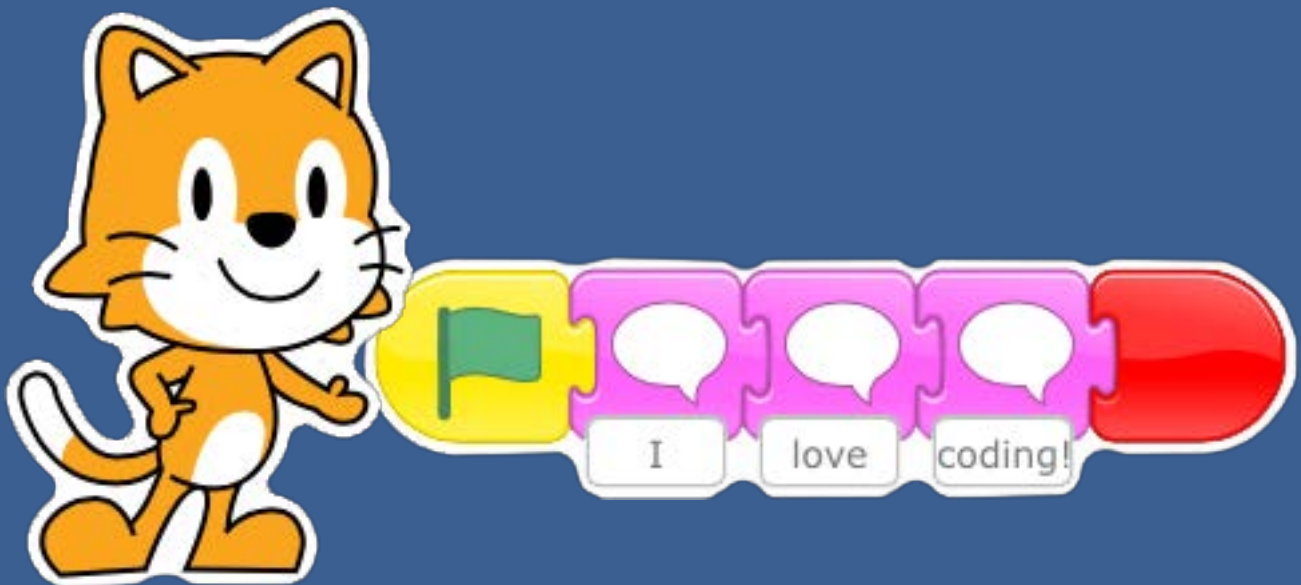


A ScratchJr Coding Curriculum for Readers

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.



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: Sequencing

Lesson 4: Characters

Lesson 5: Programming

Lesson 6: Debugging

Lesson 7: Details

Lesson 8: Repeat Loops

Lesson 9: Descriptive Language

Lesson 10: Conditionals

Lesson 11: Final Project - Writing the Jungle Dance Party

Lesson 12: Final Project - Coding the Jungle Dance Party

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

Appendix D: Collaboration Web

References

Introduction

CODING AS LITERACY (CAL) APPROACH

This curriculum introduces powerful ideas from computer science, specifically programming with ScratchJr to 2nd grade 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 *Giraffes Can't Dance* by Giles Andreae and Guy Parker-Rees.

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.

Table 1: Pacing Guide

| Lesson | Activities |
|------------------------------|--|
| Lesson 1: Foundations | <ul style="list-style-type: none"> • What is a programmer? (20 min) • Programmers and Writers (10 min) • Think Like a Programmer (10 min) • How to Write a Program (20 min) |
| Lesson 2: What is a Program? | <ul style="list-style-type: none"> • What is a Program? (5 min) • Tools of Communication (15 min) • Programmer Says (15 min) • Program the Teacher with ScratchJr Blocks (10 min) • Meet the ScratchJr App (15 min) |
| Lesson 3: Sequencing | <ul style="list-style-type: none"> • Giraffes Can't Dance (15 min) • Order Matters (10 min) • Composition Planning (15 min) • First Draft (20 min) |

| | |
|---|---|
| Lesson 4: 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) • Change Setting in ScratchJr (5 min) • Add Page in ScratchJr (5 min) • Character Share (10 min) |
| Lesson 5: Programming | <ul style="list-style-type: none"> • Dance the Hokey-Pokey (5 min) • Program the Hokey-Pokey (20 min) • Hokey-Pokey Reflection (10 min) • Share Creations (10 min) • Solve-It Assessment A (15 min) |
| Lesson 6: Debugging | <ul style="list-style-type: none"> • When We Write (15 min) • Why is the Kitten Confused? (20 min) • Free Play (15 min) • Debugging Reflection (10 min) |
| Lesson 7: Details | <ul style="list-style-type: none"> • Fast or Slow? (5 min) • Speed Block (5 min) • Freeze Dance (10 min) • Wait Time Block (5 min) • Program a Freeze Dance (10 min) • Reflection and Discussion (10 min) • Solve-It Assessment B (15 min) |
| Lesson 8: Repeat Loops | <ul style="list-style-type: none"> • Repetition in Instructions (5 min) • Pattern Dance (15 min) • ScratchJr Repeat with Numbers (20 min) • Program a Pattern Dance (20 min) |
| Lesson 9: Descriptive Language | <ul style="list-style-type: none"> • Character Voice (10 min) • ScratchJr Sound Recorder (10 min) • Bringing Your Characters to Life (15 min) • Free Play (10 min) • Solve-It Assessment C (15 min) |
| Lesson 10: Conditionals | <ul style="list-style-type: none"> • Writing an Alternative Story (15 min) • Scratch Jr Conditionals (10 min) • Start on Tap and Start on Bump (10 min) • Send Messages (15 min) • Free Play (10 min) |
| Lesson 11: Final Project - Writing the Jungle Dance Party | <ul style="list-style-type: none"> • Planning the Jungle Dance Party (20 min) • Peer Feedback (5 min) • Collaboration Web (10 min) • Begin Coding the Jungle Dance Party (25 min) |

Lesson 12: Final Project - Coding the Jungle Dance Party

- Coding the Jungle Dance Party (20 min)
- Share Creations and Deliver Cards (15 min)
- Reflections/Final Tech Circle (10 min)
- Solve-It Assessment D (15 min)

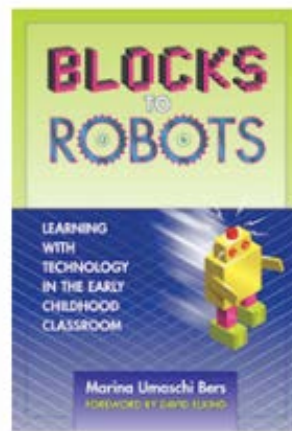
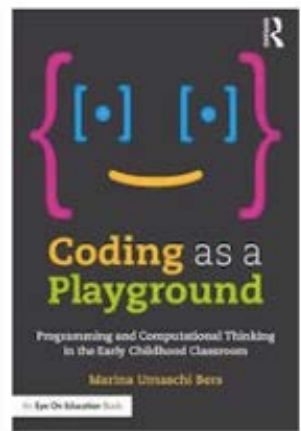
MATERIALS

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 *Giraffes Can't Dance* by Giles Andreae and Guy Parker-Rees.

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 second grade 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 *Where the Wild Things Are*) 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 **K–12 Computer Science Framework** (2016).

Table 2: Alignment of Standards

| | Powerful Ideas of Computational Thinking (CT) and Literacy Embedded in Each Lesson | Common Core ELA/ Literacy Framework (Grade 2) | Computer Science Framework Alignment (Based on the “by end of Grade 2 band”) |
|-----------------------|---|---|---|
| 1: Foundations | <p><i>CT: Design Process</i></p> <p><i>Literacy: Writing Process</i></p> | <p>CCSS.ELA-LITERACY.W.2.5 With guidance and support from adults and peers, focus on a topic and strengthen writing as needed by revising and editing.</p> <p>CCSS.ELA-LITERACY.W.2.2 Write informative/ explanatory texts in which they introduce a topic, use facts and definitions to develop points, and provide a concluding statement or section.</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>2: What is a Program?</p> | <p><i>CT: Algorithms</i></p> <p>Literacy: <i>Sequencing of a Story</i></p> | <p>CCSS.ELA-LITERACY.W.2.3</p> <p>Write narratives in which they recount a well-elaborated event or short sequence of events, include details to describe actions, thoughts, and feelings, use temporal words to signal event order, and provide a sense of closure.</p> | <p>Algorithms and Programming:</p> <p>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>Computing Systems</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>3: Sequencing</p> | <p><i>CT: Algorithms</i></p> <p>Literacy: <i>Summarizing/Retelling the Sequence of a Story</i></p> | <p>CCSS.ELA-LITERACY.W.2.3</p> <p>Write narratives in which they recount a well-elaborated event or short sequence of events, include details to describe actions, thoughts, and feelings, use temporal words to signal event order, and provide a sense of closure.</p> <p>CCSS.ELA-LITERACY.RL.2.5</p> <p>Describe the overall structure of a story, including describing how the beginning introduces the story and the ending concludes the action.</p> | <p>Algorithms and Programming:</p> <p>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:</p> <p>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>4: Characters</p> | <p><i>CT: Representation</i></p> <p>Literacy: <i>Characters</i></p> | <p>CCSS.ELA-LITERACY.RL.2.7</p> <p>Use information gained from the illustrations and words in a print or digital text to demonstrate understanding of its characters, setting, or plot.</p> | <p>Algorithms and Programming:</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>5: Programming</p> | <p><i>CT: Algorithms, Design Process</i></p> <p>Literacy: Sequencing of a Story</p> | <p>CCSS.ELA-LITERACY.W.2.5 With guidance and support from adults and peers, focus on a topic and strengthen writing as needed by revising and editing.</p> <p>CCSS.ELA-LITERACY.W.2.3 Write narratives in which they recount a well-elaborated event or short sequence of events, include details to describe actions, thoughts, and feelings, use temporal words to signal event order, and provide a sense of closure.</p> | <p>Algorithms and Programming: Program Development: People develop programs collaboratively and for a purpose, such as expressing ideas or addressing problems.</p> |
| <p>6: Debugging</p> | <p><i>CT: Debugging</i></p> <p>Literacy: Editing, Awareness of Audience</p> | <p>CSS.ELA-LITERACY.W.2.5 With guidance and support from adults and peers, focus on a topic and strengthen writing as needed by revising and editing.</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>7: Details</p> | <p><i>CT: Control Structures</i></p> <p>Literacy: Details of Language</p> | <p>CSS.ELA-LITERACY.W.2.3 Write narratives in which they recount a well-elaborated event or short sequence of events, include details to describe actions, thoughts, and feelings, use temporal words to signal event order, and provide a sense of closure.</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>8: Repeat Loops</p> | <p><i>CT: Control Structure, Modularity</i></p> <p>Literacy: <i>Repetition as a Literary Device, Repetition in Word Forms</i></p> | <p>CCSS.ELA-LITERACY.RL.2.6</p> <p>Acknowledge differences in the points of view of characters, including by speaking in a different voice for each character when reading dialogue aloud.</p> | <p>Algorithms and Programming:</p> <p>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>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>9: Descriptive Language</p> | <p><i>CT: Control Structures</i></p> <p>Literacy: <i>Descriptive Language in Writing, Characters</i></p> | <p>CCSS.ELA-LITERACY.RL.2.6</p> <p>Acknowledge differences in the points of view of characters, including by speaking in a different voice for each character when reading dialogue aloud.</p> <p>CSS.ELA-LITERACY.W.2.3</p> <p>Write narratives in which they recount a well-elaborated event or short sequence of events, include details to describe actions, thoughts, and feelings, use temporal words to signal event order, and provide a sense of closure.</p> | <p>Algorithms and Programming:</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>10: Conditionals</p> | <p><i>CT: Control Structure</i></p> <p>Literacy: <i>Cause and Effect, Making Predictions</i></p> | <p>CCSS.ELA-LITERACY.RL.2.3 Describe how characters in a story respond to major events and challenges.</p> <p>CCSS.ELA-LITERACY.W.2.3 Write narratives in which they recount a well-elaborated event or short sequence of events, include details to describe actions, thoughts, and feelings, use temporal words to signal event order, and provide a sense of closure.</p> | <p>Algorithms and Programming: 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>11: Final Project - Writing the Jungle Dance Party</p> | <p><i>CT: Algorithms, Design Process</i></p> <p>Literacy: <i>Writing Process</i></p> | <p>CCSS.ELA-LITERACY.W.2.3 Write narratives in which they recount a well-elaborated event or short sequence of events, include details to describe actions, thoughts, and feelings, use temporal words to signal event order, and provide a sense of closure.</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> |

12: Final Project - Coding the Jungle Dance Party

CT: Design Process

Literacy: *Awareness of Audience, Retelling a Story*

CCSS.ELA-LITERACY.W.2.5

With guidance and support from adults and peers, focus on a topic and strengthen writing as needed by revising and editing.

CCSS.ELA-LITERACY.W.2.6

With guidance and support from adults, use a variety of digital tools to produce and publish writing, including in collaboration with peers.

CCSS.ELA-LITERACY.SL.2.4

Tell a story or recount an experience with appropriate facts and relevant, descriptive details, speaking audibly in coherent sentences.

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.

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

- The Design Process (20 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*
- How-to-Book checklist

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 forward 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



THE DESIGN PROCESS (20 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.

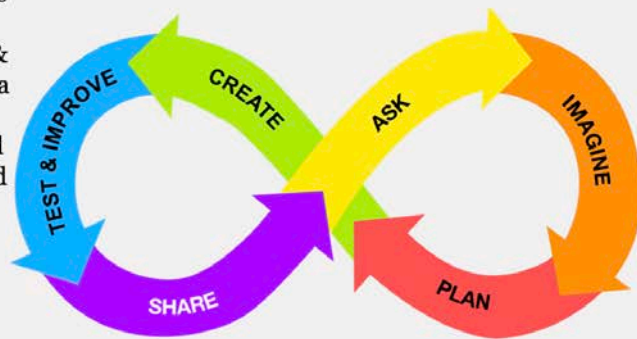
Design Process

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

OVERVIEW

Students will learn about programming and be introduced to the ScratchJr app. Once students become familiar with some of the ScratchJr programming blocks (or if they already are), they will learn more about different functions and capabilities in the app.

PURPOSE

In the previous lesson, students began learning about different ScratchJr blocks. Now they will begin to 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

- What is a Program? (5 min)
- Tools of Communication (15 min)
- Programmer Says (15 min)
- Program the Teacher with ScratchJr Blocks (10 min)
- Meet the ScratchJr App (15 min)

Powerful Idea From Literacy:

Sequencing of a Story

STUDENTS WILL BE ABLE TO...

- Understand why order matters when programming a robot or telling a story
- Identify the blocks in ScratchJr that are crucial towards a successful program (start on green, end on red)
- Add pages to their programs

PREPARATION FOR TEACHERS

- Read through the Activity Guide
- Print large ScratchJr Blocks
- Ensure all tablets have ScratchJr App installed
- Go through the Kitten 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:

- Design Journal (see Appendix C for example)
- 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 (15 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 (15 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.

PROGRAM THE TEACHER WITH SCRATCHJR BLOCKS (10 min)

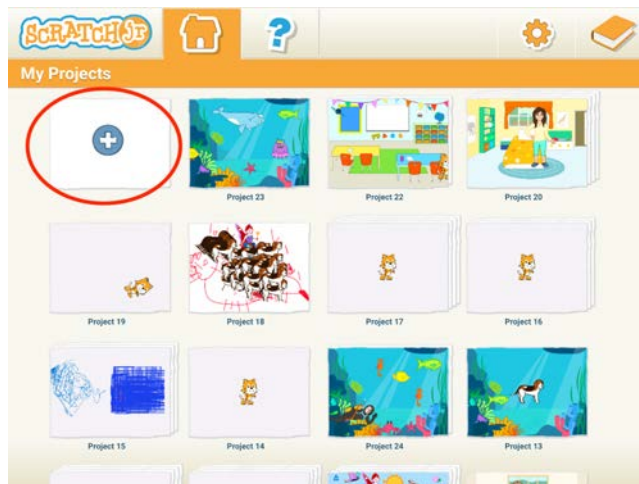
Using the Programmer Says cards, students will work together as a class to “program” their teacher to move from one part of the room to the other. Be silly! An example would be for the students to “program” their teacher to move from the front end of the room to the library area by using these blocks: 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.

Once the students have finished programming the teacher, have them write in their Design Journals what they think a program is in their own words.

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: Sequencing

Powerful Idea From Computer Science:

Algorithms

Powerful Idea From Literacy:

Summarizing/Retelling the Sequence of a Story,

OVERVIEW

Students will learn about sequencing in programming and how it relates to literacy, and why order matters in both cases. Once students become familiar with some of the ScratchJr programming blocks (or if they already are), they will learn more about different functions and capabilities in the app. Students will spend the bulk of the lesson working on an original written composition following the model from Writer's Workshop.

PURPOSE

In the previous lesson, students began learning about different ScratchJr blocks. 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. Much of this lesson will be dedicated to beginning the students' written composition. This written composition, which the students will return to throughout the curriculum, revising, adding to, and reworking, is the central piece of this integrated curriculum. Following the model from Writer's Workshop, students will return to their written composition, but in this curriculum each step of the writing process will be experienced in coordination with/ aligned with corresponding powerful ideas from computer science. Because sequencing lies at the heart of both written and programming composition, the students begin their writing in this lesson.

ACTIVITIES

- Giraffes Can't Dance (15 min)
- Order Matters (10 min)
- Composition Planning (15 min)
- First Draft of Composition (20 min)

STUDENTS WILL BE ABLE TO...

- Understand why order matters when programming or telling a story

PREPARATION FOR TEACHERS

- Read through the Activity Guide
- Print Brainstorming Worksheets

MATERIALS

FOR THE TEACHER:

- 1 copy of *Giraffe's Can't Dance* by Giles Andreae
- Large letter cards: A, R, C*

FOR STUDENTS:

- Design Journal (see Appendix C for example)
- Brainstorming Worksheet

*See Appendix A for examples

VOCABULARY

- Instruction – a direction that a robot will understand
- Order – parts of a group arranged in a specific way (e.g., smallest to largest, tallest to shortest)
- 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 3: Activities

GIRAFFES CAN'T DANCE (15 min)



Read the book *Giraffes Can't Dance* 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 if the first scene was x? How would that change the story?* The purpose of this activity is to get students to think about sequencing in narrative.

ORDER MATTERS (10 min)



This activity is called Rearrange the Letters. The purpose of this activity is to reflect on the importance of sequencing both in computer science and literacy.

For the activity, ask three students to volunteer to hold one of the three large letter cards: A, R, and C. Ask the three students to spell “A-R-C” by arranging themselves in a line. Then ask the three students to spell the word “C-A-R” by rearranging themselves. *Ask the class: What changed when the three volunteers moved their positions? Do the two words mean the same or different things?* Explain to students that letters are symbols for sounds and are strung together in different ways to make different words. When the position of the letters changed, the way we sounded out the letters and the word itself (hence the meaning of the word) also changed.

Conclude the activity by reflecting on the importance of sequencing in literacy and computer science. *Ask students: Why did the order matter in each activity?*

COMPOSITION PLANNING: INSPIRATIONAL STORY (15 min)

In *Giraffes Can't Dance*, Gerald thinks he's a bad dancer. The grasshopper helps Gerald find his music. We are going to write about times in our own lives when we've been like Gerald. Prompt Options:

- Like Gerald, think of a time when you got better at something.
- Like Gerald, think of a time when a friend helped you solve a problem.
- Like Gerald, think of a time when you got to show off your talent.

Requirement: Make sure stories have more than one character, and clear settings. This will be important later as stories are converted to programs.

BRAINSTORMING

- Have students quickly pair up and share which prompt they would like to write about and why. (5 min)
- Feel free to use planning or brainstorming worksheets you already utilize in your class. Make sure that the worksheet emphasizes the sequence or order of the story. This is a great time to incorporate transitional words such as “first, next, then, finally.”

FIRST DRAFT: INSPIRATIONAL STORY (20 min)

Have students begin the first draft of their inspirational story. Use whatever writing procedures are most comfortable for your class. Students may choose to write in pencil in their journals, etc. Emphasize to students that their stories should go in the right order.

Lesson 4: 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 four new blocks: the hide block, the show block, the grow block and the shrink block. All of these blocks alter the characters' appearance and create character actions.

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)
- Changing the Setting in ScratchJr (5 min)
- Add Page in ScratchJr (5 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
- Change the background

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 4: Activities

DESIGN A CHARACTER (15 min)

Students will use the worksheet in their Design Journal to create their own character to join in in the dance party in *Giraffes Can't Dance*. 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)

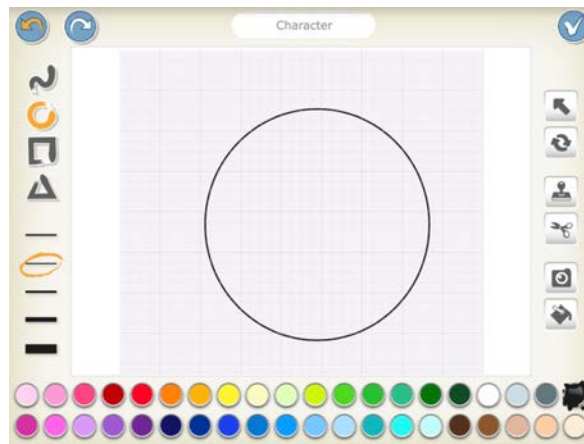


Students will use the paint editor to create their animal character in ScratchJr. They may use the character they created on their worksheet as an example, but it is okay if it ends up looking different - the paint editor can be tricky! (If you used 3D materials like clay or felt, students can upload 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:

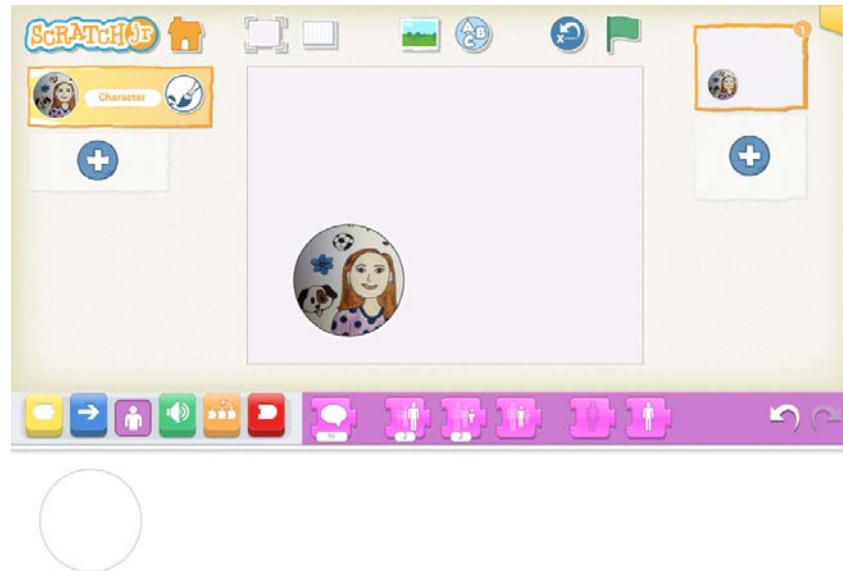
1. Add a new character and open the paint editor



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.

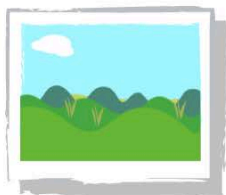


FREE PLAY WITH PURPLE BLOCKS (10 min)

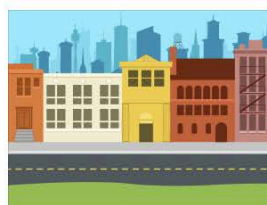
Quickly demonstrate the hide/show and grow/shrink blocks. Allow students to explore using these blocks with their created character.

CHANGE THE SETTING IN SCRATCHJR (5 min)

Show students how to change the background in ScratchJr. Let them adjust the background to fit the character they designed.



New Background

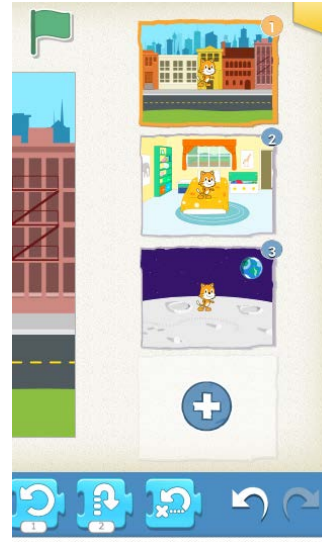
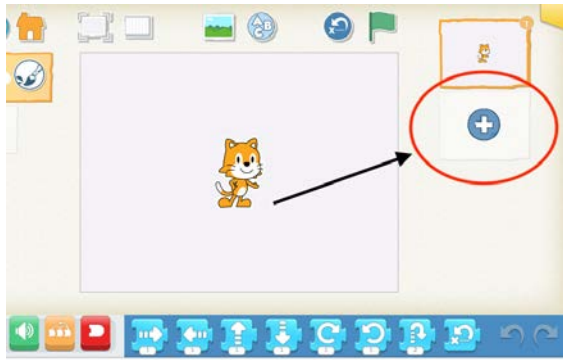


OK



ADD PAGE IN SCRATCHJR (5 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.



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 5: Programming

Powerful Idea From Computer Science:

Algorithms, Design Process

Powerful Idea From Literacy:

Sequencing of a Story

OVERVIEW

Students will learn about sequencing in programming and think about how it relates to sequencing in literacy. Students will program the ScratchJr Kitten to dance the Hokey-Pokey, or if you wish, a different children's song where students can create a program to dance to the words. At the end of the lesson, students will demonstrate their current level of understanding by completing the first Solve-It assessment.

PURPOSE

In the previous lesson, students had the opportunity to engage with ScratchJr as a class. Now they 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.

ACTIVITIES

- Dance the Hokey-Pokey (5 min)
- Program the Hokey-Pokey (20 min)
- Hokey-Pokey Reflection (10 min)
- Share Creations (10 min)
- Solve-It Assessment A (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 5: Activities

DANCE THE HOKEY-POKEY (5 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 (20 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. Have several students share out their strategies for programming. Individually or in pairs, students program the Kitten to do the Hokey-Pokey.

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 you use each programming block? What order did you put the blocks in? Why did you choose this particular order? Have students share out the number of times they used the Forward block or the Turn block. Ask students: Did the whole class use the same number of each block?*

SHARE CREATIONS (10 min)

When all groups are done with their Hokey-Pokey robot 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 snapping 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!)



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 6: 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.

PURPOSE

The parallel of editing in literacy and debugging in computer science is an asset 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

- When We Write (15 min)
- Why is Kitten Confused? (15 min)
- Free Play (20 min)
- Debugging Reflection (10 min)

STUDENTS WILL BE ABLE TO...

- Identify common errors with scanning ScratchJr programs and troubleshoot them
- Practice scanning programs with ScratchJr
- Learn strategies for debugging and editing

PREPARATION FOR TEACHERS

- Read through the Activity Guide
- 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 6: Activities

WHEN WE WRITE (15 min)



Modeled after writer's conference from Writer's Workshop, have students share their written compositions in small groups. Have each student tell the student one aspect of their composition they liked and one aspect that confused them. In framing this activity, make explicit to students that editing their written composition is similar to debugging their programming compositions.

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>

FREE PLAY (20 min)

This is a great opportunity for students to freely explore the ScratchJr app and programming blocks. 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.

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.

Lesson 7: Details

Powerful Idea From Computer Science:

Control Structures

OVERVIEW

Dances, like mentioned in *Giraffes Can't Dance*, can go at any speed and often involve pauses. Today, students will work on creating a dance program for Gerald the Giraffe, that goes slow, medium or fast and involves pauses or wait time. This lesson also 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 is an important, yet difficult, part of programming in ScratchJr. It adds an additional element to the program, the same way that details add to a story.

ACTIVITIES

- Fast or Slow? (5 min)
- Speed Block (5 min)
- Freeze Dance (10 min)
- Wait Time Block (5 min)
- Freeze Dance Program (10 min)
- Solve-It B (15 min)

Powerful Idea From Literacy:

Details of Language

STUDENTS WILL BE ABLE TO...

- Use the control speed and wait time blocks appropriately
- Add details to their written compositions

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 7: Activities

FAST OR SLOW (5 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?*

SPEED BLOCK (5 min)

Project your tablet to the class and introduce the speed block. Just like songs can go fast or slow, we also may want our programs to go fast or slow. Demonstrate a sample program and what happens when you change the speed.



FREEZE DANCE (10 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 (10 min)



Allow students to use their own tablets or work in pairs or small groups to program their own freeze dance. Remind them that they must use both the speed and wait time blocks. Students' programs are becoming more detailed, and thus more difficult, so it is okay if they need additional time or do not complete the activity.

REFLECTION AND DISCUSSION (10 min)



Allow students to share out their projects and reflect in their Design Journals. Guiding questions:

- *What was difficult or easy about this project?*
- *If your program was set to music, what song would you want? Would it be fast or slow?*
- *What is something you liked about a friend's program?*
- *What would you do differently if you could go back and edit?*

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 8: 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 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.

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)
- Pattern Dance (15 min)
- ScratchJr Repeat with Numbers (20 min)
- Program a Pattern (20 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 repetition is used in stories and songs

PREPARATION FOR TEACHERS

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

MATERIALS

FOR THE TEACHER:

- ScratchJr Block Cards

FOR STUDENTS:

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

VOCABULARY

- Loop — something that repeats over and over again

Lesson 8: 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)

PATTERN DANCE (15 min)



Choose a song or dance the students like. (Examples include the electric slide, the Cupid Shuffle, etc.) Hand out the lyrics to the class, play the song for the class, and ask students, as the song is playing, to circle repeating stanzas. The purpose of this activity is to remind students that repetition is essential in language, literature, and, as they will learn today, coding as well. You may also choose to do the dance as a class and discuss where the repeats happen and what steps you are repeating.

SCRATCHJR REPEAT WITH NUMBERS (20 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.



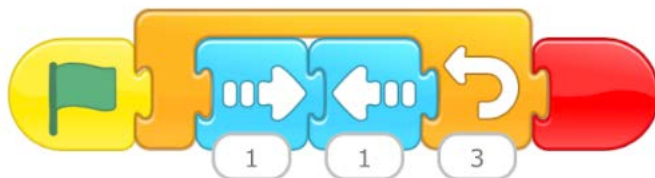
Ask the class: Think back to our activity earlier when (student's name) had to get up from the chair and walk around several times. Do you think there's an easier way I could program this? Is there a way I could make the program shorter?

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.



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.



Have students 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.



PROGRAM A PATTERN DANCE (20 min)

Have students program a verse of a song or dance you chose as a class. You may choose to use the same song you used earlier in the lesson. Tell students that they should use as few blocks as possible i.e. they should always use repeat loops when they can! Share out great examples with the class.

Lesson 9: Descriptive Language

Powerful Idea From Computer Science:

Control Structure

Powerful Idea From Literacy:

Descriptive Language in Writing, Characters

OVERVIEW

In this lesson, students will learn to enhance their characters by being introduced to the record block. To use the record block, the students must first record a sound or their voice and then add it to their program.

PURPOSE

In this lesson, students learn that programs can capture information from the outside environment and incorporate it into the program to enhance their character. These concepts are crucial to enhancing characters with details and modes of expression in literacy curriculums.

ACTIVITIES

- Character Voice (10 min)
- ScratchJr Sound Recorder (10 min)
- Bringing Your Character to Life (15 min)
- Free Play (10 min)
- Solve-It C (15 min)

STUDENTS WILL BE ABLE TO...

- Record a sound clip successfully using the Sound Recorder block

PREPARATION FOR TEACHERS

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

MATERIALS

FOR THE TEACHER:

- 1 copy of *Giraffes Can't Dance* by Giles Andreae

FOR STUDENTS:

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

VOCABULARY

- Record — to make something (like a sound) permanent so that it can be played back at a later time
- Sound — a type of energy made by vibrations in the air that we can hear

Lesson 9: Activities

CHARACTER VOICE (10 min)

Have your students take out their written compositions. Have them make a character list of all the characters/people in their story. They will return to this list at the end of the lesson. Take a few minutes with your students to discuss all of the characters in *Giraffes Can't Dance*. Create a list on your board or chart paper of all the characters. *Ask students: How do we know what the characters are thinking? How do the characters communicate with each other?* It is crucial in stories that characters find different ways to express themselves. On the board/chart paper, write down all the characters that speak throughout the story, and other ways they express themselves. Today, students will learn how to make their characters speak on ScratchJr.

SCRATCHJR SOUND RECORDER (10 min)

Demonstrate for you class how to write a program that includes the sound recorder. As always, start with the green flag. Then, 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. 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.



BRINGING YOUR CHARACTERS TO LIFE (15 min)



Have students take out their written composition drafts and their character lists from the beginning of class. Using the following guiding questions, have them add any details to their story that will help their characters express themselves better.

Guiding Questions:

- What about your character can be expressed through words or noises?
- What does your character's voice sound like? What kind of things does the character say?
- How does your character interact with other characters? Is there any places in your story where your characters talk to each other?
- What words/details can you add to your story to emphasize the personality of your character?

FREE PLAY (10 min)



Individually or in pairs, students should take this time to explore the Sound Recorder freely. By the end of this free-exploration, students should be able to record the specific sounds they want ScratchJr to play using the Sound Recorder.

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 10: Conditionals

Powerful Idea From Computer Science:

Control Structure

Powerful Idea From Literacy:

Cause and Effect, Making Predictions

OVERVIEW

In this lesson, students will further learn about cause and effect in addition to creating interaction between characters by using the “send messages” and “start on bump” and “start on tap” features.

PURPOSE

This lesson allows students to interact with their characters and make their characters interact. They will be able to connect these actions to the literacy concepts of cause and effect and making predictions. Knowing that outcomes can vary depending on the circumstances is an important concept in early childhood, as students begin to comprehend how decisions are made in everyday life.

ACTIVITIES

- Writing an Alternative Story (15 min)
- ScratchJr Conditionals (10 min)
- Start on Tap and Start on Bump (10 min)
- Send Messages (15 min)
- Free Play (10 min)

STUDENTS WILL BE ABLE TO...

- Use the Start on Bump and Start on Tap blocks successfully
- Understand and be able to use the Send Message blocks

PREPARATION FOR TEACHERS

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

MATERIALS

FOR THE TEACHER:

- 1 copy of *Giraffes Can't Dance* by Giles Andreae

FOR STUDENTS:

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

VOCABULARY

- Event — an action that causes something to happen

Lesson 10: Activities

WRITING AN ALTERNATIVE STORY (15 min)



The purpose of this activity is to have students think creatively about what could have happened in *Giraffes Can't Dance* if Gerald had done things differently. If necessary, reread the story to students.

Below are some examples from the story. Ask students to think about these hypothetical scenarios, and have several students share out their hypotheses.

- *What would have happened if the lions hadn't made fun of Gerald?*
- *What would have happened if Gerald hadn't met the cricket?*
- *If Gerald had not left the dance to look at the moon...*
- *If Gerald had been with other giraffes at the dance...*

Now the students have the opportunity to turn these suggested alternative stories into compositional texts.

Students will write their alternative stories in their Design Journals. This activity is also an opportunity to review whatever skills the students have most recently learned in writing (e.g. strategies for organization, capitalization of proper nouns, etc.).

SCRATCHJR CONDITIONALS (10 min)



Explain to students that in the programs they have learned so far, Kitten has only one choice of what instructions to follow when the program begins. Now they will learn that there are blocks that give Kitten different choices of instructions to follow depending on what happens while the program is running! For example:

- If Kitten is tapped, then Kitten will start moving!
- If Kitten receives a message, then Kitten will jump!
- If Kitten is bumped by Turtle, then Kitten will spin around!



START ON TAP AND START ON BUMP (10 min)

The first block we are going to be using today is called start on tap. With this button, a character will not start until we tap it! Guiding questions for discussion:

- Where do you think Start on Tap should go in our program?
- Should it go at the end? Why or why not?

Demonstrate the start on tap block using your same recorded program.



In addition, programs can start with the start on bump block. This block only starts a character's program when that character is "bumped" by another character.

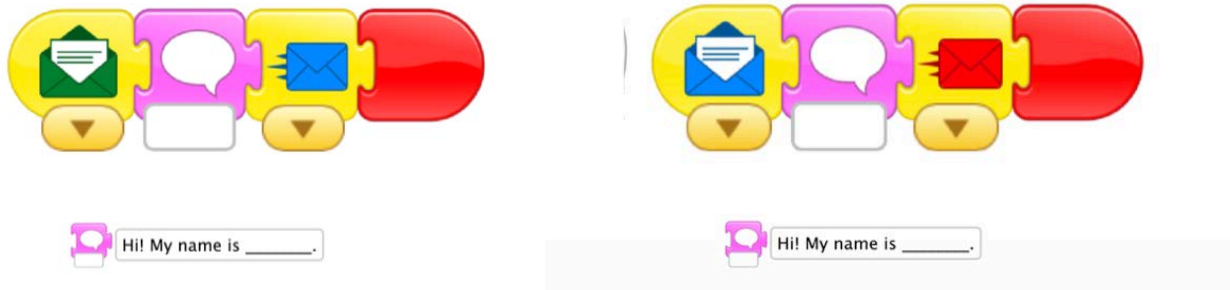


Add a character to your same recorded program (if there aren't already multiple characters) and then demonstrate the start on bump block.

SEND MESSAGES (15 min)

Characters in ScratchJr can send messages to one another. These are special messages that control when a character starts its programs. Demonstrate an example of a character sending a message to another character. The start on message block can be used in place of the green flag block.

(Example program to the right)



Guiding questions:

- If I start my character with the orange start on message block, what color do you think the send message block needs to be? Why?



FREE PLAY (10 min)



Instruct students to each select 3 characters to practice using start on tap, start on bump, and send messages. Students may choose to use characters that align with *Giraffes Can't Dance*. *Note: these blocks are tricky to use and take practice! Encourage students and reinforce the idea that mistakes and debugging are common practices in programming.

Lesson 11: Final Project - Writing the Jungle Dance Party

Powerful Idea From Computer Science:

Algorithms, Design Process

Powerful Idea From Literacy:

Writing Process

OVERVIEW

Students will begin their final project in this lesson: programming their Jungle Dance Party! During the course of this final project, students will put to use all the concepts learning during the previous lessons. Students can use parts of their programs from previous lessons, but they should be encouraged to start fresh and transfer their skills to a new context.

PURPOSE

The purpose of the final project is two-fold: first, to allow students to demonstrate the skills they have acquired throughout the previous lessons and to apply them in new, creative ways. By writing out their plans first before creating in ScratchJr, students make purposeful decisions about their projects and understand that not all ideas on paper can transfer to the actual design. Second, by “rewriting” their written planning in code, the connections between programming and writing will be brought to life. Student will experience the constraints and affordances of each medium and better understand how the two support one another.

ACTIVITIES

- Planning the Jungle Dance Party (20 min)
- Peer Feedback (5 min)
- Collaboration Web (10 min)
- Coding the Jungle Dance Party (25 min)

STUDENTS WILL BE ABLE TO...

- Compose a jungle dance party and transfer it to ScratchJr
- Decide which of their ideas can be translated to ScratchJr programs and which cannot
- Identify and show appreciation to those who have helped them with their final projects

PREPARATION FOR TEACHERS

- Read through the Activity Guide
- Ensure all tablets are charged and have ScratchJr
- Print Collaboration Web for each student (Appendix D)

MATERIALS

FOR THE TEACHER:

- 1 copy of *Giraffes Can't Dance* by Giles Andreae

FOR STUDENTS:

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

Lesson 11: Activities

PLANNING THE JUNGLE DANCE PARTY (20 min):

If your class has a standard “Pre-write” or “Planning” process, this is an excellent time to implement it. Students will be planning out their Jungle Dance Party. They will want to consider the characters, the setting, and the details. Give your students clear instructions for what a successful program will look like.. What will stay the same? What will change? What details can they add in ScratchJr? What details will it be hard to convert? Remind the students that their programs should have multiple characters and multiple pages with different backgrounds, repeat loops, send messages and any additional details they’d like to include. In addition, students may choose to set their program to music or record their voice narrating.

PEER FEEDBACK (5 min):



Have students swap plans with a peer. Give them a few minutes to talk about their story plan and then receive feedback.

- Partner 1 explains their jungle dance party planning (2 min)
- Partner 2 gives feedback (1 min)
- Partner 2 explains their jungle dance party planning (2 min)
- Partner 1 gives feedback (1 min)

COLLABORATION WEB (10 min):



A collaboration web is a tool for students to recognize peers who have helped and supported them in different ways, such as working together on a common task, lending or borrowing materials, programming together, etc.

Directions:

1. Obtain headshots of each student in the class.
2. Create individual printouts with each student’s photograph in the center of the page and the names and photographs of all the other students arranged in a circle surrounding the central photo.
3. Whenever you observe students collaborating during the final project, ask students to draw a line from their photo in the center to the photo of the other students with whom they collaborated.
4. At the end of Lesson 12, ask students to count the number of lines they have with each student. Then choose a couple students to say thank you to!

BEGIN CODING THE JUNGLE DANCE PARTY (25 min):



Give students any remaining time to start their program. Emphasize that all of the details they planned may not transfer to ScratchJr, and that’s okay! This could be a great starting off point for a discussion on which story elements work well in programs and which are more difficult. Some actions, characters, details may be more difficult to represent within the ScratchJr programming language.

Lesson 12: Final Project - Coding the Jungle Dance Party

Powerful Idea From Computer Science:

Design Process

Powerful Idea From Literacy:

Awareness of Audience, Retelling a Story

OVERVIEW

In this final lesson, students will code their jungle dance party programs. During the course of this final project, students will put to use all the concepts learned during the previous lessons. When students are finished with their projects, they will share them with each other and offer their gratitude to those who have helped them along the way.

PURPOSE

The purpose of the final project is to allow students to demonstrate the skills they have acquired throughout the previous lessons and to apply them in new, creative ways. After students finish presenting their projects to their peers, educators are encouraged to invite families and community members to view students' final projects

ACTIVITIES

- Coding the Jungle Dance Party (20 min)
- Share Creations and Deliver Cards (15 min)
- Reflections/Final Tech Circle (10 min)
- Solve-It D (15 min)

STUDENTS WILL BE ABLE TO...

- Compose a jungle dance party and transfer it to ScratchJr
- Share final projects with peers, family and community members
- Identify and show appreciation to those who have helped them with their final projects

PREPARATION FOR TEACHERS

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

MATERIALS

FOR THE TEACHER:

- 1 copy of *Giraffes Can't Dance* by Giles Andreae

FOR STUDENTS:

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

Lesson 12: Activities

CODING THE JUNGLE DANCE PARTY (20 min)



Students may continue to work on coding their final projects and incorporating any feedback they may have received from peers. After students are done working, have students write three thank you cards to the three students who have helped them the most using construction paper or another kind of nice paper.

SHARE CREATIONS AND DELIVER CARDS (15 min)

During the final presentations, have students present their Jungle Dance Party compositions and ScratchJr programs. Students can share their final projects altogether in a technology circle, or as a gallery walk, in which half of the students walk around the classroom to each project while the other half present their projects. Then the two groups switch. Students should share:



- their programs and stories
- why they chose those characters and backgrounds
- the final program and what each block represents
- anything that was hard, easy, surprising, interesting, etc. about the process.

Take photos of students' final projects! While you do this have the students deliver their cards.

REFLECTION/FINAL TECH CIRCLE (10 min)

Gather students into a circle and generate a final discussion surrounding their projects. Take this time to think about the constraints and affordances of each medium, writing and coding. Guiding questions include:

- What was difficult about coding your composition?
- Were there somethings in your writing that you could not code? Why?

EXTENDED ACTIVITY:

If more time is needed for students to finish their final projects, this reflection activity can be assigned for homework. Now that students have written a composition and created a project in ScratchJr, have students write a letter to their families explaining their projects. Ask students: What was your project about? What did you learn by playing with ScratchJr? What was your favorite thing? What was your most challenging thing? Send students' letters to families, along with pictures of their compositions, final projects and codes. If students have access to tablets at home, you may also share their final project to their parents. See instructions below:



And check out <https://www.scratchjr.org/learn/tips/share-projects> for more information on sharing.

1. Make sure both the Sending and Receiving devices are:
 1. turned on
 2. connected to the internet
 3. enabled to receive emails/AirDrops

4. pre-loaded with the ScratchJr app
2. On the Sending device, open the project you want to share. Tap the yellow rectangle in the top right corner of the screen to see the Project Information Screen
3. Type a specific name for this project (e.g. "Dance Party"). Share the project to your Receiving device using your preferred share method (AirDrop or email)
4. On the Receiving device, open ScratchJr. You should see your newly shared project in your project library, with a blue ribbon to show it hasn't been opened yet.



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.

Appendix A. Materials

Appendix A. Materials

Technology Materials:

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

Art Materials:

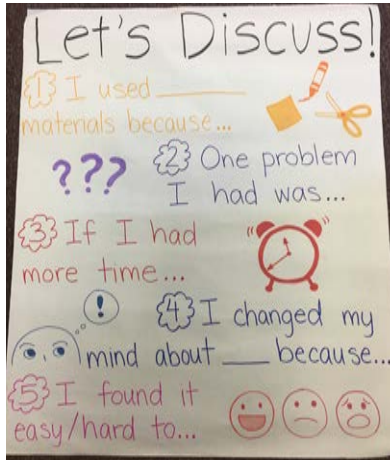
- Construction paper or other kind of decorative paper
- Markers, crayons, or colored pencils
- Masking tape

Teaching Materials:

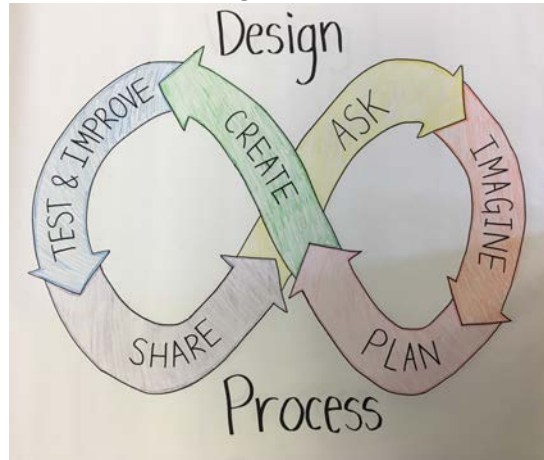
- 1 copy of *Giraffes Can't Dance* by Giles Andreae (it might be helpful to have multiple copies for students to reference during projects)
- Premade anchor charts (**see following pages for examples**)
 - Discussion sentence starters
 - Design Process
 - Writing Process
 - Why is Kitten Confused?
- Printed pictures (**see following pages for examples**)
 - Large letter cards: A, R, C
 - Scratch Jr Block Cards

Examples of Anchor Charts:

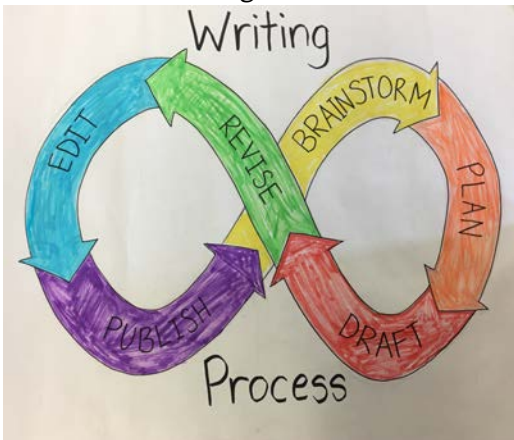
Discussion Sentence Starters



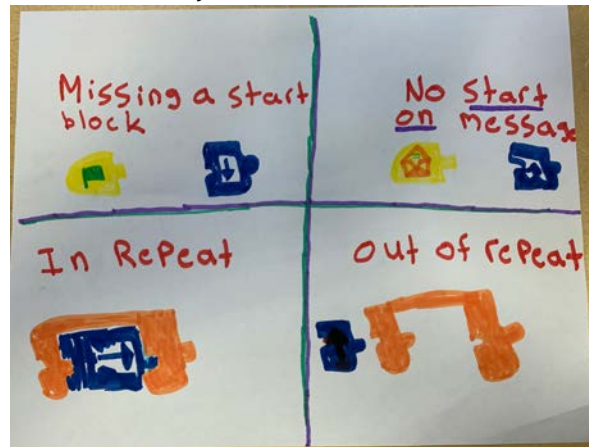
Design Process



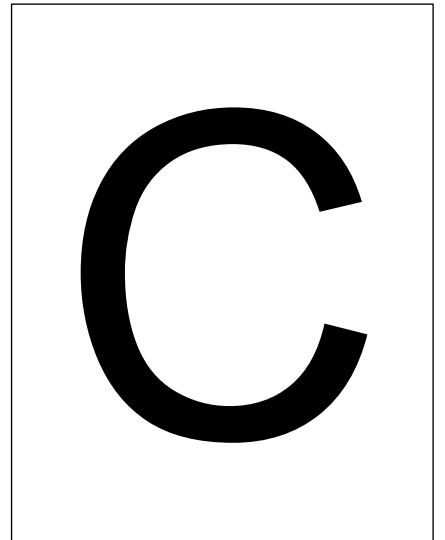
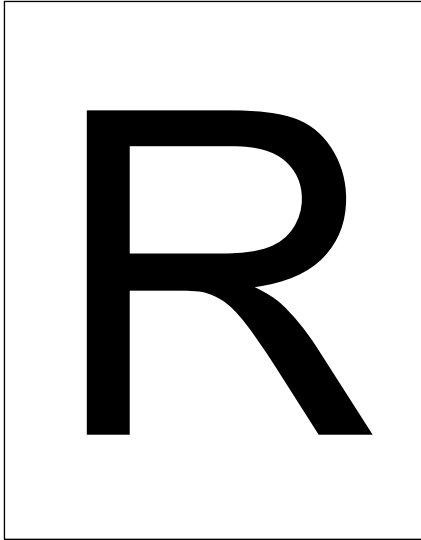
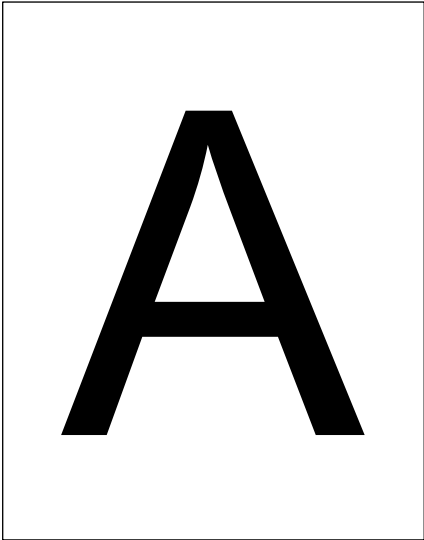
Writing Process



Why is Kitten Confused?

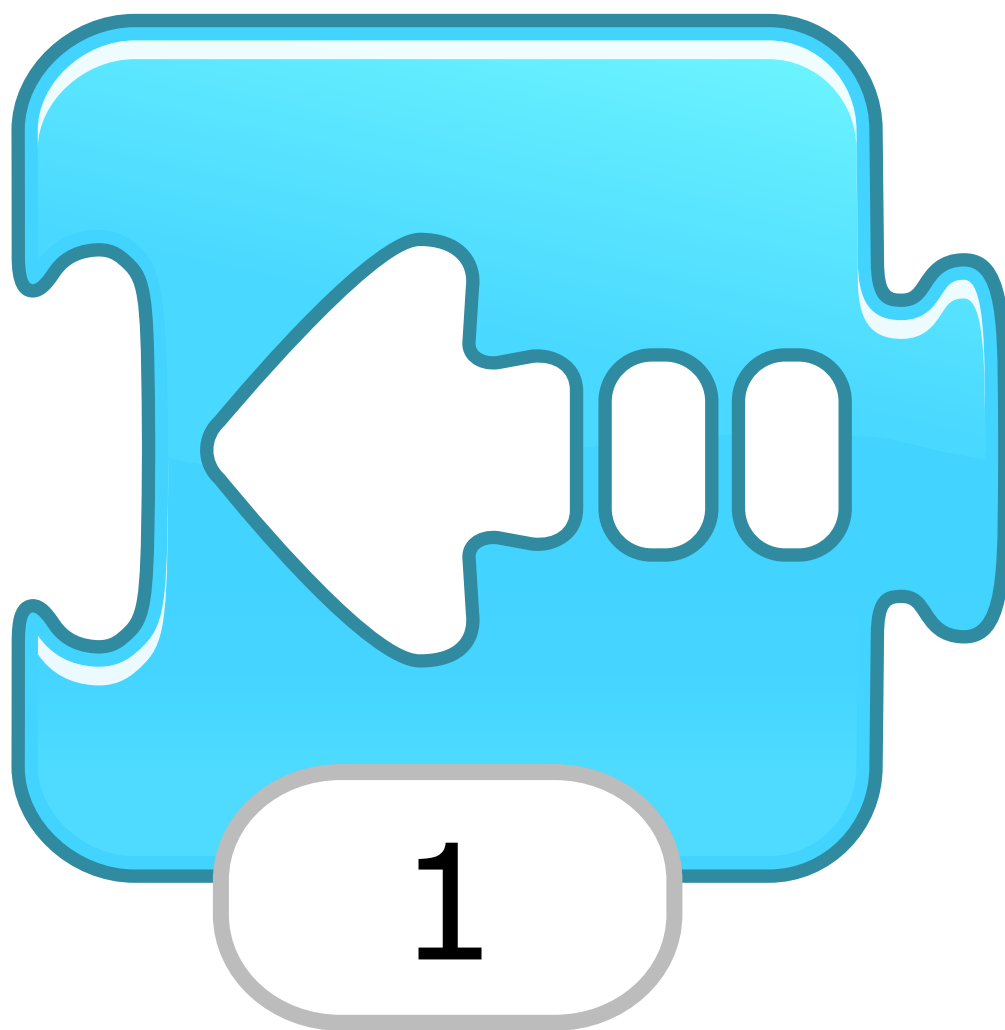


Examples of Printed Pictures:
Large letter cards: A, R, C

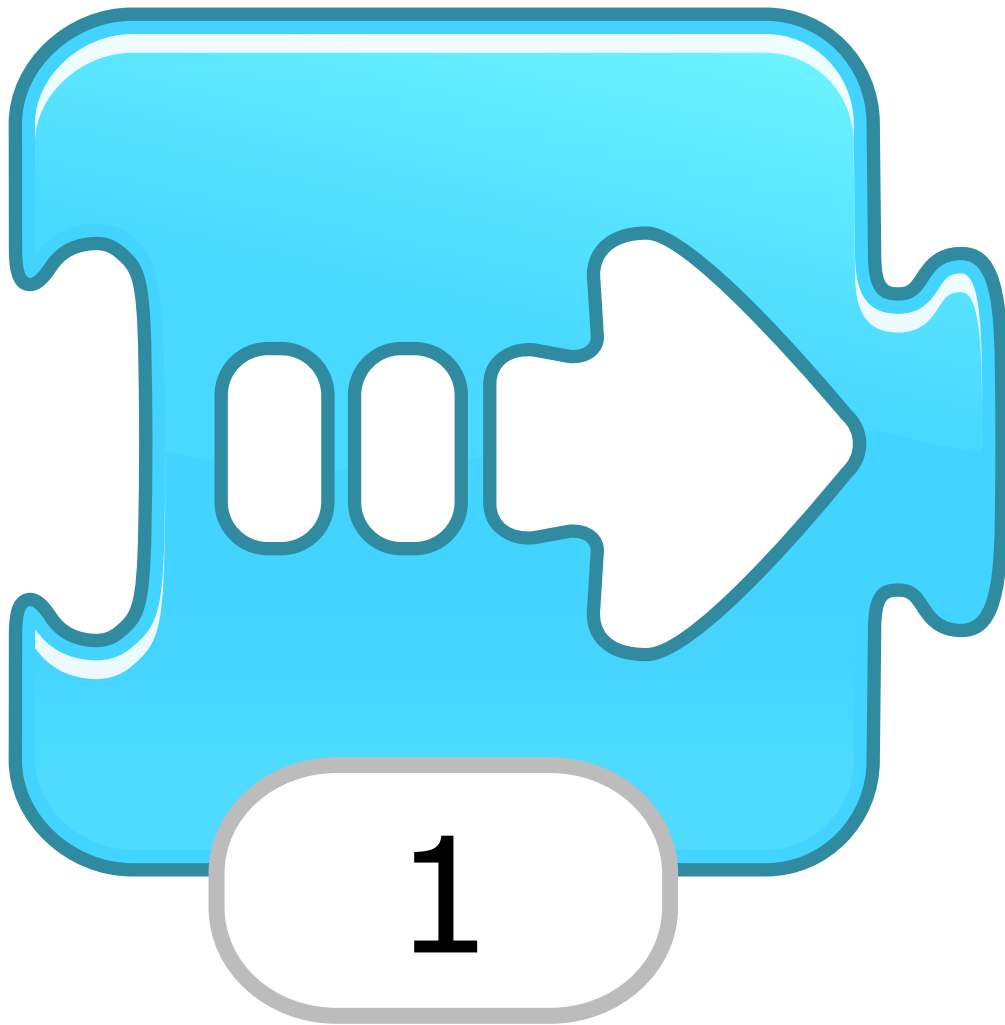


MOTION BLOCKS

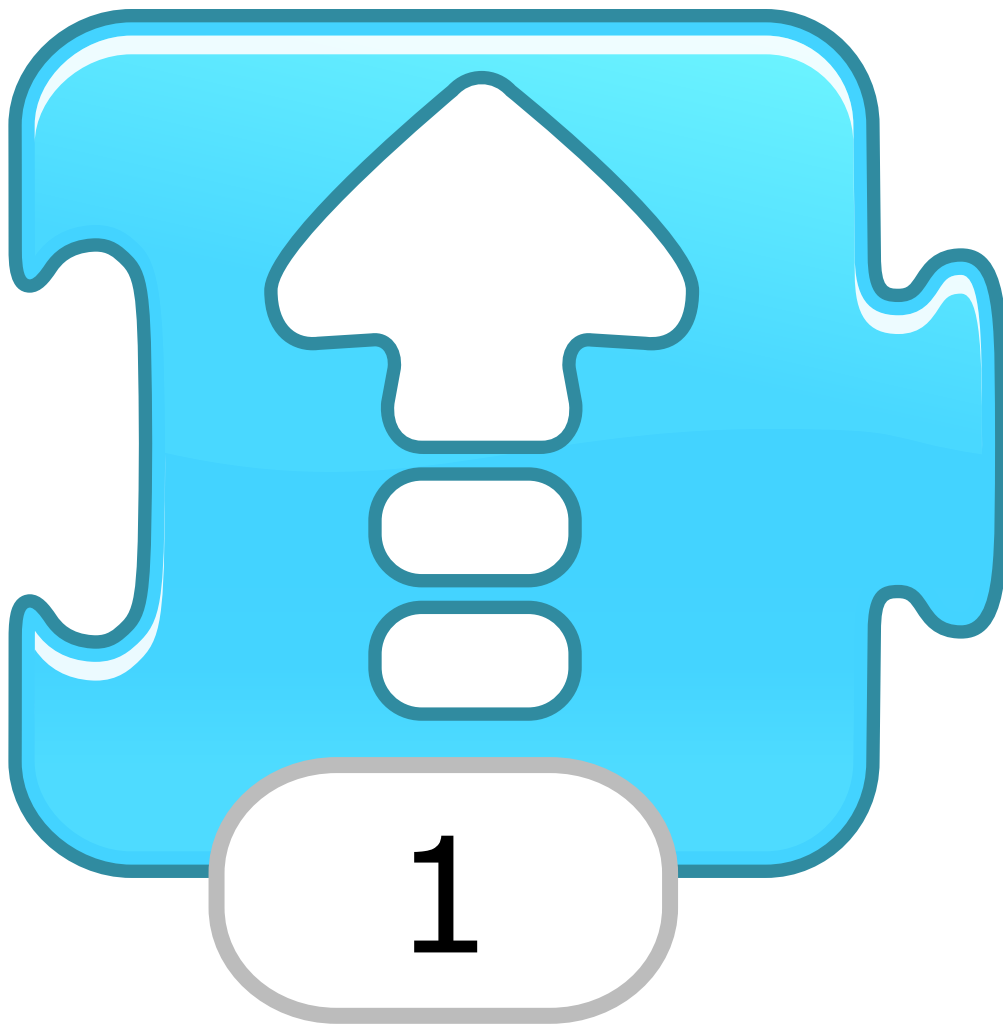
Move Left



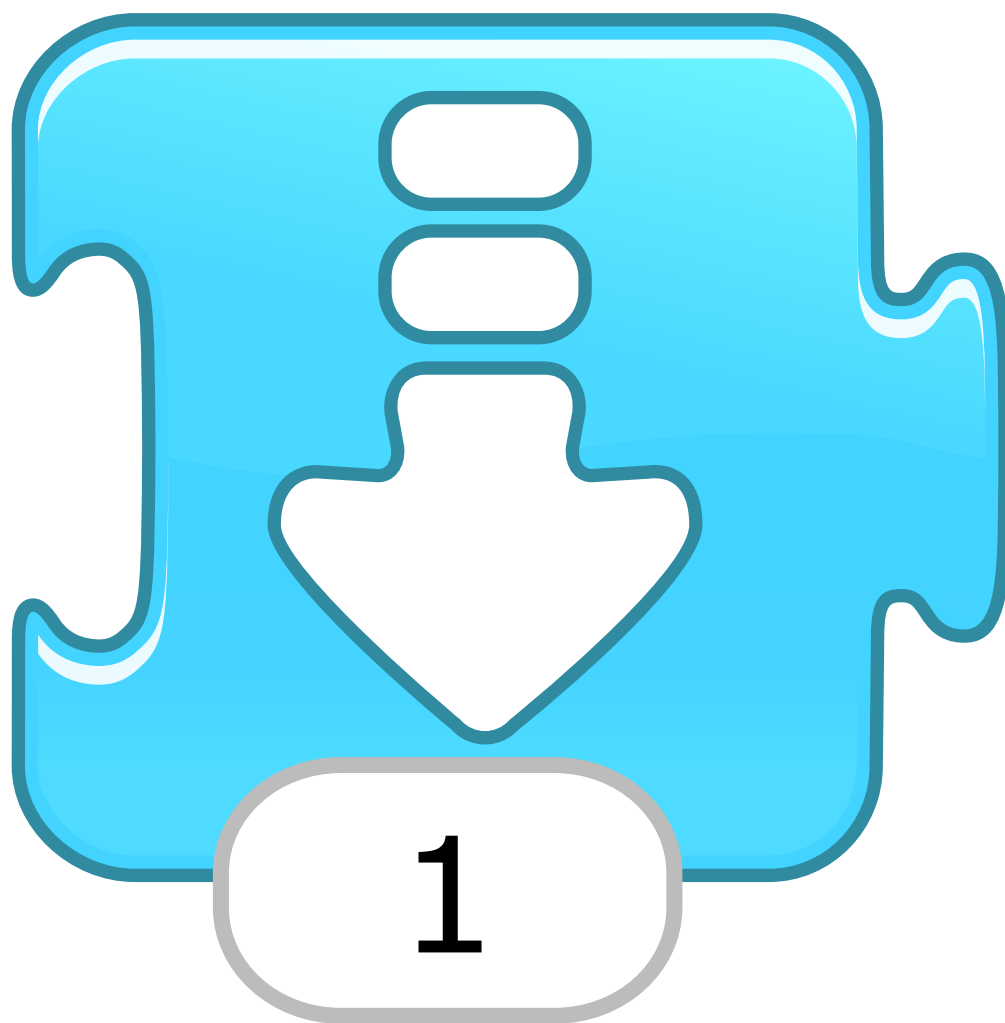
Move Right



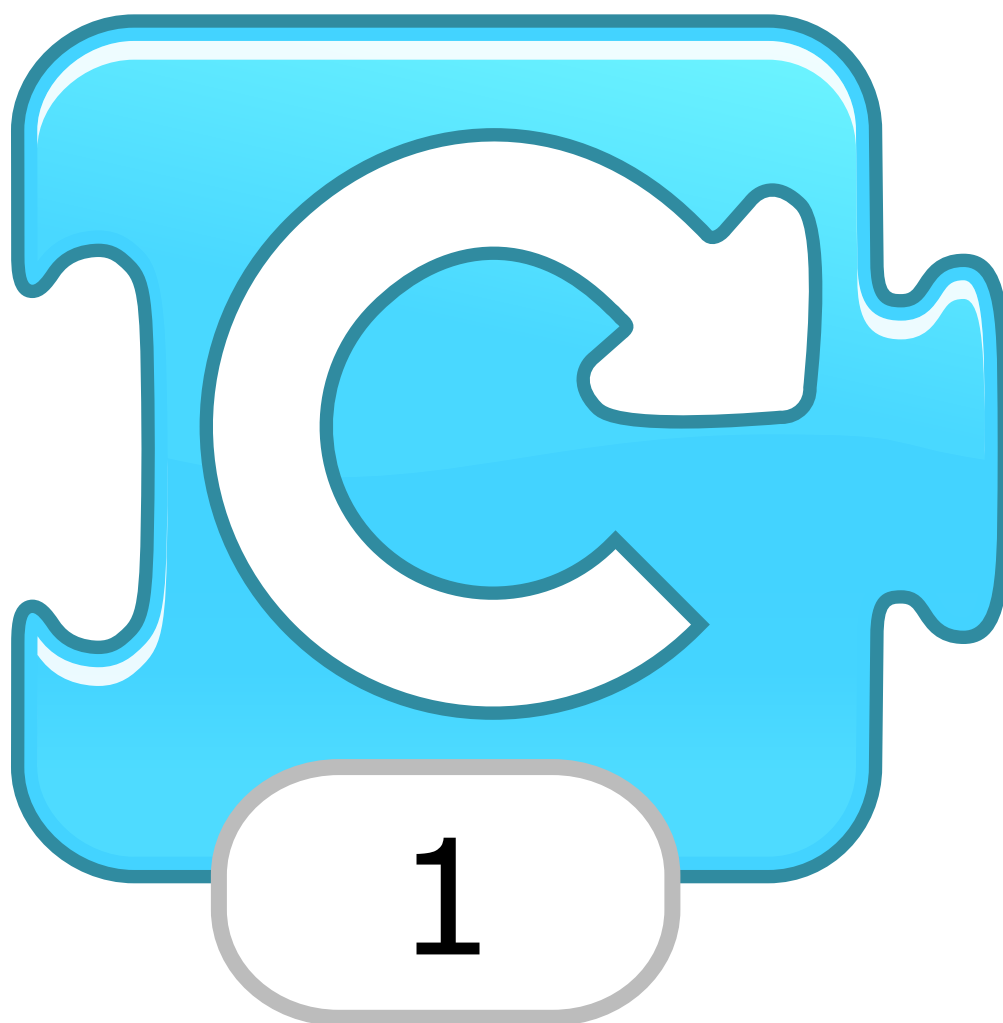
Move Up



Move Down



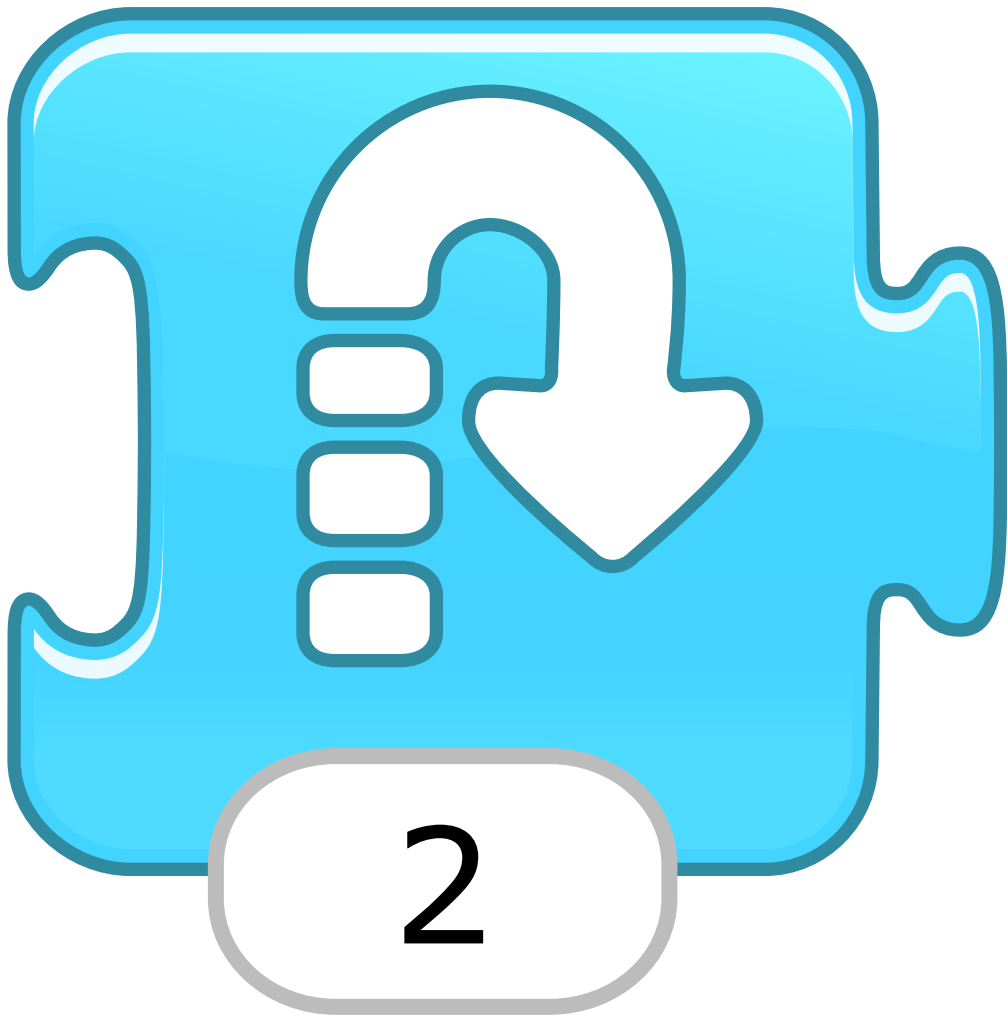
Turn Right



Turn Left



Hop



Go home



TRIGGERING BLOCKS

Start on Message



Start on Message



Start on Message



Start on Message



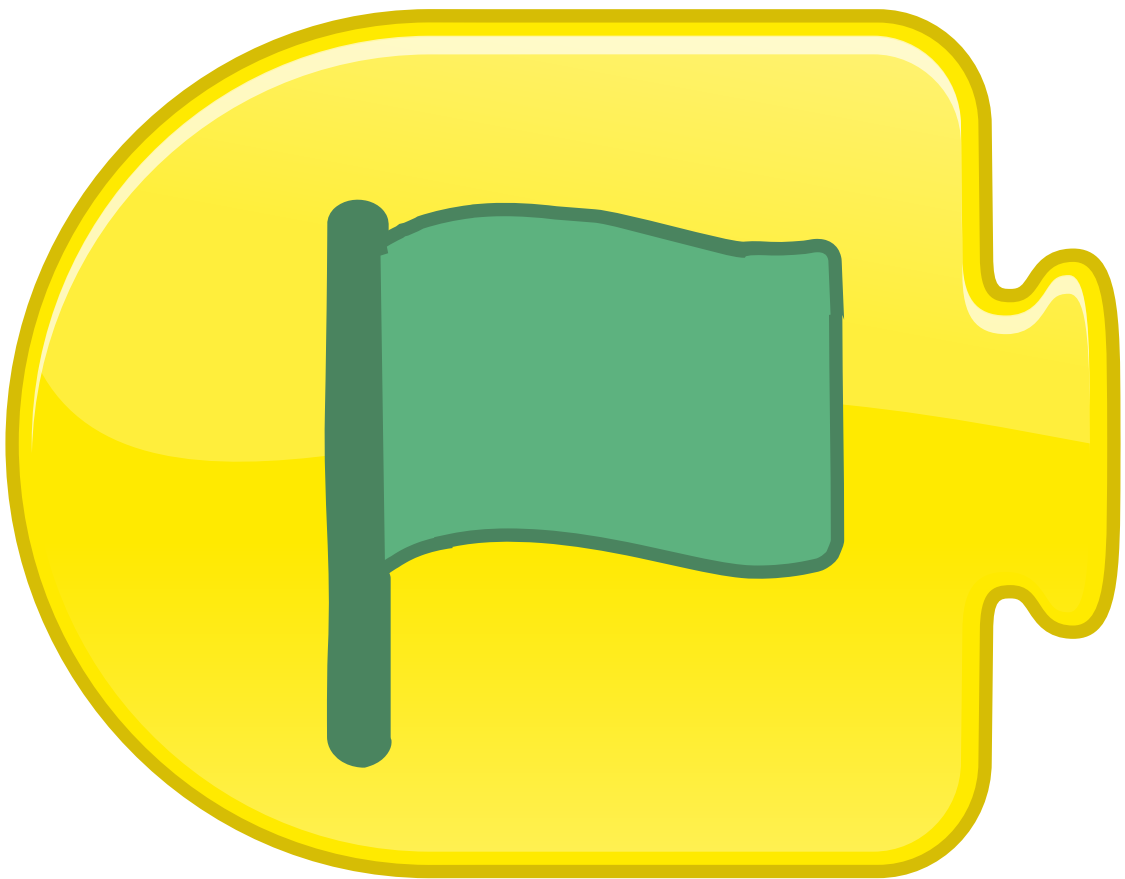
Start on Message



Start on Message



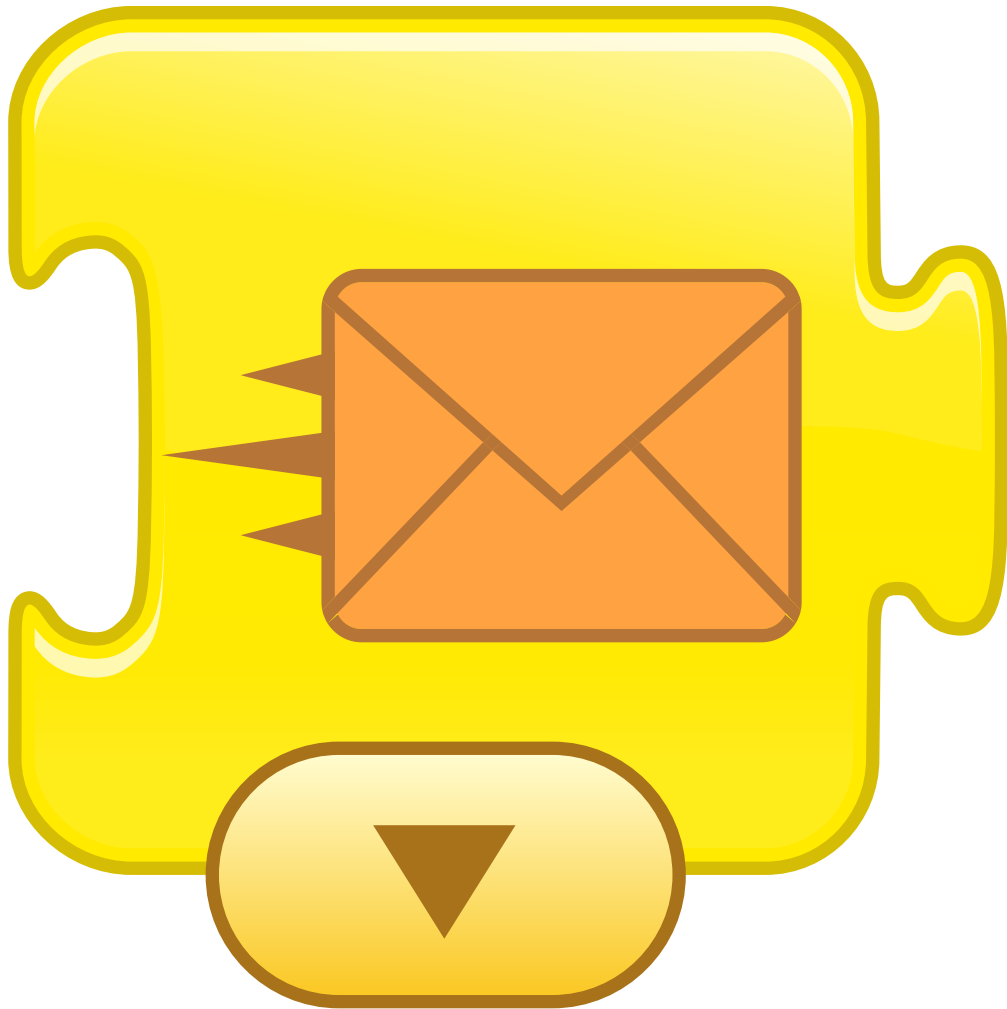
Start on Green Flag



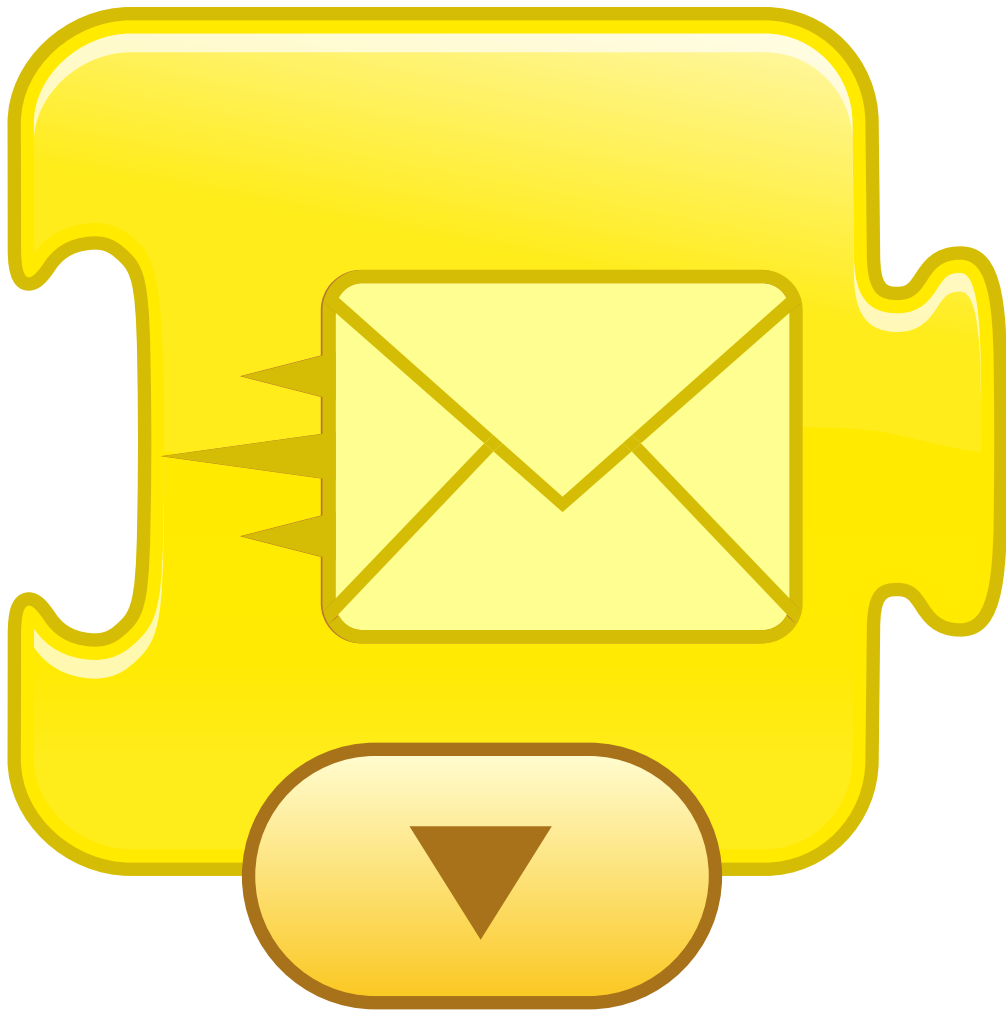
Start on Tap



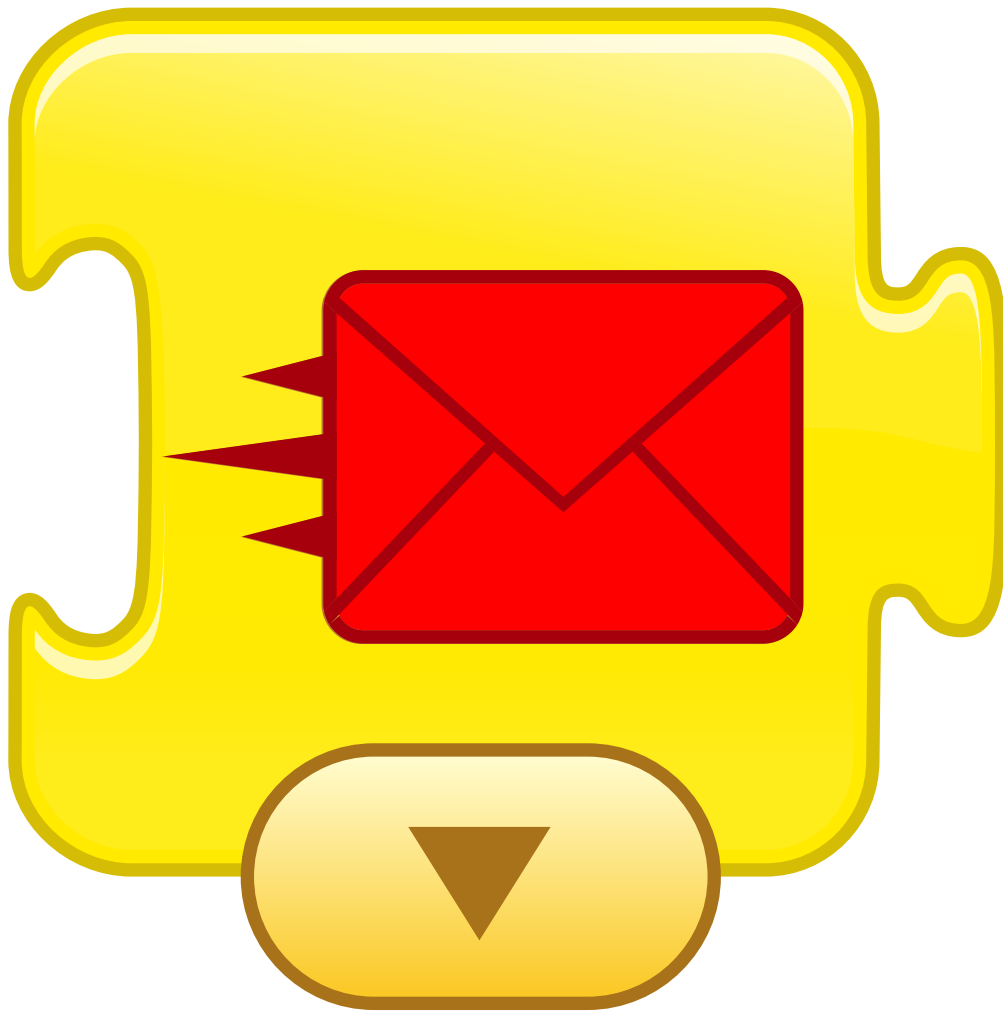
Send Message



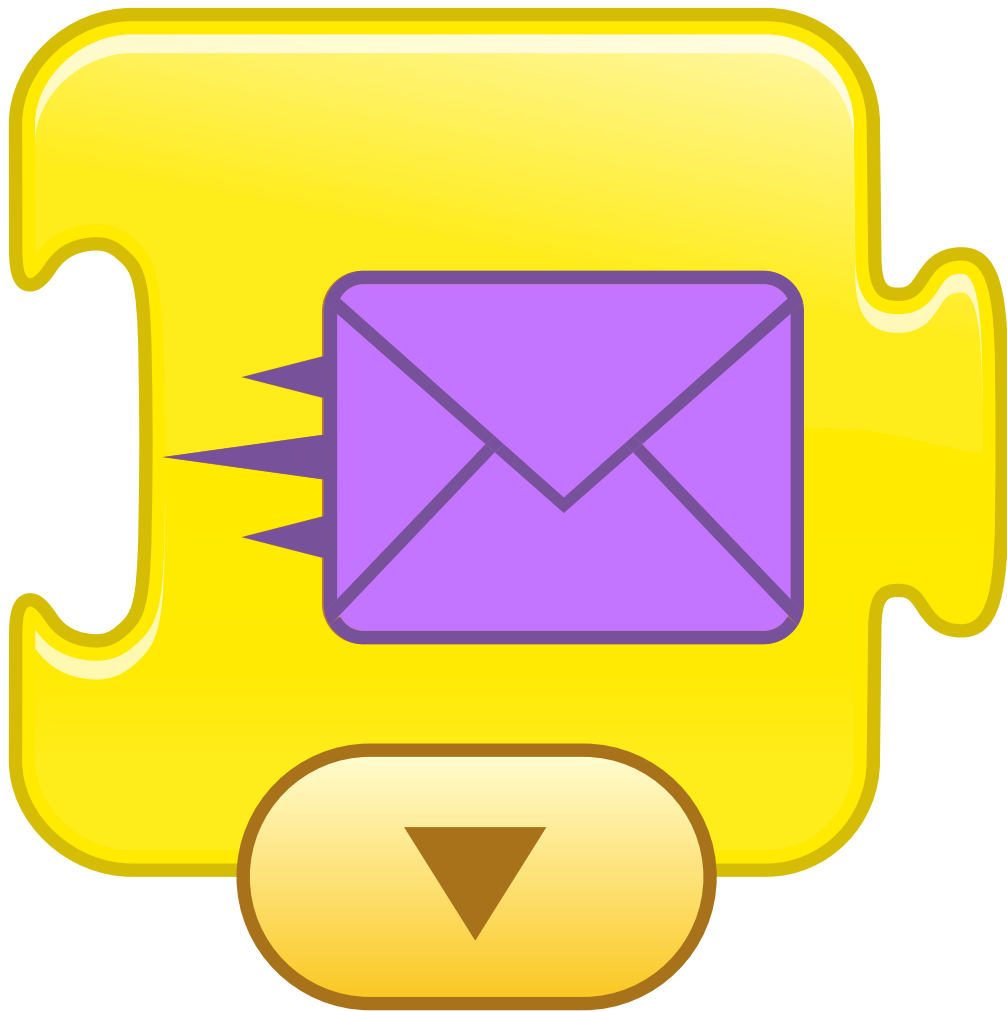
Send Message



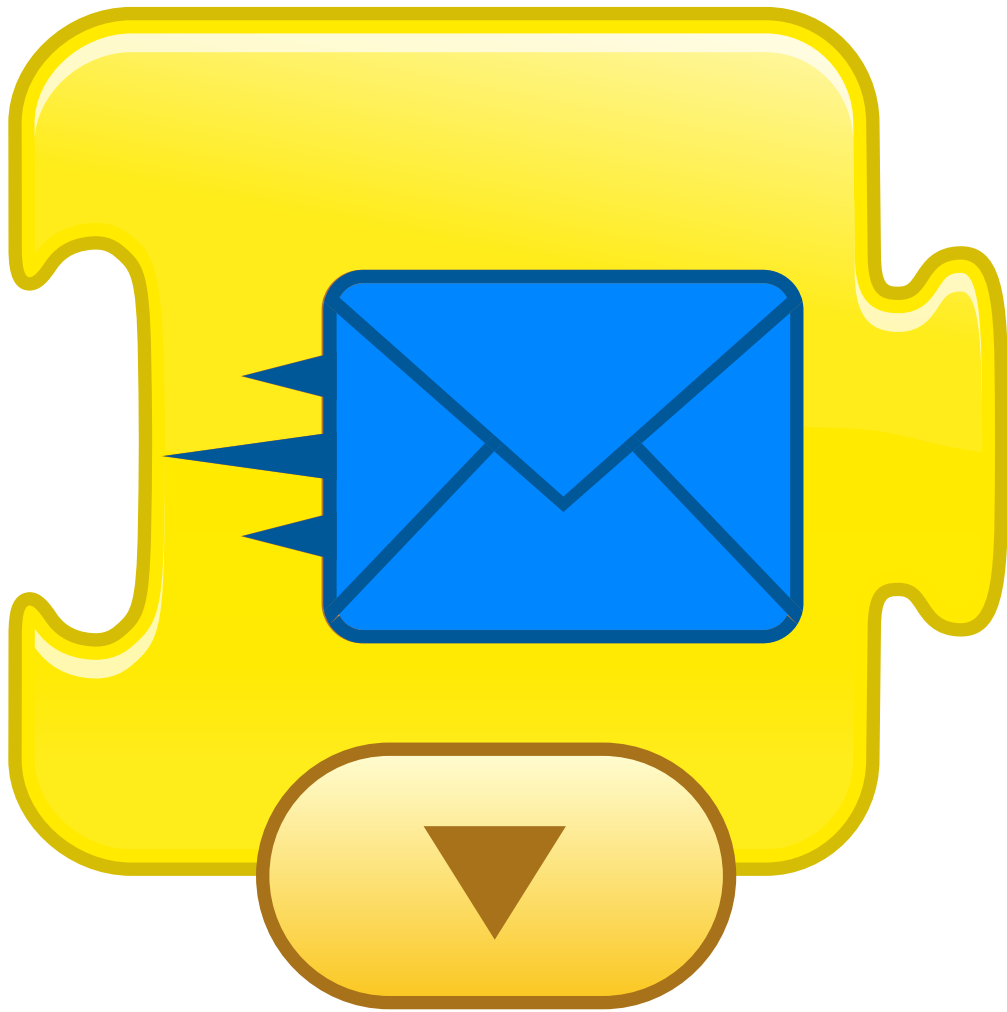
Send Message



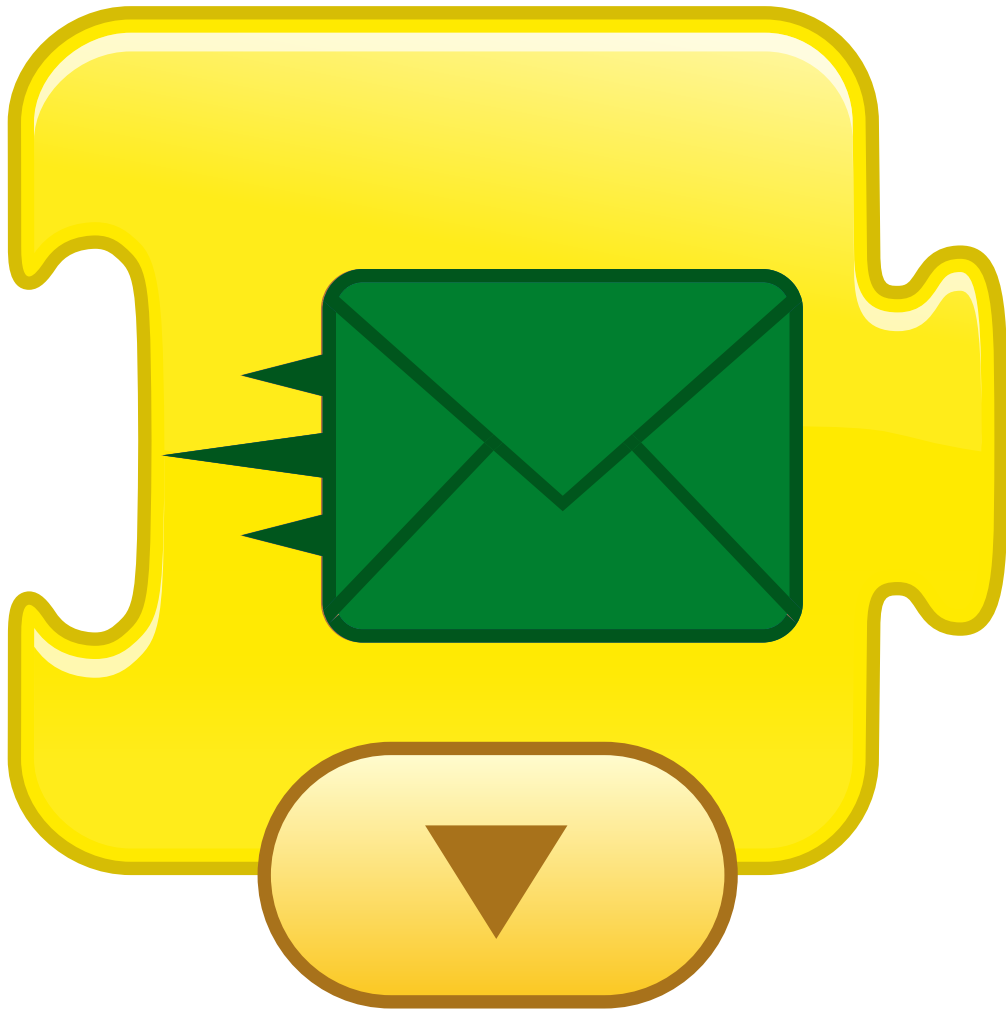
Send Message



Send Message



Send Message



Start on Bump



SOUND BLOCKS

Pop



Play Recorded Sound

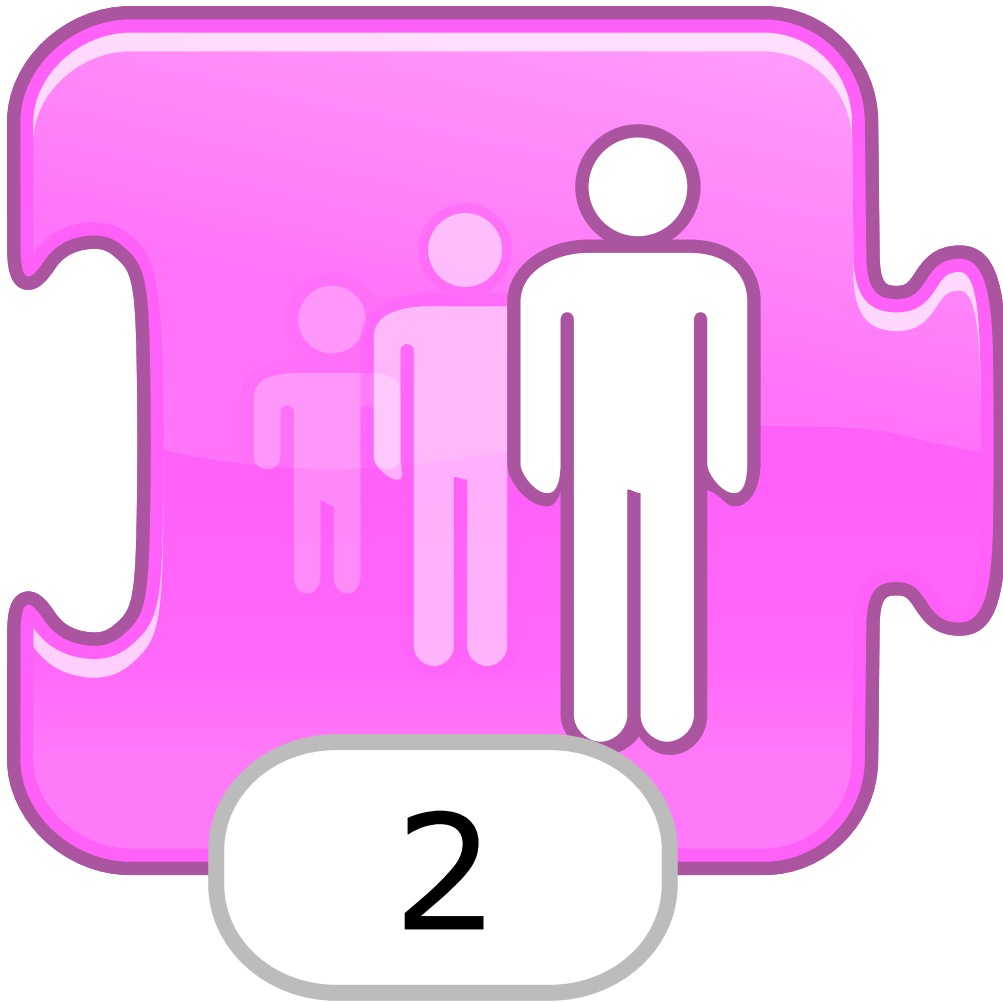


LOOKS BLOCKS

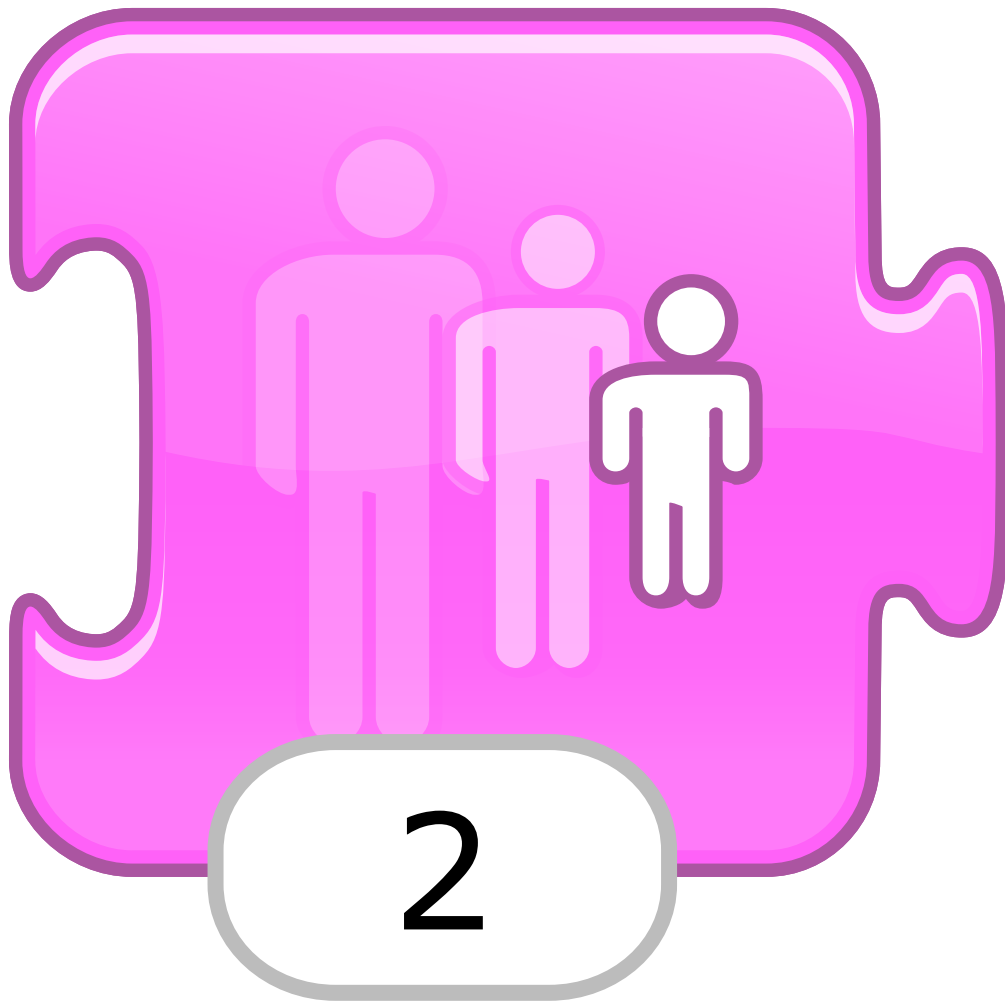
Say



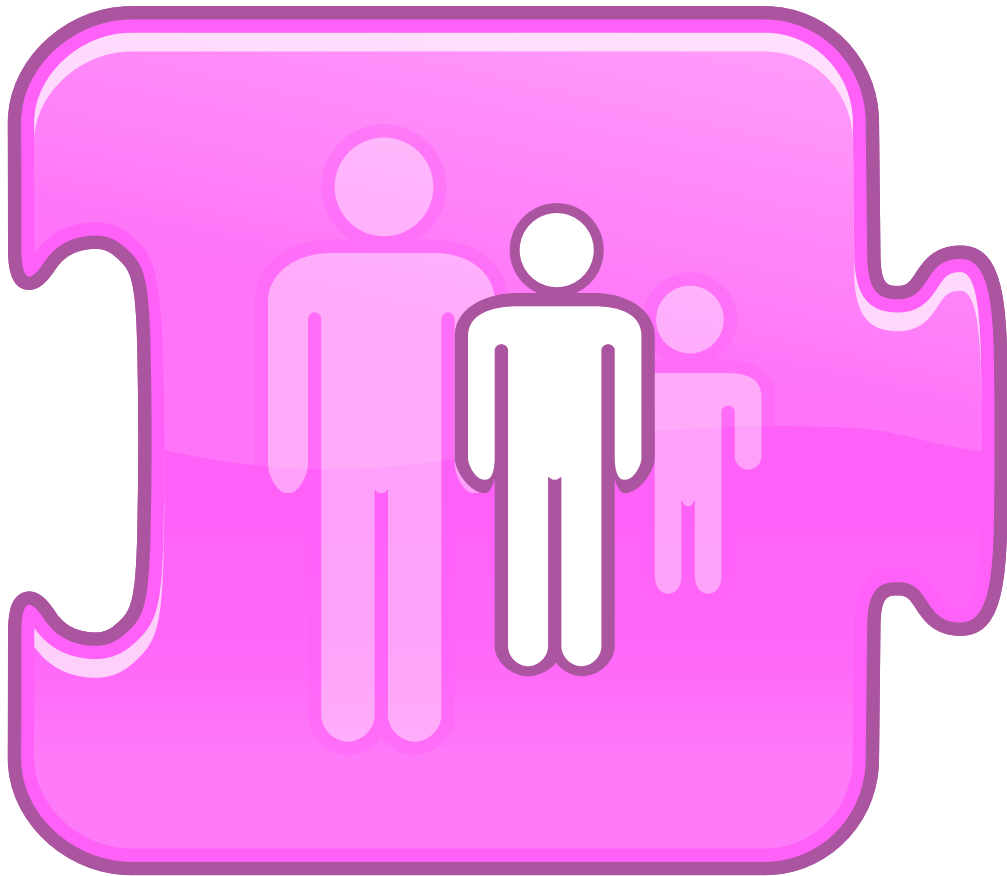
Grow



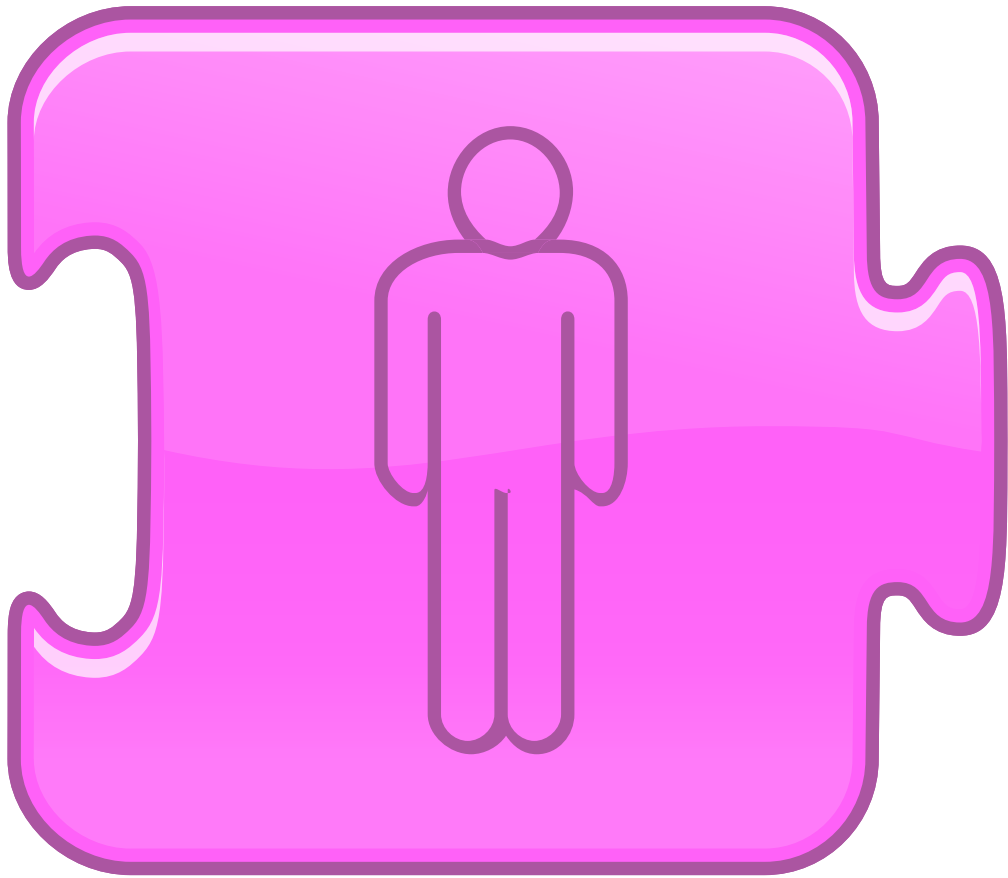
Shrink



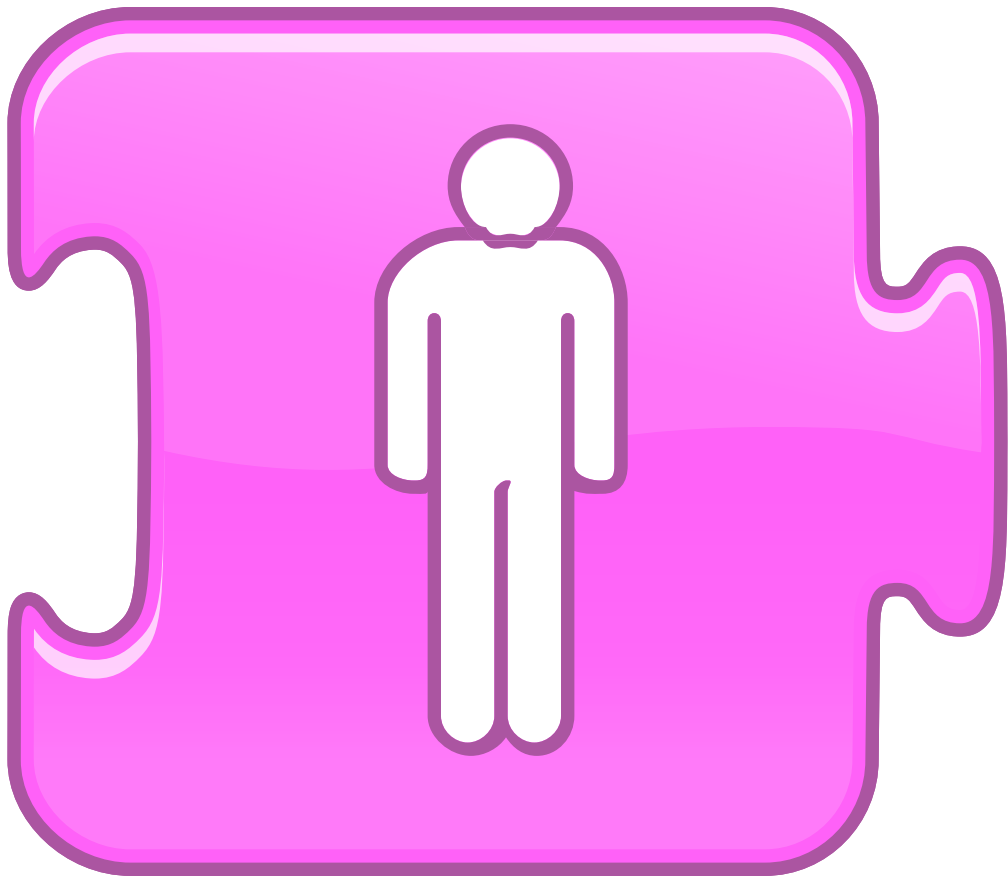
Reset size



Hide



Show



CONTROL BLOCKS

Wait



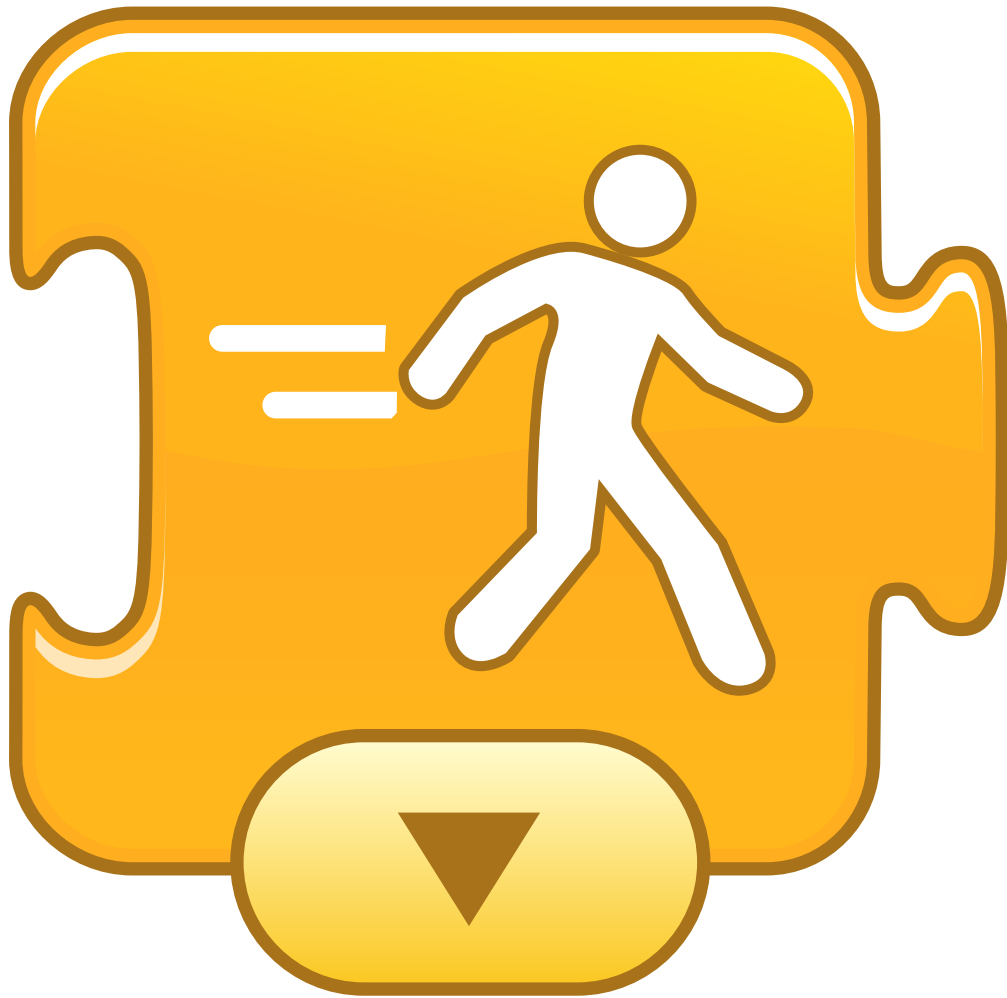
Stop



Set Speed



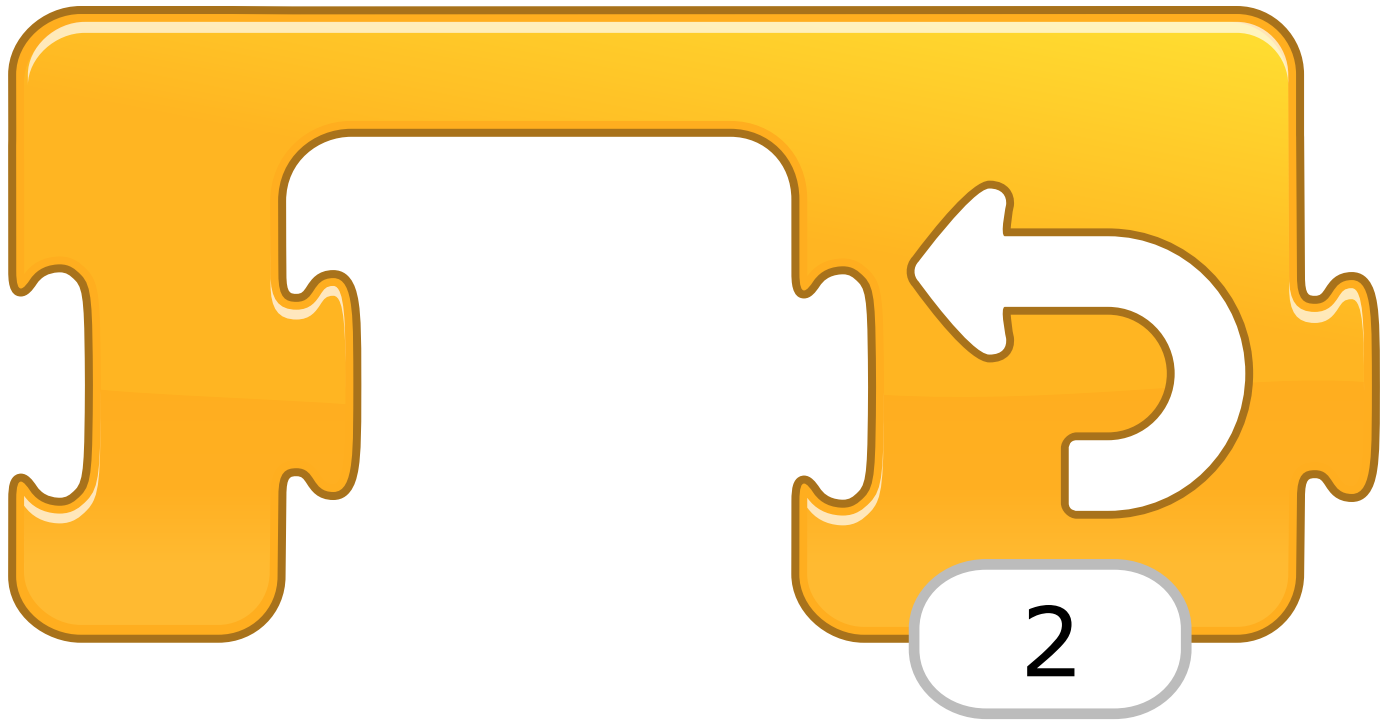
Set Speed



Set Speed



Repeat

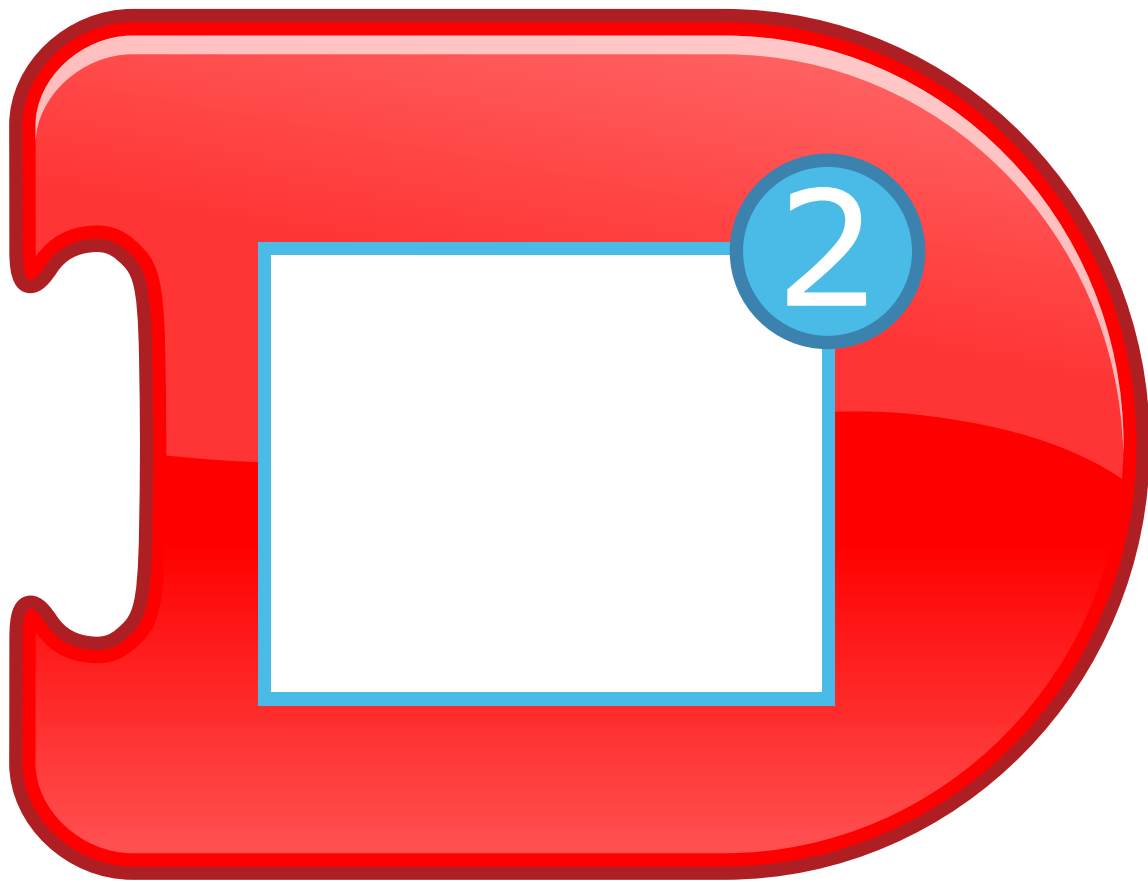


END BLOCKS

Repeat forever



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).

Repeat Forever



Runs the script over and over.

Go to Page



Changes to the specified page of the project.

Appendix B-1. Solve-It Assessment A

Name _____

Date _____

A

Teacher _____

Circle the correct answers.

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


Keeping things the same



Making new things




Breaking something




2. How many times can you go through the writing process and design process?


1 time



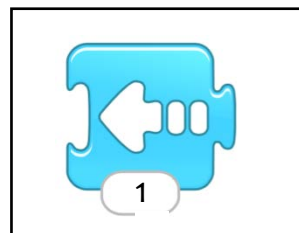
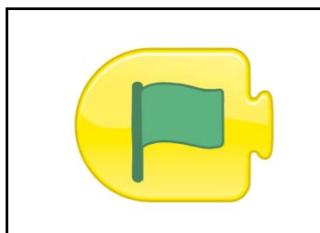
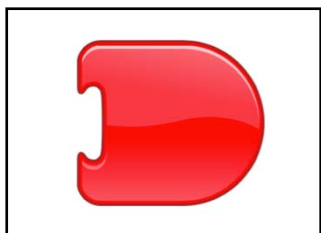
3 times



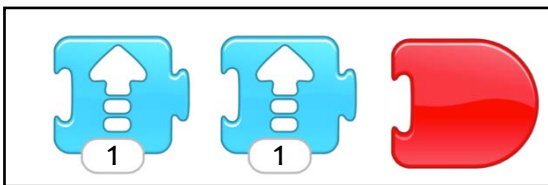
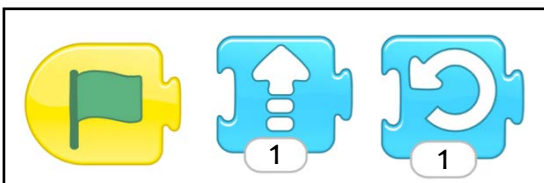
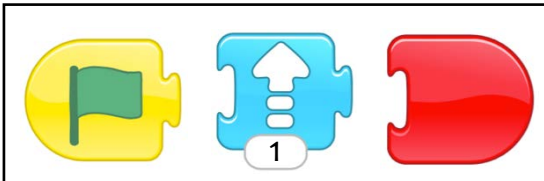
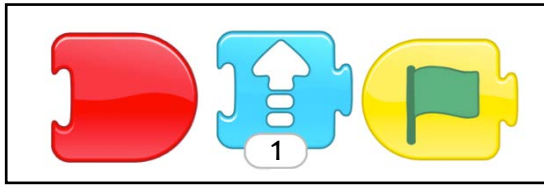
You can repeat the cycle as many times as you need



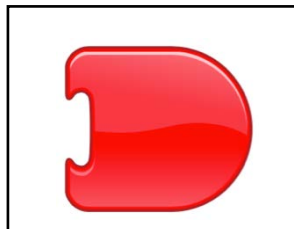
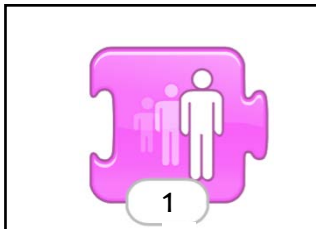
3. Which of these is an end block?



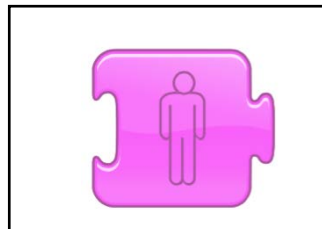
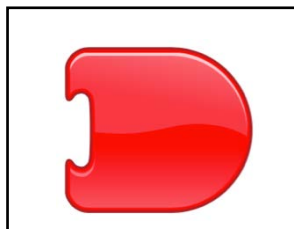
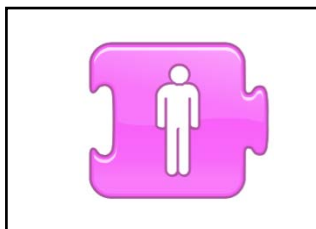
4. Which of these is a program?



5. Which block will make the character grow?



6. Which block will make the character hide?



Appendix B-2. Solve-It Assessment B

Solve-it

Name _____

Date _____

B

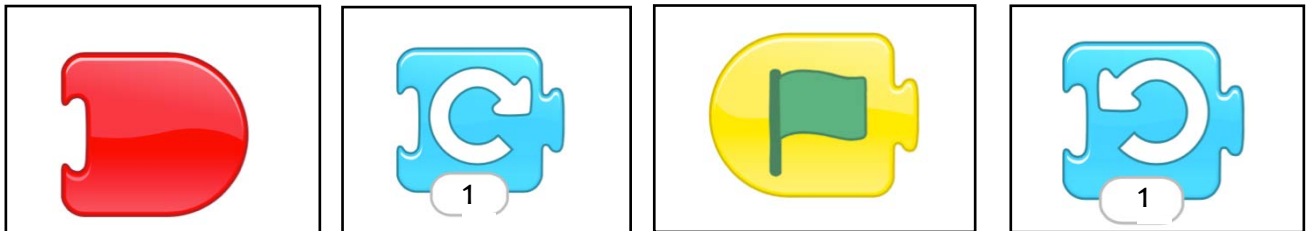
Teacher _____

Circle the correct answers.

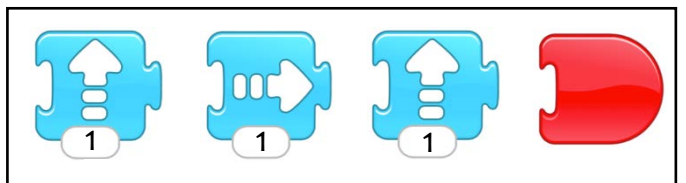
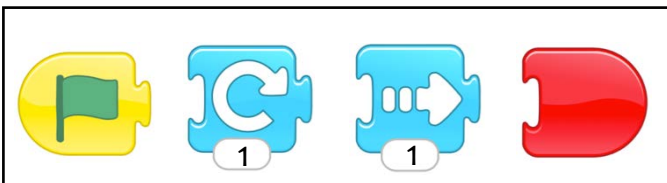
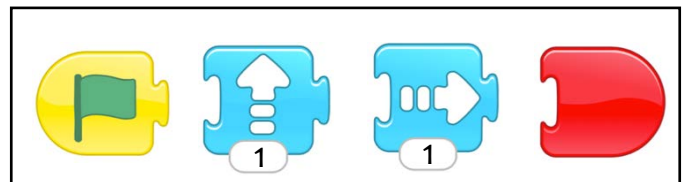
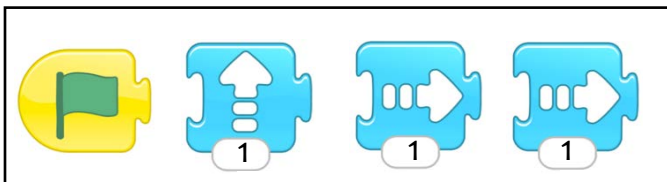
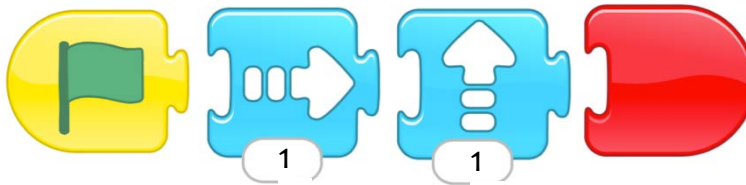
1. My character won't spin and move forward when I try to run my program.



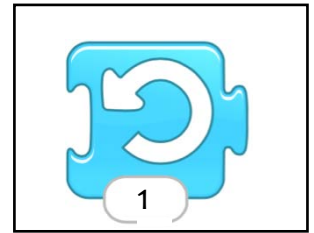
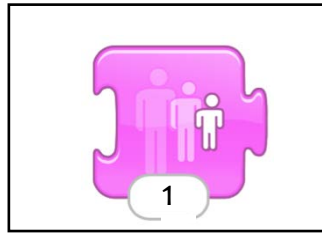
Circle one block that is needed to debug this program:



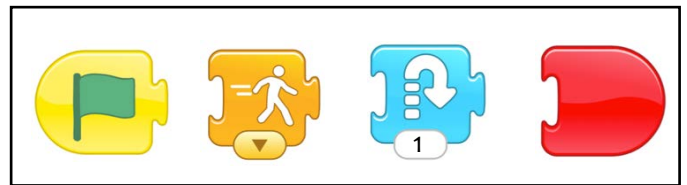
2. How can we change this program so that the character goes up before moving forward?



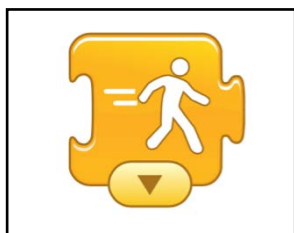
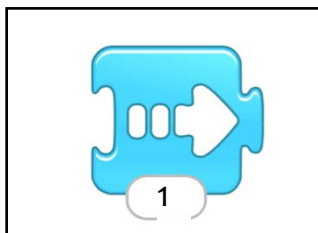
3. Which one of these blocks makes the character freeze?



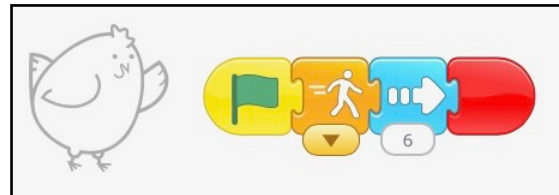
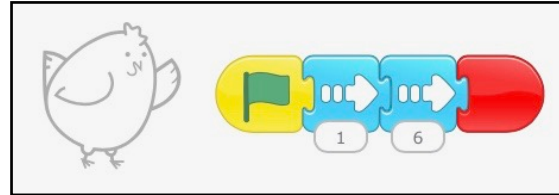
4. Which one of these programs will make the character freeze before jumping?



5. Which one of these blocks control speed?



6. Which one of these programs will make Chicken faster than Kitten?



Appendix B-3. Solve-It Assessment C

Solve-it

Name _____

Date _____



Teacher _____

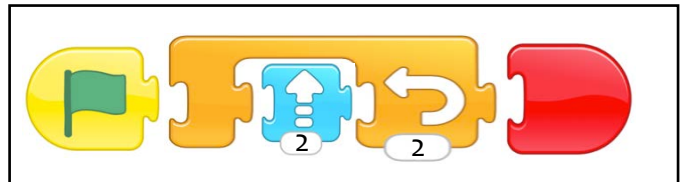
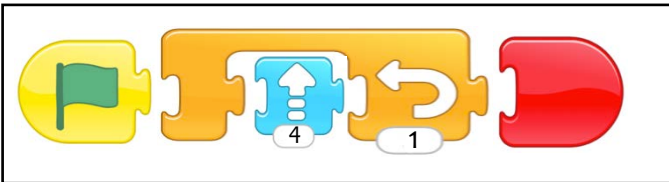
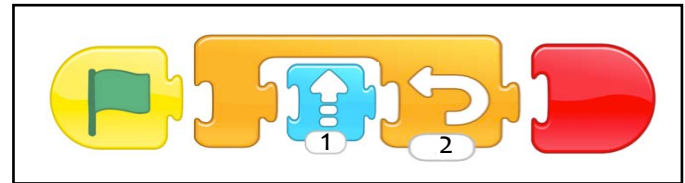
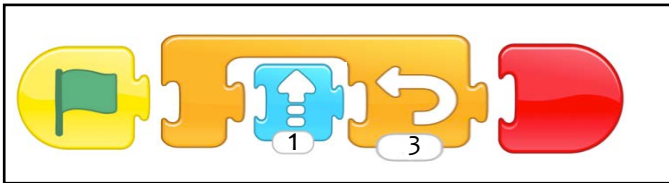
Circle the correct answers.

1. How many times will the character move backwards?

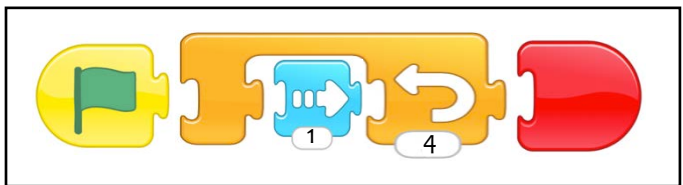
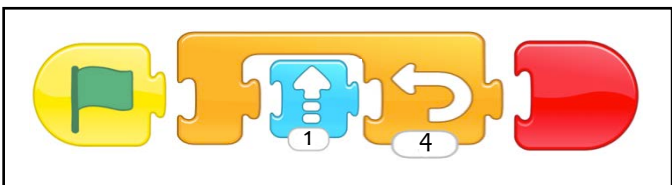
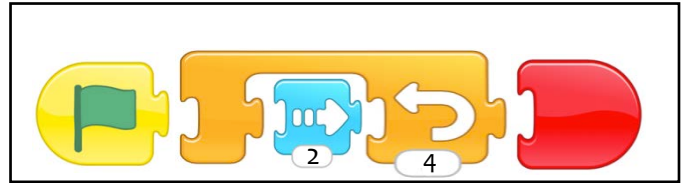
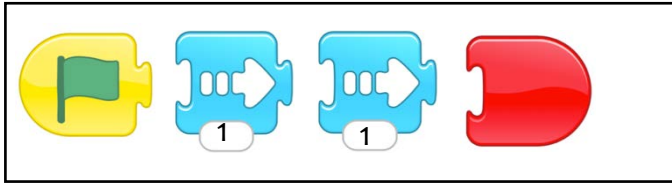
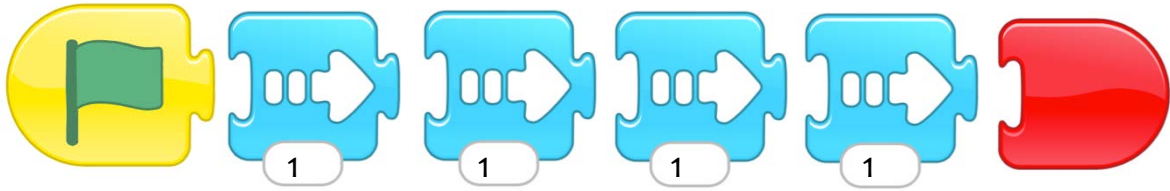


- | | | | |
|--------|---------|---------|---------|
| 1 time | 2 times | 3 times | 4 times |
|--------|---------|---------|---------|

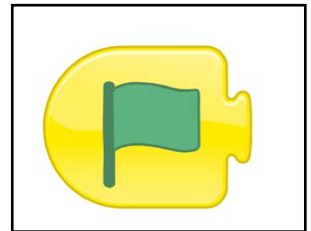
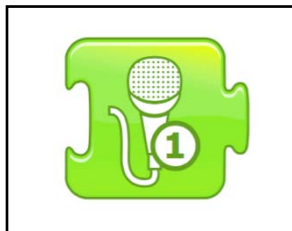
2. Which of these programs will make the character jump three times?



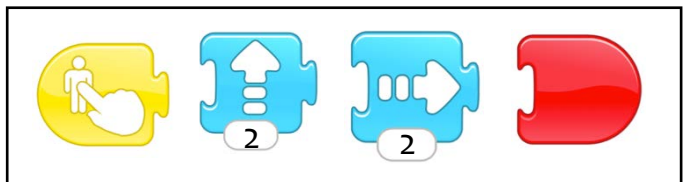
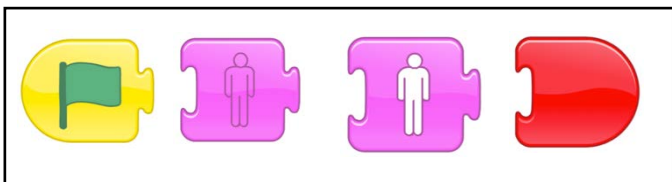
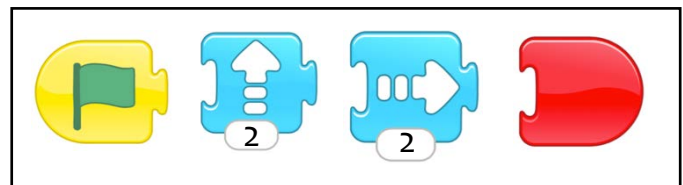
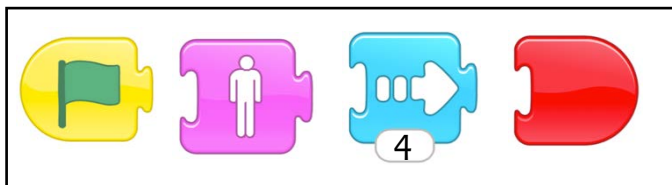
3. Which program does the same thing as this program?



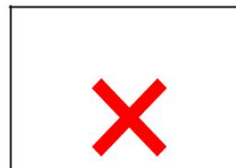
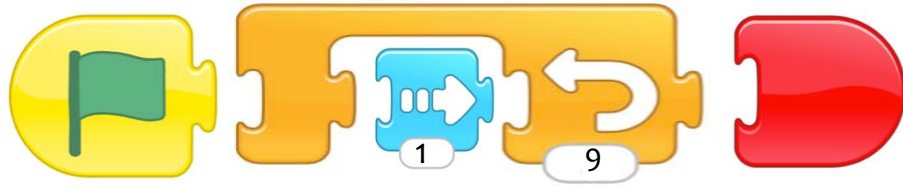
4. Which block lets you record your own sound?



5. Which program does not start until you tap the character?



6. True or False: This program makes the character move forward forever.



Appendix B-4. Solve-It Assessment D

Solve-it

Name _____

Date _____

D

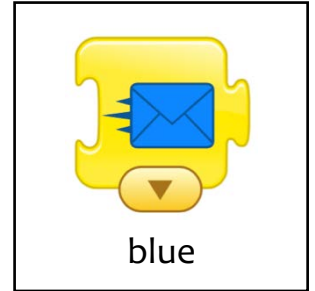
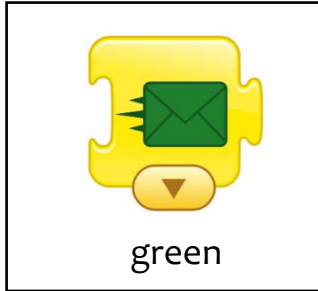
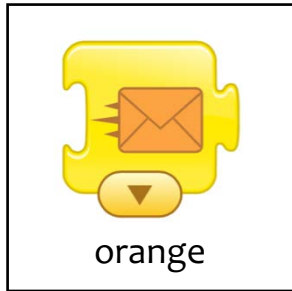
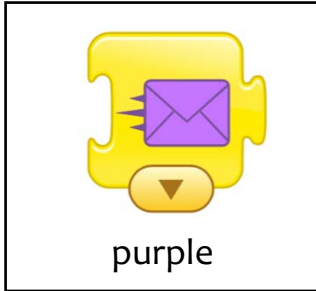
Teacher _____

Circle the correct answers.

1. The program begins with the orange start on message block.



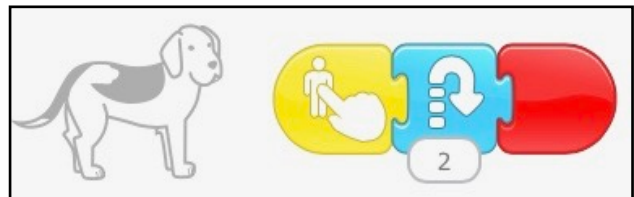
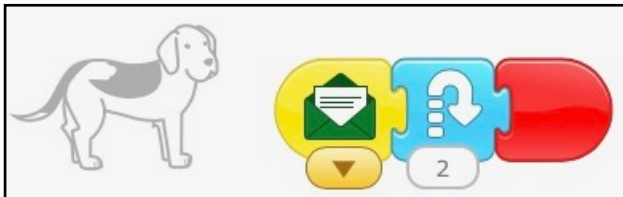
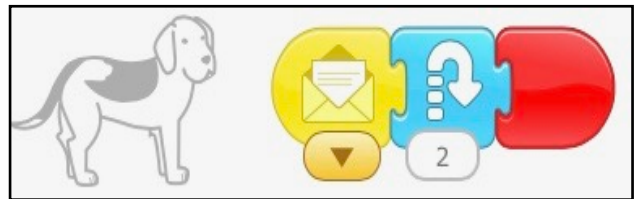
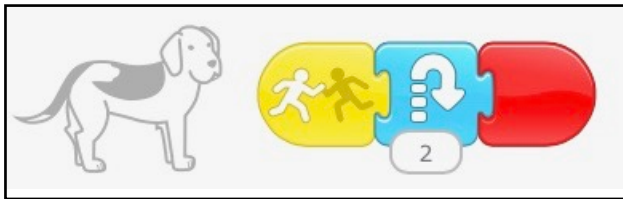
What color does the send message block need to be?



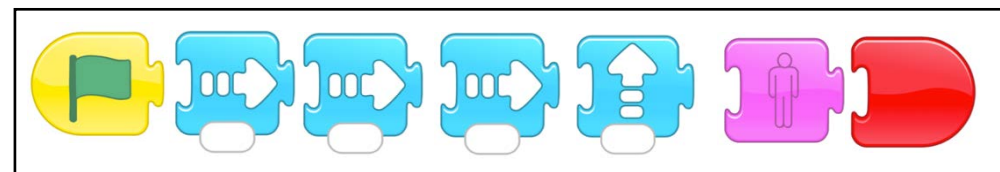
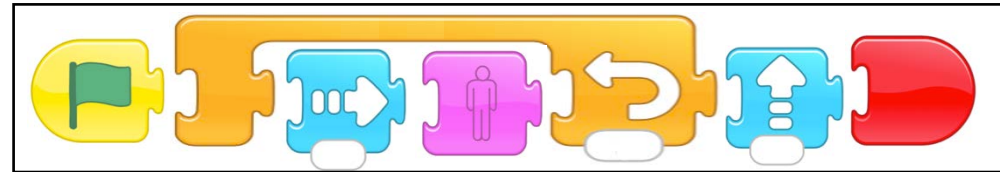
2. Which block only starts a character's program when that character is "bumped" by another character?



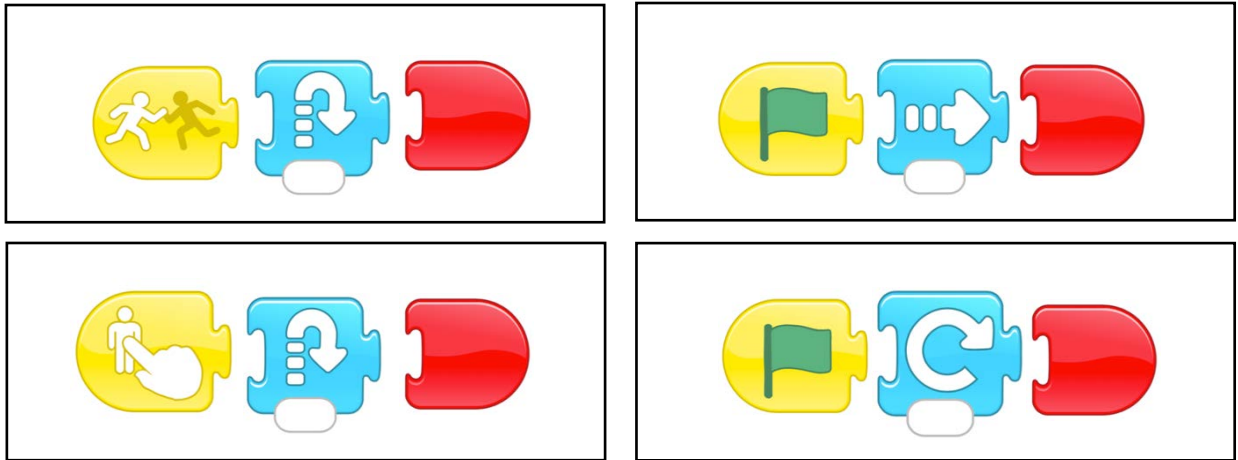
3. After Kitten jumps, I want Dog to jump. But, Dog is not jumping. How can I change this program to make Dog jump after Kitten jumps?



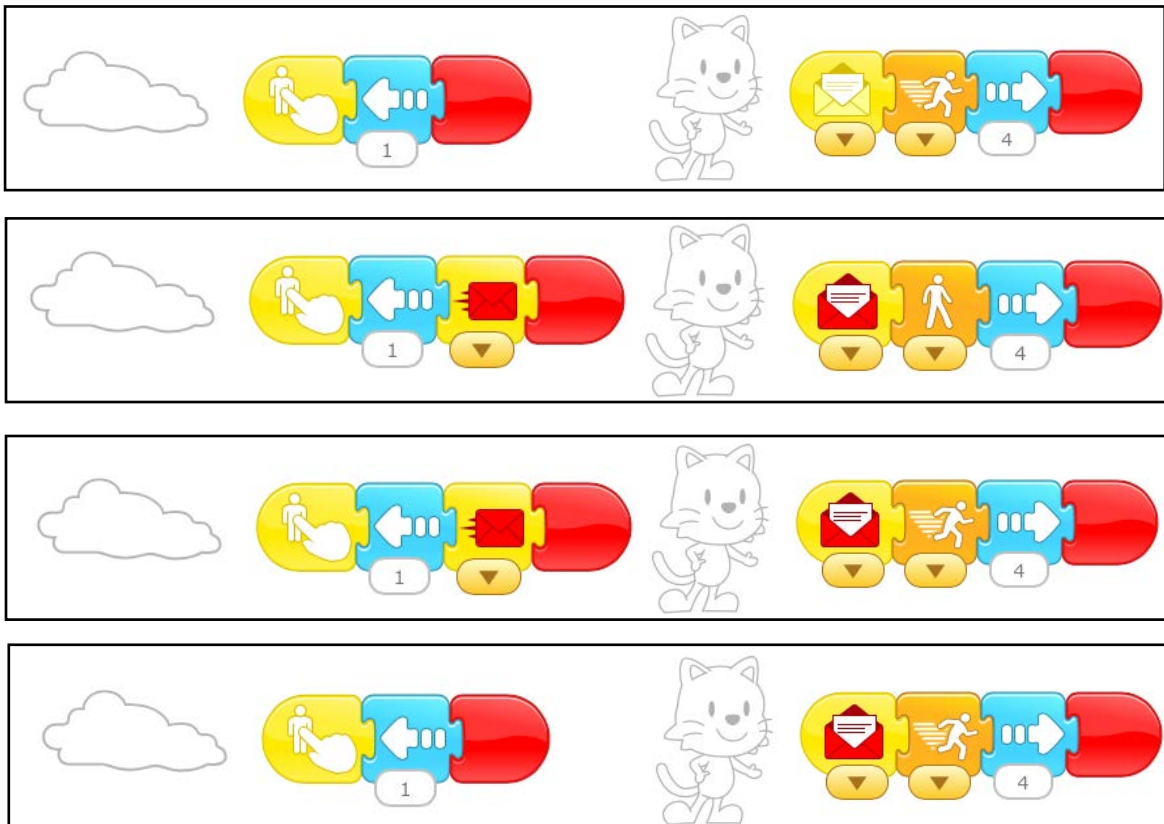
4. Which program has the character move four steps forward, one step up, and disappear?



5. When Kitten bumps into Dog, Kitten bumps into Dog. Which program makes Dog jump when Kitten bumps into Dog?

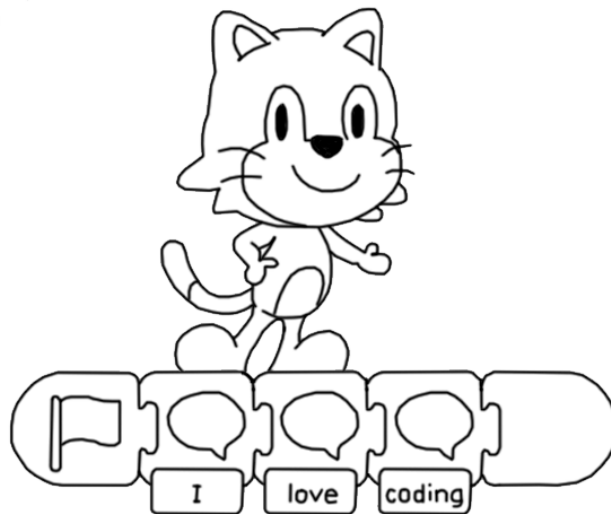
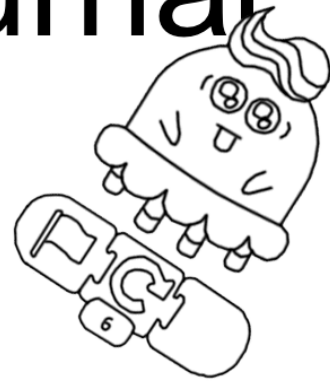
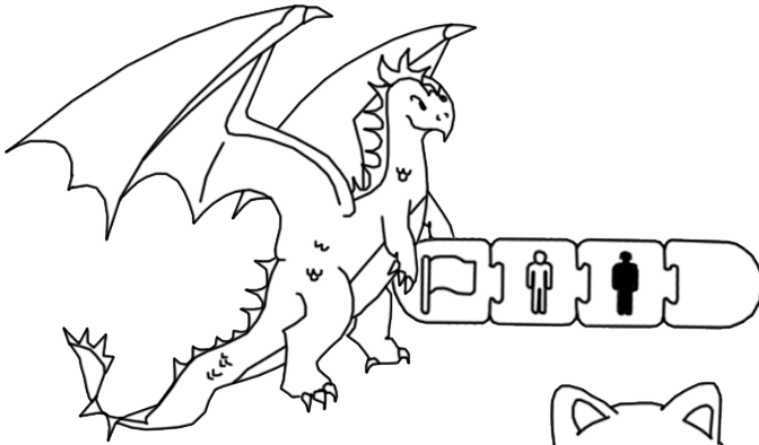


6. Which program has the storm cloud move backwards when tapped and send a message for Kitten to move forward the fastest?



Appendix C. Design Journal

_____ 's
Design Journal



Name: _____

Date: _____

Lesson 1: *Foundations*

What are some similarities between the process of designing a program and the process of writing a story?

How did you use the design process to make your How-To book?

Name: _____

Date: _____

Lesson 1:
Foundations (continued)

Write instructions for how to create the game that you designed. You can also draw pictures to help readers understand your directions.



Name: _____

Date: _____

Lesson 1:

Name: _____

Date: _____

Lesson 2:
What is a program?

Name: _____

Date: _____

Lesson 3:

Composition Planning:



Details about my idea are:

| | | |
|--|--|--|
| | | |
|--|--|--|

Name: _____

Date: _____

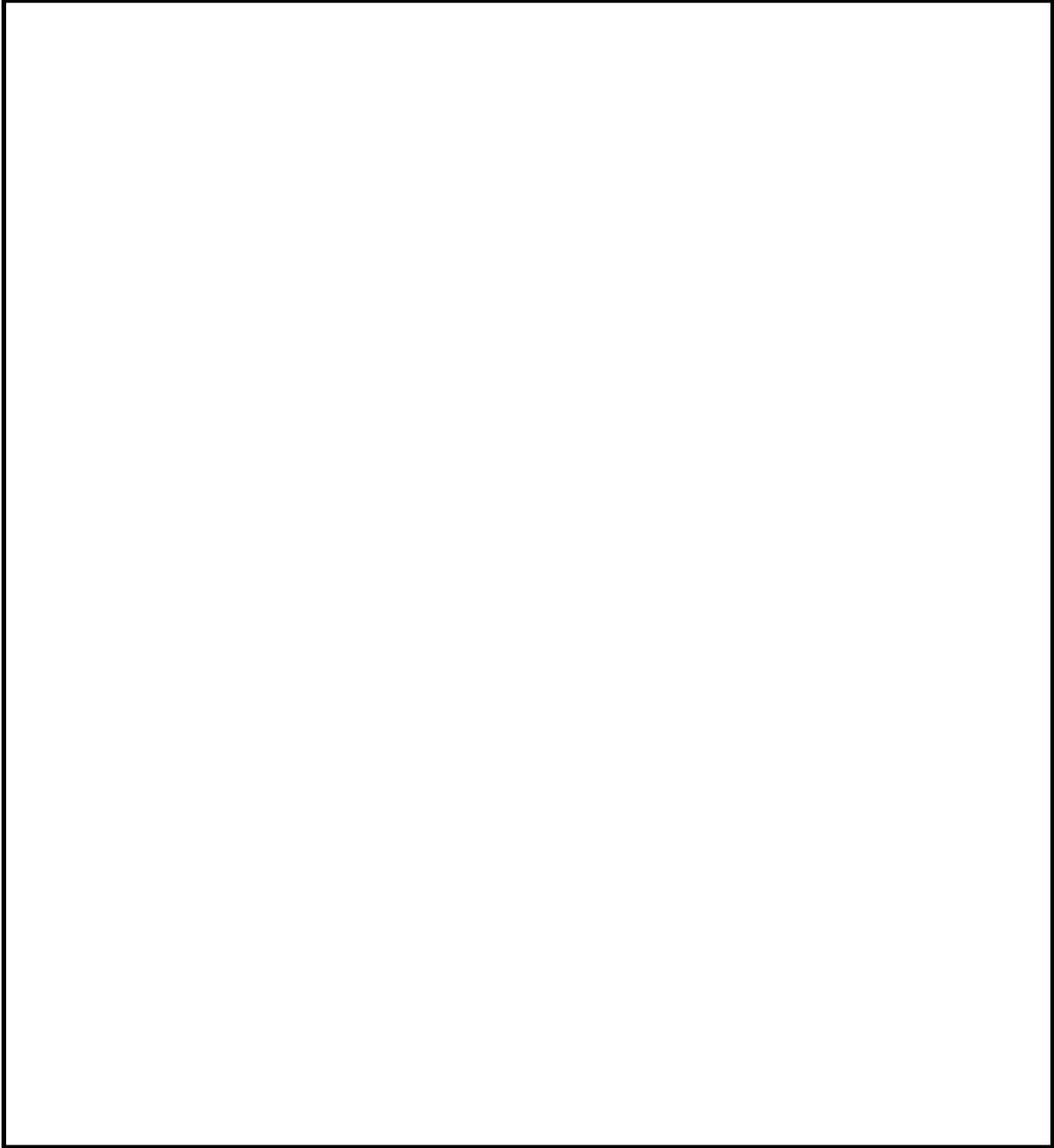
Lesson 3:

Composition First Draft:

Name: _____

Date: _____

Lesson 4: My Character

A large, empty rectangular box with a black border, intended for a drawing or a written response related to the lesson title 'My Character'.

Name: _____

Date: _____

Lesson 4:

Use this blank sheet for any extra planning or writing!

Name: _____

Date: _____

Lesson 5: *Programming*

Use the ScratchJr stickers to write your Hokey-Pokey program here. Make sure the blocks are in the right order!

Which block did you use the most? _____

How many times did you use this block? _____

Which block did you use the least? _____

How many times did you use this block? _____

Name: _____

Date: _____

Lesson 5:



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: _____

Date: _____

Lesson 6:

Name: _____

Date: _____

Lesson 7:
Debugging

Name: _____

Date: _____

Lesson 7:
Details

FAST OR SLOW?

1. _____

2. _____

3. _____

4. _____

5. _____

Reflection:

What was difficult or easy about your Freeze Dance Project?

Name: _____

Date: _____

Lesson 7:

Use this blank sheet for any extra planning or writing!

Name: _____

Date: _____

Lesson 8:

Use this blank sheet for any extra planning or writing!

Name: _____

Date: _____

Lesson 9: *Cause and Effect Part II*

Choose one of the five senses below. Draw a circle around the sense that you have chosen.

See



Hear



Smell



Taste



Touch



Describe a time of when you last used this sense:

Name: _____

Date: _____

Lesson 9:

Use this blank sheet for any extra planning or writing!

Name: _____

Date: _____

Lesson 10:

Use this blank sheet for any extra planning or writing!

Name: _____

Date: _____

Lesson 11: *Final Project*

Now that you have written about your Jungle Dance Party, start planning your Scratch Jr. Jungle Dance Party!

Ask: What activities do you want your characters to do?

Imagine: What will your project look like?

My character is: _____

Its name is _____

Draw what your character will look like:



Name: _____

Date: _____

Describe the kinds of blocks you used in your program:

Test and Improve: Before programmers finish a project, they need to test and improve their work. Use this checklist to see how your program is coming along!

Now it's time to improve and fix your "bugs"! Feel free to make as many changes as you want to improve your program or your robot's decorations.

What changes did you make to your program?

Name: _____

Date: _____

Lesson 12:

Name: _____

Date: _____

Lesson 12:

Describe your final Scratch Jr. program. What was it like? Was it a story? A game? Something else?

What characters and backgrounds did you use in your program? How did using those parts make your program come to life?

What was your **favorite** thing about working on your program?

What was the **hardest** part about working on your program?

Name: _____

Date: _____

Lesson 12:
Final Projects

Name: _____

Date: _____

Name: _____

Date: _____

Student Perceptions Survey

1. How much did you like doing these activities?



Not at all!



Not really



It was okay



Liked it



Loved it!

2. How much do you think the whole class liked doing these activities?



Not at all!



Not really



It was okay



Liked it



Loved it!

3. How much did you learn about coding?



Nothing!



A little bit



A fair amount



Quite a bit



Learned a lot!

4. How important do you think the things you learned are going to be to you later in life?



Not at all important!



Probably not



Maybe



Probably



Very important!

5. How would you feel if you had the chance to do this again?



Awful



Not very good



Okay



Pretty good



Great!

6. What was your favorite part about these activities?

Appendix D. Collaboration Web

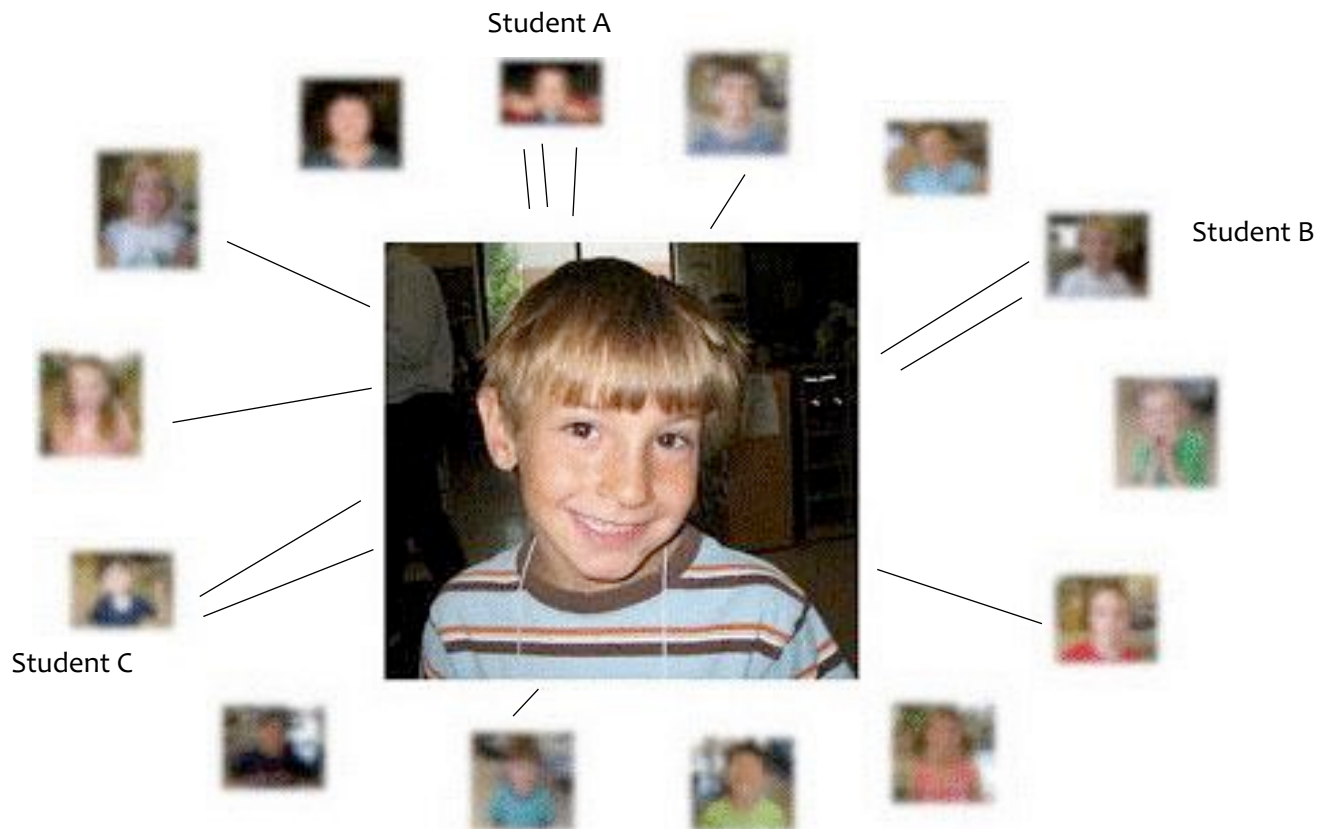
Appendix D. Collaboration Web

A collaboration web is a tool for students to recognize peers who have helped and supported them in different ways, such as working together on a common task, lending or borrowing materials, programming together, etc. Students will create a Collaboration Web during Lesson 11: The Jungle Dance Party Project and then later write thank you letters to the three peers with whom they have collaborated the most.

Directions:

1. Obtain headshots of each student in the class.
2. Create individual printouts with each student's photograph in the center of the page and the names and photographs of all the other students arranged in a circle surrounding the central photo.
3. Whenever you observe students collaborating during the final project, ask students to draw a line from their photo in the center to the photo of the other students with whom they collaborated.
4. During Lesson 12, ask students to count the number of lines they have with each student. Ask students to write thank you letters to the three students who have the most lines drawn to their photos.

Sample Collaboration Web:



The student in the center will write thank you letters to Students A, B, and C.

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.