SCRATCH LESSON PLAN

Getting Started with Scratch, Experiencing Creative Learning: Paper Planes, Turtle Graphics, and Computational Concepts

In this lesson, we will explore the computational concepts of decomposition and sequence through both unplugged and digital activities in Scratch. First, we will attempt to fold a paper plane using only written/verbal instructions from the facilitator. Next, we will reflect on the importance of clear and detailed instructions and sequential order, as well as the process of breaking a task down into smaller instructions. Lastly, we'll build on our understanding of sequence and decomposition as we attempt to create a shape drawing in Scratch using the Pen Tool extension. Using the Pen extension to create a visual artifact enables learners to observe results, debug, and iterate, all while having fun and gaining real-world problem solving and critical thinking skills!

Audience: Classroom Teachers, Instructional Technology Specialists, Library Media Specialists, Informal Learning Environments

Time: Approx 1 hour total

- Part 1: Unplugged Activity - Paper Planes - 10 min
- Part 2: Making Connections (Reflection, Connection, and Transition) - 10 min
- Part 3: Scratch Activity - Turtle Graphics in Scratch - 30 min
- Part 4: Reflect and Share - 10 min

This is example timing for one 60 minute session, but you may opt to spread out over two separate sessions if learners need/want more time to tinker or you want to take more time to reflect with your class after each section.

For aligned standards, please see the last page of this lesson.

Objectives (Learners Will):

- Analyze a task and determine how to break it into smaller individual steps (decomposition)
- Articulate the significance of details and the sequential order of instructions (sequence)
- Evaluate problems/identify bugs, test solutions, and iterate on a program
- Express creativity and individuality within the context of a structured activity (through optional opportunities for individual creativity/expression)
- Reflect on the design process, individually and/or collectively
- Communicate and share their projects with their learning community

Setting Expectations: Get Ready to Get Stuck! Celebrate Growth Mindset & Variety!

- As a facilitator, know going into this activity that **frustration/getting stuck and the need to problem-solve are intentionally part of this activity**! So, too, is variation and reflection.
- As you'll see, we are **giving intentionally vague directions** when folding the plane, for example, and not providing visual clues in order to **create dialog around the importance of clear and detailed instructions in proper order**. This objective is reinforced when learners turn to Scratch to create written/block instruction for their program. Our purposeful vagueness also gives learners a chance to reflect on what additional knowledge they had to employ to understand the activity and discuss how to change the instructions to add key details.
- Whether you are "coding" a sandwich build, a dance, a paper plane, or a Scratch project, **the computer can only perform the actions you code. Decomposition and sequence are important!**
- We want to **celebrate a culture of getting unstuck**. Did you "FAIL"? "FAIL" is a First Attempt In Learning. Celebrate what was learned from missteps: Did you forget to put the pen down? Did you use other blocks than those we suggested? Did your code or shape look different than a neighbor? Great! Let's talk about it and learn from it.
- While the result of this lesson is a paper plane or a shape, how we get there and what they look like may vary. And that is great! This leads to our intentional reflection that **there is no one right answer/solution in this lesson**. Some planes may fly farther or straighter than others, just like some programs may be more efficient or may be better in loops, etc., but that doesn't necessarily mean one is right and one is wrong or better than another.

Localize This Lesson

While this lesson focuses on paper planes and turtle graphics as the vehicles to explore decomposition and sequencing, you can **take these principles and apply it to other activities that are of interest to your learner**. For example, you could:

- Code a sandwich: In pairs, have one person write pseudocode (written language instructions) for making a sandwich like a peanut butter and jelly sandwich. The other participant serves as the "robot" and must follow the instructions exactly (if they don't say to open the jar or take the bread out of the bag, you can't…and hilarity will ensue).
- Try our unplugged Lego maze activity in pairs or small groups to practice writing an algorithm.
- Code a dance. Repeating steps can lead to conversation about loops that relates to creating patterns with shapes.
- Read *How to Code a Sandcastle* by Josh Funk and discuss learner observations. One of our Scratch Team members even made a project inspired by this book. What would you create/code?

Part 1: Unplugged Activity - Paper Planes

Decomposition is when learners break a complex problem or system into smaller parts like a series of steps for a task, and sequencing focuses on the proper order in which to perform those steps. Making a paper plane provides hands-on experience with both of these concepts.

The Set-Up

You don't need to be an expert in paper plane folding. Learning and debugging alongside your learners adds to the fun and educational experience!

The facilitator can lead an entire class in the activity and reflection. Or the class can be broken up into small groups, with one person designated as the group facilitator (in charge of reading the instructions) while the rest perform the steps of the activity. Then, all the groups may come together for reflection or reflect in their small groups.

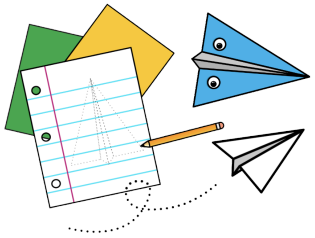
Materials Needed:

- A piece of paper per individual or group (the size of the paper and the material or condition of the paper we are leaving intentionally vague; can be a discussion/observation point)

Paper Planes (10 minutes)

Read out the pre-written set of instructions below. The facilitator is acting as the programmer, and the learner acts as the computer. Try not to provide learners with any more information than what is written in the instructions (some of which are intentionally vague). It is helpful (except in the case of accommodations) that you do not show images of how to fold.

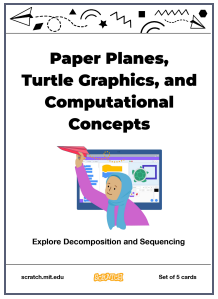
Step 1: Read the instructions *(to the right)* aloud. →
Step 2: The learner(s) should try to fly the plane and determine if the process worked. *(See the Debugging and Reflections sections below.)*

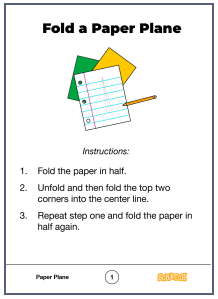


1. **Fold the paper in half.**
2. **Unfold and then fold the top two corners into the center line.**
3. **Repeat step one and fold the paper in half again.**

Resources:

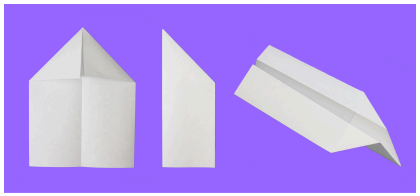
- Paper Planes, Turtle Graphics, and Computational Concepts Lesson Coding Cards (Student-Facing Cards) - printable cards students can use to follow along with the lesson)



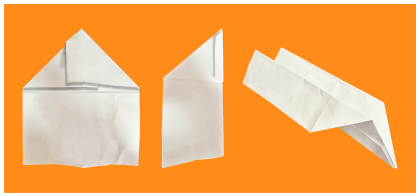


Important Facilitation Reminders

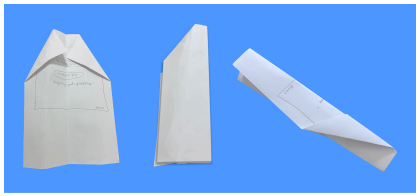
- These instructions are intentionally vague! Remember, the facilitator is acting as the programmer, and the learner(s) as the computer. The computer takes everything literally and only knows what you give it, and in this case, we aren't providing additional information or reference images unless needed for accommodations.
- Even though these instructions are vague, the facilitator and learner(s) may come into this activity with some pre-knowledge of what planes look like that could inform how the instructions are interpreted. That can make for a great reflection point. Did anyone need to rely on additional knowledge to interpret a vague instruction?
- Once planes have been created, highlight and celebrate the similarities and differences and talk about it! See the debugging and reflection prompts.
- Did you have fun? Was this engaging? Excellent! Success looks like a great reflection discussion.



Perhaps coming to the lesson with some pre-knowledge about paper planes, this learner folded it as the programmer envisioned.



This learner folded the paper in half along the short side, then folded the corners to that middle fold creating an overlap with the triangles.



This learner folded the paper along the long side, but folded the corners in only slightly, leaving the plane nose flat versus pointed.

Opportunities for Individual Creativity/Expression

Option 1:

Think of a simple activity you know how to do well (kick a ball, make a food, go through a morning routine, etc.) **and write out the steps to describe it to a computer.** Share these steps with a peer, in a small group, or with the class. Determine if they are ordered correctly and clearly enough/contain enough detail that one could follow without asking for additional clarification.

Option 2:

Write a message on your paper before folding. What message would you share with learners in a different part of the world or with your future self or with a classmate if you could throw your plane far enough to reach them?

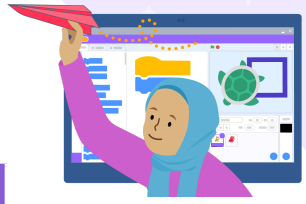
Option 3:

Personalize your plane so it can be identified amongst the other fliers by adding designs, your name, characters, etc.

Debugging or Ah-Ha Moment Prompts

These can be answered out loud or written down.

- Does your plane fly? How far? What was the flight path (straight, in a circle, up then down)?
- Does the size of the paper matter? Does the material (tissue paper, construction paper, printer paper…) matter? Does the condition of the paper matter (is it crisp or wrinkled)? Experiment! Test different materials and conditions and see the results.
- Can you debug any issues? How could the instructions be written in a different way to help you achieve different results?
- What additional steps would you add to make your plane fly straighter or fly further? Share your thoughts!* *(*Note: This is an opportunity to empower the learner(s) to become the teacher and share any additional knowledge or tinkering ideas they have.)*



Part 2: Making Connections

Reflection Prompts (10 minutes)

These can be answered out loud or written down.

- Does your plane look like the others in the room? Compare similarities and differences.
- Did you interpret one or more instructions differently?
- Why might the instructions have been understood differently? Did anyone need to rely on additional knowledge to interpret a vague instruction?
- What was fun about this activity?
- What struggles or frustrations did you have during this activity?
- If you wrote a message or decorated your plane, what was the significance of what you chose?

Connection and Transition Point

When we created the paper plane, we ended up with a physical artifact that we can observe in order to debug or iterate on it. Now, we'll be attempting to create a shape in Scratch and use the Pen extension to create a visual artifact. Being able to see the trail of the object as it moves through the code sequence can make it easier to identify problems (debug) and iterate on the program.

In Scratch, we are giving the computer only written instructions and no other references. That is the environment we were trying to replicate while doing the unplugged activity when we only provided verbal or written text instructions and no reference images of how to fold.

Key Points:

- Details matter (Fold along the long or short side?)
- Sequential order matters
- We can't make assumptions about knowledge (for example, we can't assume the computer knows to do something). The programmer has to be very explicit in our instructions.

Part 3: Scratch Activity - Turtle Graphics in Scratch

As learners now turn to coding shape drawings in Scratch, they are creating what is known as “turtle graphics” (named after a class of educational “turtle-like” robots that often came with pen mechanisms, which allowed the programmer to create a design on a physical sheet of paper). These digital pen drawings are created when a cursor (the "turtle") draws on a plane (like the Scratch stage). Turtle graphics employ geometry concepts and focus on direction, location, and repetition/looping.

The Set-Up

You don’t need to be an expert in drawing with the pen in Scratch! We suggest the facilitator walk through the activity from the perspective of the learner. As facilitators, it can be powerful to model getting stuck and not having all the answers right away. Debug and iterate along with learners, and participate in the play and tinkering. By walking through the exercise from the learner’s point of view first, facilitators develop empathy for Scratchers approaching a new topic, and it might spark debugging and reflection questions of your own.

The facilitator can lead an entire class in the activity and reflection. Or the class can be broken up into small groups or pairs to practice pair programming. Then, all the groups may come together for reflection or reflect in their small group.

Materials Needed:

- A computer operating Scratch. The Scratch portion of this activity can be done with the online or offline editor (the Pen extension is available in both).

Resources:

- Turtle Graphics: Using Pen Blocks (Video) - Prior to facilitating the activity, you may want to watch our video. You may also opt to share it with learners.
- Paper Planes, Turtle Graphics, and Computational Concepts Lesson Coding Cards (Student-Facing Cards) - printable cards students can use to follow along with the lesson

Turtle Drawing (30 minutes)

Step 1: Open the Scratch interface and pick any sprite from the sprite library. *(See facilitation notes below.)*

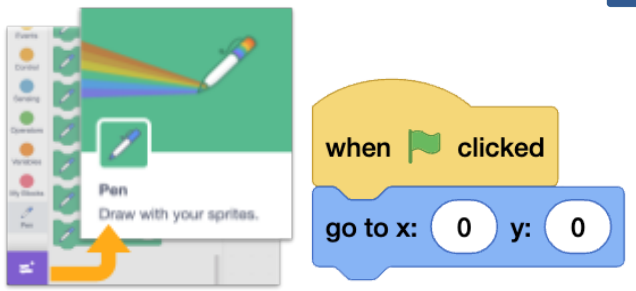
Step 2: Add the Pen extension by clicking on the extension menu in the lower-left corner of the project editor and choosing "Pen." With Pen blocks, your sprite will act as the "pen" (the turtle) and it can draw lines on the stage, which acts as the "paper" (the plane).

Step 3: Add a starter script *(to the right)* to position your sprite center stage each time the program is run. →

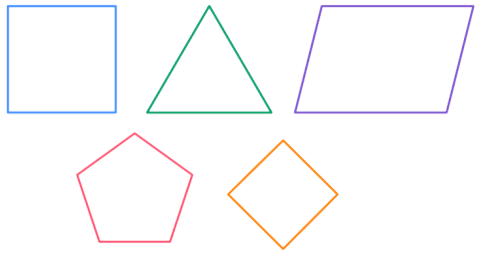
Step 4: Now, look at these shapes *(to the right)* and pick one to recreate in Scratch using the Pen blocks. →

Step 5: Study your shape. What do you notice? Are the sides the same length or different length? Are the angles the same or different? Are there any points where the steps you'd take might be the same/repeat?

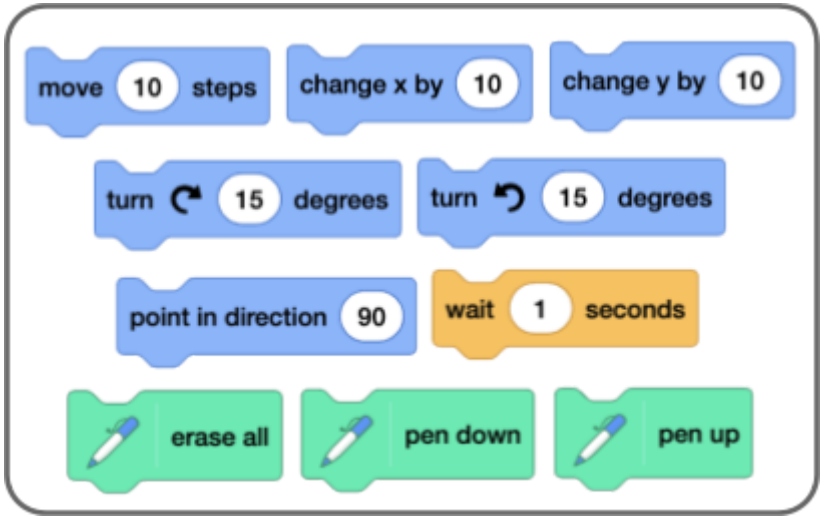
Step 6: Focus on a few Motion and Pen blocks *(to the right)* to recreate the shape in Scratch. You don't have to use all of these blocks, and you can use any of these blocks as many times as you need. *(If learners are not familiar with Cartesian Coordinates, you may opt not to have "change x by" and "change y by" as block options. Or include them and let them tinker with them! You might also want to share an angle chart for those new to angles. See additional facilitation notes below.)* →



Scratch Extension Menu and project starter script.



Shapes to Recreate



Motion and Pen Blocks to Use

Step 7: Imagine if I was poised here with my pen in the air to draw a shape. What is the first thing you would tell me to do? What is the second thing? Third?... Keep in mind, when we folded the plane, we needed to have clear instructions and break the task down into smaller steps. Use the Scratch blocks to create each step.

Step 8: Experiment to see what each block does by clicking on each block in the block palette or on the script area. Play with the values and see what effect that has. Then, combine the blocks to start your program. There is often more than one solution!

Step 9: Test and debug.

Important Facilitation Reminders

- It may be valuable to have learners work in pairs so they can help each other through debugging.
- Allow learners to explore those motion and Pen blocks before explaining what any of them do. Ask learners to limit themselves to these blocks to start but they can use any of these blocks multiple times. They can add additional blocks after they create their initial shape.
- Some sprites are large and may hide the pen line if it is short. Some sprites are round and it is hard to tell the direction they face. It is okay if the learner chooses a sprite like that. Use it as an opportunity to debug. Let learners experiment and tinker and get stuck for a bit!
- Complex shapes and circles or spirals can be more challenging to draw, so encourage beginners to start with the simple shapes shown and more advanced learners can challenge themselves to create more complex shapes.
- Once the pen is down, it is down until the program is told to raise the pen. Clicking the green flag does not automatically erase the lines on the stage or reset the pen up or down. If and where those blocks are placed in the sequence has a great effect on the success of the program. Challenge learners to think about the steps they take when putting pen to paper: they need to put the pen to the paper in order for lines to be recorded. If they don't raise the pen when they change positions to a new starting point, lines will be drawn as they move.

- Ask probing questions to help the learners debug on their own and resist stepping in with guidance or possible solutions too quickly. *(See the debugging and reflection prompts below, or the strategies in our debugging resource below.)*
- Does a learner's program involve extra/possibly unnecessary blocks? That is fine! We aren't looking for the most efficient code in this exercise. There is no one "right" solution. The important thing is to have practiced breaking tasks apart, thinking about sequential order, and problem solving.
- Did you have fun? Was this engaging? Excellent! Success looks like having some code that is at least partially working and actively participating in debugging and reflection.

Resources:

- Getting Started Guide (Written Guide) - If you are new to Scratch and just getting started, this resource has helpful information.
- Scratch Ideas Page (Webpage) - This is a great place to short tutorials and Coding Cards.
- Debugging Reflection (Worksheet) - Dive into the practice of debugging with learners and use this reflection sheet to help them explore.
- Debugging Strategies Posters (Printable Posters)

Opportunities for Individual Creativity/Expression

Option 1:

Choose a shape to recreate from a selection of shapes. **Once they have created one shape, challenge them to try another**.

Option 2:

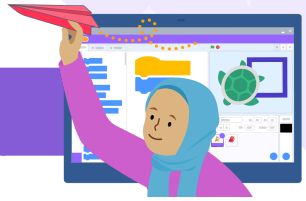
Customize the shape using additional blocks (such as set pen size and color).

Option 3:

Create a paper plane sprite or turtle sprite by drawing one using the Paint Editor tools

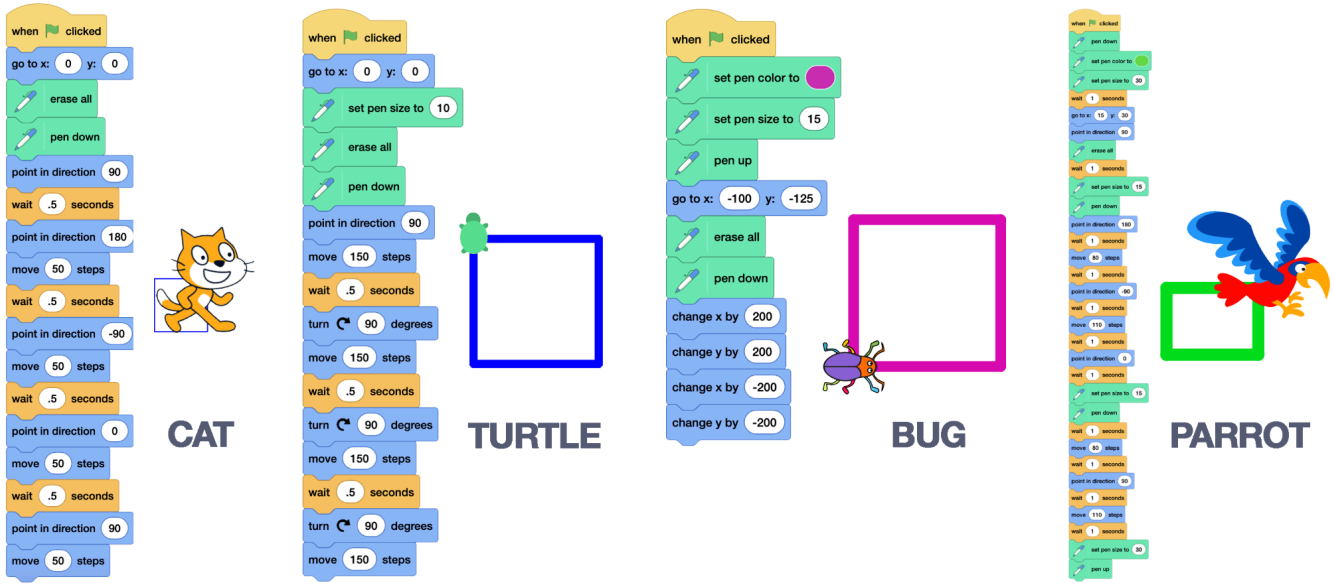
- Create a Sprite with the Scratch Paint Editor (Video Tutorial)
- Create a Sprite with the Paint Editor (Written Guide)

Debugging or Ah-Ha Moment Prompts



These can be answered out loud or written down.

- Did your shape come out as expected? If not, can you spot the problem and debug?
- Do you need to slow down the action? How can you use the wait block to help? Or try this debugging strategy: Put a “play sound until done” block between individual steps or at key points in the program. Choose a different sound each time. Did the problem occur before or after a specific sound? Now you know where to start looking to debug.
- Sequential order matters: Where have you used blocks like erase all, pen up, and pen down in your sequence? What happens if you place these blocks at the beginning or end of a sequence or in a different order?
- There is often more than one solution or path to accomplish a task. Compare your code with other solutions drawing the same shape (in pairs or a small group). Was your solution similar or different? Analyze the other solution(s) and discuss why you chose the blocks you did.



*Examples of Different Script Results: The **Cat’s code** uses specific degrees, compared to the **Turtle’s code** that is general enough to be used in a loop in the next iteration. The **Bug’s code** is a more advanced solution that bypasses angles completely. And the **Parrot’s code** is an example of a learner who may be a beginning coder or new to turtle drawing. They have grasped the main concepts and successfully created a shape, but their program has extraneous code that could be removed in the next iteration.*

Part 4: Reflect and Share

Reflect (10 minutes)

These can be answered out loud or written down.

- How did you break the task down into individual steps? What details were important to include?
- Did you choose to give specific instructions (point in direction _) or more general instructions (turn _ degrees)? Why?
- How does the ability to see the path the sprite took aid in debugging?
- What was fun about this activity?
- What struggles or frustrations did you have during this activity?

Share Option #1: Create a Class Studio to Gather Shared Projects

Studios are a space on Scratch where users can come together to make, share, and collect projects related to a particular theme, idea, or prompt. Set up a class studio* for your learners and add their original asset projects. Learners are encouraged to take time to look at projects and read/listen/interact with them to learn more about their peers.

Resources:

- Teacher Account Guide (Written Guide) - This resource contains information on setting up teacher accounts and student accounts, managing classes, and class studios.
- Scratch Studios Guide (Written Guide) - Information on setting up and managing studios generally.

**Note: Learners will need a Scratch account and access to the online Scratch editor to participate in this option.*

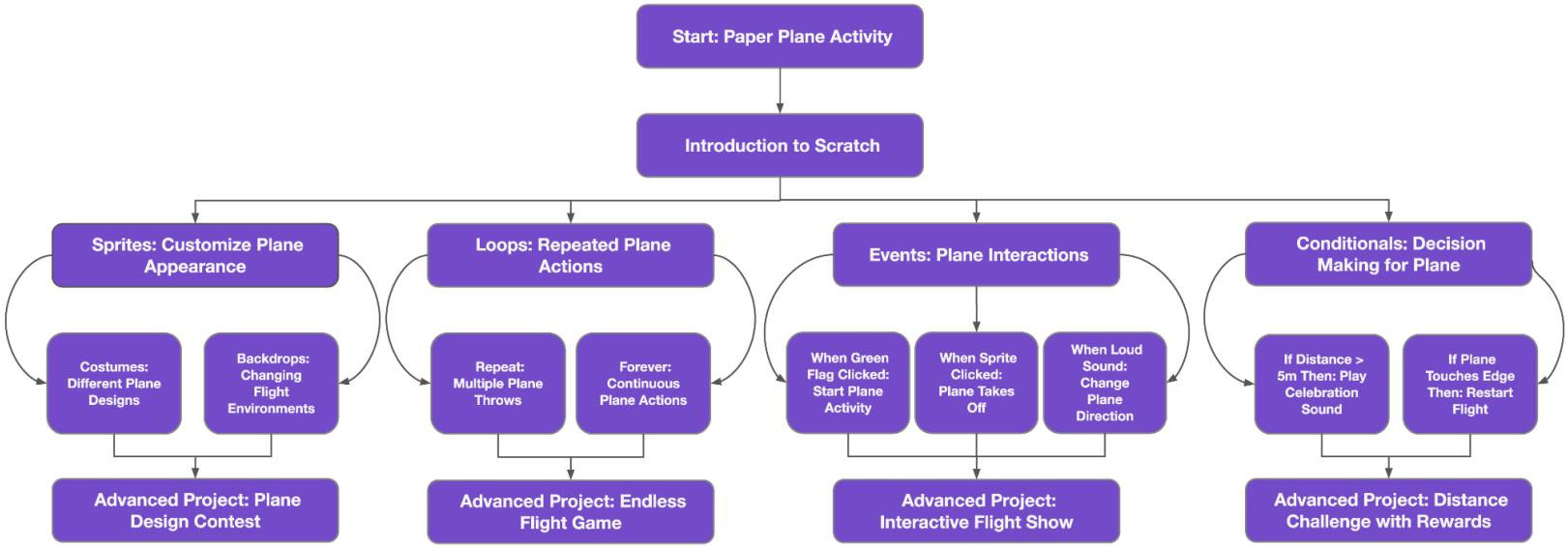
Share Option #2: Gallery Walk

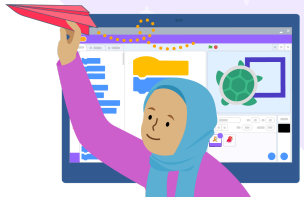
Have each participant's project open on their computer or other device. Participants can walk around a room, or take turns sharing their screen in a virtual space, to experience each other's creations. Another option is to display one project at a time on a large screen. Learners are encouraged to take time to look at projects and read/listen/interact with them to learn more about their peers.

More Things to Try

- How could we use a loop like “repeat” to speed up our coding for repeated steps? Do you need to adjust the blocks you use (from more specific to more general) to transition steps to a loop?
- How can we use loops to make more complex patterns/shapes from our single shape?
- How might you fill in the shape? (Some options could include repeatedly drawing lines within the bounds of the shape, or changing to a larger pen size.)
- Explore different art styles such as abstract art versus more geometric shapes. What might be the advantages of starting with a geometric shape when learning to draw with a turtle?
- Connect each instruction to a Makey Makey input so you can draw the shape in Scratch by touching bananas, foil covered toilet paper tubes, etc.
- For advanced learners: While this activity ensures a low floor to get started, there are endless possibilities/a high ceiling. Learners can explore more advanced features in Scratch and more complex computational concepts while still keeping the paper plane at the center. Objects can be a powerful tool to think with and help learners build strong connections to creative learning, computational thinking, and Scratch.

The chart below is an example of possible directions one could explore:





Additional Resources

- The official Scratch YouTube Channel
- Example Turtle Graphics Projects:
  - "Drawing Practice - Pen Blocks" project
  - "Turtle Drawing" project
  - "Turtle Art - Blue Sky, Green Grass, Sun" project
  - "Starlings trails" project
  - "Paper Crafts - Scratch Week 2023" studio

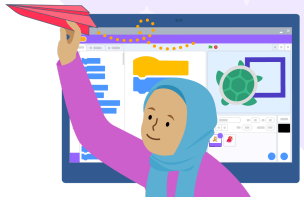
  (Please note projects could be unshared or changed at any time; try searching the platform for "turtle graphics," "turtle drawing," or "pen drawing".)
- Unlock the Block: Stamp Block (blog post and video) - for those interested in this block in the Pen Blocks category
- Possible External Related Resources:
  - Logo and Turtle Graphics History
  - Turtle Graphics examples using a version of Logo focused on Turtle Geometry
  - Wind and Air Explorations: Paper Airplanes from the Exploratorium

  (Please note that this is not our content and it could be unshared or changed at any time.)

Standards Aligned

CSTA Standards	ISTE Standards	CASEL Framework	RITEC Indicators
Link to full standards	Link to full standards	Link to full standards	Link to full standards
• 1B-AP-08 Compare & refine algorithms • 1B-AP-10 Create programs • 1B-AP-11 Decompose problems • 1B-AP-15 Test and debug	• 1.5.c Decompose Problems • 1.5.d Algorithmic Thinking • 1.6.b Creative Communicator	• Self-awareness • Relationship Skills	• Autonomy • Competence • Emotions • Creativity • Diversity, equity and inclusion

This lesson also fulfills all three of the ISB Indicators of Playful Learning (Choice, Delight, Wonder), developed by the Pedagogy of Play (PoP) research project at Harvard University.



Tip: If you'd like to translate this guide, **click here to make a copy** of this Google doc.