. A literate person knows that reading and writing are tools for meaning making and, ultimately, tools of power because they support new ways of thinking.

The CAL approach proposes that programming, as a literacy of the 21st century, engages new ways of thinking and new ways of communicating and expressing ideas, as well as new ways of problem solving and working with others. CAL understands the process of coding as a semiotic act, a meaning making activity that engages children in both developing computational thinking, as well as promoting personal expression, communication, and interpretation. This understanding shapes this curriculum and our strategies for teaching coding.

The curriculum is organized around **powerful ideas** from both computer science and literacy. The term powerful idea refers to a central concept or skills within a discipline that is simultaneously personally useful, inherently interconnected with other disciplines, and has roots in intuitive knowledge that a child has internalized over a long period of time. The **powerful ideas from computer science** addressed in this curriculum include: algorithms, design process, representation, debugging, control structures, modularity, and hardware/software. The **powerful ideas from literacy** that will be placed in conversation with these powerful ideas from computer science are: the writing process, recalling, summarizing and sequencing, using illustrative and descriptive language, recognizing literary devices such as repetition and foreshadowing, and using reading strategies such as predicting, summarizing, and evaluating.

The CAL approach allows students to make connections between coding and literacy and use the two platforms to express their thoughts and ideas. These powerful ideas of literacy and computer science are explored in the context of a curriculum that draws on the well-known children's book *Knuffle Bunny* by Mo Willems.

Each lesson contains a variety of activities to introduce children to programming and literacy skills and concepts. Lessons are aligned to academic frameworks of Common Core and K-12 computer science standards. Teachers are encouraged to use this curriculum as a guiding resource and to adapt lessons and activities to their needs of their students. Activities in this curriculum include:

- Warm up games to playfully introduce or reinforce concepts
- Design challenges to introduce the powerful ideas from computer science
- Writing activities to introduce the powerful ideas from literacy
- Work individually or in pairs on designing and creating projects
- Technology circles to share and reflect on activities
- Free-explorations to allow students to tinker and expand their skills

The culmination of the unit is an open-ended project to share with family and friends. Just as young children can read age-appropriate books, computer programming can be made accessible by providing young children with appropriate tools such as ScratchJr.

PACING

This 12-hour curriculum unit is designed to take place over the course of a few months with one or two sessions per week (i.e. 1-2 hours each week for 2-3 consecutive months). This curriculum provides suggested time allotments, but they should be adapted to suit the needs of each classroom.

To supplement the structured challenges, free-exploration is allotted throughout the curriculum. These open-ended sessions are vital for children to fully understand the complex ideas behind their robotic creations and programs. The free-exploration sessions also serve as a time for teachers to observe students' progress and understandings. These sessions are as important for learning as the lessons themselves! In planning and adjusting the timeframe of this curriculum, free-exploration sessions should not be left by the wayside. Free-exploration provides opportunities for playing with materials and ideas. This will help build a solid foundation.

Lesson	Activities
Lesson 1: Foundations	<ul style="list-style-type: none"> • What is a Programmer? (10 min) • The Design Process (10 min) • Programmers and Writers (10 min) • Think Like a Programmer (10 min) • How-to-Books (20 min)
Lesson 2: What is a Program?	<ul style="list-style-type: none"> • What is a Program? (5 min) • Tools of Communication (20 min) • Programmer Says (20 min) • Meet the Scratch Jr App (15 min)
Lesson 3: Taking Care of Our Materials	<ul style="list-style-type: none"> • How to Treat Our Materials (15 min) • Conventions of Tablet Usage (15 min) • Procedure Practice (15 min) • Left to Right on Paper and Scratch Jr (15 min)

Lesson 4: Sequencing	<ul style="list-style-type: none"> • Knuffle Bunny (10 min) • Order Matters (20 min) • Program the Teacher with ScratchJr Blocks (15 min) • Solve-It Assessment A (15 min)
Lesson 5: Characters	<ul style="list-style-type: none"> • Design A Character (15 min) • Create Your Character in ScratchJr (15 min) • Free Play with Purple Blocks (10 min) • ScratchJr Sound (10 min) • Character Share (10 min)
Lesson 6: Programmer and Author	<ul style="list-style-type: none"> • Introduction (10 min) • Change the Setting in ScratchJr (5 min) • Add Page in ScratchJr (10 min) • Be a Programmer (20 min) • Solve-It Assessment B (15 min)
Lesson 7: Programming	<ul style="list-style-type: none"> • Dance the Hokey-Pokey (10 min) • Program the Hokey-Pokey (25 min) • Hokey-Pokey Reflection (10 min) • Share Creations (15 min)
Lesson 8: Debugging	<ul style="list-style-type: none"> • Why is Kitten Confused? (15 min) • Speed Block (5 min) • Free Play (15 min) • Debugging Reflection (10 min) • Solve-It Assessment C (15 min)
Lesson 9: Details	<ul style="list-style-type: none"> • Introduction (5 min) • Fast or Slow? (10 min) • Freeze Dance (15 min) • Wait Time Block (5 min) • Freeze Dance Program (25 min)
Lesson 10: Repeat Loops	<ul style="list-style-type: none"> • Repetition in Instructions (5 min) • Toothbrush Exercise (15 min) • ScratchJr Repeat with Numbers (25 min) • Solve-It Assessment D (15 min)
Lesson 11: Final Project - Planning Story Time	<ul style="list-style-type: none"> • Knuffle Bunny (10 min) • Story Time Introduction (10 min) • Story Map Worksheet (20 min) • Characters and Setting (20 min)
Lesson 12: Final Project - Coding Story Time	<ul style="list-style-type: none"> • Coding Story Time (30 min) • Presentation Mode (5 min) • Peer Feedback (10 min) • Share Creations (15 min)

Table 1: Pacing Guide

MATERIALS

This curriculum is based on ScratchJr, so the main material necessary for the students is iPads, Androids or Chromebooks (check here <https://www.scratchjr.org/about/faq> for devices compatible with ScratchJr), so children are able to code. In addition, there are ScratchJr block pages that can be printed to help with student comprehension. More information is provided in lessons that use these pages. This curriculum also uses the book *Knuffle Bunny* by Mo Willems.

Other materials used in the curriculum are inexpensive crafts and recycled materials. The use of crafts and recycled materials, a practice already common in other domains of early childhood education, provides opportunities for children to use materials they are already comfortable with.

PEDAGOGICAL FRAMEWORK: POSITIVE TECHNOLOGICAL DEVELOPMENT and DIALOGIC INSTRUCTION

The theoretical foundation of this curriculum, called **Positive Technological Development** (PTD), was developed by Prof. Marina Umaschi Bers and can be found in her books: *Blocks to Robotics: Learning with Technology in the Early Childhood Classroom* (Bers, 2008), *Designing Digital Experiences for Positive Youth Development: From Playpen to Playground* (Bers, 2012), and *Coding as a Playground: Programming and Computational Thinking in the Early Childhood Classroom* (Bers, 2018). More information is included in the References section at the end of this curriculum.



The PTD framework guides the development, implementation and evaluation of educational programs that use new technologies to promote learning as an aspect of positive youth development. The PTD framework is a natural extension of the computer literacy and the technological fluency movements that have influenced the world of education but adds psychosocial and ethical components to the cognitive ones. From a theoretical perspective, PTD is an interdisciplinary approach that integrates ideas from the fields of computer-mediated communication, computer-

supported collaborative learning, and the Constructionist theory of learning developed by Seymour Papert (1993) and views them in light of research in applied development science and positive youth development.

As a theoretical framework, PTD proposes six positive behaviors (six C's) that should be supported by educational programs that use new educational technologies, such as the ScratchJr app. These are: **content creation, creativity, communication, collaboration, community building, and choices of conduct**. The six C's of PTD are highlighted in the activities throughout the curriculum with their respective icons:

CONTENT CREATION The engineering design process of building and the computational thinking involved in programming foster competence in computer literacy and technological fluency. The use of Design Journals document for the children themselves, as well as for teachers and parents, their own thinking, their learning trajectories and the project's evolution over time.



CREATIVITY by making and programming personally meaningful projects, problem solving in creative playful ways and integrating different media such as robotics, motors, sensors, recyclable materials, arts and crafts, and a tangible programming language. Final ScratchJr projects that represent a theme found in the overall early childhood curriculum are a wonderful way to engage children in the creative process of learning.



COLLABORATION by engaging children in a learning environment that promotes working in teams, sharing resources and caring about each other while working with their ScratchJr programs. Collaboration is defined here as getting or giving help with a project, programming together, lending or borrowing materials, or working together on a common task. While working on their final ScratchJr projects, children create a collaboration web: a tool used to foster collaboration and support. Each child receives a printout with their photograph in the center of the page and the names and photographs of all the other children in the class arranged in a circle surrounding the central photo (see Appendix D for an example). Throughout the activity, with the teacher's prompting, each child draws a line from their own photo to the photos of the other children with whom they have collaborated. Children then write or draw "thank you cards" to the children with whom they have collaborated the most.



COMMUNICATION through mechanisms that promote a sense of connection between peers or with adults. For example, technology circles, when children stop their work, put their projects on the table or floor, and share their learning process. Technology circles present a good opportunity for problem solving as a community. Some teachers invite all the children to sit together in the rug area for this. It can also be helpful to make a "Program Parking Lot" for all the tablets to go while they are not being worked on, so children have empty hands and can focus at the technology circles. Each classroom will have its own routines and expectations around group discussions and circle times, so teachers are encouraged to adapt what already works in their class for the technology circles in this curriculum.



COMMUNITY BUILDING through scaffolded opportunities to form a learning community that promotes contribution of ideas. Final projects done by children are shared with the community via an open house, demo day, or exhibition. These open houses provide authentic opportunities for children to share and celebrate the process and tangible products of their learning with family and friends. Each child is given the opportunity not only to run their program, but to play the role of teacher as they explain to their family how they built, programmed, and worked through problems.



CHOICES OF CONDUCT which provide children with the opportunity to experiment with “what if” questions and potential consequences, and to provoke examination of values and exploration of character traits while working with technology. As a program developed following the PTD approach, the focus on learning about coding is as important as helping children develop an inner compass to guide their actions in a just and responsible way.



In alignment with the Positive Technological Development (PTD) framework, this curriculum approaches literacy from the perspective of dialogic instruction. **Dialogic instruction** is a theory of learning (and teaching) premised on the belief that students engage with literacy instruction best when there are opportunities for them to engage in authentic, open-ended interpretation of texts. If a student does not have a voice, a position, or an evaluation of the text, then what good are literary skills? Only when she needs these tools for her own purpose, to help her achieve her own interpretation, and to convince others of it, will she have a reason and motivation (beyond getting a good grade) to acquire the tools being taught. This curriculum, in adherence with the theory of dialogic instruction, strives to place the student in the position of interpreter, with opportunities for authentic, open-ended interpretation of texts. This aligns with the curriculum’s approach to coding where students are given opportunities for open-ended coding tasks that encourage them to explore their own expressive ideas.

CLASSROOM MANAGEMENT

Teaching programming in an early childhood setting requires careful planning and ongoing adjustments when it comes to classroom management issues. These issues are not new to the early childhood teacher, but they may play out differently during iPad activities because of the novelty of the materials themselves. Issues and solutions other than those described here may arise from classroom to classroom; teachers should find what works in their particular circumstances. In general, provide and teach a clear structure and set of expectations for using materials and for the routines of each part of the lessons (technology circles, clean up time, etc.). Make sure the students understand the goal(s) of each activity. Posters and visual aids can facilitate children’s attempts to answer their own questions and recall new information.

GROUP SIZES

The curriculum refers to whole-group versus pair or individual work. In fact, some classrooms may benefit from other groupings. Whether individual work is feasible depends on the availability of supplies, which may be limited for a number of reasons. However, an effort should be made to allow students to work in as small groups as possible, even individually. At the same time, the curriculum includes numerous opportunities to promote conversations which are enriched by multiple voices, viewpoints, and experiences. Some classes may be able to have these discussions as a whole group. Other classes may want to break up into smaller groups to allow more children the opportunity to speak

and to maintain focus. Some classes structure ScratchJr time to fit into a “center time” in the schedule, in which students rotate through small stations around the room with different activities at each location. This format gives students more access to teachers when they have questions and lets teachers tailor instruction and feedback as well as assess each students’ progress more easily than during whole-group work. It is important to find a structure and group size for each of the different activities (instruction, discussions, work on the challenges, and the final project) that meet the needs of the students and teachers in the class.

ALIGNMENT OF ACADEMIC FRAMEWORK

This curriculum is designed for kindergarten and covers many foundational computer science and engineering skills. These academic frameworks are taught through a series of powerful ideas: algorithms, modularity, control structures, representation, hardware/software, design process, and debugging. Each powerful idea has activities and materials (in this case, the activities are tailored to fit the theme of *Knuffle Bunny*) that encourage mastery of the powerful ideas from computational thinking (CT) and matches them with corresponding powerful ideas from literacy. This curriculum contains activities that specifically address the following literacy concepts and skills: the writing process, recalling, summarizing and sequencing, using foreshadowing, and using reading strategies such as predicting, summarizing, and evaluating.

Each lesson in this curriculum unit is aligned with standards from the **Common Core English Language Arts (ELA)/Literacy Framework**. The Common Core framework is “a set of standards that were created to ensure that all students graduate from high school with the skills and knowledge necessary to succeed in college, career, and life, regardless of where they live” (National Governors Association Center for Best Practices & Council of Chief State School Officers, 2010). Lessons in this curriculum are also aligned with the nationally recognized computer science frameworks **K–12 Computer Science Framework** (2016).

	Powerful Ideas of Computational Thinking (CT) and Literacy Embedded in Each Lesson	Common Core ELA/Literacy Framework (Kindergarten)	Computer Science Framework Alignment (Based on the “by end of Grade 2 band”)
1: Foundations	<i>CT: Design Process</i> <i>Literacy: Writing Process</i>	CCSS.ELA-LITERACY.W.K.5 With guidance and support from adults, respond to questions and suggestions from peers and add details to strengthen writing as needed.	Algorithms and Programming: Algorithms: People follow and create processes as part of daily life. Many of these processes can be expressed as algorithms that computers can follow.
2: What is a Program?	<i>CT: Algorithms</i> <i>Literacy: Tools of Communication</i>	CCSS.ELA-LITERACY.W.K.6 With guidance and support from adults, explore a variety of digital tools to produce and publish writing, including in collaboration with peers.	Algorithms and Programming: Program Development: People develop programs collaboratively and for a purpose, such as expressing ideas or addressing problems.

<p>3: Taking Care of Our Materials</p>	<p><i>CT: Hardware/Software</i> <i>Literacy: Book Handling</i></p>	<p>CCSS.ELA-LITERACY.W.K.6 With guidance and support from adults, explore a variety of digital tools to produce and publish writing, including in collaboration with peers.</p>	<p>Computing Systems Hardware and Software: A computing system is composed of hardware and software. Hardware consists of physical components, while software provides instructions for the system. These instructions are represented in a form that a computer can understand.</p> <p>Devices: People use computing devices to perform a variety of tasks accurately and quickly. Computing devices interpret and follow the instructions they are given literally.</p>
<p>4: Sequencing</p>	<p><i>CT: Algorithms, Control Structure</i> <i>Literacy: Sequence, Summarizing/Retelling</i></p>	<p>CCSS.ELA-LITERACY.W.K.3 Use a combination of drawing, dictating, and writing to narrate a single event or several loosely linked events, tell about the events in the order in which they occurred, and provide a reaction to what happened.</p>	<p>Algorithms and Programming: Algorithms: People follow and create processes as part of daily life. Many of these processes can be expressed as algorithms that computers can follow.</p> <p>Control: Computers follow precise sequences of instructions that automate tasks. Program execution can also be nonsequential by repeating patterns of instructions and using events to initiate instructions.</p>
<p>5: Characters</p>	<p><i>CT: Representation</i> <i>Literacy: Characters</i></p>	<p>CCSS.ELA-LITERACY.RL.K.2 With prompting and support, identify characters, settings, and major events in a story.</p>	<p>Algorithms and Programming Variables: Information in the real world can be represented in computer programs. Programs store and manipulate data, such as numbers, words, colors, and images. The type of data determines the actions and attributes associated with it.</p>

<p>6: Programmer and Author</p>	<p><i>CT: Design Process, Algorithms, Control Structures</i></p> <p>Literacy: <i>Writing Process, Sequence</i></p>	<p>CCSS.ELA-LITERACY.RL.K.6 With prompting and support, name the author and illustrator of a story and define the role of each in telling the story.</p> <p>CCSS.ELA-LITERACY.W.K.3 Use a combination of drawing, dictating, and writing to narrate a single event or several loosely linked events, tell about the events in the order in which they occurred, and provide a reaction to what happened.</p>	<p>Algorithms and Programming: Program Development: People develop programs collaboratively and for a purpose, such as expressing ideas or addressing problems.</p>
<p>7: Programming</p>	<p><i>CT: Algorithms, Design Process</i></p> <p>Literacy: <i>Writing Process</i></p>	<p>CCSS.ELA-LITERACY.W.K.2 Use a combination of drawing, dictating, and writing to compose informative/explanatory texts in which they name what they are writing about and supply some information about the topic.</p>	<p>Algorithms and Programming: Program Development: People develop programs collaboratively and for a purpose, such as expressing ideas or addressing problems.</p>
<p>8: Debugging</p>	<p><i>CT: Debugging</i></p> <p>Literacy: <i>Editing, Awareness of Audience</i></p>	<p>CCSS.ELA-LITERACY.W.K.5 With guidance and support from adults, respond to questions and suggestions from peers and add details to strengthen writing as needed.</p>	<p>Computing Systems Troubleshooting: Computing systems might not work as expected because of hardware or software problems. Clearly describing a problem is the first step toward finding a solution.</p>
<p>9: Details</p>	<p><i>CT: Representation</i></p> <p>Literacy: <i>Descriptive Language in Writing, Characters</i></p>	<p>CCSS.ELA-LITERACY.W.K.2 Use a combination of drawing, dictating, and writing to compose informative/explanatory texts in which they name what they are writing about and supply some information about the topic.</p>	<p>Algorithms and Programming Variables: Information in the real world can be represented in computer programs. Programs store and manipulate data, such as numbers, words, colors, and images. The type of data determines the actions and attributes associated with it.</p>

<p>10: Repeat Loops</p>	<p><i>CT: Control Structure, Modularity</i></p> <p><i>Literacy: Repetition as a Literary Device, Repetition in Word Forms</i></p>	<p>CCSS.ELA-LITERACY.W.K.5 Recognize common types of texts (e.g., storybooks, poems)</p>	<p>Algorithms and Programming Modularity: Complex tasks can be broken down into simpler instructions, some of which can be broken down even further. Likewise, instructions can be combined to accomplish complex tasks.</p>
<p>11: Final Project - Planning Story Time</p>	<p><i>CT: Design Process, Representation, Control Structure</i></p> <p><i>Literacy: Character, Sequence, Writing Process</i></p>	<p>CCSS.ELA-LITERACY.W.K.3 Use a combination of drawing, dictating, and writing to narrate a single event or several loosely linked events, tell about the events in the order in which they occurred, and provide a reaction to what happened.</p> <p>CCSS.ELA-LITERACY.RL.K.2 With prompting and support, retell familiar stories, including key details.</p>	<p>Algorithms and Programming: Algorithms: People follow and create processes as part of daily life. Many of these processes can be expressed as algorithms that computers can follow.</p> <p>Control: Computers follow precise sequences of instructions that automate tasks. Program execution can also be nonsequential by repeating patterns of instructions and using events to initiate instructions.</p> <p>Variables: Information in the real world can be represented in computer programs. Programs store and manipulate data, such as numbers, words, colors, and images. The type of data determines the actions and attributes associated with it.</p>
<p>12: Final Project - Coding Story Time</p>	<p><i>CT: Algorithms, Design Process</i></p> <p><i>Literacy: Writing Process, Awareness of Audience</i></p>	<p>CCSS.ELA-LITERACY.W.K.5 With guidance and support from adults, respond to questions and suggestions from peers and add details to strengthen writing as needed.</p>	<p>Algorithms and Programming: Algorithms: People follow and create processes as part of daily life. Many of these processes can be expressed as algorithms that computers can follow.</p>

Lesson 1: Foundations

Powerful Idea From Computer Science:

Design Process

OVERVIEW

Students will learn about the Design Process and the Writing Process and understand how both processes are similar in nature but serve different purposes. Activities in this lesson encourage students to think and act like programmers and writers.

PURPOSE

While this lesson does not involve using the Scratch Jr app, the activities set up an important foundation for how students engage in key computer science and literacy skills, such as brainstorming ideas, planning out a project, reviewing and revising ideas, and sharing ideas with peers.

ACTIVITIES

- What is a Programmer? (10 min)
- The Design Process (10 min)
- Programmers and Writers (10 min)
- Think Like an Programmer (10 min)
- How-to-Book (20 min)

STUDENTS WILL BE ABLE TO...

- Define program and programmers
- Compare and contrast the Design Process and Writing Process
- Use the Design and Writing Processes to write a How-to-Book

Powerful Idea From Literacy:

Writing Process

PREPARATION FOR TEACHERS

- Read through the Activity Guide
- Create anchor charts of the Design Process and Writing Process*
- Print Design Journals (one for each student - to be used throughout the entire unit)

MATERIALS

FOR THE TEACHER:

- Anchor chart of Design Process*
- Anchor chart of Writing Process*

FOR STUDENTS:

- Design Journal (see Appendix C for example)

*See Appendix A for examples

VOCABULARY

- Cycle — something that moves in a circle (i.e. the seasons, a baseball field (compare to a football field that goes forwards and backwards) the Design Process, the Writing Process)
- Design — a plan for a building or invention
- Program — a complete set of instructions for a computer
- Programmer — someone who writes programs

Lesson 1: Activities

WHAT IS A PROGRAMMER? (10 min)



Ask students: What do you think is a programmer? Has anyone ever heard of programmers? What do you think they do?

Explain to students that programmers do many different things - they work with computers to write *programs*. Programs are instructions for computers! So, every time students play games on tablets or use computers at school, they are using the work of programmers. Programmers often write their programs to help solve different types of problems that can be fixed by computers.

As a class, brainstorm different problems that computers can help solve. Feel free to write these problems on a piece of chart paper. Prompt students to think about the various different ways computers are present in their life (e.g. communication, calculations, information, etc.)



THE DESIGN PROCESS (10 min)

Explain to students that in order for programmers to write solutions, they need to go through the steps of the Design Process. To help them understand the Design Process better, give the students a problem for them to solve.

Ask students: If there's a bug on the ceiling, how can I reach the bug and catch it?

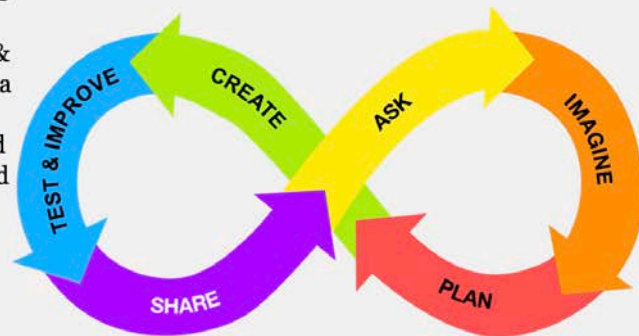
Help the students imagine solutions by suggesting materials you can use in the classroom - a chair, cup, paper - to introduce them to ways the problem can be addressed. As they walk through imagining solutions and creating a plan, let them know that they just began the steps of Design Process. Continue to walk through the rest of the steps of the Design Process using the example you have chosen. Engage with the preschoolers through a design process song detailing each stage of the series.

Design Process

When making projects, engineers follow a series of steps called the **Design Process**. It has 6 steps: ASK, IMAGINE, PLAN, CREATE, TEST & IMPROVE, and SHARE. The Design Process is a **cycle** – there's no official starting or ending point. You can begin at any step, move back and forth between steps, or repeat the cycle over and over!

Design Process song

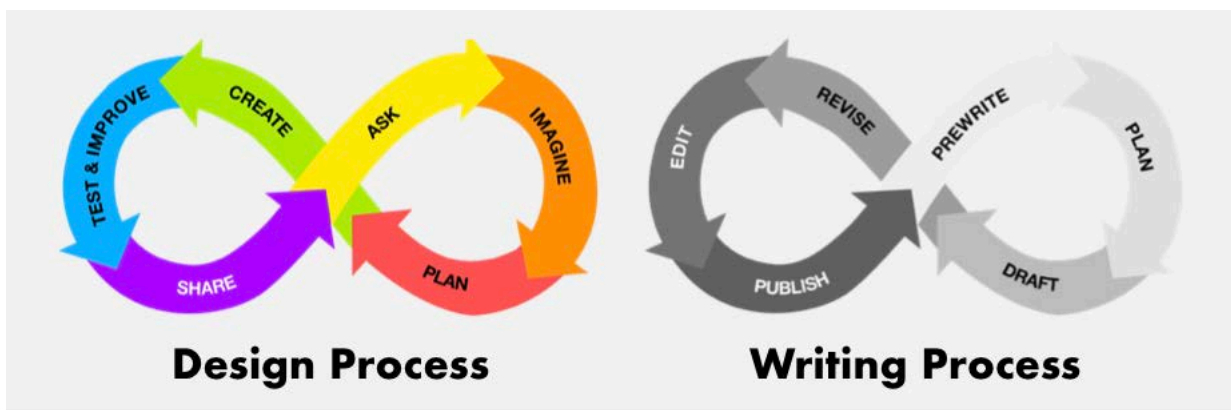
(to the tune of "Twinkle, Twinkle")
Ask and imagine, plan and create,
Test and improve and share what we make.
(Repeat)



PROGRAMMERS AND WRITERS (10 min)

Show students the Design Process and the Writing Process side by side. Explain to students that both are creative processes that require imagination, planning, creating, revising, feedback, and sharing. Both programmers and writers turn ideas into projects that are shared with others. Ask students what other activities require a process (e.g., cooking, painting, getting good at a sport, etc.). Lead student-centered discussion on the similarities and differences between programmers and writers.

Writing Process: Just as programmers use the Design Process to design and create projects, writers use the Writing Process to brainstorm ideas, write a draft, make revisions, and share their writing with others. The Writing Process is also a cycle - there's no official starting or ending point, and you can move back and forth between steps!



THINK LIKE A PROGRAMMER (10 min)

Explain to students that everyone in the class is going to start thinking like a programmer! Ask students: *Have you ever used a computer or tablet before? What did you use them for? Did you play a game? Was it fun?* Explain to students that programmers don't just create games or programs without thinking of a plan first. The purpose of this activity is to engage students in thinking about design and how programmers must think creatively in order to engage audiences in their creations.

HOW-TO-BOOKS: BUILDING A PROGRAM (20 min)

How-to-Books are a low-stress entry point into writing. After all, all students know how to do something and the structure of a How-to-Book is fairly simple. In addition, pictures can easily take the place of words. We even suggest that each step in a how-to book should be accompanied by a sketch or picture. Pass out the Design Journals. Ask students to create a "How-to-Book", or really a "How-to-Book", outline for designing an app or game. Ask students to include specific details so that someone else can learn about their app or game by reading these instructions. Depending on the students' writing level, this activity may need more framing. A wonderful resource for How-to-Books can be found at: <https://www.education.com/lesson-plan/creating-a-how-to-book/>.

Lesson 2: What Is a Program?

Powerful Idea From Computer Science:

Algorithms

Powerful Idea From Literacy:

Tools of Communication

OVERVIEW

Students will learn about programming and be introduced to the ScratchJr app. Students will participate in numerous activities that emphasize the importance of clarity in both writing and computer science. Once students have a grasp on what a program is, they will be introduced to the app as a class.

PURPOSE

In the previous lesson, students were introduced to the concept of programming and the design process. Now they will begin to think more deeply about what a program is and how it is written. Learning the process behind a program as well as why the program is written are both crucial for gaining a broader understanding of computer science. This lesson gives the students strong foundational knowledge before engaging with ScratchJr.

ACTIVITIES

- What is a Program? (5 min)
- Tools of Communication (20 min)
- Programmer Says (20 min)
- Meet the ScratchJr App (15 min)

STUDENTS WILL BE ABLE TO...

- Understand that programs are instructions for computers
- Understand that those instructions must be very clear in order to be executed properly
- Know the function of the start, end and motion blocks

PREPARATION FOR TEACHERS

- Read through the Activity Guide
- Print large ScratchJr Blocks
- Download ScratchJr onto a class set of compatible tablets
- Familiarize yourself with ScratchJr <https://www.scratchjr.org>
- Go through the Programmer Says cards and take out only the blocks listed in the Materials section

MATERIALS

FOR THE TEACHER:

- Large ScratchJr block cards: Begin and End, Blue Motion Blocks

FOR STUDENTS:

- Tablet with ScratchJr

VOCABULARY

- Program — a complete set of instructions for a computer
- Sequence — the order of instructions that a robot will follow exactly (often used interchangeably with algorithm)

Lesson 2: Activity

WHAT IS A PROGRAM? (5 min)

A program is a sequence of instructions that the robot acts in order. Each instruction has a specific meaning, and the order of the instructions affects the robot's overall actions. This is an example of a ScratchJr Program:



TOOLS OF COMMUNICATION (20 min)

Have students sit in a circle and play a game of “Telephone”, in which one student thinks of a message and whispers it to the person sitting next to them, who then whispers to the person next to them, and so on and so forth until the message gets to the last person. Ask the last person and the first person to say their messages out loud and compare the two messages. *Ask students: Were the two messages the same? Why or why not? What are some other ways we could use to pass along a message?*

Repeat the game one final time, this time by giving each student a typed and printed version of the message. Have a few students read out their printed message. *Ask students: How was this better than the last two rounds? Are all students able to receive the same information? (Yes)*

At the end of the activity, explain to students how this mirrors the evolution of writing technology from oral societies to scribal writing to post-printing press. Help students draw the connection to the evolution of computers and robotic technologies. More specifically, explain to students that if we had to program robots without writing, it would be messy, but we can use computer writing to program robots, and that is called **code**.



PROGRAMMER SAYS (20 min)

In order to program in ScratchJr, students first need to learn ScratchJr's language: the programming blocks! This activity is played like the traditional “Simon Says” game, in which students repeat an action if Simon says to do something. Briefly introduce each programming instruction and what it means (use only the blocks listed in the Materials section in this lesson).

Have the class stand up. Hold up one big ScratchJr icon at a time and say, “Programmer says to _____”. Go through each individual instruction a few times until the class seems to get it. Once students are familiar with each instruction, ask for volunteers to be the Programmer who gives the class full programs to run through (e.g. Begin, Spin, Forward, End). Just like in the real “Simon Says” game, the Programmer can try to be tricky! For example, if the Programmer forgets to give a Begin or End instruction, should the class still move? Just like Simon Says, if the Programmer forgets to say, “Programmer says to _____”, then students should sit down! This will help reinforce the concept that ScratchJr is programmed by humans.

MEET THE SCRATCHJR APP (15 min)

Gather students closely on the carpet or project a tablet in order to look at the ScratchJr App as a group. Throughout this demonstration, be very intentional in showing students how you drag blocks around the screen, tap different features. Demonstrate the programming area, stage, block categories and programming script. Check out the ScratchJr interface guide for additional resources: <https://www.scratchjr.org/learn/interface>

Show students how to start a new project:



Show students the green begin block (circled in black) and the programming area (circled in red):



Demonstrate a sample program. Intentionally tap on each blue block one a time and narrate how the kitten reacts to it. Then, you may choose to show snapping the blocks together to create a continuous program:



Show students the red end block:



EXTENDED ACTIVITY:

Tell students: *Just like there are pages in a book, we can have pages in a program.*

- Show students where to add pages (located on the far right of the screen):



- Introduce the “go to page” block:
 - Tell students: This block is just like turning the page in a book! It will turn the page in our programs. So, instead of using the red end block, when you want to turn the page of your program, use this block.
- Pass out tablets. Students may have to work in pairs or small groups. Students will try and recreate their setting drawing in ScratchJr.

Lesson 3: Taking Care of Our Materials

Powerful Idea From Computer Science:

Hardware/Software

Powerful Idea From Literacy:

Book Handling

OVERVIEW

This lesson will introduce the concept of taking care of materials such as books and tablets. It will also cover important conventions of text and programs such as going from left to right, holding materials right side up and being gentle with handling supplies.

PURPOSE

While this lesson does not involve using the ScratchJr app, the activities set up an important foundation for how students engage with materials, both in literacy and in programming.

ACTIVITIES

- How to Treat Our Materials (10 min)
- Conventions of Tablet Usage (10 min)
- Procedure Practice (10 min)
- Left to Right on Paper and ScratchJr(15 min)

STUDENTS WILL BE ABLE TO...

- Hold a book or tablet right side up
- Read and program from left to right
- Take care of their materials

PREPARATION FOR TEACHERS

- Ensure all tablets have ScratchJr App installed
- Create anchor charts for How to Treat Books and Tablets
- Create procedures in your classroom for using and handling tablets

MATERIALS

FOR THE TEACHER:

- Paper for anchor chart
- Optional class set of books
- Class set of tablets

FOR STUDENTS:

- Tablet with ScratchJr

Lesson 3: Activities

HOW TO TREAT OUR MATERIALS (15 min)



Tell the class that today we are going to talk about treating our books and tablets with respect and kindness, just like we treat each other with respect and kindness. Ask students: *Who has ever seen a phone or tablet with a cracked screen? Who has seen a book with ripped pages, missing pages or stains or marks?* Feel free to show examples if you have books in your class that have been treated poorly.

Discuss why it's important to maintain our materials - if we rip books or break tablets, you may not get to use them again, or our class might run out. We want to treat our tablets the same way we treat our books OR be even more careful with them.

Create an anchor chart as a class that shows good and bad ways to treat books and tablets. Come up with some class norms surrounding tablet usage. Examples:

- Keep tablets flat on the floor or table.
- Always carry with two hands if you are walking around.
- Keep tablets put away at snack time.
- Use gentle hands with tablets and books

CONVENTIONS OF TABLET USAGE (15 min)



This activity will be a side by side comparison of a book and ScratchJr to cover topics such as left to right and right side up. Feel free to incorporate strategies or concepts your students have been working on in literacy. Have student give up a thumbs up when something is correct and a thumbs down if something is incorrect. Examples:

- I should hold my tablet like this (upside down). *This may be redundant if your tablets auto swap rotation.*
- When I'm done with my tablet I should toss it onto the ground!
- If I get mad, I should slam my tablet onto the table!
- I should walk very carefully with my tablet held in two hands.
- I should drink my milk and eat my snack right next to my tablet!

PROCEDURE PRACTICE (15 MIN):



This is an opportunity for you to practice with your students any procedures in your classroom related to tablet usage. You may want to have students practice retrieving their tablets from storage and bringing to their desk, practice walking holding the tablet safely, putting away tablets, plugging tablets in for charging etc. You may want to practice how you will get students' attention when you need to make an announcement while they are on tablets. We recommend a call to action that requires students to place both hands in the hair.

LEFT TO RIGHT ON PAPER AND SCRATCHJR (15 min)

There are many different ways to teach young readers that text is read from left to right. Teachers will most likely have this established in their classroom. Feel free to use your chosen method here. We want to make a connection that just as a book is read from left to right, so will programs on tablets. We recommend doing quick read aloud and demonstrating intentionally with your finger that you start at the left and move toward the right. Then, show class an example of ScratchJr and state that it functions the same way. We will dive more into this topic in the next lesson.

Lesson 4: Sequencing

Powerful Idea From Computer Science:

Algorithms, Control Structure

Powerful Idea From Literacy:

Sequence, Summarizing/Retelling

OVERVIEW

Students will learn about sequencing in programming and think about how it relates to sequencing in literacy. In both cases, they will think about why order matters. Once students become familiar with the ScratchJr programming blocks, they will work together as a class to write a group program.

PURPOSE

Students will engage in goal-oriented programming, in which students purposefully choose actions in a specific order to achieve a particular outcome. Understanding that order matters is an important skill for students not only in computer science and literacy, but also in their everyday lives as they learn to tie their shoelaces, reflect on the day's activities, plan a family vacation, and more.

ACTIVITIES

- Knuffle Bunny (10 min)
- Order Matters (20 min)
- Program the Teacher with ScratchJr Blocks (15 min)
- Solve-It Assessment A (15 min)

STUDENTS WILL BE ABLE TO...

- Understand why order matters when writing a program or telling a story
- Apply their knowledge of the Start, End and Motion blocks to “program” movements for their teacher

PREPARATION FOR TEACHERS

- Read through the Activity Guide
- Print Solve-It Assessment A (one for each student)
- Ensure all tablets are charged and are working and have ScratchJr installed

MATERIALS

FOR TEACHER:

- 1 copy of *Knuffle Bunny* by Mo Willems

FOR STUDENTS:

- Tablet with ScratchJr

VOCABULARY

- Instruction — a direction or order
- Order — the arrangement of a group of things according to a particular pattern or sequence
- Program — a complete set of instructions for a computer to follow
- Sequence — a particular order in which things or events follow one another

Lesson 4: Activities

KNUFFLE BUNNY (15 min)



Read the book *Knuffle Bunny* as a class; if needed, read the book a second time. Lead a student-centered discussion that reviews the events of the story. You can prompt the students: *Who can summarize the main events in this story? What would have happened if Trixie didn't go on the errand with her daddy? What would have happened if her mom was at the laundromat and asked where Knuffle Bunny was before they left the laundromat?* The purpose of this activity is to get students to think about sequencing in narrative.

ORDER MATTERS (25 min)



Tell the students that just like order mattered in the book, order matters in programming as well. Programs are instructions that are followed by computers or machines. We all use instructions (use whatever word your students are familiar with i.e. directions, procedures, etc.) every day, both here at school and when you're at home.

Ask students: What are some instructions that we have to follow here at school?

Example: How do we line up at the door for lunch? First, you have to stand up. Then, you have to push in your chair. Then, you have to walk to the line. Would you push in your chair before standing up?

Ask students: Who can think of other step by step instructions that we use at school? What about at home?

Examples: Brushing your teeth, tying shoes.

After students have brainstormed step by step instructions for a few different activities, ask students: *What would happen if you tried to line up for lunch but didn't stand up first? What would happen if you didn't put toothpaste on your toothbrush until after you brushed your teeth? Why did the order matter in each activity?* Choose one of the examples and have students act out the example in all the different orders that they can think of.

Conclude the activity by bringing it back to how computers have to follow instructions as well - reflect on the importance of sequencing in literacy and computer science.

PROGRAM THE TEACHER WITH SCRATCHJR BLOCKS (20 min)



Using the Programmer Says cards, students will work together as a class to “program” their teacher to move from one part of the room to the other. Be silly! An example would be for the students to “program” their teacher to move from the front end of the room to the library area by using these blocks: Begin, Forward, Turn Left/Right, Forward, Forward, End. The goal of this game is for students to practice sequencing as a class before working individually or in their small groups. Before the teacher-computer moves, students can make predictions about where the teacher-computer will end up. It may be helpful to let the students make mistakes in order to foster discussion on sequencing and debugging.

SOLVE-IT ASSESSMENT A (15 min)

On the Appendix B-1 you will find assessment A. Please hand out one copy of the assessment to each child in your class.

Instructions:

- Read each question and option out loud to the group. Students can ask to have questions or options read out loud up to 3 times.
- Instruct children to circle only 1 answer per question.
- Make sure students answer the questions by themselves. Students should not be discussing or copying answers.

Lesson 5: Characters

Powerful Idea From Computer Science:

Representation

OVERVIEW

Students will learn about characters in literacy and learn how to create characters in ScratchJr and use five new blocks: the hide block, the show block, the grow block, the shrink block, and the sound recorder block. All of these blocks alter the characters' appearance and create character actions. At the end of the lesson, students will have a chance to share their character creations with the rest of the class.

PURPOSE

The character feature in ScratchJr is crucial. The character will carry out the programs created by students, but it is also an opportunity for children to get creative, insert themselves into their programs and tell stories.

ACTIVITIES

- Design A Character (15 min)
- Create Your Character in ScratchJr (15 min)
- Free Play with Purple Blocks (10 min)
- ScratchJr Sound (10 min)
- Character Share (10 min)

Powerful Idea From Literacy:

Characters

STUDENTS WILL BE ABLE TO...

- Define character, name the characters in a story and add characters in ScratchJr
- Use the grow, shrink, hide and show blocks successfully
- Record sounds on the sound recorder and include them in their programs

PREPARATION FOR TEACHERS

- Read through the Activity Guide
- Ensure all tablets are charged and are working and have ScratchJr installed

MATERIALS

FOR STUDENTS:

- Design Journal (see Appendix C for example)
- Tablet with ScratchJr

VOCABULARY

- Character — who the story is about

Lesson 5: Activities

DESIGN A CHARACTER (15 min)



Students will use the worksheet in their Design Journal to create their own stuffed animal character, like Knuffle Bunny. You may want students to draw in pencil first and then go back and use crayons or other craft materials. This is also a great opportunity to use any additional art materials you have like modeling clay, felt, or recycled materials etc. Guiding questions:

- Do you have a favorite animal?
- Make sure you include lots of details about your animal so that everyone can see exactly what it is like!

CREATE YOUR CHARACTER IN SCRATCHJR (15 min)

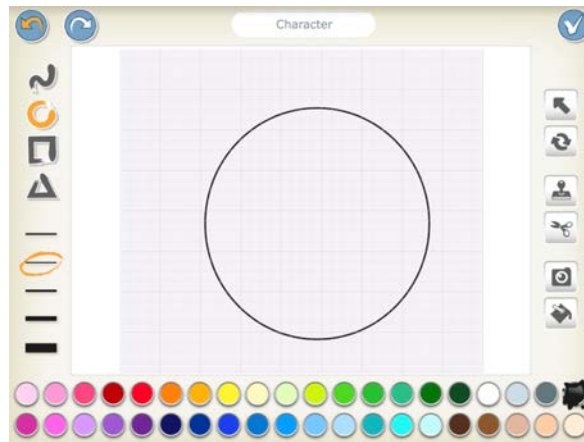


Students will use the paint editor to create their animal character in ScratchJr. They may use the character they created on their worksheet as an example, but it is okay if it ends up looking different - the paint editor can be tricky! (If you used 3D materials like clay or felt, students can upload a picture of their character directly into ScratchJr, see instructions below). For students who create a character quickly, ask them what details they may need to add to describe their character. If the student wishes to use a photo for their character, follow these directions:

1. Add a new character and open the paint editor



2. Select the camera tool and then select a shape for the camera to fill.



3. Select the correct photo, then tap the check mark to return to the stage.



FREE PLAY WITH PURPLE BLOCKS (10 min)

Quickly demonstrate the hide/show and grow/shrink blocks. Allow students to explore using these blocks with their created character.

SCRATCHJR SOUND (10 min)

Demonstrate for you class how to write a program that includes the sound. As always, start with the green flag. Then, tap the green blocks and show the students that they can play a “pop” sound, or record their own sound. To record, tap the green microphone to show the sound recorder. Practice recording a sound as a class and adding it to your program. Then, finish with the red end block.



1. Start with the green begin flag:
2. Play or record your sound: (This may take a few tries in order for students to correctly capture their character intro, and that’s okay!)





3. End on the red end block.

CHARACTER SHARE (10 min)



Save time at the end of the lesson for students to be able to share their characters with the rest of the class. Guiding Questions:

- Describe your character. What are they like?
- What was easy about creating your character? What was difficult?
- What is the setting that your character is in? Why are they there?

Lesson 6: Programmer and Author

Powerful Idea From Computer Science:

Design Process, Algorithms, Control Structures

Powerful Idea From Literacy:

Writing Process, Sequence

OVERVIEW

In previous lessons, students have learned about sequencing and characters, both of which help to draw parallels between literacy and coding. In this lesson, students will solidify their understanding of these parallels by directly looking characteristics of both programmers and authors.

PURPOSE

This lesson allows students to connect their understanding of an author with their understanding of a programming and draw parallels. They will also see how ScratchJr pages feature allows their programs to have a narrative like in a story.

ACTIVITIES

- Introduction (10 min)
- Change Setting in Scratch Jr (5 min)
- Add Page in Scratch Jr (10 min)
- Be a Programmer (20 min)
- Solve-It Assessment B (15 min)

STUDENTS WILL BE ABLE TO...

- Understand parallels between authors and programmers
- Understand that both programs and stories need a logical order (beginning, middle, and end) to make sense

PREPARATION FOR TEACHERS

- Read through the Activity Guide
- Print picture of Mo Willems (optional)*
- Print Solve-It Assessment B (one for each student)
- Ensure all ScratchJr Tablets are working and charged

MATERIALS

FOR THE TEACHER:

- 1 copy of *Knuffle Bunny* by Mo Willems

FOR STUDENTS:

- Tablet with ScratchJr

*See Appendix A for example

VOCABULARY

- Author — writer of a story, poem, book, etc.
- Setting — the time and place a story takes place

Lesson 6: Activities

INTRODUCTION (10 min)

Ask students: Who remembers what a program is? What is a programmer? Let the students know that today, they will get to become programmers. But before they get to become programmers, talk to them about other people that get to create stories.

Ask students: Does anyone know what an author is? (This may be something your class has already covered - if so, this is a great way to check for understanding and make the comparison to a programmer.)

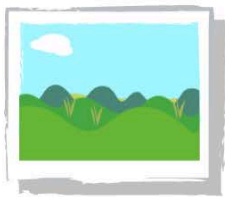
Tell students that an author writes the words of a story just like a programmer writes step-by-step instructions. Reread the book *Knuffle Bunny* as a class. Be sure to emphasize the author's name on the title page and cover. Reinforce by asking the students "Who is the author of *Knuffle Bunny*?" with the sentence starter "The author is _____."

The author of *Knuffle Bunny* is Mo Willems. Being the author of a book means creating and writing the words of the book. Let the students know that on top of becoming a programmer, they will get the chance to also be an author.

Optional: Project or print a picture of Mo Willems so that they see it is a real person. Oftentimes, kids have a difficult time making the jump to understanding that books are written by people!

CHANGE THE SETTING IN SCRATCHJR (5 min)

Have students take out their tablets. Show them how to change the background in ScratchJr. Let them adjust the background to fit the character they designed in the previous lesson. Tell them that just as stories have different settings during different parts of the story, they can program ScratchJr to have different settings for different parts of code.



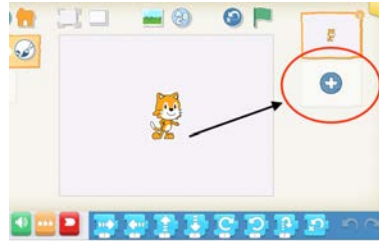
New Background



OK

ADD PAGE IN SCRATCHJR (10 min)

Show students how to add a new page to their ScratchJr program. This can be like adding new pages in a story. Different pages can have different backgrounds and new characters.



Once they have added pages, introduce the “go to page” block . Tell students: This block is just like turning the page in a book! It will turn the page in our programs. So, instead of using the red end block, when you want to turn the page of your program, use this block.



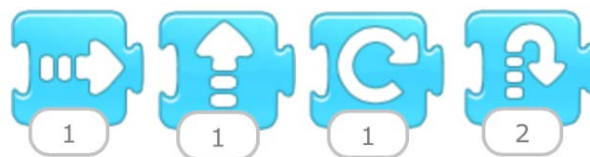
BE A PROGRAMMER (20 min)

Tell students that they are going to get to tell a short story today using three pages. The first page will be the beginning of the story, the second page will be the middle, and the third page will be the end of the story. Encourage students to use the motion and purple blocks in their story, as well as the end next page block to go to each page of the story.

Before students start programming, demonstrate simple blocks for the class. They can watch or follow along on their own tablets.

1. Making a program:

1. Drag blocks into the programming area and tap on them to show how the character moves. Some important motion blocks are left, right, up, down, rotate left/right, and jump.



2. Demonstrate how to change the number on a block to move multiple times.



3. Show how to snap blocks together to make a sequence of movements.
4. Demonstrate how to create a program using start and end blocks that runs by clicking the green flag. Note that start and end blocks are necessary when using presentation mode or when running a program with multiple parts (so they start simultaneously).



Let students explore ScratchJr on their own for about 15 mins. Encourage them to make programs with a beginning, middle and end and with different numbers on their blue movement blocks. Select a few programs to share out at the end of class.

SOLVE-IT ASSESSMENT B (15 min)

On the Appendix B-1 you will find assessment B. Please hand out one copy of the assessment to each child in your class.

Instructions:

- Read each question and option out loud to the group. Students can ask to have questions or options read out loud up to 3 times.
- Instruct children to circle only 1 answer per question.
- Make sure students answer the questions by themselves. Students should not be discussing or copying answers.

Lesson 7: Programming

Powerful Idea From Computer Science:

Algorithms, Design Process

Powerful Idea From Literacy:

Writing Process

OVERVIEW

In this lesson, students will program the ScratchJr Kitten to dance the Hokey-Pokey. This programming project will give the students a chance to exercise their current knowledge of the movement blocks and the programming structure. Additionally, students will be practicing sequencing and retelling in programming and think about how it relates to sequencing and summarizing in literacy.

PURPOSE

Students will engage in goal-oriented programming, in which students purposefully choose their ScratchJr blocks and place them in a specific order to achieve a particular outcome. The project will help solidify their understanding of sequencing and order as it pertains to the Hokey-Pokey.

ACTIVITIES

- Dance the Hokey-Pokey (10 min)
- Program the Hokey-Pokey (25 min)
- Hokey-Pokey Reflection (10 min)
- Share Creations (15 min)

STUDENTS WILL BE ABLE TO...

- Tell and retell a story clearly and effectively (the Hokey-Pokey)
- Identify common errors with creating programs and troubleshoot them effectively
- Learn strategies for debugging and editing

PREPARATION FOR TEACHERS

- Ensure all ScratchJr Tablets are working and charged
- Prepare Discussion Sentence Starters anchor chart*
- Print out Solve-It A for each student

MATERIALS

FOR TEACHERS:

- Discussion Sentence Starters anchor chart*

FOR STUDENTS:

- Design Journal (see Appendix C for example)
- Tablet with ScratchJr

*See Appendix A for example

VOCABULARY

- Program — a complete set of instructions for a computer to follow

Lesson 7: Activities



DANCE THE HOKEY-POKEY (10 min)

Explain to students that today they will create a program in ScratchJr to do the Hokey-Pokey. Sing and dance the Hokey-Pokey as a class to make sure everyone knows and remembers it.

*You put your right hand in,
You put your right hand out,
You put your right hand in,
And you shake it all about,*

*You do the hokey pokey
and you turn yourself around
That what it's all about. (clap, clap!)*

- 2) left hand*
- 3) right foot*
- 4) left foot*
- 5) head*
- 6) whole self*

*You put your Kitten in, you put your Kitten out,
You put your Kitten in, and you shake it all about.
You do the Hokey Pokey, and you turn your Kitten around.
And that's what it's all about. (Clap, clap.)*

PROGRAM THE HOKEY-POKEY (25 min)



Take out tablets and remind students of any rules or procedures. Tell students that we are going to be programming the Kitten in ScratchJr to do the Hokey-Pokey. Together as a class, brainstorm which motion blocks should be used for which parts of the song. Make the connection that just like how our instructions gave us directions one step at a time, we are creating a program one block at a time. Have students follow along on their own tablets, as the class creates a program together.

HOKEY-POKEY REFLECTION (10 min)

In their Design Journals, ask students to record their Hokey-Pokey programs by drawing the ScratchJr blocks in their program. Ask students: *How many times did we use each programming block? What order did we put the blocks in? Why did we choose this particular order?*

SHARE CREATIONS (15 min)



When all students are done with their Hokey-Pokey programs, ask the whole class to play their programs at once and dance the Hokey-Pokey! This is the first time that students engage in goal-oriented programming. Using the Discussion Sentence Starters anchor chart, ask students about their challenges of programming: *What problems did you have when you were putting blocks together? Did*

you ever feel frustrated or disappointed? Why did you feel that way? Note down students' responses on a piece of paper so that you can come back to these points in the next lesson.

EXAMPLE PROGRAM (though there is no "correct" answer!)



Lesson 8: Debugging

Powerful Idea From Computer Science:

Debugging

Powerful Idea From Literacy:

Editing, Awareness of Audience

OVERVIEW

In this lesson, students learn the importance of communicating effectively to an audience. Students engage in this learning by retelling a story to their peers and “edit” their story when their audience is confused and needs more clarification. Students connect this idea to when a ScratchJr program does not turn out the way they had expected. The process of figuring out what went wrong and how to fix things is called debugging. At the end of the lesson, students will demonstrate their current level of understanding by completing a Solve-It assessment.

PURPOSE

The parallel of editing in literacy and debugging in computer science is an asset to students’ development in both fields. This lesson aims to align the two processes by engaging the students in the activities of editing and debugging side-by-side.

ACTIVITIES

- Why is Kitten Confused? (15 min)
- Speed Block (5 min)
- Free Play (15 min)
- Debugging Reflection (10 min)
- Solve-It Assessment C (15 min)

STUDENTS WILL BE ABLE TO...

- Identify common errors with ScratchJr programs and troubleshoot them
- Use speed block correctly
- Learn strategies for debugging and editing

PREPARATION FOR TEACHERS

- Read through the Activity Guide
- Print Solve-It Assessment C (one for each student)
- Prepare Anchor Chart on Why is Kitten Confused (reference back to the Hokey-Pokey lesson)

MATERIALS

FOR THE TEACHER:

- Why is Kitten Confused? anchor chart

FOR STUDENTS:

- Design Journal (see Appendix C for example)
- Tablet with ScratchJr

VOCABULARY

- Debug — to find and solve a problem in a computer program
- Edit — to make changes to something

Lesson 8: Activities

WHY IS KITTEN CONFUSED? (15 min)



In previous lessons, students shared challenges of programming the Hokey-Pokey. Check back on your notes from that discussion and prepare an anchor chart noting 4-5 of these challenges on the left side of the chart, leaving the right side empty for students to provide solutions in this activity. Present the anchor chart to students. Explain to students how in the previous lesson, students encountered different challenges. Other examples can be found at: <https://www.scratchjr.org/learn/tips>
Ask students to brainstorm 1-2 solution for every problem.

Explain to students that **debugging** is a method used to understand how to fix things when engineers program robots, and the robots do not work. By identifying these problems and different solutions to solve them, students are debugging.

Debugging is a word used in computer science to describe when people find errors in their computer programs and use different strategies to solve the problem. While the word “bug” was used in other scientific fields, the word “debugging” is attributed to Admiral Grace Hopper, who back in the 1940s found a moth stuck inside the computer (computers used to be that big!), which caused an error in the system. She was able to resolve the error by taking out the bug, hence the word “debugging”!

For further activity ideas and examples of pictures, check out the following resources:

- <https://www.computerhope.com/issues/ch000984.htm>
- <https://www.npr.org/sections/alltechconsidered/2015/11/23/457129179/the-future-of-nanotechnology-and-computers-so-small-you-can-swallow-them>

SPEED BLOCK (5 min)

Once students have a grasp on how to debug, introduce a new block to them. Tell the students that the Speed Block will allow them to change the speed of their character.



FREE PLAY (15 min)

This is a great opportunity for students to freely explore the ScratchJr app and programming blocks, especially the speed block. Encourage students to try and make mistakes and to practice debugging! By the end of this activity, students should feel comfortable running a complete program with the speed block.

DEBUGGING REFLECTION (10 min)

Pass out students' Design Journals. Ask students to reflect on one of the problems they had in ScratchJr. *What was the problem? Why wasn't the Kitten understanding what they wanted it to do?* Students can reflect in their Design Journals by drawing a picture of how they debugged, or if they can, write about their problem solving strategy.

SOLVE IT ASSESSMENT C (15 min)

On the Appendix B-1 you will find assessment C. Please hand out one copy of the assessment to each child in your class.

Instructions:

- Read each question and option out loud to the group. Students can ask to have questions or options read out loud up to 3 times.
- Instruct children to circle only 1 answer per question.
- Make sure students answer the questions by themselves. Students should not be discussing or copying answers.

Lesson 9: Details

Powerful Idea From Computer Science:

Representation

Powerful Idea From Literacy:

Descriptive Language in Writing, Characters

OVERVIEW

In the previous lesson, students were introduced to the speed block. In this lesson, students will be able to apply this block and other complex motion blocks by coding a dance. The dance will involve movement that is slow, medium, or fast and has pauses or wait time. This lesson involves a lot of movement for students - a great opportunity for students to stay engaged and energized! Speed is an important detail that can be added to programs.

PURPOSE

Speed and wait time are two important, yet difficult, parts of programming in ScratchJr. Both features add an additional element to the program, the same way that details add to a story.

ACTIVITIES

- Introduction (5 min)
- Fast or Slow? (10 min)
- Freeze Dance (15 min)
- Wait Time Block (5 min)
- Freeze Dance Program (25 min)

STUDENTS WILL BE ABLE TO...

- Use the control speed and wait time blocks together appropriately

PREPARATION FOR TEACHERS

- Read through the Activity Guide
- Ensure all tablets are working and have ScratchJr
- Print out Solve-It B for each student

MATERIALS

FOR THE TEACHER:

- A few songs of your choosing and speakers

FOR STUDENTS:

- Design Journal (see Appendix C for example)
- Tablet with ScratchJr

VOCABULARY

- Edit — to make changes to something
- Detail — a small part of fact

Lesson 9: Activities

INTRODUCTION (5 min)

Remind students about the Speed Block that they were introduced to in the last class. Tell them that today, they will be thinking about the speed of songs, characters, and their own dancing!



FAST OR SLOW (10 min)

In this activity, students will listen to quick clips of songs and determine if they are fast or slow. You may choose to have students have a hand signal for fast and slow, write their answer in their Design Journals, write their answer on white boards, etc. If students disagree about a song, use the disagreement to prompt discussion. Ask students:

- *Why did you think the song was fast? Why did you think it was slow?*
- *What are some descriptive words that we could use to describe the song?*



FREEZE DANCE (15 min)

Freeze Dance is a great game to get students moving and engage their creativity. When music plays, students dance and when the music pauses, they must freeze immediately. You may choose to use the same songs you used in the Fast or Slow activity or chose other songs your students enjoy. As the teacher, control the music and press pause at will to make students freeze. Make sure you reinforce class norms around safety and being cautious with bodies.

WAIT TIME BLOCK (5 min)

Explain to students that we can also make our characters in ScratchJr freeze, just like we did in freeze dance. We do this using the wait time block. The wait time block causes our programs to freeze briefly.



Optional math connection: The changeable number at the bottom of the wait time block signifies the amount of time that the program will pause or wait. The number is in tenths of seconds. For students who know their multiples of ten, this is a great opportunity to make a math connection.

PROGRAM A FREEZE DANCE (25 min)



Allow students to use their own tablets to follow along while you lead the class in programming a freeze dance. Remind them that the program must use both the speed and wait time blocks. Programs are becoming more detailed, and thus more difficult, so it is okay if the class needs additional time or does not complete the activity.

Lesson 10: Repeat Loops

Powerful Idea From Computer Science:

Control Structure, Modularity

Powerful Idea From Literacy:

Repetition as a Literary Device, Repetition in Word Forms

OVERVIEW

In this lesson, students will think about different instances in daily life which repetition occurs. They will then think about these instances in the context of ScratchJr and learn how they can program repetition loops to simplify their programs. At the end of the lesson, students will demonstrate their level of understanding by completing the last Solve-It assessment.

PURPOSE

In this lesson, students understand the importance of repetition both in computer science and literature. Students will learn about a new instruction that makes ScratchJr repeat programming instructions infinitely or a given number of times. Students also think about repetition as a literary device and the purpose it serves in a text, as well as repetition in word structure as a review of foundational phonic and word recognition skills.

ACTIVITIES

- Repetition in Instructions (5 min)
- Toothbrush Exercise (15 min)
- ScratchJr Repeat with Numbers (25 min)
- Solve-It Assessment D (15 min)

STUDENTS WILL BE ABLE TO...

- Identify patterns in code sequences and rewrite codes using repeat loops
- Use ScratchJr repeat blocks to make a program that loops a certain number of times
- Understand how there is repetition in instructions

PREPARATION FOR TEACHERS

- Read through the Activity Guide
- Print Toothbrush Sequence Card*
- Print Solve-it Assessment D (one for each student)
- Ensure all tablets are charged and have ScratchJr

MATERIALS

FOR THE TEACHER:

- ScratchJr Block Cards
- Toothbrush Sequence Cards*

FOR STUDENTS:

- Tablet with ScratchJr

*See Appendix A for examples

VOCABULARY

- Loop — something that repeats over and over again

Lesson 10: Activities



REPETITION IN INSTRUCTIONS (5 min)

Chose a student to come to the front of the class and sit in a chair. Ask the student to stand up, walk 2 steps forward, walk 2 steps backward, then sit down. As soon as they sit down ask them to stand up, walk 2 steps forward, walk 2 steps backward, then sit down. Repeat this process 2 more times. Finally, ask the class:

- Is there an easier way I could have gotten x to follow these instructions?
- What would it be? (“Stand up walk 2 steps forward, walk 2 steps backward, then sit down and repeat this whole process 4 times)



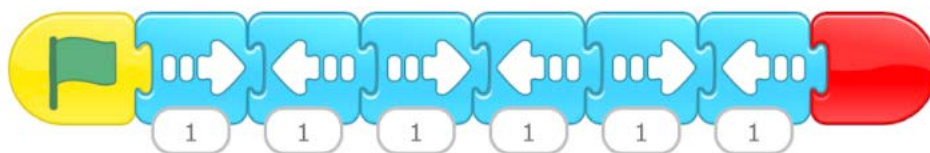
TOOTHBRUSH EXERCISE (15 min)

Have students think about the way they brush their teeth. *Ask students: Are there actions that you have to repeat? (e.g. moving the toothbrush from left to right) Are there motions that only happen once? (e.g. squeezing out toothpaste)* Hand out Toothbrush Sequence Pictures to the children. Working in pairs, have students arrange the pictures in order, noting which pictures may be repeated. Then have the children come up with instructions for brushing your teeth based on the pictures they arranged and act it out to ensure they have covered all the steps. This exercise helps students understand repeat loops while also revisiting the concept of sequencing. Make sure to remind students that even when they’re focusing on which parts of the action repeat, they should always be thinking about why the order matters as well!

Once pairs finish, have several students share out their programs. As a class, discuss how the programs were similar or different.

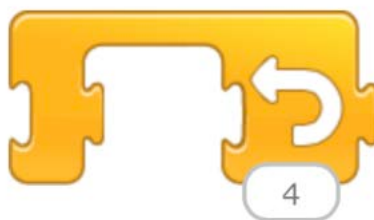
SCRATCHJR REPEAT WITH NUMBERS (25 min)

Project your tablet screen to the class or gather everyone close enough so that they can see your screen. Create a basic program that has the Kitten going forward, backward, forward, backward, forward backward.

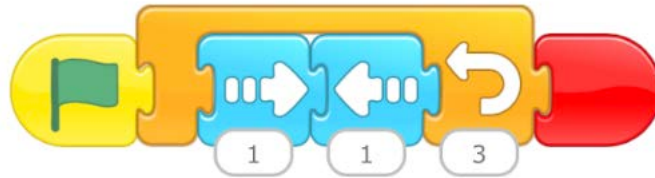


Ask the class: Think back to our activity earlier when (student’s name) had to get up from the chair and walk around several times. Do you think there’s an easier way I could program this? Is there a way I could make the program shorter?

Introduce the Repeat Block. Demonstrate changing the number in the bottom right. Note that this is how you change how many times the pattern will repeat.



Create your same program of moving forward and backward, but this time use the repeat block.

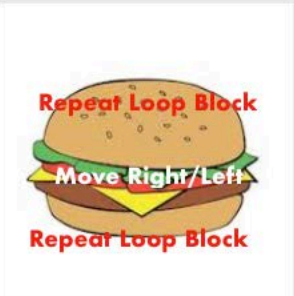


After students have been introduced to the block, allow them to explore their own programs using the repeat blocks. The emphasis here should be on proper syntax i.e. making sure that what students want to repeat is within the repeat block sandwich.

What is a Repeat Loop?

The orange repeat loop block is like the bread of a sandwich. The programming blocks put inside of them are like the filling. ScratchJr will only repeat the commands that are placed inside of the Repeat Loop Sandwich. Any blocks outside of the sandwich will not be repeated.

Parameters are used to tell how many times to repeat the program, or when to stop repeating.



SOLVE IT ASSESSMENT D (15 min)

On the Appendix B-1 you will find assessment D. Please hand out one copy of the assessment to each child in your class.

Instructions:

- Read each question and option out loud to the group. Students can ask to have questions or options read out loud up to 3 times.
- Instruct children to circle only 1 answer per question.
- Make sure students answer the questions by themselves. Students should not be discussing or copying answers.

Lesson 11: Final Project - Planning Story Time

Powerful Idea From Computer Science:

Design Process, Representation, Control Structure

Powerful Idea From Literacy:

Character, Sequence, Writing Process

OVERVIEW

In this lesson, students will begin planning their own stories for their final projects. Their planning process will be broken down by the Story Map worksheet which will help them to identify their characters, setting, and sequence of the story. Students won't engage with ScratchJr in this lesson, as the main focus is on details, planning, and characterization.

PURPOSE

This lesson will emphasize the plan aspect of the Design Process and Writing Process. Students will learn to identify the beginning, middle and end of their story, while also planning out the details of their characters and settings. This lesson will allow them to be creative while also organizing their thoughts for their projects.

ACTIVITIES

- Knuffle Bunny (10 min)
- Story Time Introduction (10 min)
- Story Map Worksheet (20 min)
- Characters and Setting (20 min)

STUDENTS WILL BE ABLE TO...

- Identify the characters, setting, beginning, middle and end of their stories
- Use the Writing Process and Design Process to plan their story before they begin programming

PREPARATION FOR TEACHERS

- Read through the Activity Guide
- Make sure all tablets are fully charged with ScratchJr downloaded

MATERIALS

FOR THE TEACHER:

- 1 copy of *Knuffle Bunny* by Mo Willems

FOR STUDENTS:

- Design Journal (see Appendix C for example)
- Tablet with ScratchJr

*See Appendix A for examples

VOCABULARY

Character — who the story is about

Setting — the time and place a story takes place

Lesson 11: Activities



KNUFFLE BUNNY (10 min)

Together as a class, reread the *Knuffle Bunny*. Talk about how Trixie and Knuffle Bunny go everywhere together. *Ask students: Other than the laundromat, where else might Trixie go with Knuffle Bunny? What kind of adventures do you think they might go on?* The purpose of this discussion is to prompt the students to think creatively about different activities that characters can do in stories.

STORY TIME INTRODUCTION (10 min)


Tell students that for the last two lessons, they will be working on making their own story about themselves and their own stuffed animal! They will get to use the stuffed animal character that they made earlier in the curriculum, to make a story about an adventure that they go on with their character, just like Trixie and Knuffle Bunny. Talk to the students about the characters, setting, and dialogue that happens in *Knuffle Bunny*. Point out to the students that they've learned how to make characters, setting and dialogue in Scratch Jr too! They will be able to use these tools to make up a new story. First, they will have to plan their story.


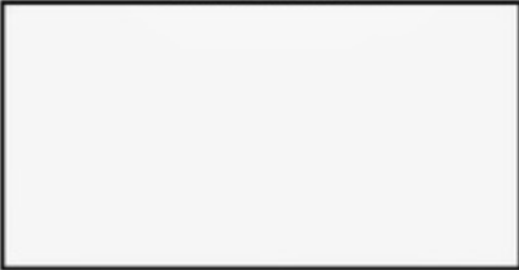





STORY MAP WORKSHEET (20 min)

Explain to students that writing programs and stories requires lots of planning! Have the students take out their Design Journals for this lesson to find their Story Map Worksheet. This worksheet will help them plan out the key parts of their story so that they can be ready to program it in the next class.

Name _____

Story Map 

Setting	Characters	
		
Beginning	Middle	End
		

© 2014 Katie Ryan

CHARACTERS AND SETTING (20 min)



Once the students have a plan for their story, give them time on Scratch Jr to start making all the characters in their story. The students have already made their stuffed animal character, so they will now need to make a character for themselves and any other characters they choose to add into their story. Help students add characters by introducing the add character button, if they have not already discovered it.



Once the students have made their characters, also remind them that they can change the setting for their story and add pages, like we learned earlier in the curriculum. Encourage students to design new settings or use multiple settings to enhance their story.

Lesson 12: Final Project - Coding Story Time

Powerful Idea From Computer Science:

Algorithms, Design Process

Powerful Idea From Literacy:

Writing Process, Awareness of Audience

OVERVIEW

In the previous lesson, students worked to plan their stories and design their characters and settings. In this lesson, they will program the actual story. Students will utilize all programming skills that they have learned throughout the previous lessons to compose a detailed and complete story. Once the students have finished their projects, they will give and receive peer feedback and then share with the class.

PURPOSE

The purpose of this lesson is to allow the students to use all programming skills that they have learned throughout the past lessons in an integrative final project. Students will practice communication skills through peer feedback and presentations. Additionally, telling a story will enforce sequencing and retelling which are crucial skills to both programming and literacy.

ACTIVITIES

- Coding Story Time (30 min)
- Presentation Mode (5 min)
- Peer Feedback (10 min)
- Share Creations (15 min)

STUDENTS WILL BE ABLE TO...

- Program a story with a beginning, middle and end using a wide variety of blocks
- Give thoughtful and constructive peer feedback
- Accept feedback and think about how to improve their project

PREPARATION FOR TEACHERS

- Read through the Activity Guide
- Make sure all tablets are fully charged with ScratchJr downloaded

MATERIALS

FOR STUDENTS:

- Design Journal (see Appendix C for example)
 - Tablet with ScratchJr
- *See Appendix A for examples

VOCABULARY

Feedback — thoughts and suggestions about how to improve a piece of work

Lesson 12: Activities

CODING STORY TIME (30 min)



Have students take out their Design Journals and look back on their Story Maps from last class. They have already made their characters and setting, so now it's time to code the actual story! Encourage students to use all the blocks that they've learned throughout the curriculum and remember that their story must have a beginning, middle, and an end!

PRESENTATION MODE (5 min)

Once students are finished with their stories, gather them for a final meeting. Either having the students gather around a tablet, or by projecting the tablet onto the screen, show the students how to put their projects into presentation mode. Presentation mode is located in the upper toolbar of the screen. Tell the students that when we share our projects with each other, we will put the projects in presentation mode.



PEER FEEDBACK (10 min)

Pair students up in small groups or partners. Have children take turns sharing with each other their stories and then giving feedback. Make sure each student tells their partner one thing they liked and one thing that could be improved!

- Give examples of helpful feedback such as: "I like how you...."
- Discuss various problems students faced and how they fixed them.
- Do any students have tips to help their classmates solve problems they faced?



SHARE CREATIONS (15 min)

In groups, or as a whole class, have students go around in a circle and take turns sharing their stories with one another. Encourage them to narrate their stories as each event happens, making sure to point out the beginning, middle and end. As they share, prompt students to reflect on the project.

Guiding questions:

- What was your favorite part about programming your story?
- How was your story similar to Knuffle Bunny? How was it different?
- What would you add to your story if you had more time?

Appendix A. Materials

Appendix A. Materials

Technology Materials:

- Tablet with ScratchJr App downloaded
- Tablet charger
- Speakers for playing music

Art Materials:

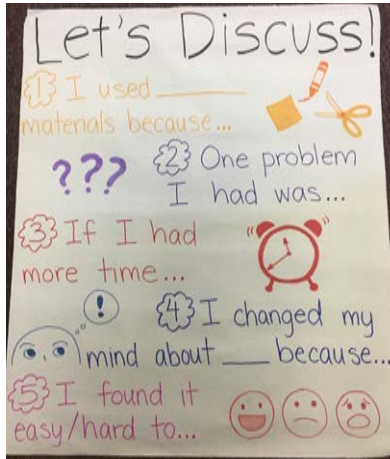
- Construction paper or other kind of decorative paper
- Markers, crayons, or colored pencils
- Masking tape

Teaching Materials:

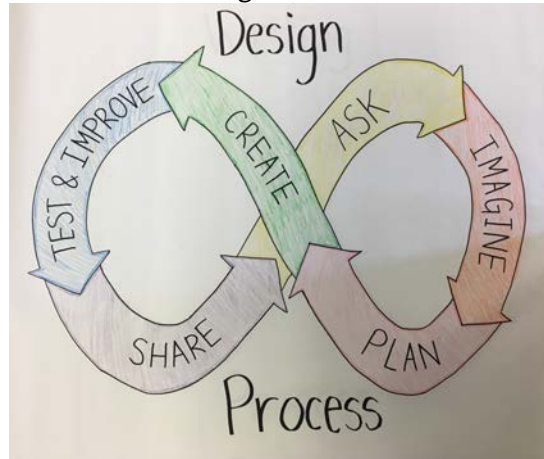
- 1 copy of *Knuffle Bunny* by Mo Willems (it might be helpful to have multiple copies for students to reference during projects)
- ScratchJr Stickers
- Premade anchor charts (**see following pages for examples**)
 - Discussion Sentence Starters
 - Design Process
 - Writing Process
 - Why is Kitten Confused?
 - How to Treat Our Materials
- Printed pictures (**see following pages for examples**)
 - Toothbrush Sequence Cards
 - Scratch Jr Block Cards

Examples of Anchor Charts:

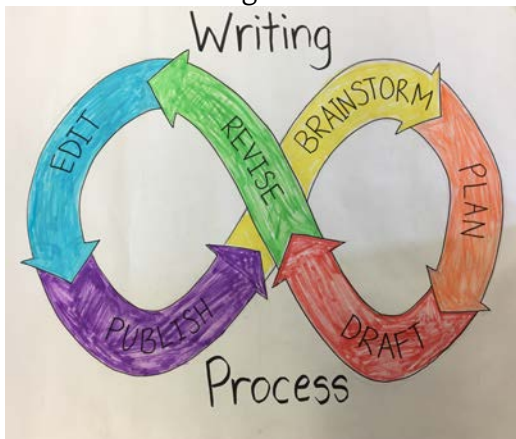
Discussion Sentence Starters



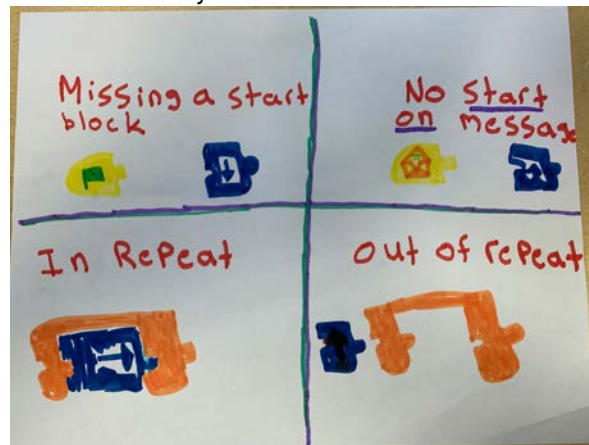
Design Process



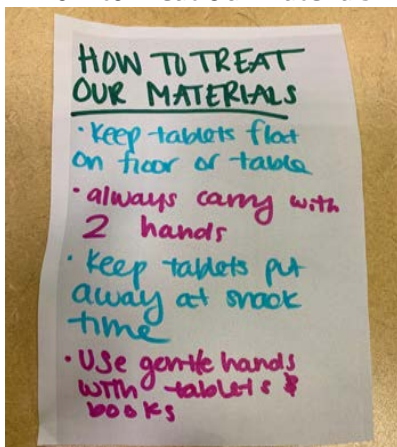
Writing Process















Why is Kitten Confused?



How to Treat Our Materials

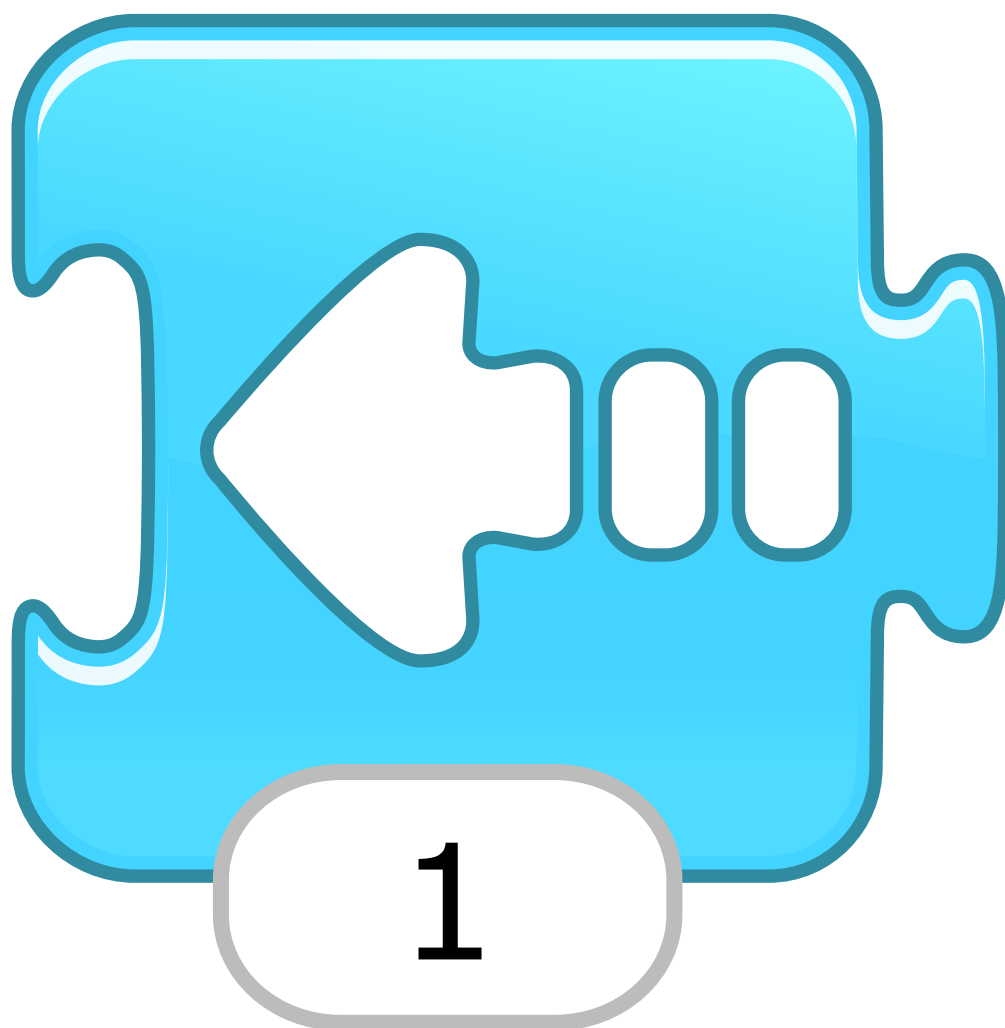


Examples of Printed Pictures:
Toothbrush Sequence Pictures

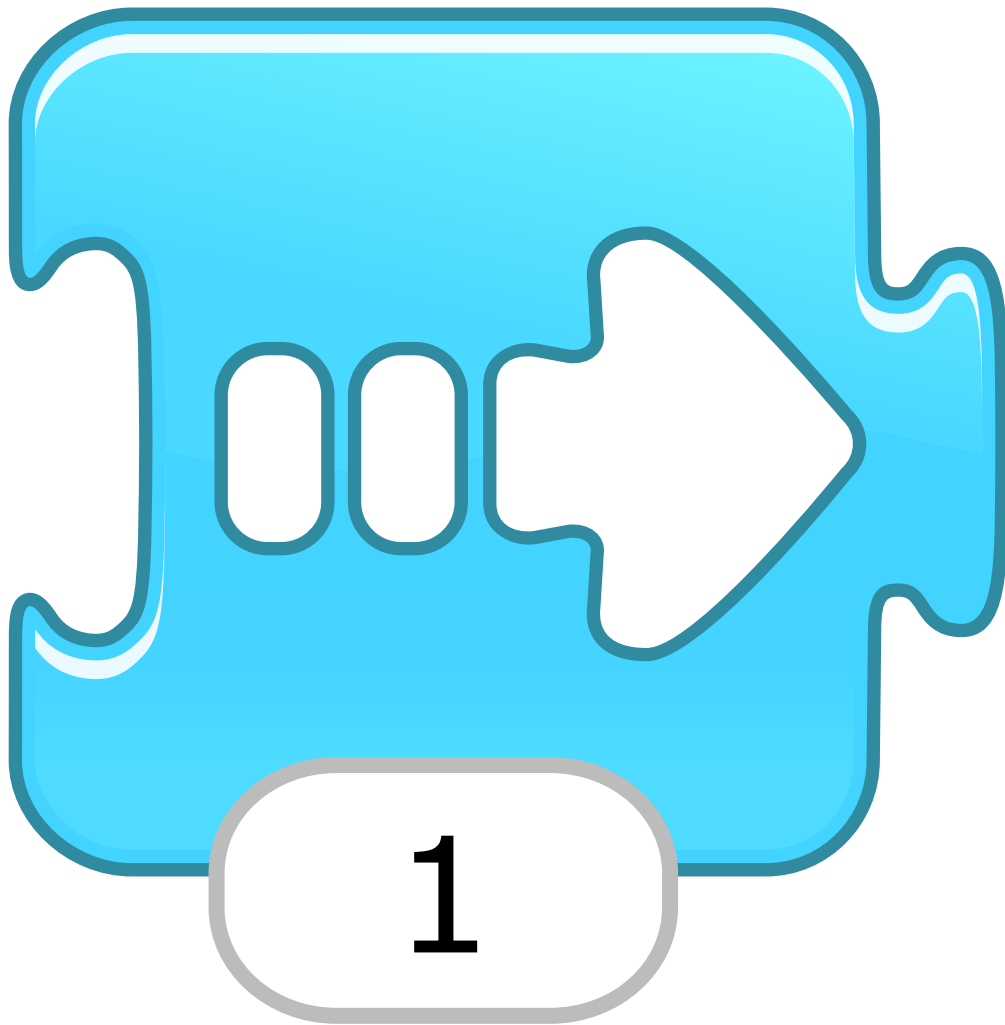
<p>turn on tap</p> 	<p>take toothbrush</p> 	<p>apply toothpaste</p> 
<p>rinse brush</p> 	<p>brush top</p> 	<p>brush bottom</p> 
<p>brush side</p> 	<p>brush side</p> 	<p>spit in sink</p> 
<p>rinse mouth</p> 	<p>rinse brush</p> 	<p>wipe face</p> 

MOTION BLOCKS

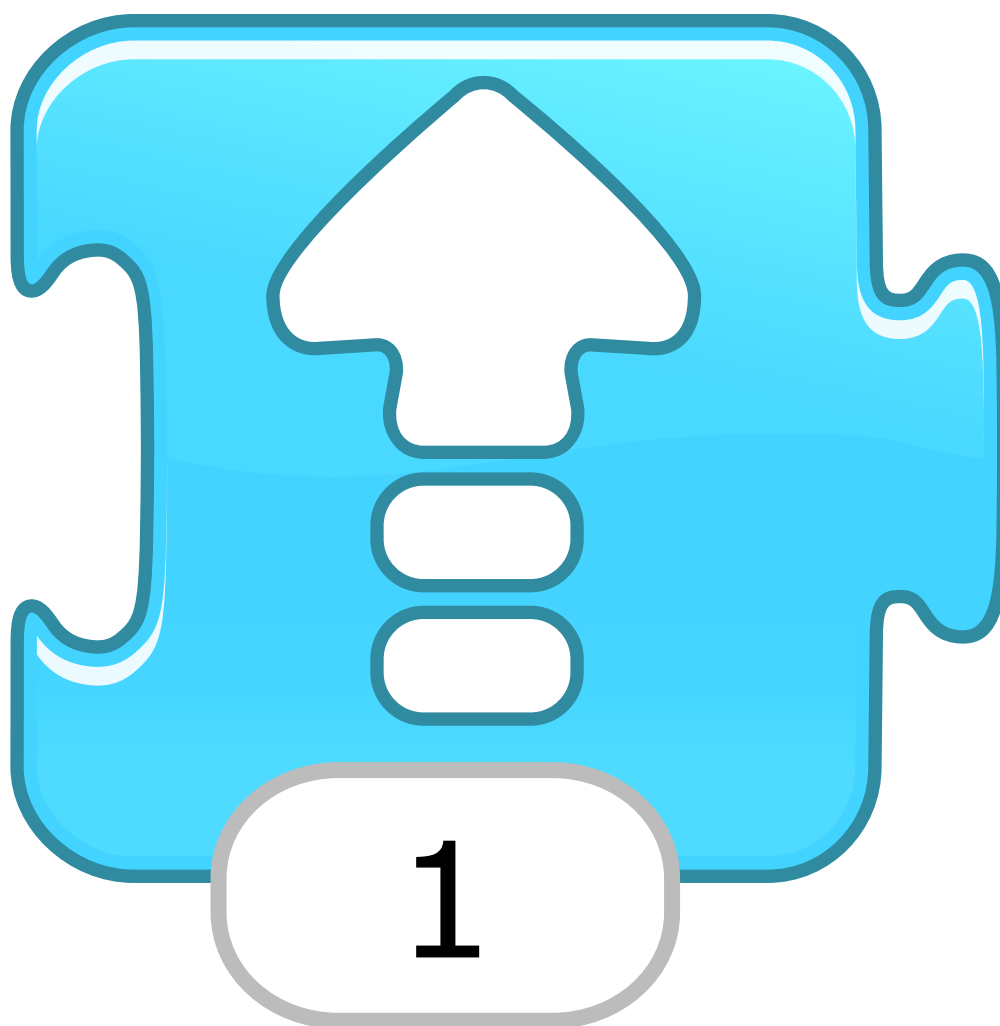
Move Left



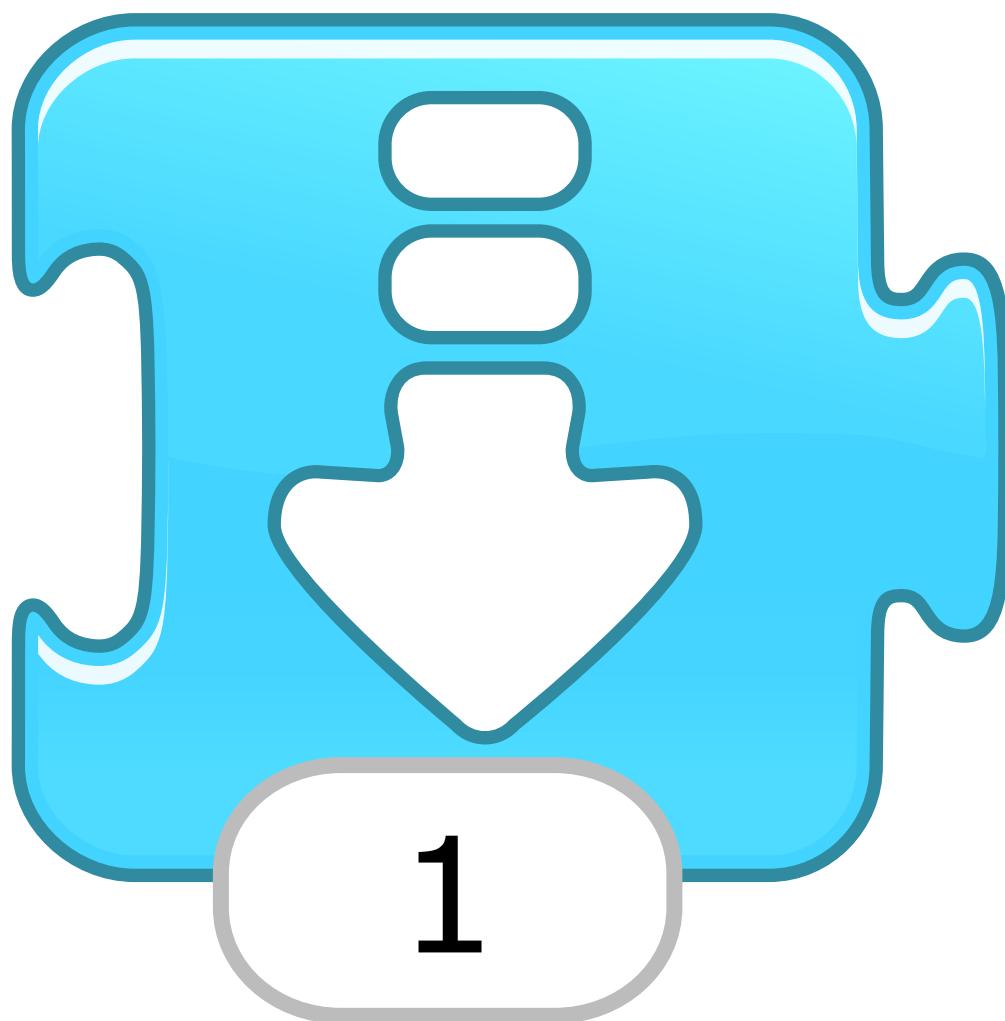
Move Right



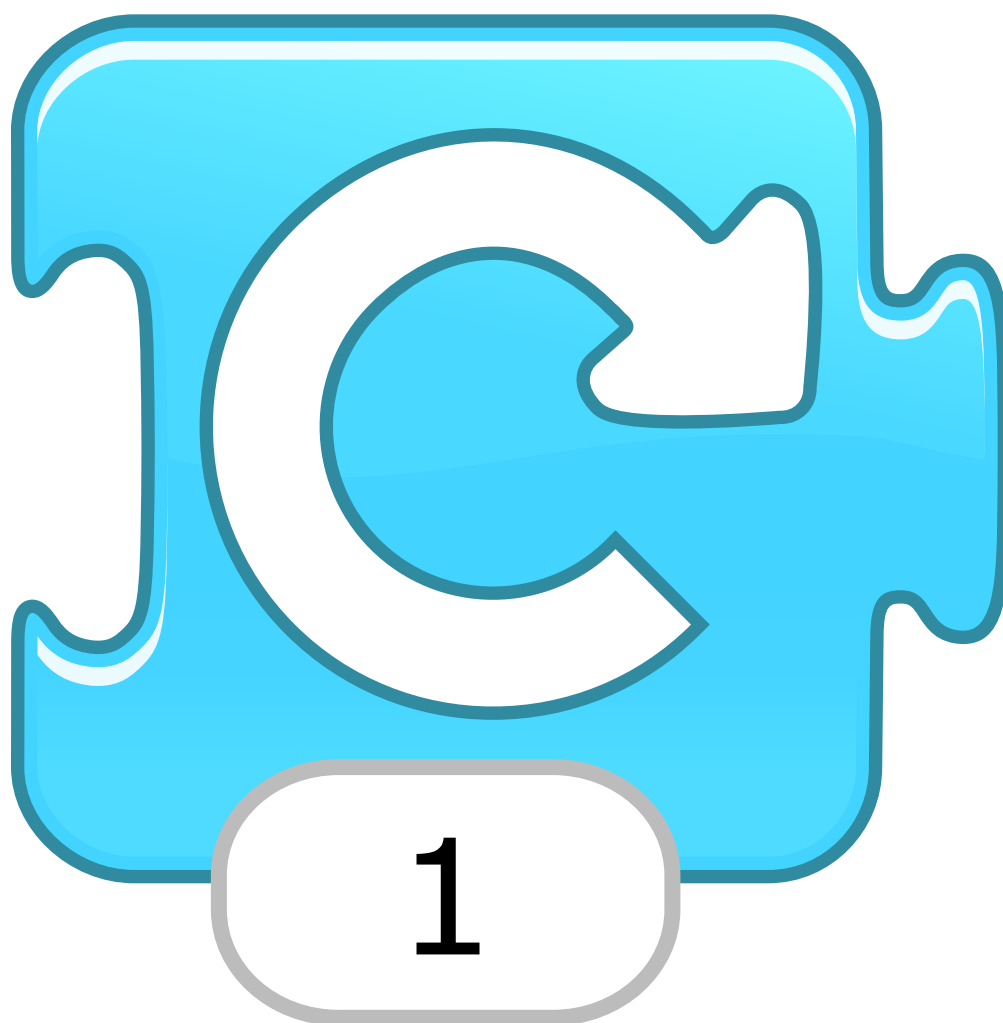
Move Up



Move Down



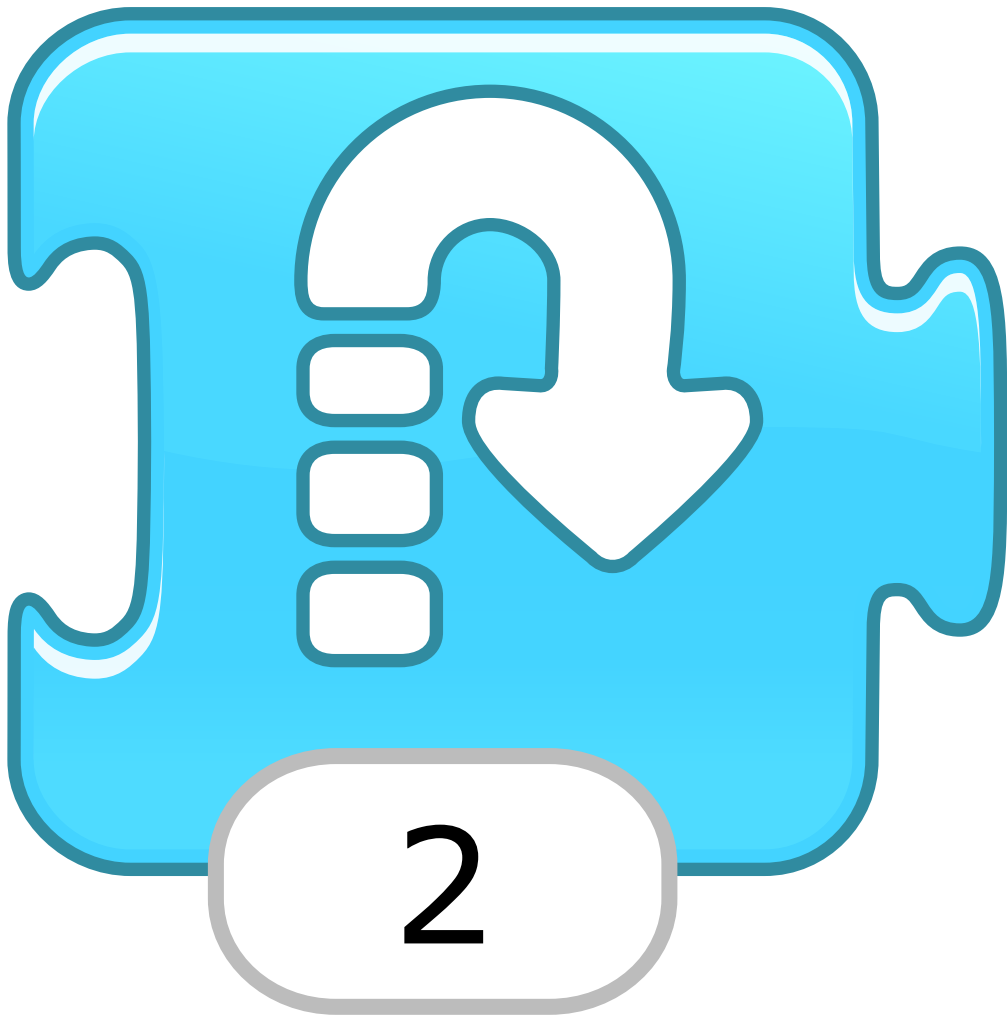
Turn Right



Turn Left



Hop

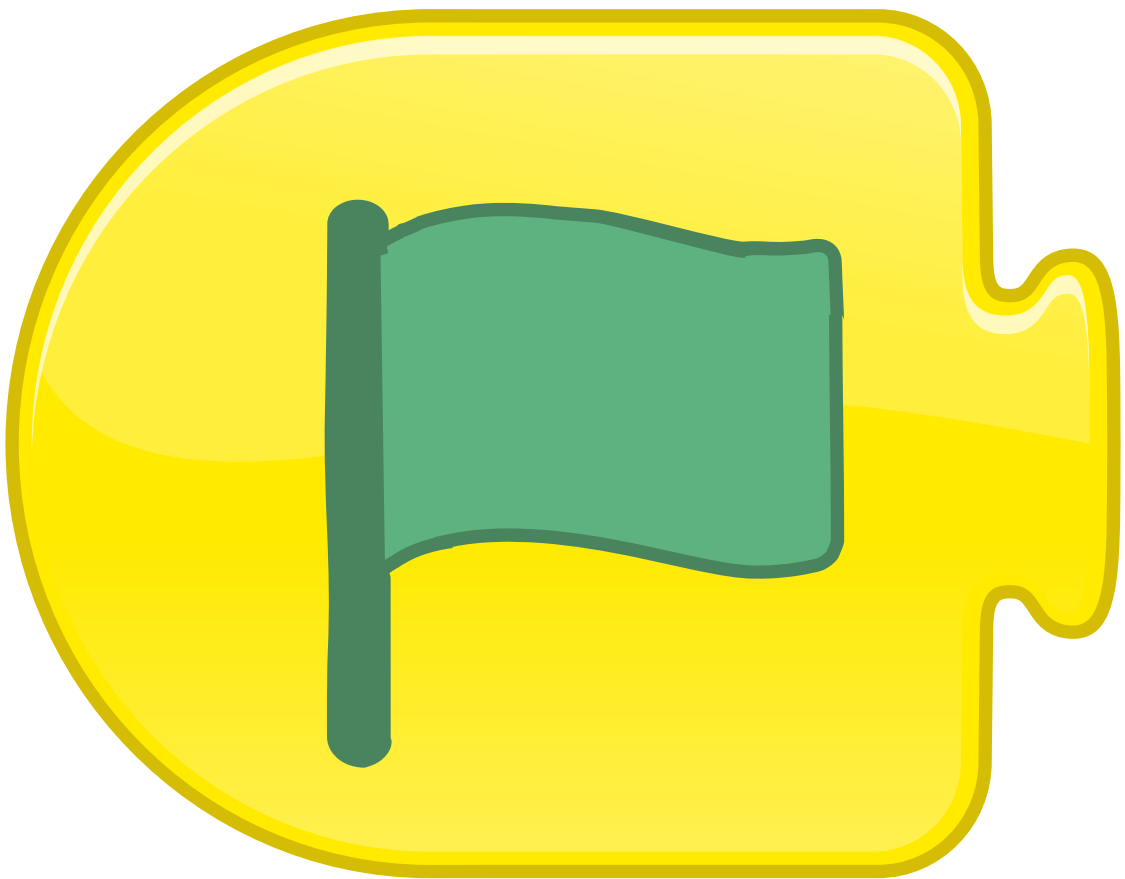


Go home

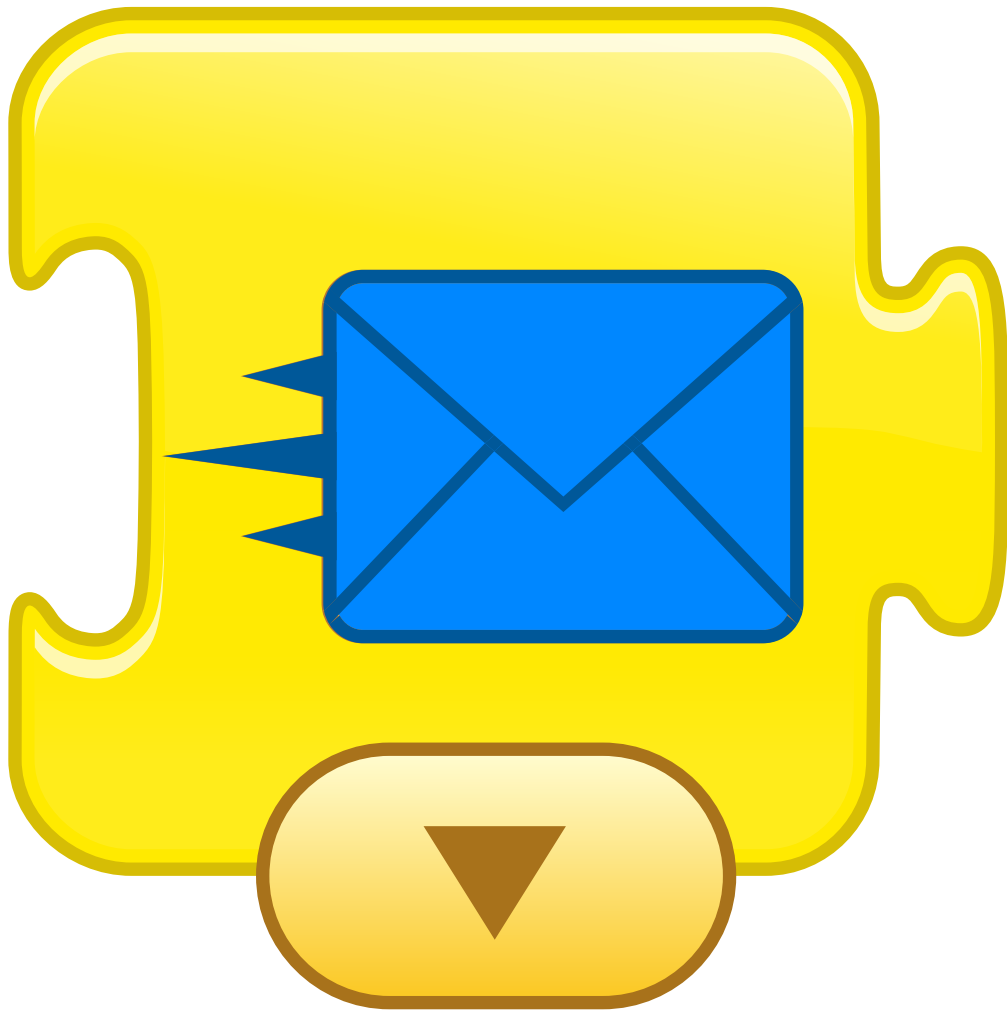


TRIGGERING BLOCKS

Start on Green Flag



Send Message



SOUND BLOCKS

Pop



Play Recorded Sound

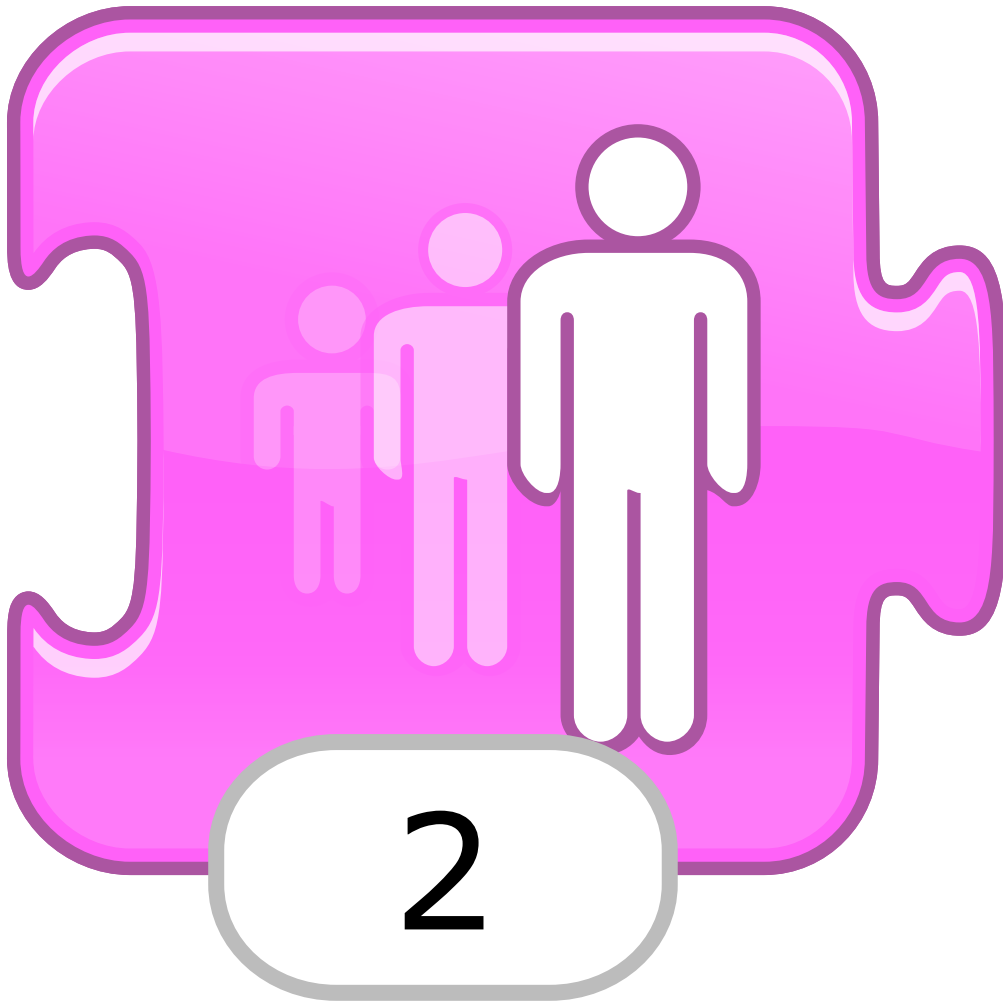


LOOKS BLOCKS

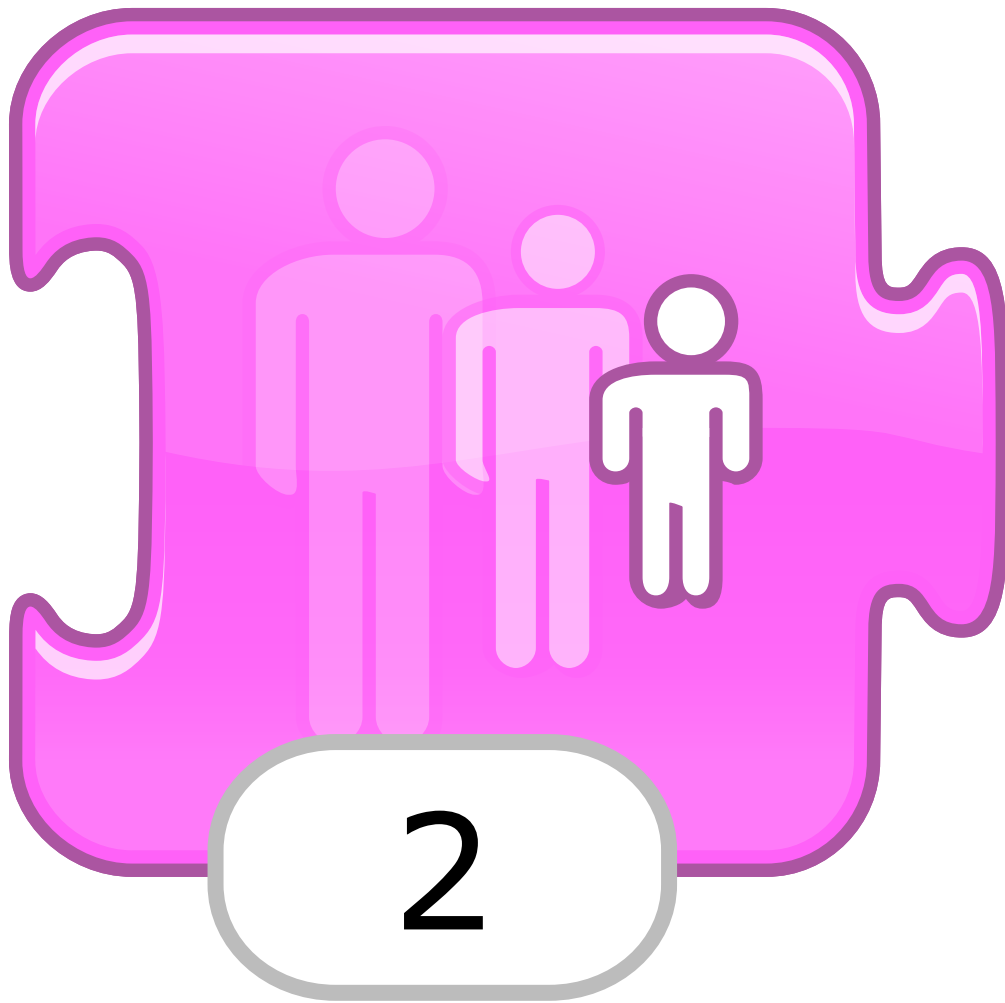
Say



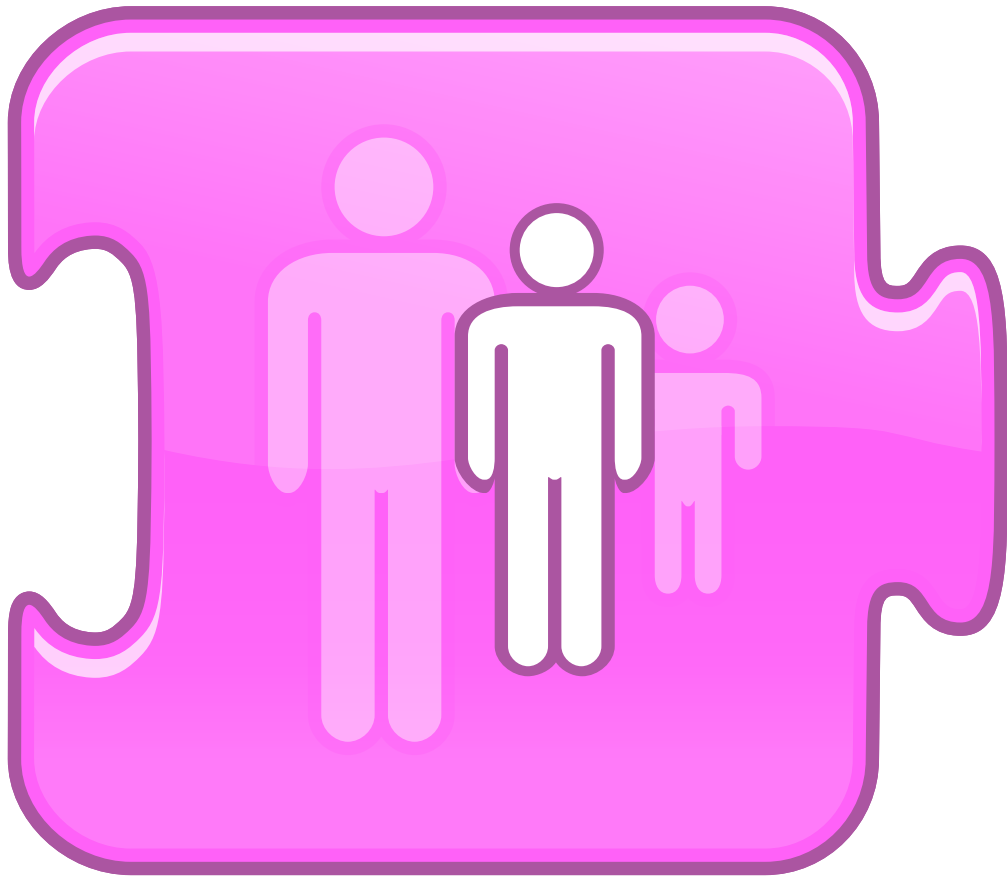
Grow



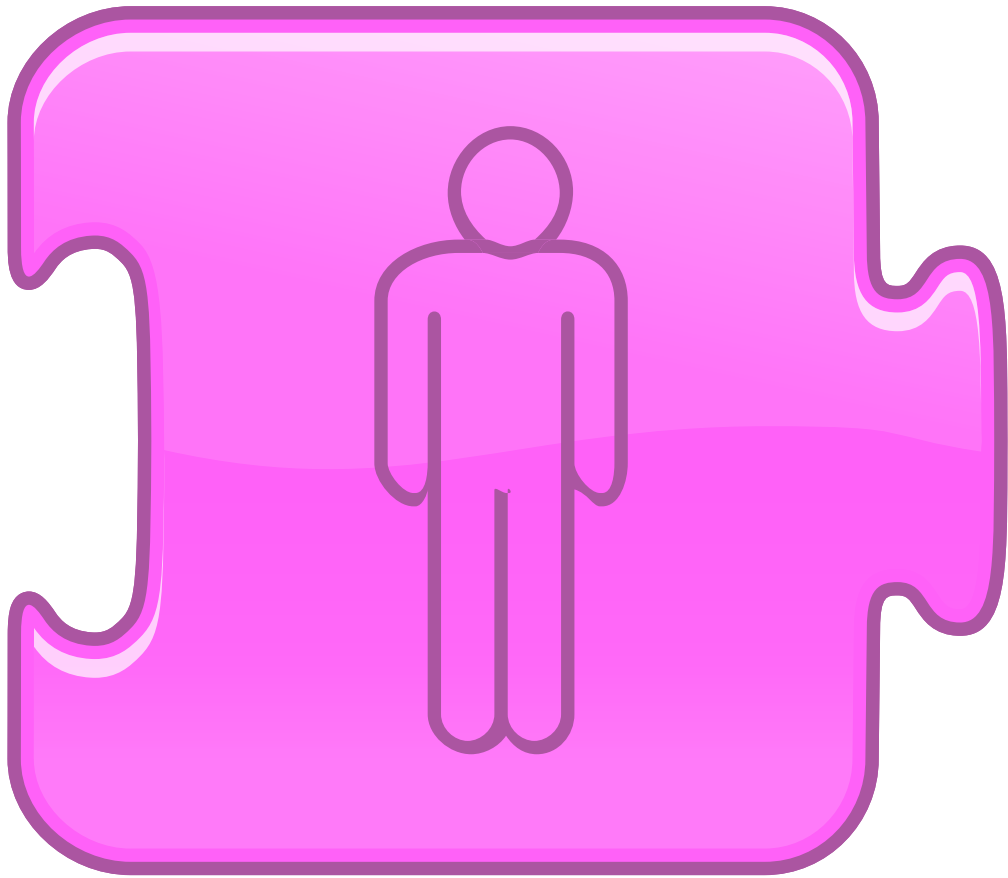
Shrink



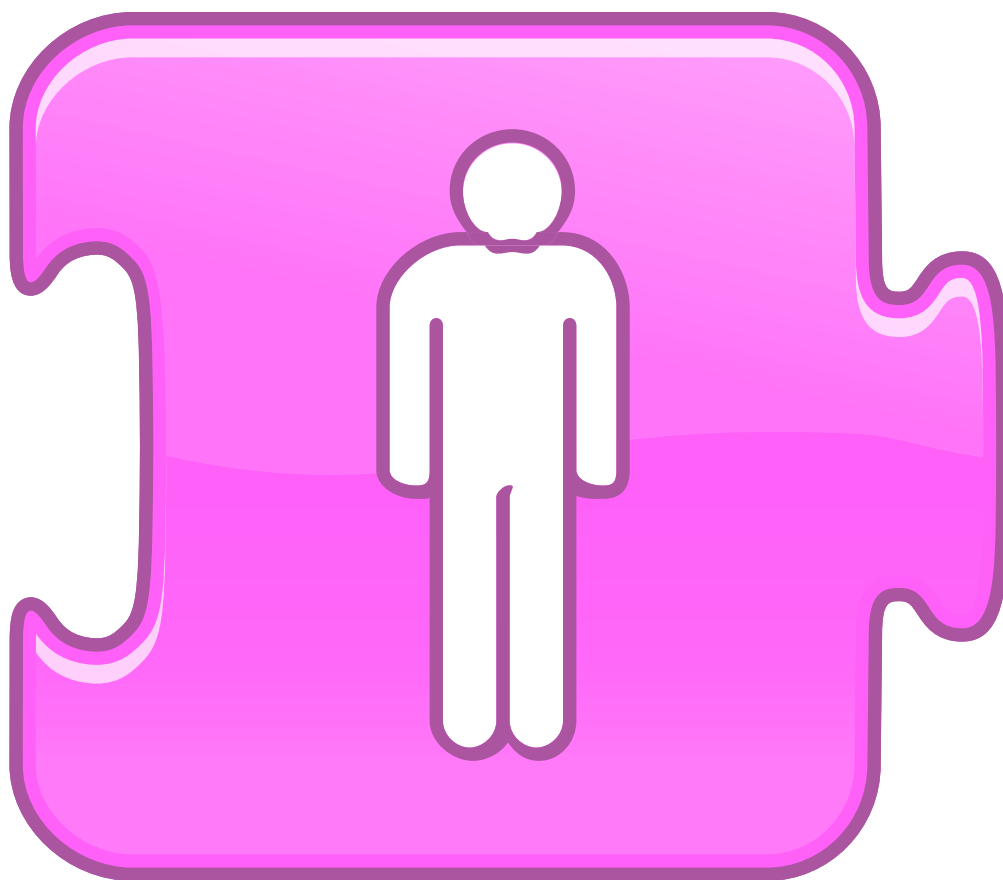
Reset size



Hide



Show



CONTROL BLOCKS

Wait



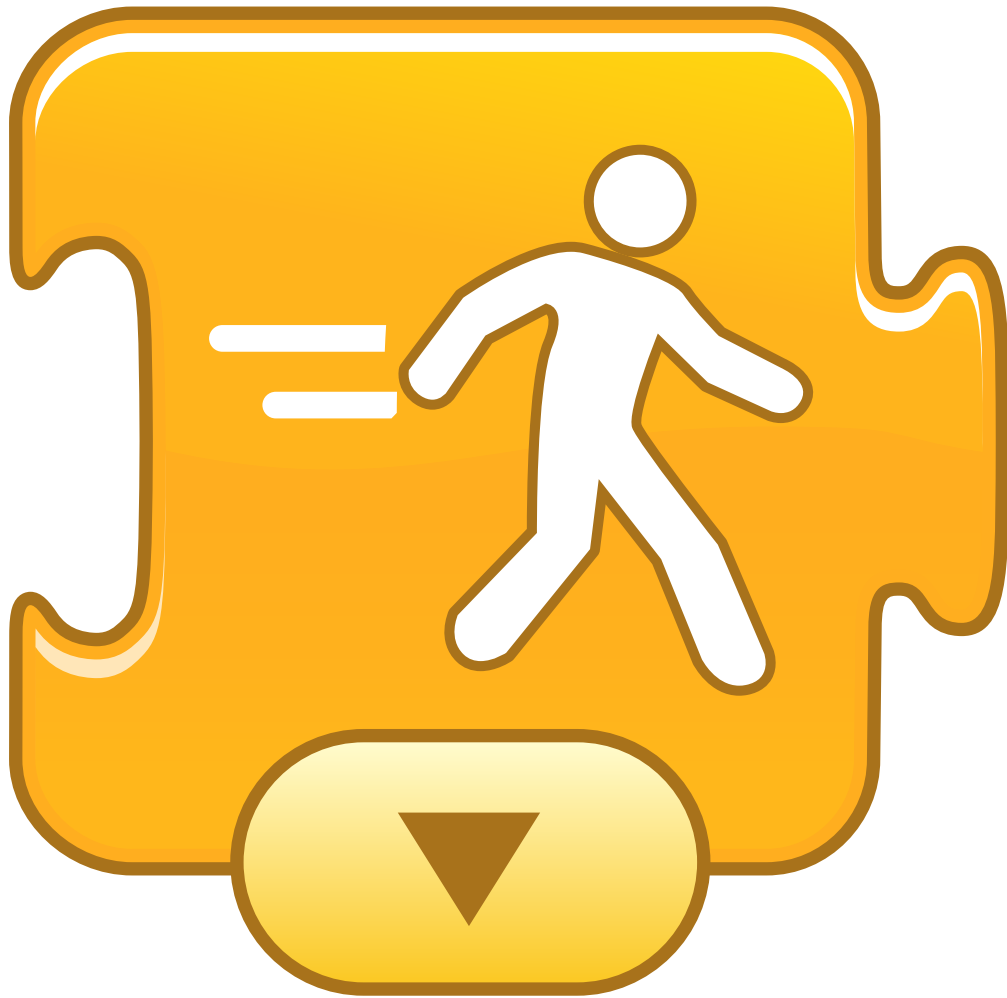
Stop



Set Speed



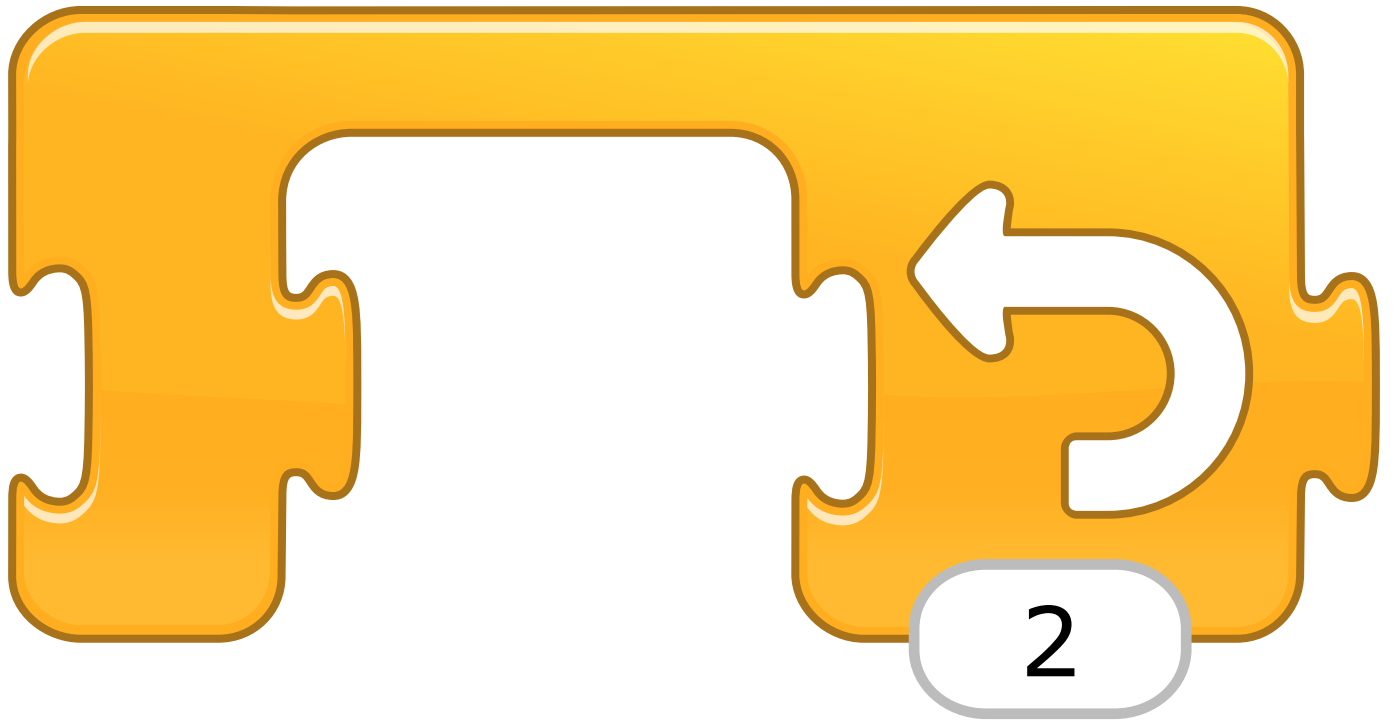
Set Speed



Set Speed

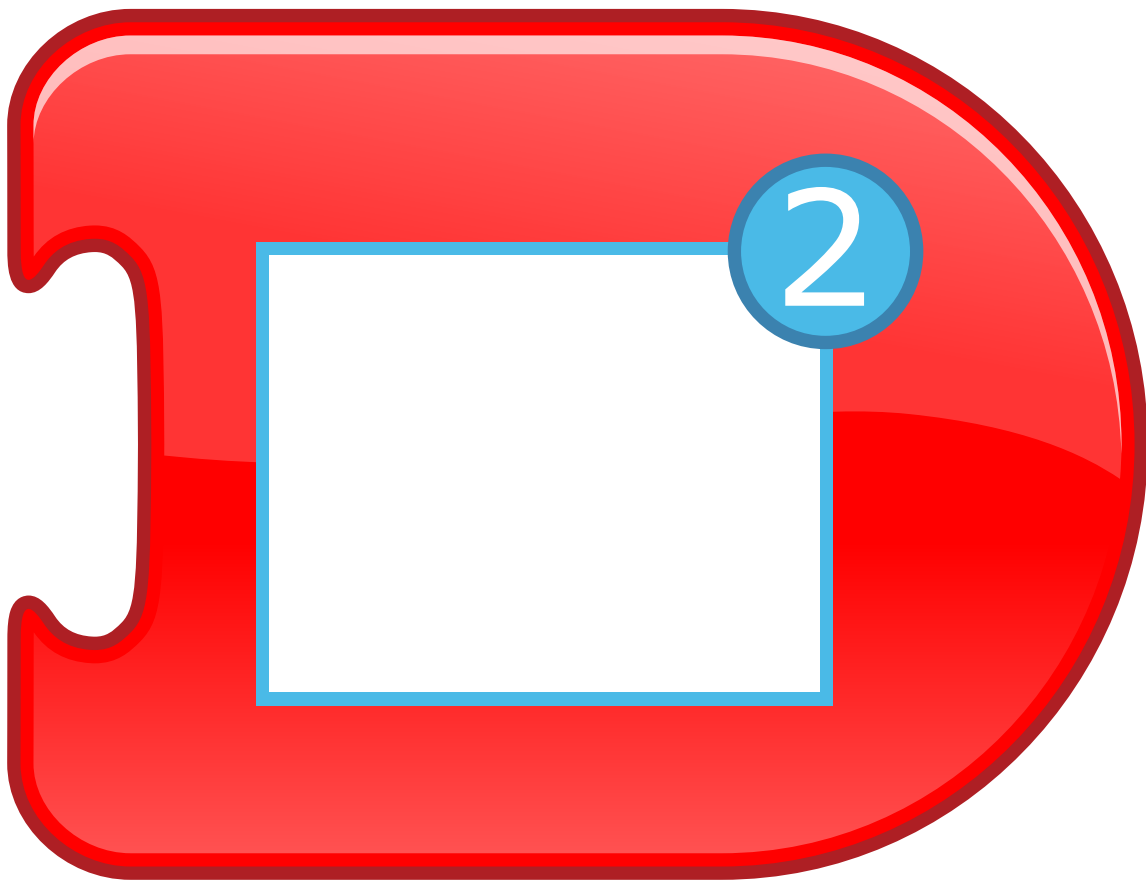


Repeat



END BLOCKS

Go to page



End



TRIGGERING BLOCKS

Start on Green Flag



Starts the script when the Green Flag is tapped.

Start on Tap



Starts the script when you tap on the character.

Start on Bump



Starts the script when the character is touched by another character.

Start on Message



Starts the script whenever a message of the specified color is sent.

Send Message



Sends a message of the specified color.

MOTION BLOCKS

Move Right



Moves the character a specified number of grid squares to the right.

Move Left



Moves the character a specified number of grid squares to the left.

Move Up



Moves the character a specified number of grid squares up.

Move Down



Moves the character a specified number of grid squares down.

Turn Right



Rotates the character clockwise a specified amount. Turn 12 for a full rotation.

Turn Left



Rotates the character counterclockwise a specified amount. Turn 12 for a full rotation.





Block Descriptions

Hop



Moves the character up a specified number of grid squares and then down again.

Go Home



Resets the character's location to its starting position. (To set a new starting position, drag the character to the location.)

LOOKS BLOCKS

Say



Shows a specified message in a speech bubble above the character.

Grow



Increases the character's size.

Shrink



Decreases the character's size.

Reset Size



Returns the character to its default size.

Hide



Fades out the character until it is invisible.

Show



Fades in the character until it is fully visible.

SOUND BLOCKS

Pop



Plays a "Pop" Sound

Play Recorded Sound



Plays a sound recorded by the user.



This work is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-sa/4.0/).

CONTROL BLOCKS

Wait



Pauses the script for a specified amount of time (in tenths of seconds).

Set Speed



Changes the rate at which certain blocks are run.

Stop



Stops all the characters' scripts.

Repeat



Runs the blocks inside a specified number of times.

END BLOCKS

End



Indicates the end of the script (but does not affect the script in any way).

Go to Page



Changes to the specified page of the project.

Repeat Forever



Runs the script over and over.

Appendix B-1. Solve-It Assessment A

Name _____

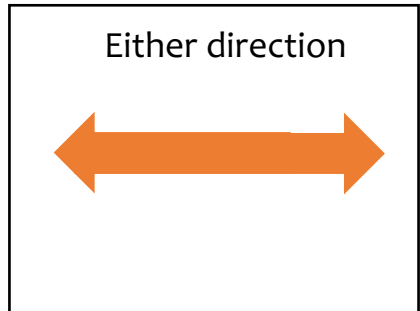
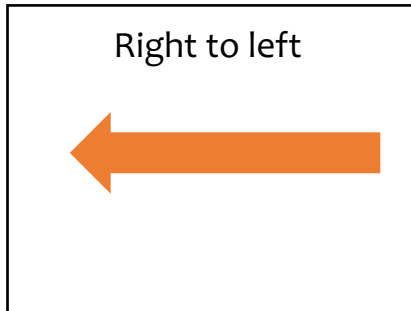
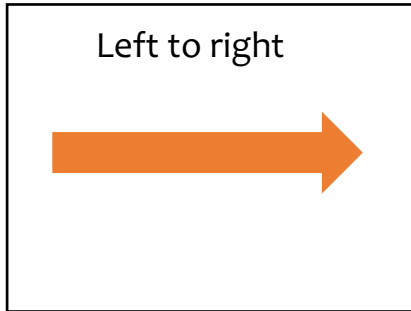
Date _____

A

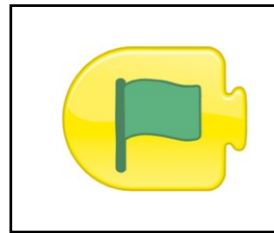
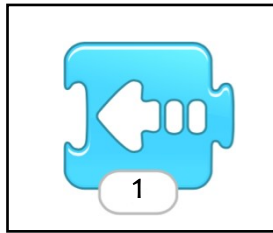
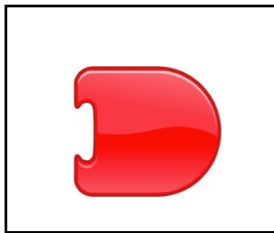
Teacher _____

Circle the correct answers.

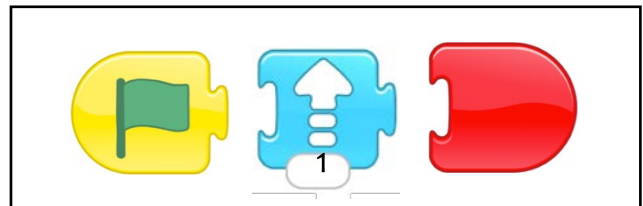
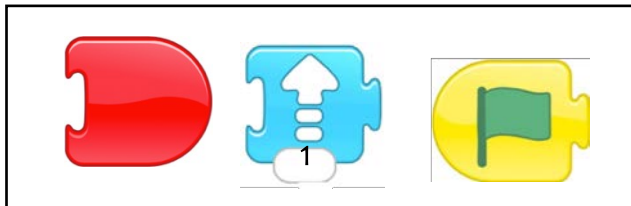
1. What direction do you read a program?



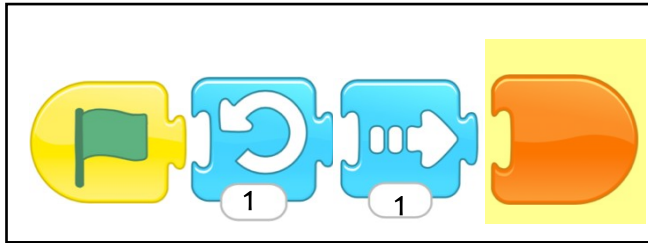
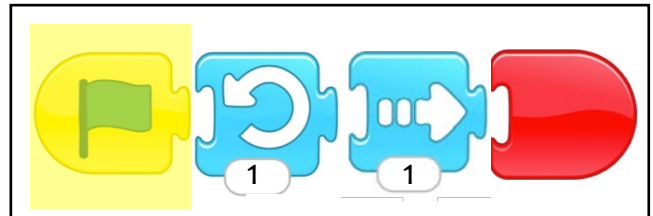
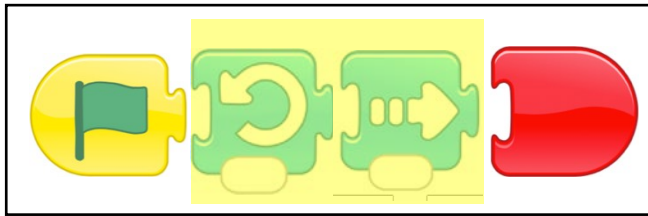
2. Which block lets you know when it is the end of a program?



3. Which of these is a program?



4. What is the middle of this program?



NONE

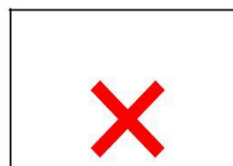
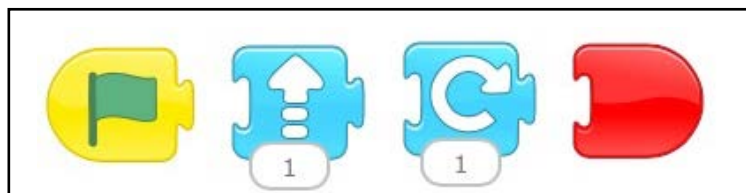
5. Which of these things would you need a design process for?

Keeping things the same

Making new things

Breaking something

6. True or False: This program makes your character go up and then spin.



Appendix B-2. Solve-It Assessment B

Solve-it

Name _____

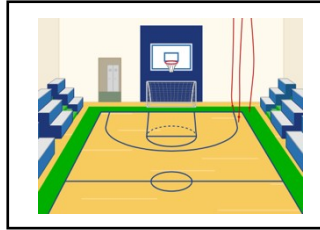
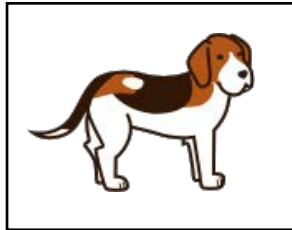
Date _____

B

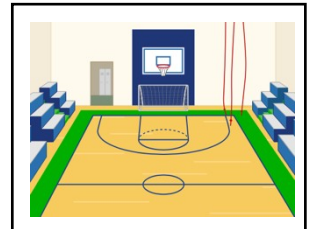
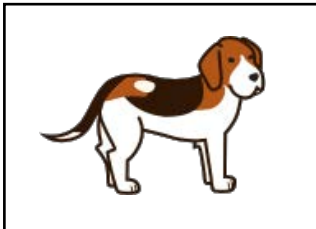
Teacher _____

Circle the correct answers.

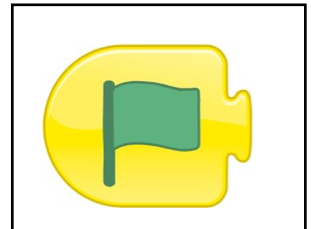
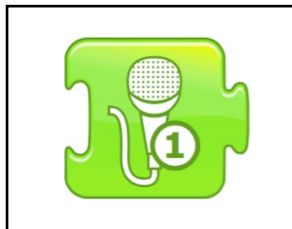
1. Which one is a character?



2. Which one is a setting?



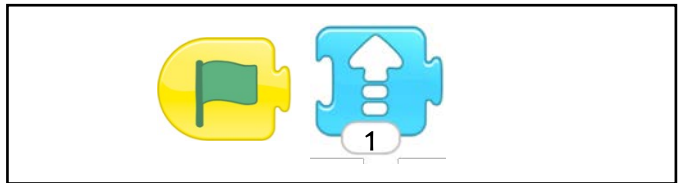
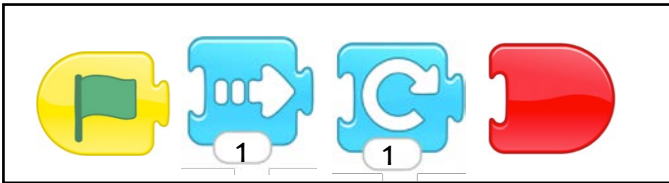
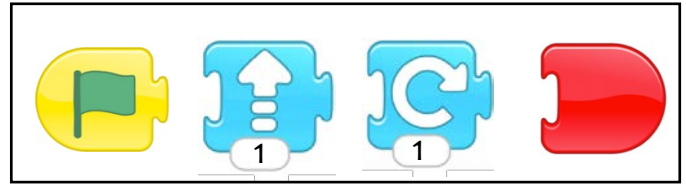
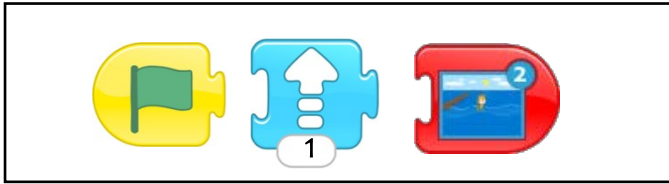
3. Which block lets you record your own sound?



4. True or False: This block makes your character get smaller.



5. Which one of these programs will make the character go to the next page after moving up?



6. Which of these lets you change the setting on ScratchJr?



Appendix B-3. Solve-It Assessment C

Solve-it

Name _____

Date _____



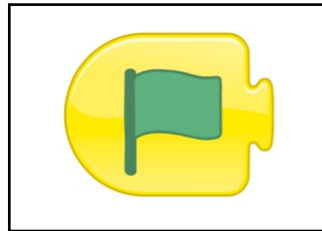
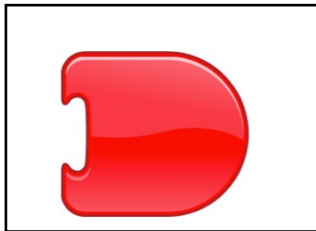
Teacher _____

Circle the correct answers.

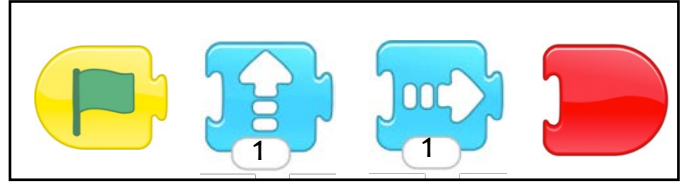
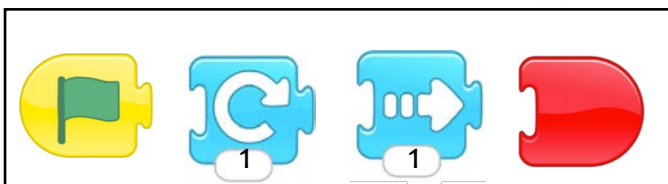
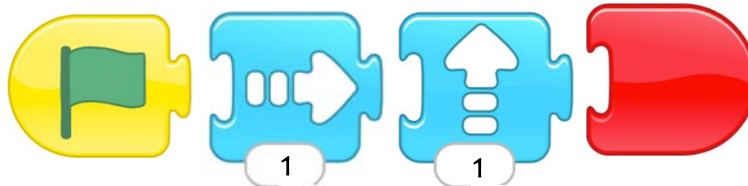
1. My character won't spin and move forward when I try to run my program by pressing the green flag.



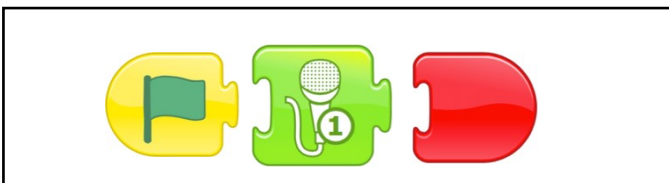
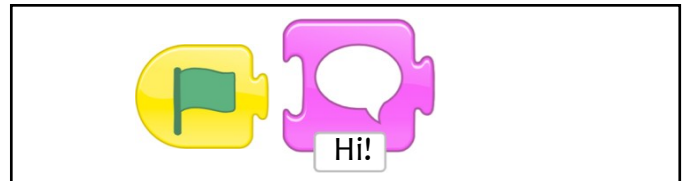
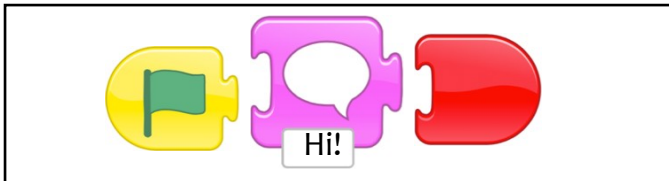
Circle one block that is needed to debug this program:



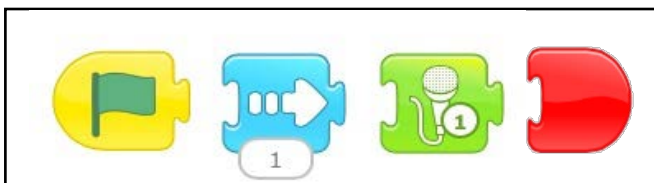
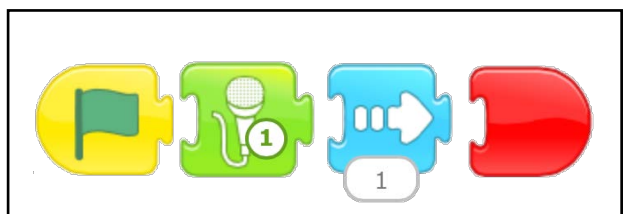
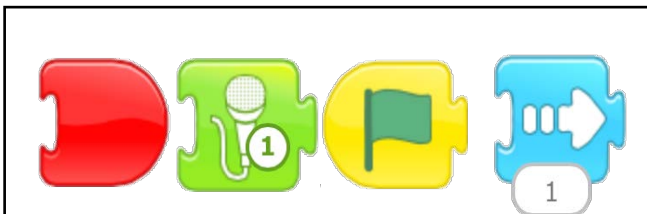
2. How can we change this program so that the character goes up before moving forward?



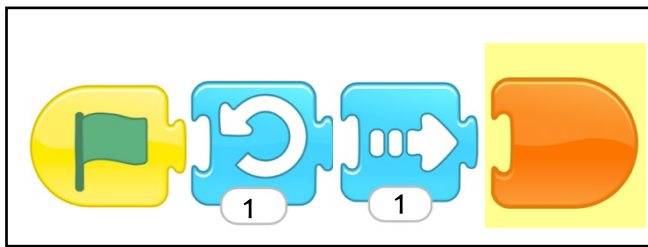
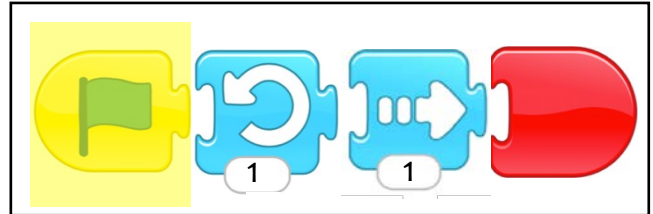
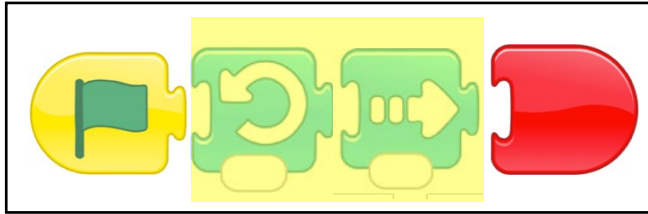
3. How can we change this program so my recorded sound plays?



4. How can we change this program so that it plays a sound and then goes right?

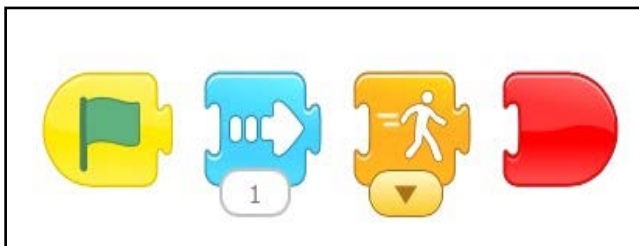


5. What is the beginning of this program?



NONE

6. My character is moving too slowly. How can I make it move faster?



Appendix B-4. Solve-It Assessment D

Solve-it

Name _____

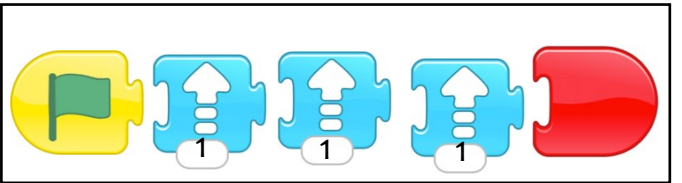
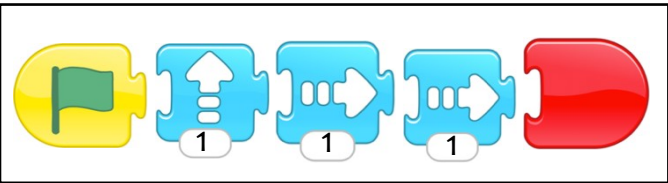
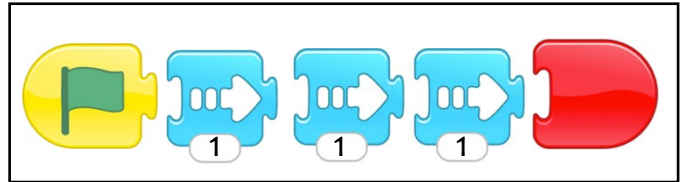
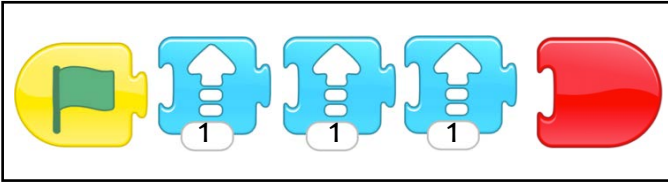
Date _____

D

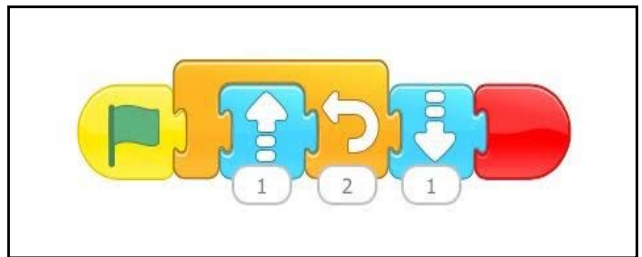
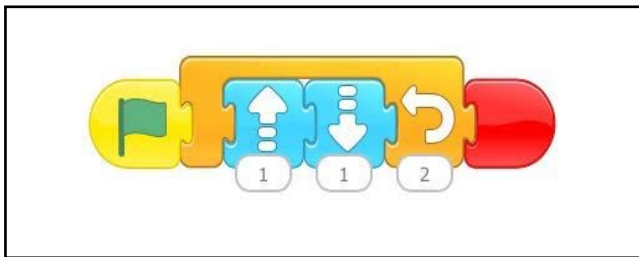
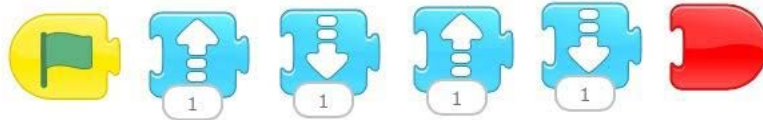
Teacher _____

Circle the correct answers.

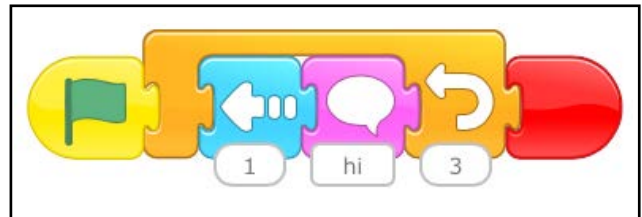
1. My character moved 3 steps backward. Now, how do I move it back to its original spot?



2. What is another way I can write my program to make my character do the same thing?



3. How can I make my character move backwards and then make a pop sound 3 times?



NONE

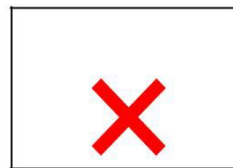
4. Which of these makes my character wait 10 seconds before doing the next instruction in the program?



5. I want my character to wait 5 seconds before playing my recorded sound. How can I do this?

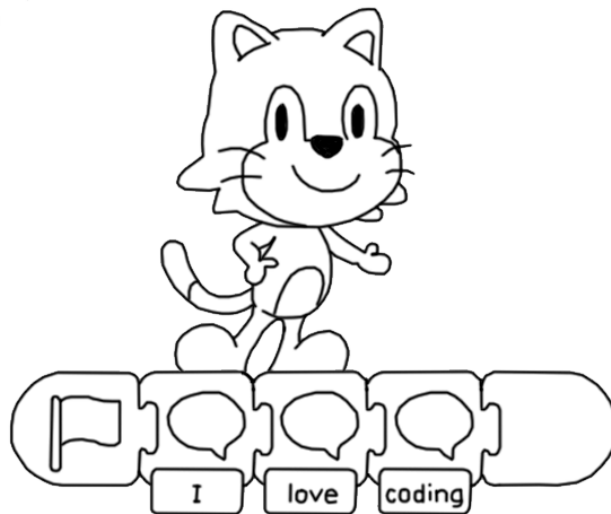
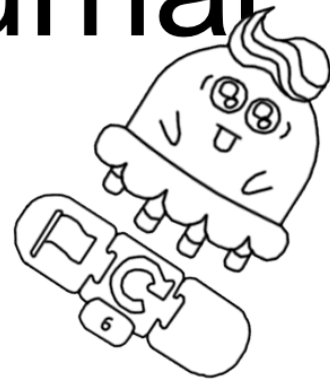
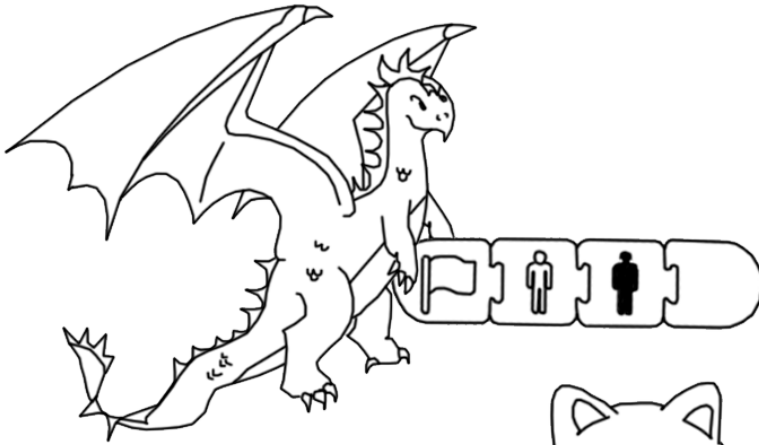


6. True or False: My character will move forwards and backwards three times, and then disappear.



Appendix C. Design Journal

_____ 's
Design Journal

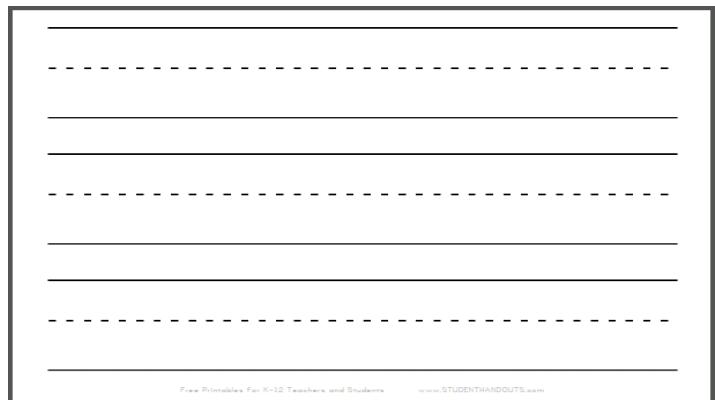
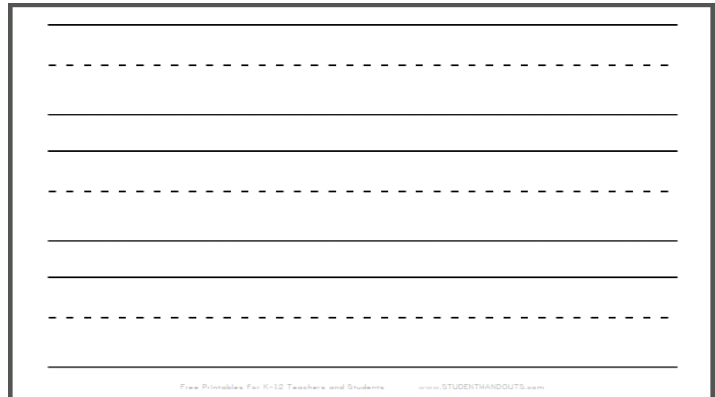
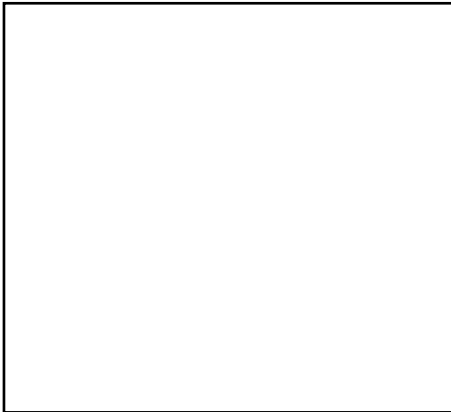


Name: _____

Lesson 1: Foundations

How To Worksheet

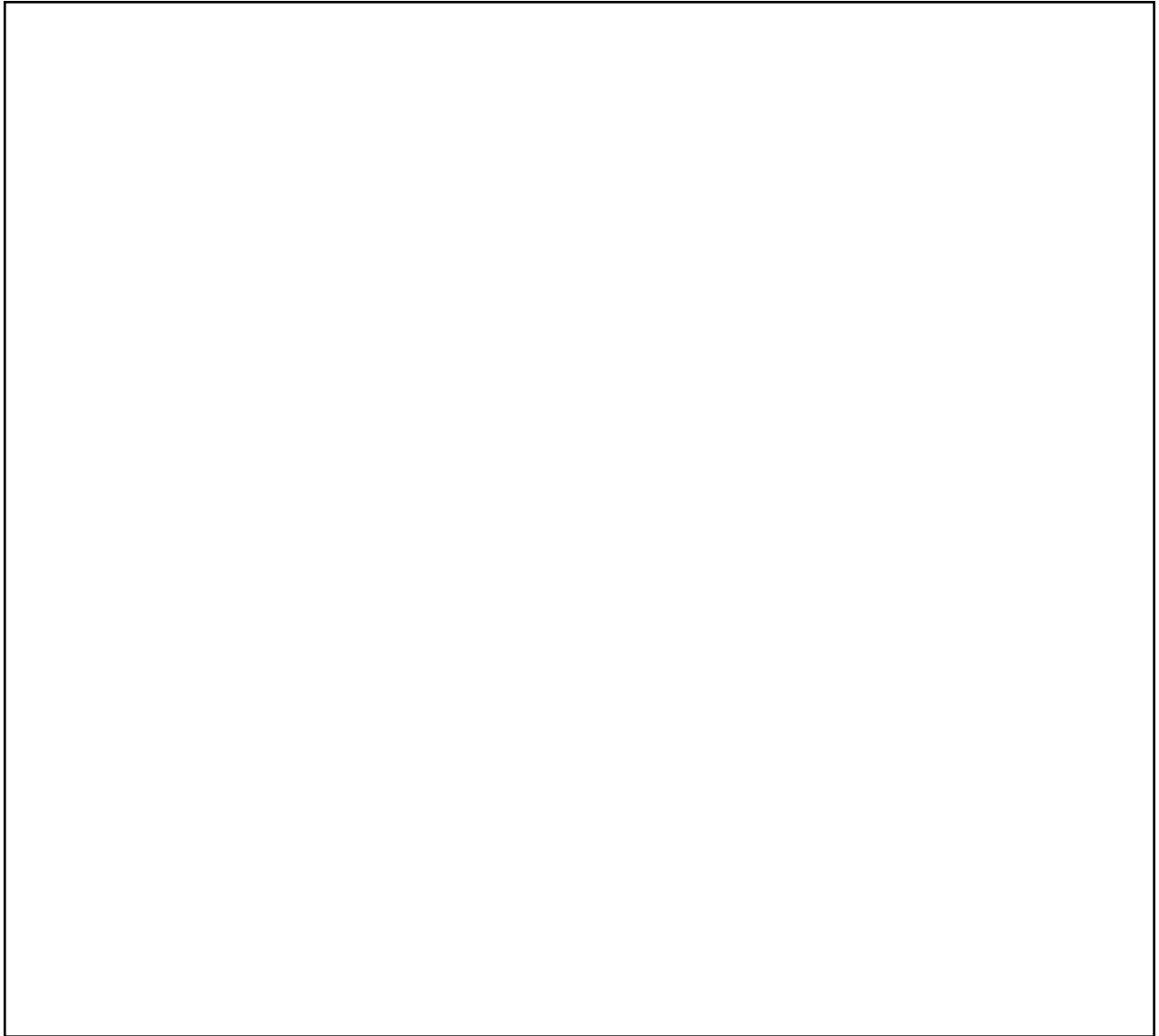
How To _____



Name: _____

Lesson 5: Characters

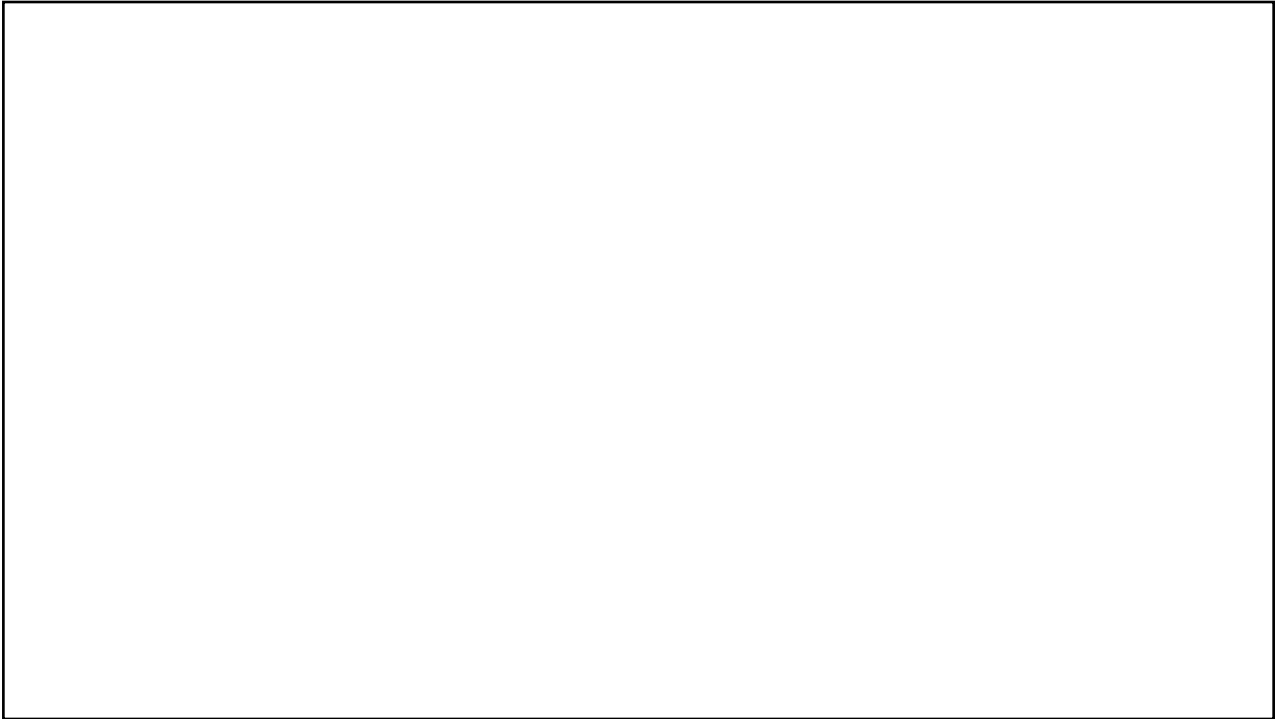
Design Your Own Character



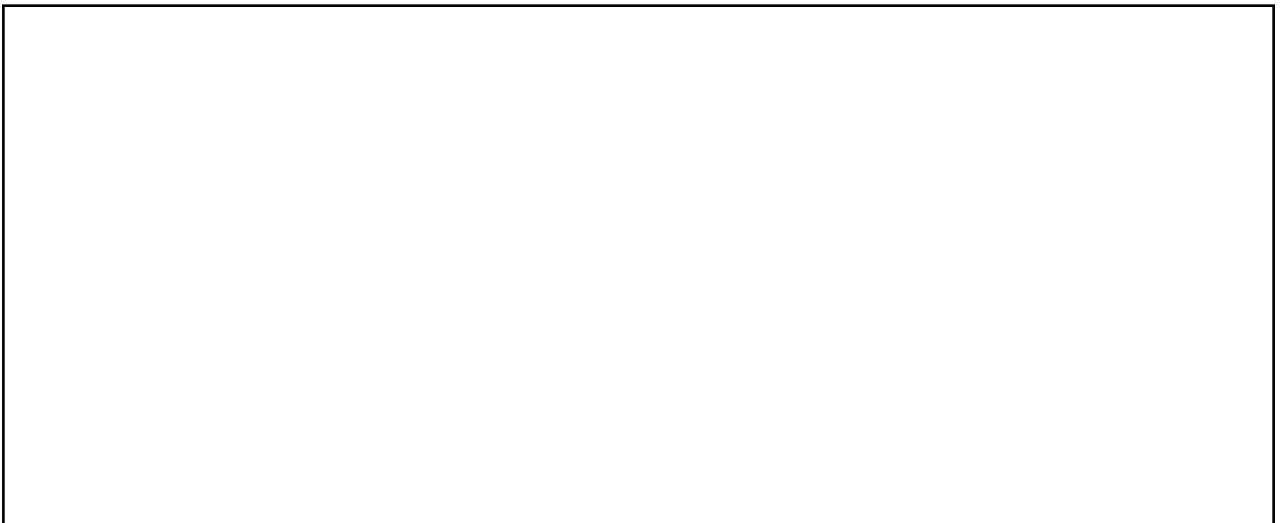
Name: _____

Lesson 7: Programming

Use Scratch Jr stickers to write out your Hokey-Pokey program:



What was the hardest part of your program?



Name: _____

Lesson 8: Debugging



What was one problem you had with Scratch JR?



What were some things you tried to help solve the problem?



Which solution worked best?

Name: _____

Lesson 9: Details

1.  **OR** 

2.  **OR** 

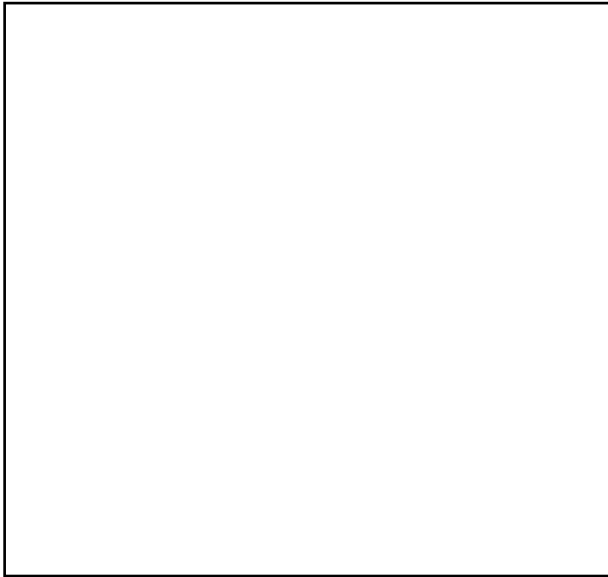
3.  **OR** 

4.  **OR** 

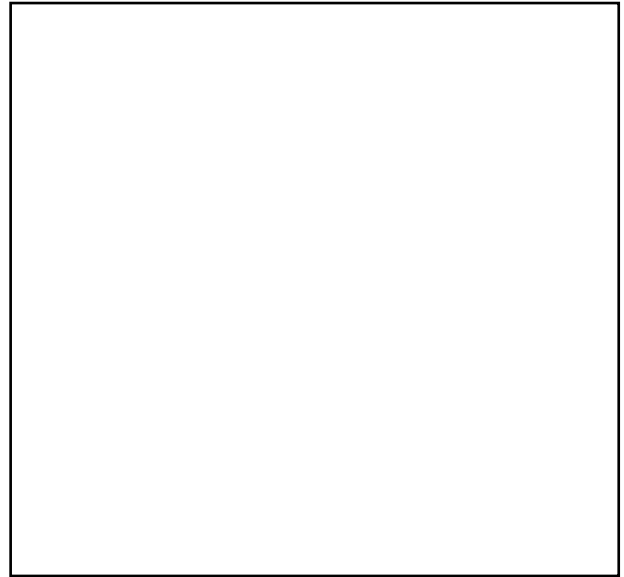
Name: _____

Lesson 11: Story Time Story Map

Setting



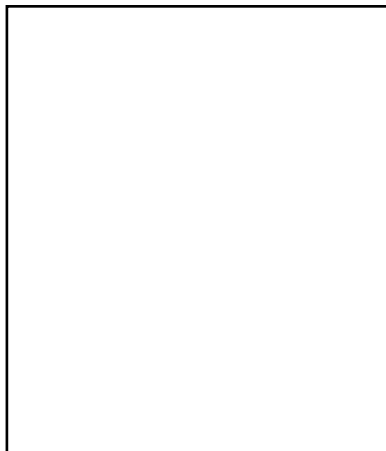
Characters



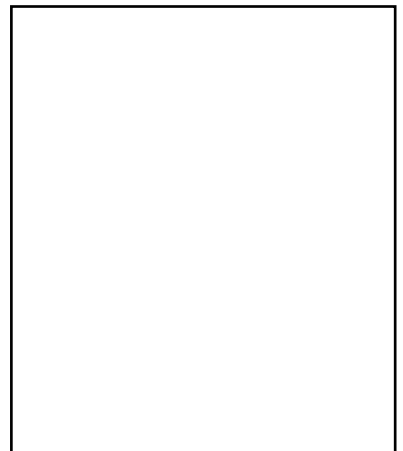
Beginning



Middle



End



References

- Bers, M. U. (2008). *Blocks to robots: Learning with technology in the early childhood classroom*. New York, NY: Teachers College.
- Bers, M. U. (2012). *Designing digital experiences for positive youth development: From playpen to playground*. Cary, NC: Oxford.
- Bers, M. U. (2018). *Coding as a Playground: Programming and Computational Thinking in the Early Childhood Classroom*. New York, NY: Routledge Press.
- Darragh, J. C. (2006). *The environment as the third teacher*. Retrieved from <http://www.eric.ed.gov/PDFS/ED493517.pdf>
- International Society for Technology in Education (2017). *ISTE Standards for Students*. Retrieved from <https://www.iste.org/standards/for-students>
- K–12 Computer Science Framework. (2016). Retrieved from <http://www.k12cs.org>.
- Massachusetts Department of Education (2016). *Digital Literacy and Computer Science Framework*. Retrieved from <https://www.doe.mass.edu/frameworks/dlcs.pdf>
- National Governors Association Center for Best Practices & Council of Chief State School Officers (2010). *About the Standards*. Retrieved from <http://www.corestandards.org/about-the-standards/>
- Papert, S. (1993). *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books.
- Virginia Department of Education (2017). *Computer Science Standards of Learning (SOL)*. Retrieved from http://www.doe.virginia.gov/testing/sol/standards_docs/computer-science/index.shtml
- Vygotsky, L. S. (2012). *Thought and Language*. Cambridge, MA: The MIT Press.



© 2018, DevTech Research Group at Tufts University.