

CoolThink@JC - Computational Thinking Education Programme

Course Code & Title

**CF1E Computational Thinking
Development - Level 1**

Unit Code & Topic

L1U8.7-8.8

Computational Arts with Scratch



香港教育大學
The Education University
of Hong Kong



Massachusetts Institute of Technology



Funded by

香港賽馬會慈善信託基金
The Hong Kong Jockey Club Charities Trust
同心 同步 同進 RIDING HIGH TOGETHER

This material is for educational purposes of the
Computational Thinking Education Programme only.
Please don't disclose to any third party.

These teaching and learning materials were co-created by The Education University of Hong Kong, and Massachusetts Institute of Technology, and funded by The Hong Kong Jockey Club Charities Trust.

Table of Contents

1. Synopsis	1
2. Unit Intended Learning Outcomes (UILOs)	1
3. Mapping with the Computational Thinking Framework	2
4. Learning Prerequisites	4
5. Pedagogical Approach	4
6. Lesson Plan	4
7. Instructional Details	8
8. Assessment	8
9. Screen Design and Code	12
Appendix 1 L1U8.7-8.8 Teacher's Guide: Lesson 1	14
Appendix 2 L1U8.7-8.8 Teacher's Guide: Lesson 2	17
Appendix 3 L1U8.7-8.8 Teacher's Guide: Lesson 3	19
Appendix 4 L1U8.7-8.8 Teacher's Guide: Lesson 4	21

1. Synopsis

Students will create computational art using Scratch, while learning to see patterns and to practice abstraction and modularization. Students will learn to make a complex shape, a snowflake, by drawing multiple squares and rotating. They will learn to create custom blocks to first draw the square, and then to draw the snowflake. Finally students will practice further abstraction by adding an input parameter to both custom blocks to specify the size of the shape. The focus for Lesson 1 is repetition, using a loop to draw the square. The focus for Lesson 2 and 3 is abstraction and modularization, as students create their own blocks, or procedures. The focus for Lesson 4 is data manipulation and elementary data structures, as students add an input parameter to their custom blocks.

2. Unit Intended Learning Outcomes (UILOs)

After completing this unit, students will be able to:

- UILO₁: create a Scratch project that uses the Pen feature to draw computational art;
- UILO₂: apply the computational thinking concept of repetition to draw multiple shapes that in turn create more complex and interesting shapes;
- UILO₃: use the computational thinking practice of abstraction and modularization by creating a custom block, or procedure, to draw a shape;
- UILO₄: use the computational thinking practice of abstraction and modularization by adding an input parameter to a custom block, to allow for shapes of different sizes; and
- UILO₅: demonstrate understanding that coding can be a fun and creative experience by using Scratch to create visual art.

3. Mapping with the Computational Thinking Framework

The following tables show the alignment of this unit with the intended learning outcomes of the computational thinking framework. The entries indicate the expected relevance of this unit to each outcome:

- ✓✓✓ : High relevance
 ✓✓ : Some relevance
 ✓ : Low relevance

Computational Thinking Concepts

L1U8.7-8.8: Computational Arts with Scratch		
1. Sequences	✓✓✓	Sequences are important to make the drawing look the way you want it to.
2. Events		
3. Repetition	Lesson 1 focus	Students use loop to draw a square and a snowflake shape.
4. Conditionals		
5. Parallelism		
6. Naming	✓✓✓	Students name their blocks and the input parameters for those blocks descriptively.
7. Operators		
8. Manipulation of data and elementary data structures	Lesson 4 focus	Students add an input parameter to their “DrawASquare” and “DrawASnowflake” blocks to specify size.

Computational Thinking Practices

L1U8.7-8.8: Computational Arts with Scratch		
1. Reusing and remixing	✓✓	Students reuse and remix code blocks from the subsequent lesson.
2. Being incremental and iterative	✓✓	Students start with a square, and iterate on that to draw a snowflake, eventually specifying different sizes and locations.
3. Abstracting and modularizing	Lesson 2 & 3 focus	Students create their own custom blocks to draw squares and snowflakes.
4. Testing and debugging	✓✓	Students need to test their projects to make sure it draws a shape as expected.
5. Algorithmic thinking		

Computational Thinking Perspectives

L1U8.7-8.8: Computational Arts with Scratch		
1. Expressing	✓✓	Students may experiment with their snowflake shapes.
2. Connecting	✓✓	Students learn about angles when drawing complex shapes.
3. Questioning	✓✓	Students see that they can create interesting visual art with Scratch.
4. Computational identity	✓✓✓	Students see that they can make projects that involve art and computation.
5. Digital empowerment	✓✓✓	Students are empowered by making fun art projects they can share with others.

4. Learning Prerequisites

Students should have experience with Scratch, from the previous units of the Coolthink@JC curriculum.

5. Pedagogical Approach

The suggested pedagogical approaches of this unit are as follows:

- Class discussion; and
- Guided individual work.

6. Lesson Plan

This unit consists of 4 lessons of 35 minutes.

Lesson 1

Time	Activity
5 mins	Introduction to Unit <ol style="list-style-type: none"> 1. Explain that in this unit, students will learn some tools to make art with Scratch. 2. Demonstrate examples of computational art: https://scratch.mit.edu/projects/282230147/ https://scratch.mit.edu/projects/307233164/
10 mins	Introduction of Drawing Shapes <ol style="list-style-type: none"> 1. Demonstrate the Drawing blocks in Scratch. <ol style="list-style-type: none"> (1) Explain how to add the Pen extension in Scratch and the usage of the blocks. (2) Show students how to draw a line, change color, pen up, pen down, etc. 2. Ask students how they might draw a square with Scratch. <ol style="list-style-type: none"> (1) Ask for volunteers for different approaches. (2) Demonstrate making a square.
20 mins	Coding Activity <p>Students complete the <i>Student Guide: Lesson 1</i> to make a square, first without, and then with a loop.</p>

Lesson 2

Time	Activity
10 mins	Introduction to More Complex Shapes <ol style="list-style-type: none"> 1. Demonstrate adding a “turn 60 degrees” block after drawing a square. Copy/paste the “repeat” block to draw a square and show the resulting shape. 2. Ask students what to do to add another turn and another square. 3. Ask students how they might continue this to make a circular shape. <ol style="list-style-type: none"> (1) Students might suggest copy/paste the turn and repeat. (2) Complete the shape. (3) Guide students to add a “repeat” block to simplify the code. 4. Introduce how to make a custom block, also known as a procedure. Make the “DrawASquare” block.
20 mins	Coding Activity Students complete the <i>Student Guide: Lesson 2</i> to explore making a snowflake shape using a set of rotated squares. The guide will instruct them in learning how to create their own custom block to draw a square.
5 mins	Wrap-up Review custom blocks (procedures) and using them break complex tasks (making shapes) into smaller blocks of code that can be reused.

Lesson 3

Time	Activity
10 mins	Review of Custom Blocks <ol style="list-style-type: none">1. Review the “DrawASquare” block and the concept of simplifying code and calling the blocks to perform a task.2. Ask students to explain how they made a snowflake shape using a set of rotated squares.3. Ask students how they might make several snowflakes in different locations on the stage in a project.<ol style="list-style-type: none">(1) Students might suggest copying/pasting code, or using a loop.(2) Ask students if it makes sense to make a custom block to make a snowflake when they want to make several snowflakes, just like they made a block to draw a square.(3) Ask students how they would make a custom block to make a snowflake.
20 mins	Coding Activity <p>Students complete the <i>Student Guide: Lesson 3</i>, creating the “DrawASnowflake” block.</p>
5 mins	Wrap-up <p>Explain to students that they will use these tools to make snowflakes of different sizes in Lesson 4. Ask them to think about how they might do that.</p>

Lesson 4

Time	Activity
10 mins	Introduction to Input Parameters <ol style="list-style-type: none"> 1. Ask students how they might make bigger snowflakes using bigger squares. Ask students what they would need to change in their code blocks. 2. Demonstrate changing the “DrawASquare” block so the sprite moves 150 steps, and make a larger square and subsequent snowflake. 3. Ask students how they could make a project with many different sized snowflakes. 4. Show students how to add an input parameter for size to the “DrawASquare” and “DrawASnowflake” blocks.
20 mins	Coding Activity Students complete the <i>Student Guide: Lesson 4</i> , to make their computational art with snowflakes of varying sizes.
5 mins	Wrap-up <ol style="list-style-type: none"> 1. Ask students to add their projects to the teacher’s studio. 2. Review custom blocks and input parameters, focusing on how it enables abstraction of a particular task.
Post-Lesson Activities	(Optional) Create your own computational art with what you have learnt in this unit.

7. Instructional Details

The guidelines of the four lessons are presented in the following appendices.

Lesson 1:

1. Teacher's Guide: Lesson 1
2. Student Guide: Lesson 1

Lesson 2:

1. Teacher's Guide: Lesson 2
2. Student Guide: Lesson 2

Lesson 3:

1. Teacher's Guide: Lesson 3
2. Student Guide: Lesson 3

Lesson 4:

1. Teacher's Guide: Lesson 4
2. Student Guide: Lesson 4

8. Assessment

The following table maps the assessment modes and corresponding Unit Intended Learning Outcomes (UILOs) in Section 2.

Assessment Modes	UILO
1. Self -assessment: (1) Online multiple-choice questions (2) Survey of learning attitudes	UILO _{1,2,3,4,5}

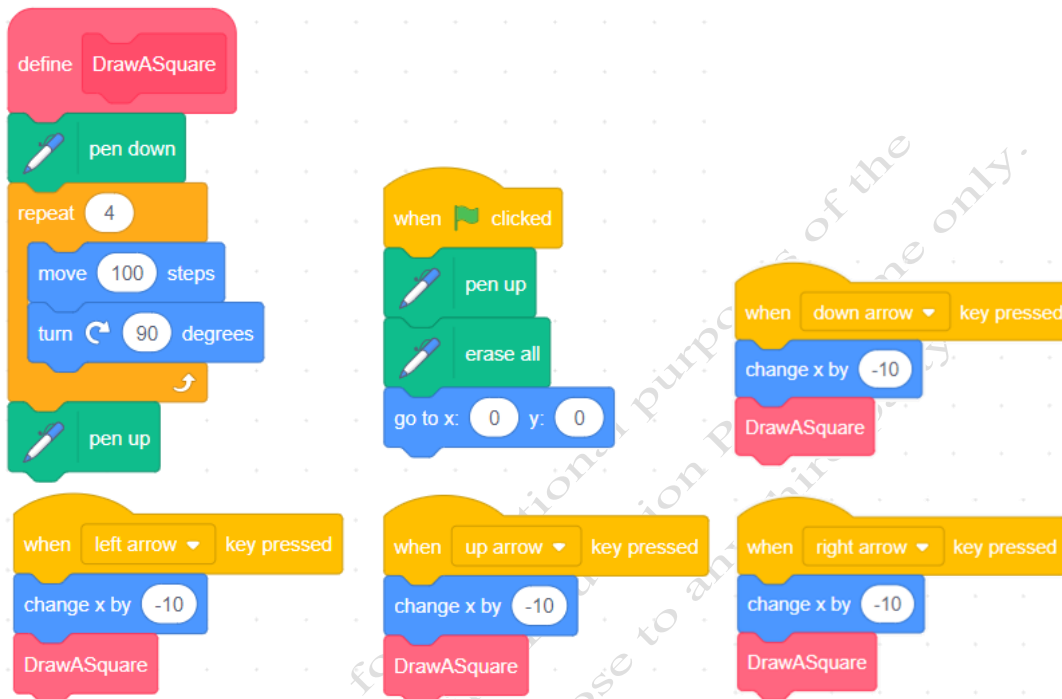
Self-assessment

Self-assessment allows students to reflect on their learning performance. There are two sorts of self-assessment questions.

Online Multiple-Choice Questions

Online multiple-choice questions are adopted to assess students' understanding of the key concepts of the activity, as shown below.

The following blocks refer to Questions 1-2:



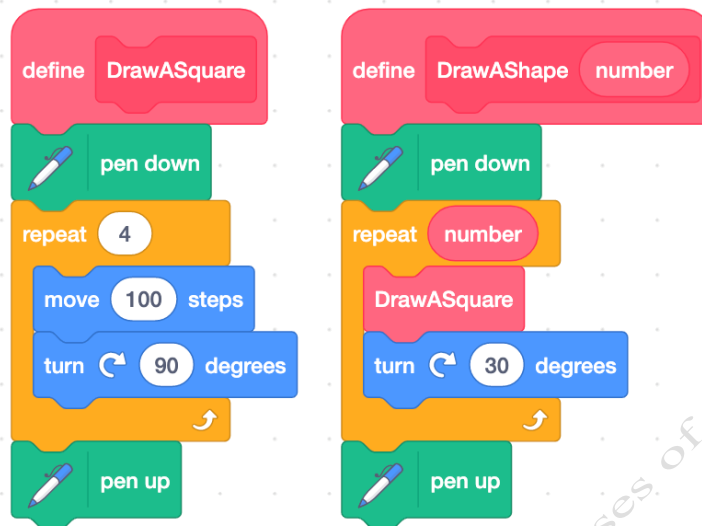
1. A student makes a project with the blocks above. What happens when the user click the green flag?
 - A. The sprite draws a snowflake.
 - B. The sprite draws four squares.
 - C. The sprite draws one square.
 - D. The stage is cleared and the sprite moves to the center of the stage.

(Answer: D)

2. What happens if the user presses the right arrow 3 times?
 - A. The sprite moves to the right 30 steps.
 - B. The sprite moves to the centre of the stage.
 - C. The sprite draws 3 squares that are beside each other on the stage.
 - D. The sprite draws 3 squares that are above each other on the stage.

(Answer: C)

3. A student makes a project with the following custom blocks.



If the student calls **drawAShape 5**, what happens?

- A. A snowflake with a size of 5 is drawn.
- B. 5 snowflakes are drawn.
- C. A shape with 5 rotated squares is drawn.
- D. Nothing is drawn because the pen is up.

(Answer: C)

Survey of Learning Attitudes

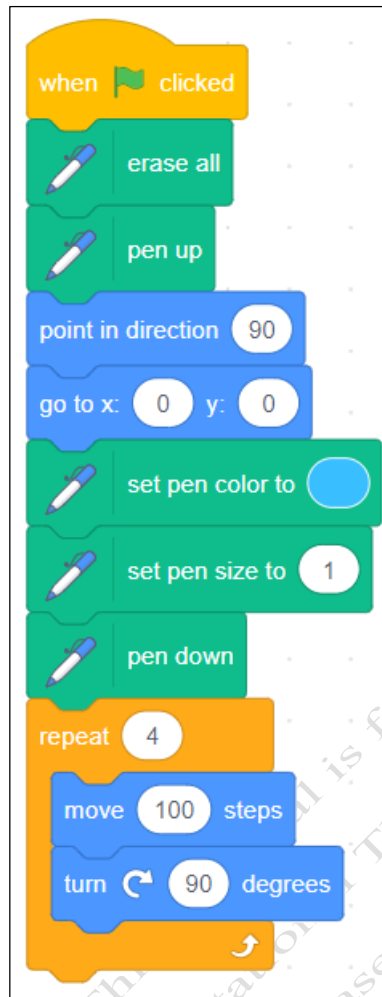
To evaluate students' attitudes, perceptions, and understandings of coding, they are required to complete the following questionnaire by putting a "✓" in the appropriate boxes.

After completion of this unit, I think...	Disagree	Somewhat disagree	Neutral	Somewhat agree	Agree
I feel happy that I can express myself through programming.					
The learning activities of this unit are interesting and I like them.					
I am excited to show this app with friends and family.					

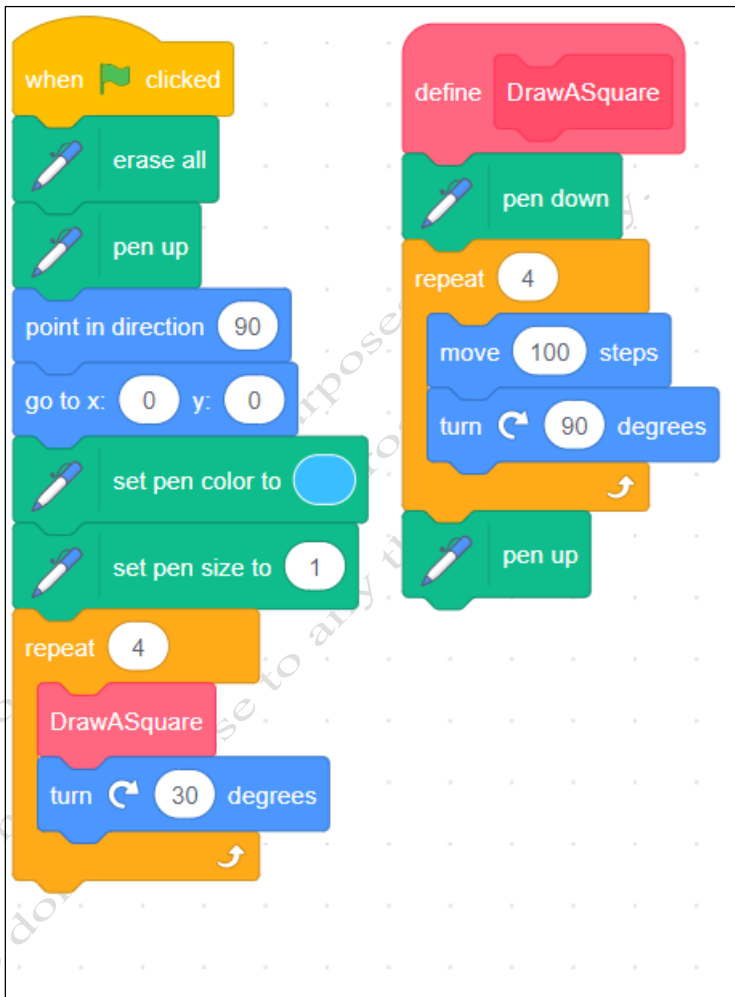
9. Screen Design and Code

Blocks

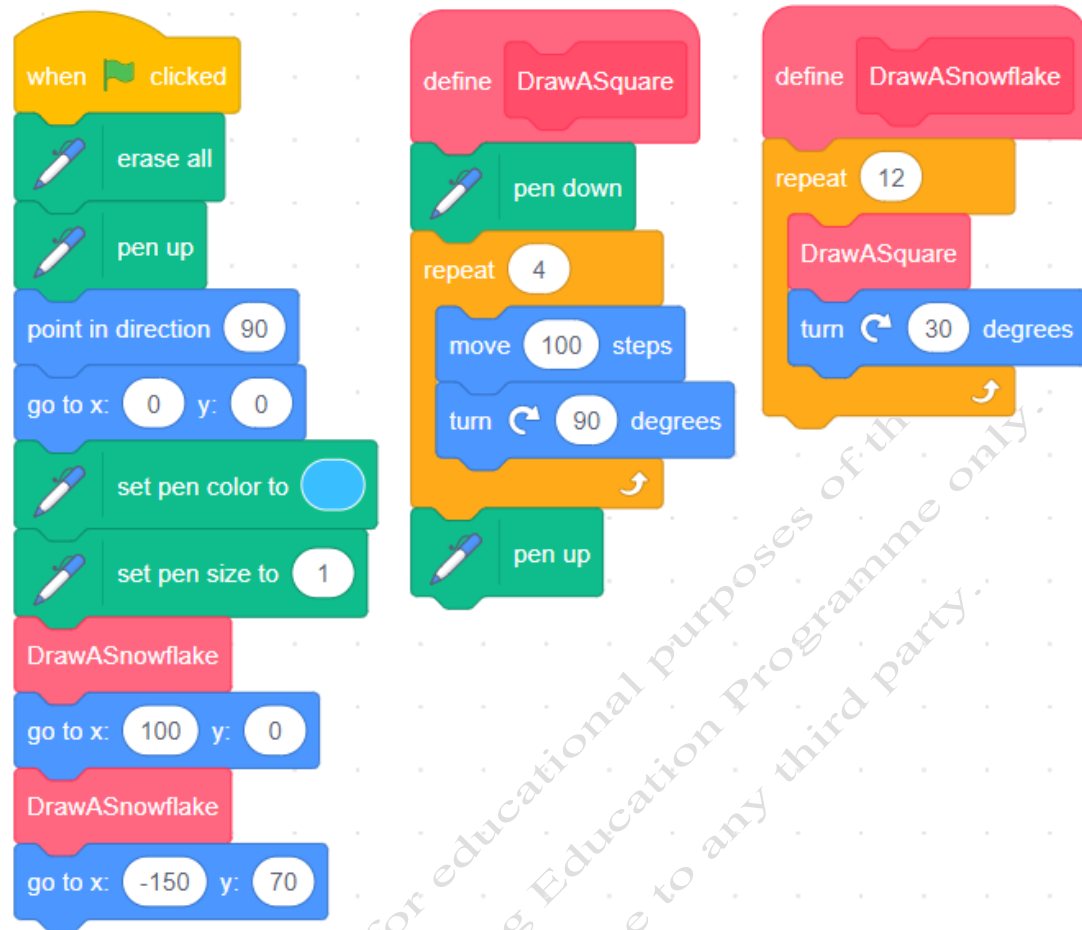
Lesson 1:



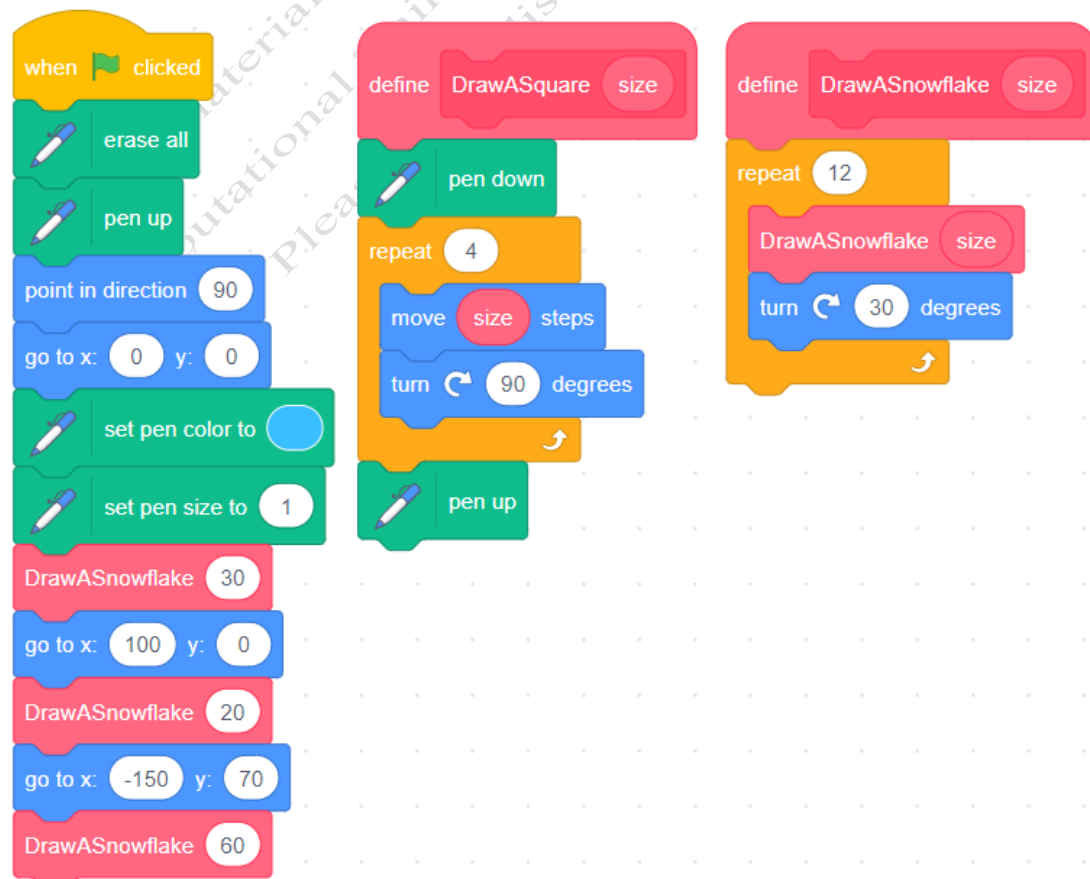
Lesson 2:



Lesson 3:



Lesson 4:



Appendix 1

L1U8.7-8.8 Teacher's Guide: Lesson 1

Teaching Objectives

1. Teach students how to use the Pen blocks in Scratch.
2. Focus: Show students how to use a loop to draw a square in Scratch.

Lesson Outline

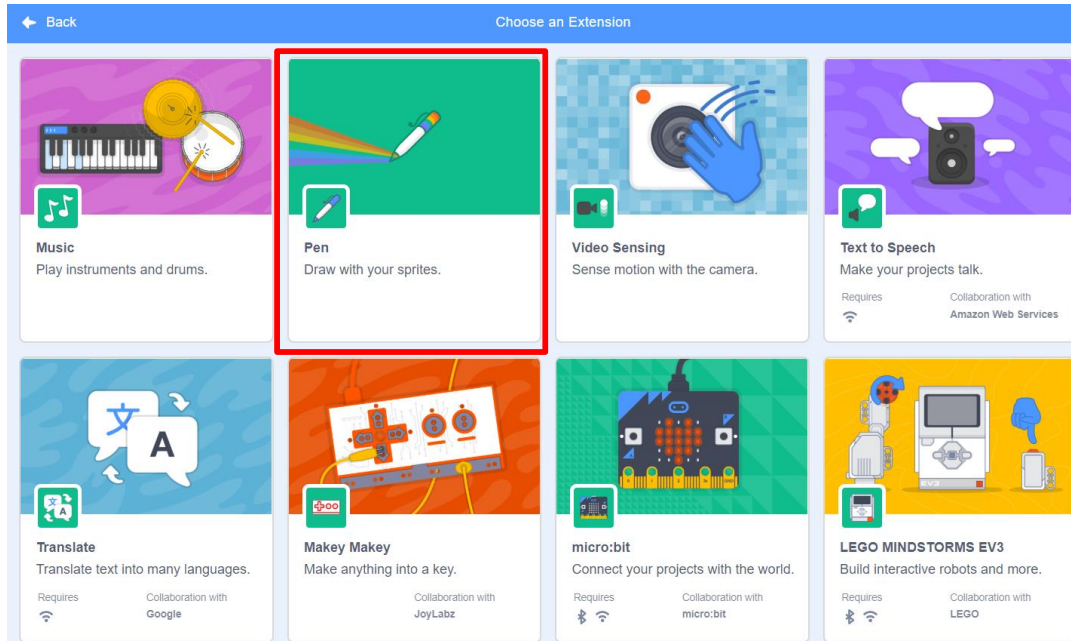
Introduction to Unit (5 minutes)

1. Explain that in this unit, students will learn some tools to make art with Scratch.
2. Demonstrate examples of computational art:
<https://scratch.mit.edu/projects/282230147/>
<https://scratch.mit.edu/projects/307233164/>

Introduction of Drawing Shapes (10 minutes)

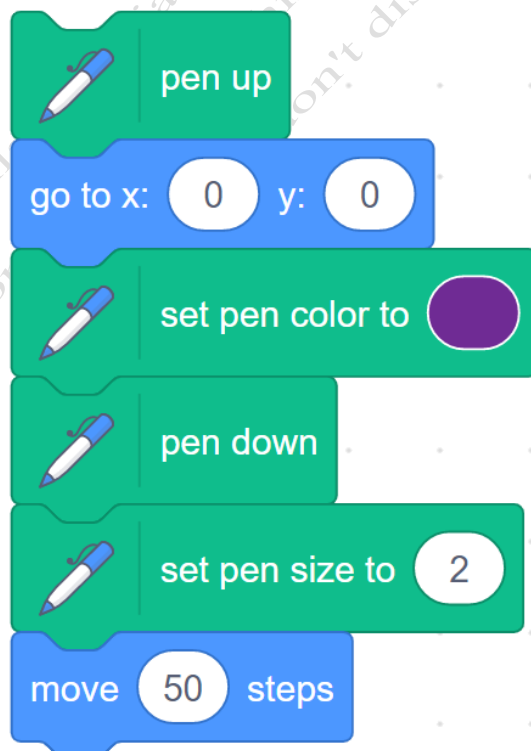
1. Show students the Pen tools in Scratch.
 - (1) Explain how to add the Pen extension in Scratch.
 - i. Click on the “Add Extension” icon.

ii. Choose Pen.

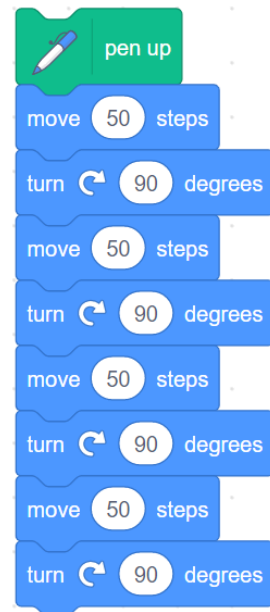


- (2) Demonstrate the Pen blocks below in Scratch. Show students how to draw a line, change color, pen up, pen down, etc.

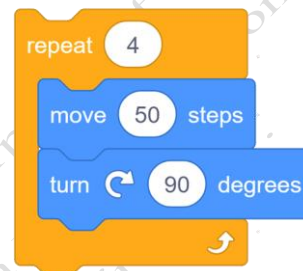
If pen is down and the sprite moves, it will draw a line as it moves, in the pen colour and size that has been set. “go to x:0 y:0” block moves the sprite to the centre of the stage. Because the pen is up, nothing is drawn. Then the pen is set down, and the colour and size is set. The sprite moves to the right, so a line is drawn, 50 steps in length.



2. Ask students how they might draw a square with Scratch.
 - (1) Ask for volunteers for different approaches.
 - (2) Demonstrate making a square. Draw a line, turn 90 degrees. Repeat 4 times.



sequential steps



using loop

Coding Activity (20 minutes)

Students complete the *Student Guide: Lesson 1* to make a square, first without, and then with a loop. Students may experiment with different size squares, changing the color, etc.

Appendix 2

L1U8.7-8.8 Teacher's Guide: Lesson 2

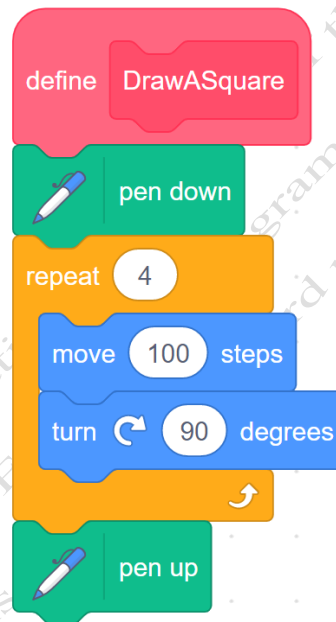
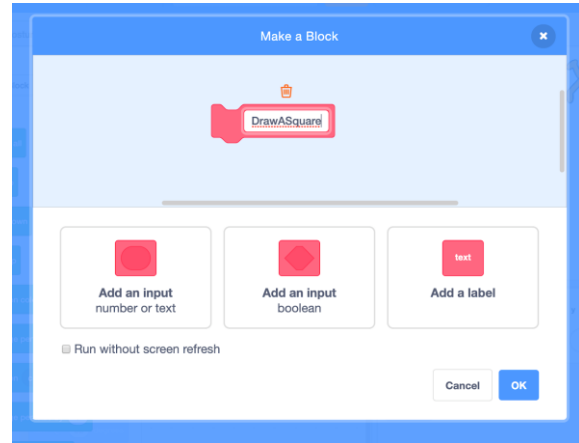
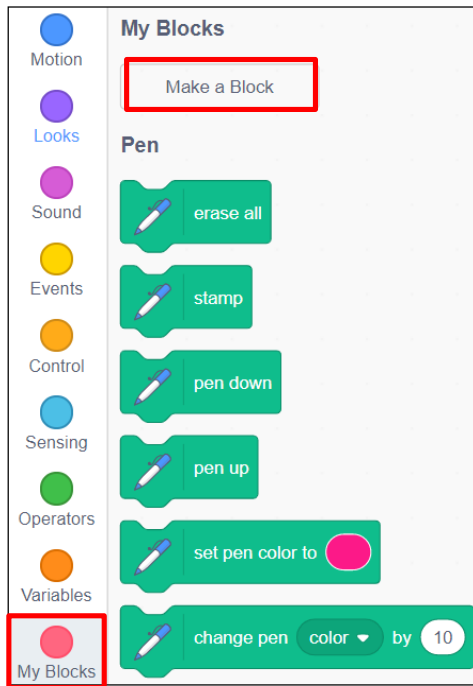
Teaching Objectives

1. Teach students how to make their own custom block to perform a task.
2. Focus: Help students to understand that making custom blocks is a way of abstracting and modularizing code.
3. Show students how to create more complex shapes by repeating a simple shape over and over.

Lesson Outline

Introduction to More Complex Shapes (10 minutes)

1. Demonstrate adding a “turn 60 degrees” block after drawing a square. Copy and paste the “repeat” block to draw a square and show the resulting shape.
2. Ask students what to do to add another turn and another square.
3. Ask students how they might continue this to make a circular shape.
 - (1) Students might suggest copy/paste the turn and repeat.
 - (2) Complete the shape.
 - (3) Guide students to add a “repeat” block to simplify the code.
 - i. Ask students “Can you see a pattern in the code?”
 - ii. What can we do when we see blocks repeated?
 - iii. Add a loop, and ask students “How many times do we repeat?” (6)
4. Introduce how to make a custom block, also known as a procedure. Make the “DrawASquare” block with a set side size of 100.



Coding Activity (20 minutes)

Students complete the *Student Guide: Lesson 2* to explore making a snowflake shape using a set of rotated squares. The guide will instruct them in learning how to create their own custom block to draw a square.

Wrap-up (5 minutes)

Review custom blocks (procedures) and using them break complex tasks (making shapes) into smaller blocks of code that can be reused. Explain that creating your own blocks is an important computational thinking practice, called abstraction and modularization. You are abstracting the process of drawing a square, so you can call the block whenever you want a square. It modularizes the process by hiding it in its own block.

Appendix 3

L1U8.7-8.8 Teacher's Guide: Lesson 3

Teaching Objectives

1. Guide students in finding patterns in code.
2. Focus: Expand students' understanding of abstraction and modularization by helping them create a "DrawASnowflake" block.

Lesson Outline

Review of Custom Blocks (10 minutes)

1. Review the "DrawASquare" block and the concept of simplifying code and calling the blocks to perform a task. Remind students that they are performing an important computational thinking practice, abstraction and modularization, by creating a custom block.
2. Ask students to explain how they made a snowflake shape using a set of rotated squares. Let students look over their code and verbalize the process of drawing a snowflake.
3. Ask students how they might make several snowflakes in different locations on the stage in a project.
 - (1) Students might suggest copying/pasting code, or using a loop.
 - (2) Ask students if it makes sense to make a custom block to make a snowflake when they want to make several snowflakes, just like they made a block to draw a square.
 - (3) Ask students how they would make a custom block to make a snowflake.

Coding Activity (20 minutes)

Students complete the *Student Guide: Lesson 3*, creating the “DrawASnowflake” block. This gives them more practice and deepens their understanding of custom blocks. They will also see the one custom block can be used within another custom block, as “DrawASnowflake” calls the “DrawASquare” block.

Wrap-up (5 minutes)

Explain to students that they will use these tools to make snowflakes of different sizes in Lesson 4. Ask them to think about how they might do that.

This material is for educational purposes only.
Computational Thinking Education Programme only.
Please don't disclose to any third party.

Appendix 4

L1U8.7-8.8 Teacher's Guide: Lesson 4

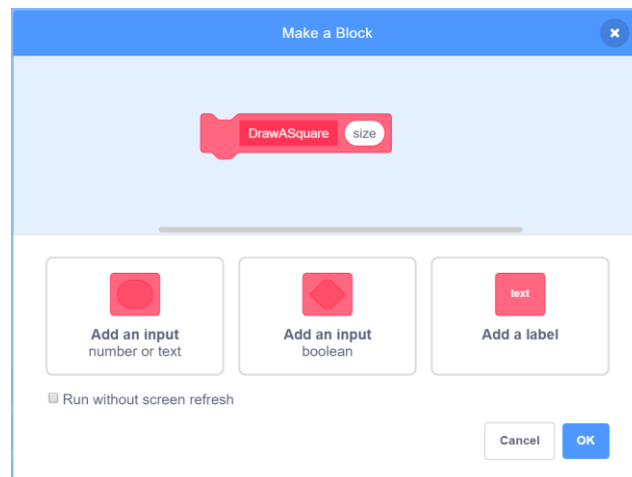
Teaching Objectives

1. Focus: Teach students how to add an input parameter to a custom block, so that variable information can be passed to the block.
2. Continue to develop students' understanding of abstraction and modularization with use of an input parameter in a custom block.
3. Motivate students to explore and create interesting computational art by learning drawing features in Scratch.

Lesson Outline

Introduction to Input Parameters (10 minutes)

1. Ask students how they might create bigger snowflakes using bigger squares. Ask students what they would need to change in their code blocks. Students should be able to see that the “move x steps” block can be changed to a different value, and result in a square/snowflake of a different size.
2. Demonstrate changing the “DrawASquare” block so the sprite moves 150 steps, and make a larger square and subsequent snowflake. Change it to a different number, run the program, and see that the snowflake size changes.
3. Ask students how they could make a project with many different sized snowflakes. Students might suggest making different custom blocks for different sized snowflakes. Ask students how difficult it would be to make 100 different sized snowflakes. Would it be hard to make 100 different custom blocks?
4. Show students how to add an input parameter for size to the “DrawASquare” and “DrawASnowflake” blocks.



Using an input parameter allows you to abstract a custom block even further, making it more versatile. Now the block can generate a square of any size.

Coding Activity (20 minutes)

Students complete the *Student Guide: Lesson 4*, to make their computational art with snowflakes of varying sizes. Students will add an input parameter to both the “DrawASquare” block and the “DrawASnowflake” block. Encourage students to be creative and update their projects to make snowflakes of varying sizes at different locations on the stage.

Wrap-up (5 minutes)

1. Ask students to add their projects to the teacher’s studio.
2. Review the use of custom blocks and input parameters, focusing on how it enables abstraction of a particular task.

Post-lesson Activities

(Optional) Create your own computational art with what you have learnt in this unit.