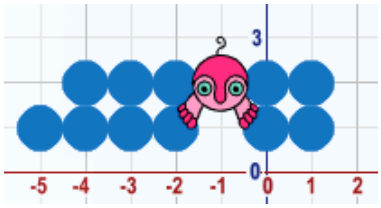
Module 6:
Coordinates And Geometry

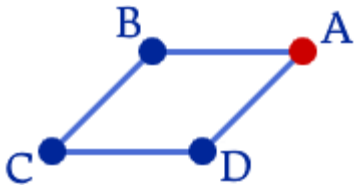
Investigation 1

Emerging Shapes



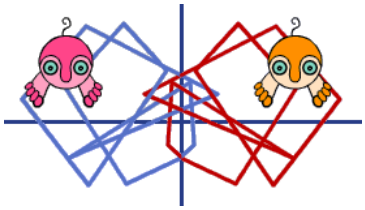
Investigation 2

Coordinate Shapes



Investigation 3

Transformations

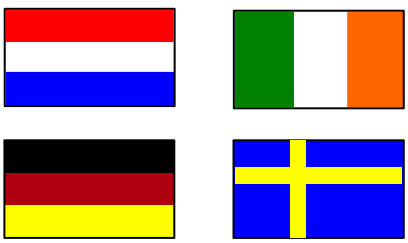


Introduction to Module 6

Module 6 explores coordinates across all four quadrants and requires pupils to use their knowledge of variable to explore scale. Pupils develop their knowledge of coordinates through different contexts including transformations (reflection and translation) and mathematical shapes which emerge by exploiting the random block. This module could potentially be linked with several different areas of the Key Stage 2 curriculum beyond mathematics and computing such as geography and English.

Geography: Locational Knowledge

Within the first investigation pupils have the opportunity to experiment with coordinates in order to create flag patterns which could be linked with specific countries that pupils are learning about within the geography curriculum.



English: Spelling

Within the second investigation pupils code and decode words using letters positioned at different positions on the coordinates grid. This could be linked to spelling particular words specified within the English curriculum such as those with silent letters or tricky homophones.

Aisle/Isle
Cereal/Serial
Draft/Draught

Key Vocabulary and Concepts Covered by Module 6

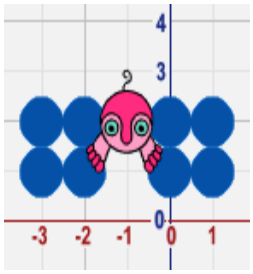
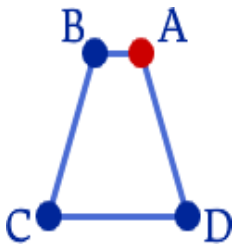
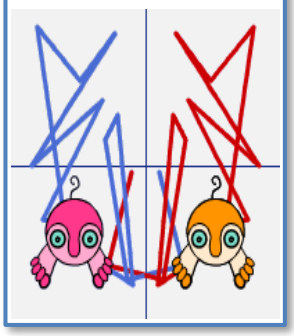
Computing

- Repetition
- Variable
- Definitions
- Debugging
- Decomposition
- Selection
- Logical Reasoning

Mathematics

- Coordinates
- Randomness
- Reflection
- Translation
- Scale Factor
- Division & rounding
- Regular and irregular polygons

Map Of Module 6

	Activity 1	Activity 2	Activity 3	Activity 4
Investigation 1 **Emerging Shapes** 	**Restless Fleeeee** Starter project: **61-Fleeeee Dots**	**Unplugged and Hands-on: Envisage and Explore**	**Introducing Scale** Continue with: **61-Fleeeee Dots**	**Dotty Patterns** Continue with: **61-Fleeeee Dots**
Investigation 2 **Coordinate Shapes** 	**Letters and Coordinates** Starter project: **62-Grid Letters**	**Busy Fleeeee and Clever Points** Starter project: **62-Coordinates**	**Tricky Triangles** Continue with: **62-Coordinates** or start with: **62-Coordinates INT1**	**Quirky Quadrilaterals** Continue with: **62-Coordinates** or start with: **62-Coordinates INT2**
Investigation 3 **Transformations** 	**Mimic Meeeee** Starter project: **63-Mimic Meeeee**	**Shadows, Translations & Reflections** Continue with: **63-Minic Meeeee** or start with: **63-Mimic Meeeee INT**	**Through the Looking Glass** Starter project: **63-Looking Glass**	

The **red** dashed line indicates the core activities which are important to complete before moving on to the next activities.

For activities which require pupils to continue with a project from a previous lesson you can alternatively use the suggested INT project for those pupils who do not have a project to continue with or if you wish all pupils to begin from the same point.

Module 6 Connections to KS2 Computing Curriculum

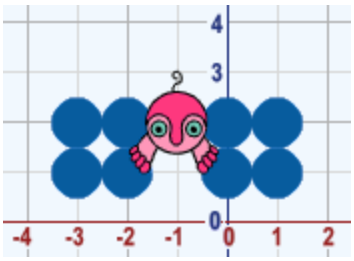
Curriculum Objectives	Link with ScratchMaths
Design, write and debug programs that accomplish specific goals	▶ During this module pupils are required to build programs that create emergent shapes from random dots, place and connect points on a coordinate grid and also that mimic the behaviour of another sprite through translation and reflection.
Solve problems by decomposing them into smaller parts	▶ Pupils are required to use decomposition in order to understand the mathematics within different operations and expressions they are using in their scripts, as well as when building the mimicking behaviour, to help develop their understanding of how the two sprites are linked.
Use selection and repetition in programs	▶ Pupils are required to use selection in their scripts through using different keyboard commands to move their sprite and stamp different letters. ▶ Pupils use two forms of repetition in drawing their emergent shapes made up of a repeated number of random dots, to be able to continuously mimic the behaviour of another sprite and also in an extension to animate their sprite in different ways.
Work with variables	▶ Pupils use variables when working with different coordinate grid intervals and also to enable sprites to snap to specific grid points.
Using logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs	▶ Pupils are required to use logical reasoning when envisaging the missing values in scripts to create different emerging shapes. ▶ Within an extension activity pupils are also asked to examine and explain the differences between several different algorithms for drawing emerging flag patterns.

Module 6: Investigation 1
Emerging Shapes

This investigation develops reading and setting coordinates by exploring and building scripts which produce emergent shapes. The start of the investigation uses a backdrop (with axis intervals marked) and a script which utilises a 1-1 mapping between coordinates and pixels. As the investigation develops pupils engage with scale, converting standard *pixel coordinates* to *grid coordinates* (different intervals on the x and y axis).

We have deliberately selected a new sprite – the Fleeeee (with 5Es in honour of our pedagogical strategy!) – instead of reusing the Beetle to make it clear that we are focusing on coordinates and moving *differently* by using the blocks **go to x: … y: …** and reporter blocks **x position** and **y position**. Pupils engage with patterns in coordinates, for example multiples of 20 when creating a dot pattern.

- **Activity 6.1.1** – Restless Fleeeee
- **Activity 6.1.2** – Unplugged and Hands-on: Envisage and Explain
- **Activity 6.1.3** – Introducing Scale
- **Activity 6.1.4** – [Extension] Dotty Patterns



Scratch projects **61-Fleeeee Dots** **61-Fleeeee Dots FINAL** **61-Fleeeee Dots Extension FINAL**

Links to Primary National Curriculum

Curriculum Objectives	Link with ScratchMaths
Mathematics Identify, describe and represent the position of a shape following a reflection or translation. Describe movements between positions as translations of a given unit to the left/right and up/down. Solve problems involving similar shapes where the scale factor is known or can be found. Describe positions on the full coordinates grid. (KS3) Work with experiments that involve randomness.	▶ Pupils are required to envisage and build scripts which create a reflected emergent shape in the y axis and then in the x axis. ▶ Pupils are required to envisage and build scripts which create an emergent shape that has been translated left, right, up and/or down by a specified amount. ▶ Pupils are required to use different grid intervals and adjust the scale of their emergent shape scripts appropriately (using variables) to match the different grids. ▶ Pupils are required to engage with the full coordinate grid to specify the position and size of their emergent shapes. ▶ Pupils employ randomness to create their emergent shapes using randomly positioned dots within a set area.

Module 6 • Investigation 1 • Activity 6.1.1
Restless Fleeeee

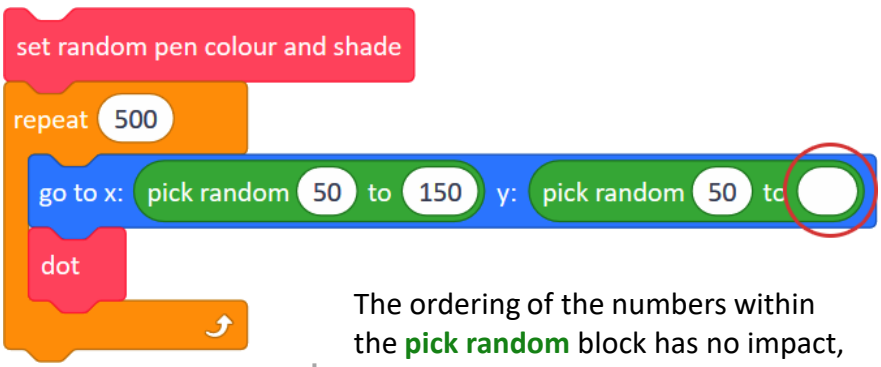
Learning Objectives

Explore how to create emergent shapes within a defined area using coordinates.
Envisage how to recreate emergent shapes by identifying maximum and minimum coordinates

Activity Instructions

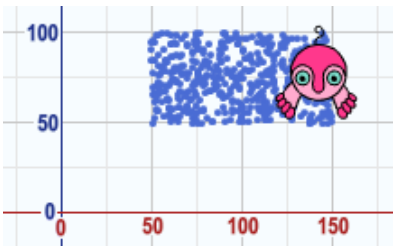
Pupils open project **61-Fleeeee Dots**, **Save as a copy** (online) or **Save as** (offline) and rename.

1. Pupils explore the project, its backdrop, the **Fleeeee** sprite, its *setup script* and the definitions of the **dot** and **pick random pen colour and shade** blocks.
2. **[Extension]** Pupils explore the costumes of **Fleeeee** and the blocks **blink**, **nod**, **walk** and **run** and the way how they are defined. [Do not run **walk** and **run** in parallel as the animations would interfere with each other.]
3. Pupils envisage, discuss and run the **go to x: … y: …** script in the scripts area. What happens if they run this 'jumping script' again and again? They add the **repeat** block around it and the **when** *space* **key pressed** hat block and experiment with different numbers of repetitions, e.g. 10 or 100 or 500…
4. In the classroom presentation there are several pictures of emerging rectangles and corresponding scripts with one or several missing coordinates. Pupils fill in the missing inputs of the **pick random … to …** blocks and run the script.



5. **[Extension]** In the classroom presentation there are more complex and interesting pictures of emerging shapes. Pupils build scripts to create them. This is an example, similar to the flag of Switzerland.

Mathematics Connections



Ask questions to connect the emergent shape with the **go to x: y:** block. *What is the length/width of the rectangle? How can this be calculated from the* ***go to x: y:*** *block?*

The ordering of the numbers within the **pick random** block has no impact, however it might be helpful to use the conventions as we would see on a coordinate axis.

Envisage the length and width of the emergent shape before running the script. *What are the coordinates of the top right corner or the bottom left corner? How do you know?*

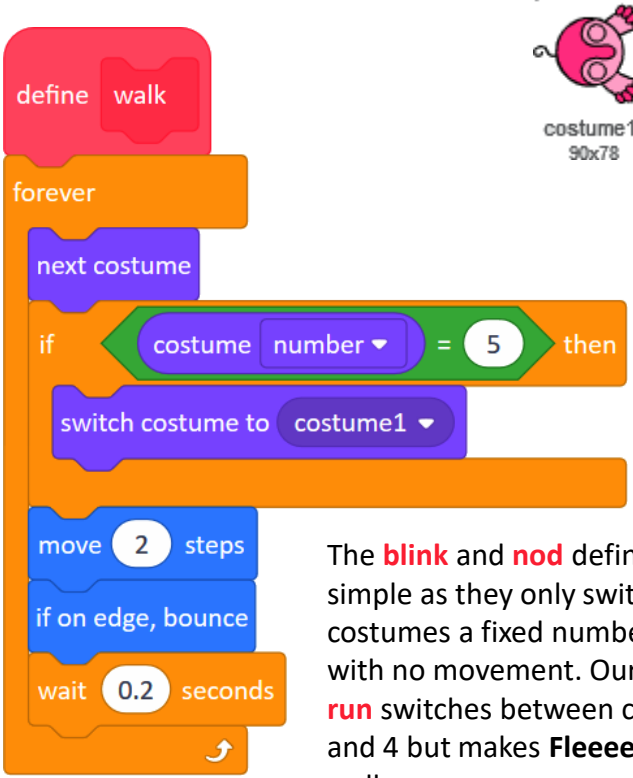
2 We started using sprites with multiple costumes in Module 1 (Activity 1.2.3) for stamping *alternating flower patterns*. In Activity 1.3.3 we used a sprite with six different costumes and started using the **switch costume to** *name* using the specific *names* of the costumes.

In Module 3 we exploited multiple costumes for different purposes: in activities 3.1.2 and 3.1.3 to express different behaviours of **Nano** and **Tera** and in 3.1.4 to make **Pico** move and look **as if he was walking**.

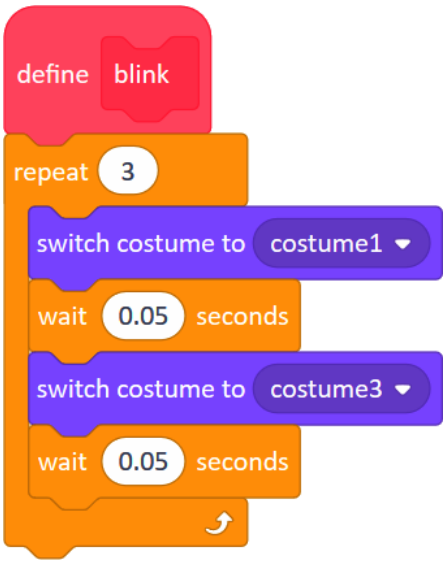
There we employed two important ideas:

- If we used **next costume** and the current costume of a sprite was its last one it automatically switched to the first costume (we used this feature in Module 4 as well).
- Inside the walking **forever** loop we were using the **if on edge, bounce** block to make sure that the sprite bounced whenever it run into an edge.

Costumes of the **Fleeeee** sprite are slightly different: there are five of them and they may be used for several different animations: the first four, when displayed in the **forever** loop 1 – 2 – 3 – 4 – 1 … will make **Fleeeee** look as if it is walking. Therefore we cannot use simple **next costume** loop here, as we have to 'skip' costume number 5.



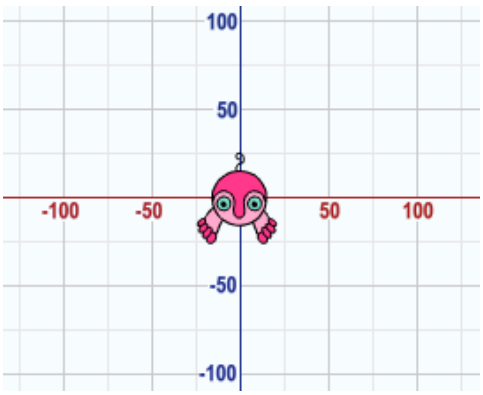
The **blink** and **nod** definitions are simple as they only switch a pair of costumes a fixed number of times with no movement. Our new block **run** switches between costumes 2 and 4 but makes **Fleeeee** move as well.

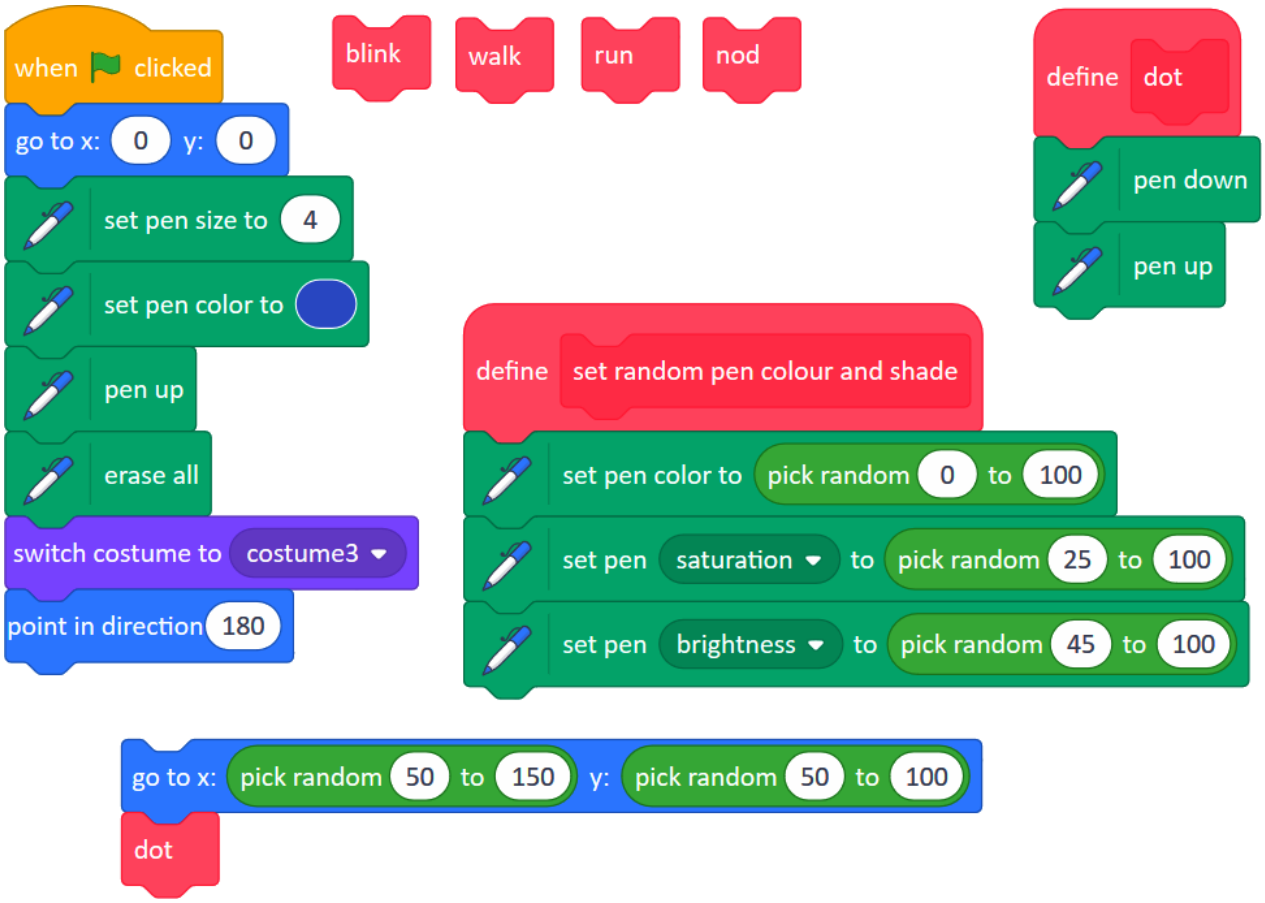


ADDITIONAL SUPPORT

1 Both **dot** and **set random pen colour and shade** are new blocks prepared for pupils so that they can quickly start exploring further activities with the **Fleeeee** sprite. Pupils have created the same or similar blocks themselves in earlier investigations.

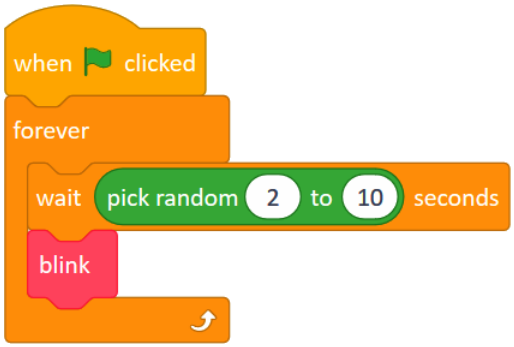
The other four new blocks – **blink**, **nod**, **walk** and **run** are explored and used only in **step 2 [Extension]** of this activity.

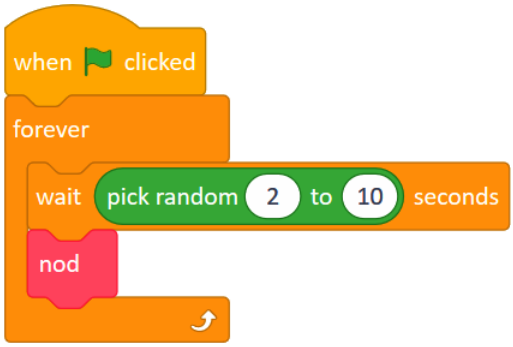




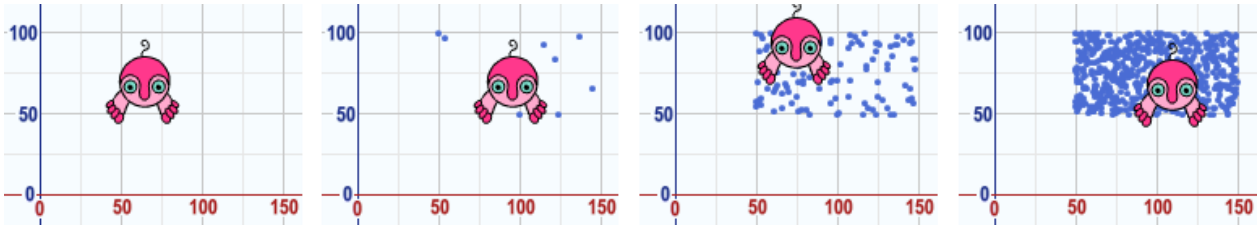
2 All of these new blocks are different animations based on the fact that the **Fleeeee** sprite has five similar but not identical costumes. The animations and their definitions are commented in more detail in the **Connections to Y5 ScratchMaths** on the previous page. They are not required to be used in this investigation. However, if pupils want to, they may have some *random animations effects* happening in parallel. For example, they may want the **Fleeeee** to have and run the following or similar extra scripts:

continued ▶

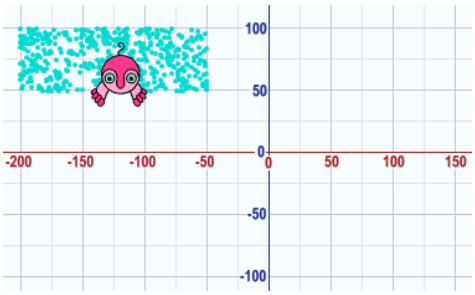


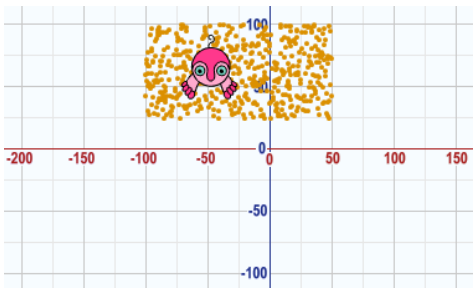


3

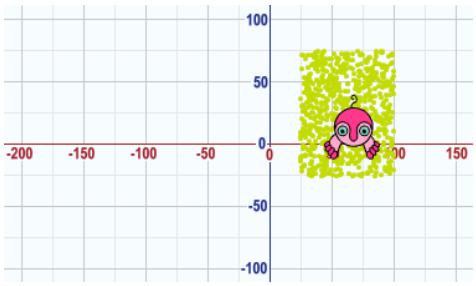


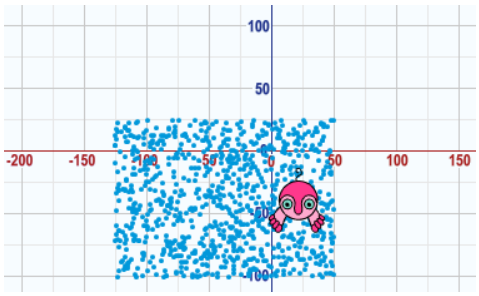
4 The values in the red circles are the missing numbers from the classroom presentation.





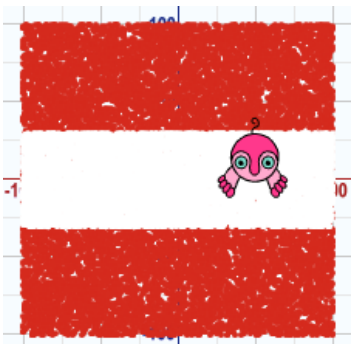
continued ▶



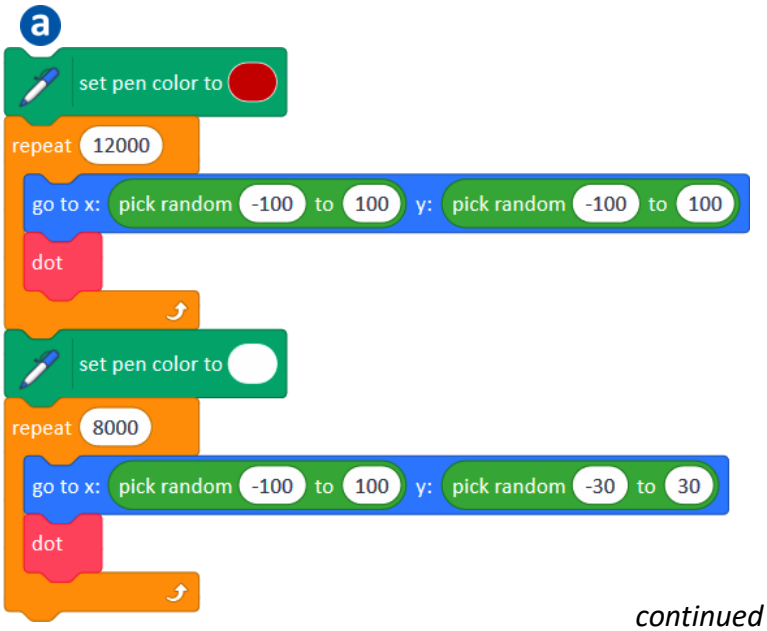


5 Note that when creating the following pictures of different emerging shapes:

- More than one **repeat** *random jumping loop* is applied in the same area of the stage, with different pen colours.
- It is desirable to increase the number of jumps to several hundreds or several thousands. To save time while running such script, you may decide to go to the **Edit menu** and select **Turbo mode**. Alternatively, you may turn the Turbo mode on (or off) by Shift + clicking the green flag. Please note - in other activities it is desirable to observe each step done in the stage so do not forget to switch Turbo mode off when not needed.
- Below is the first solution, see also solutions (b) and (c) on next pages.

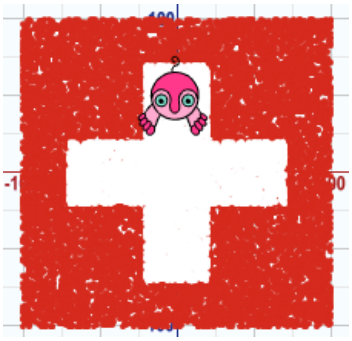


a

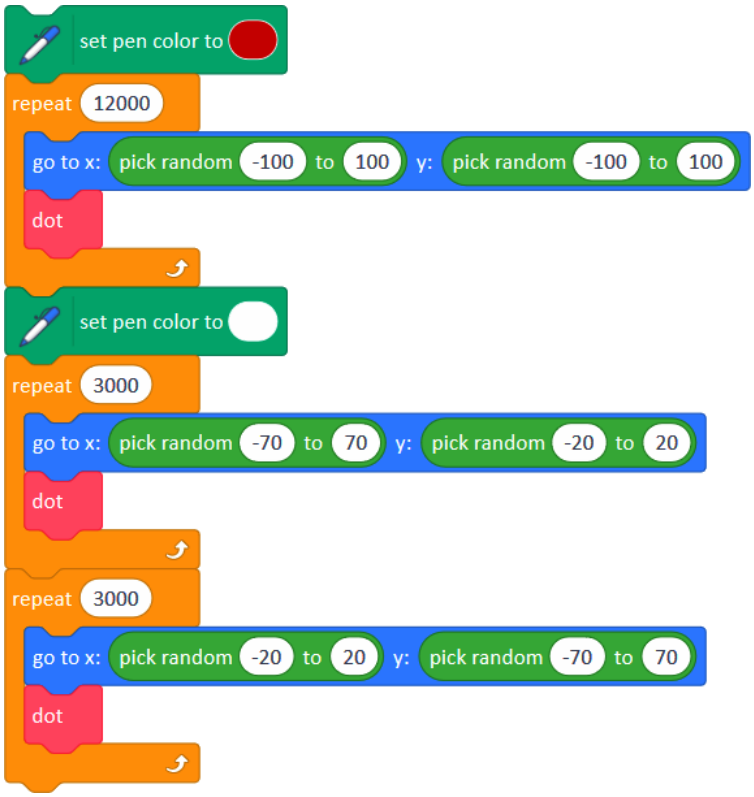


continued ▶

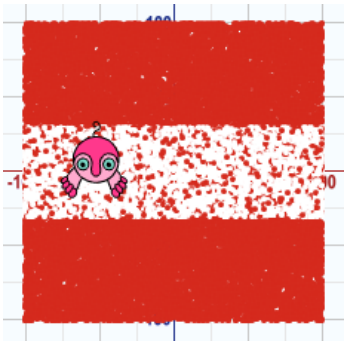
ADDITIONAL SUPPORT CONTINUED



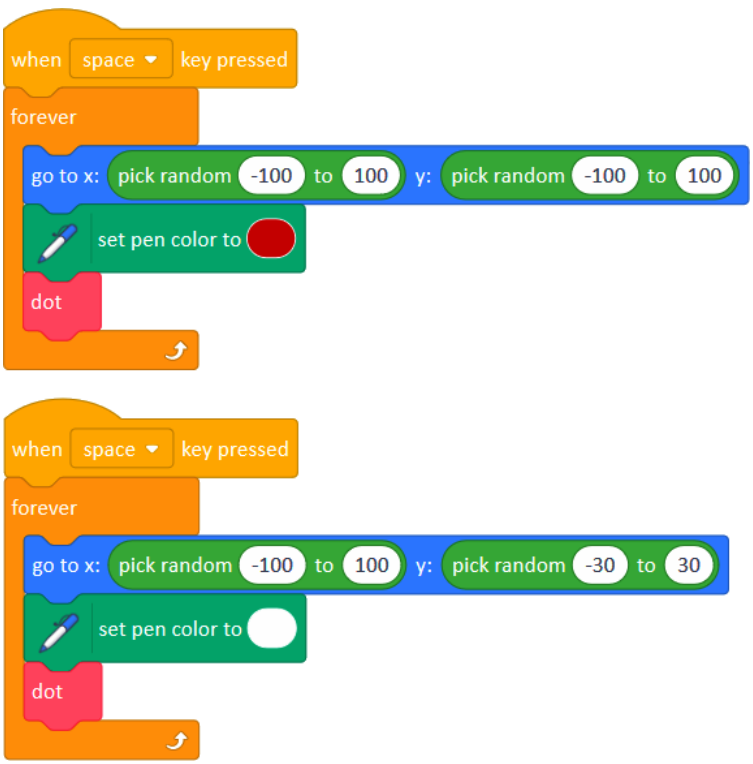
Can you think of any other flags you could create in a similar way?



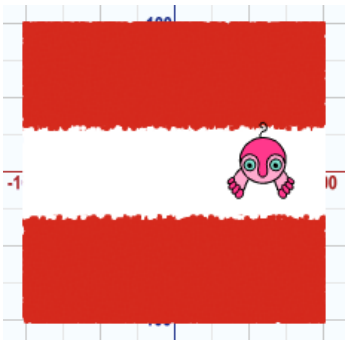
Compare solution (a) of (5) with two following strategies (algorithms) – (b) and (c). Try to understand the details that make those three algorithms similar and yet considerably different.



b



EXTENSION SUPPORT

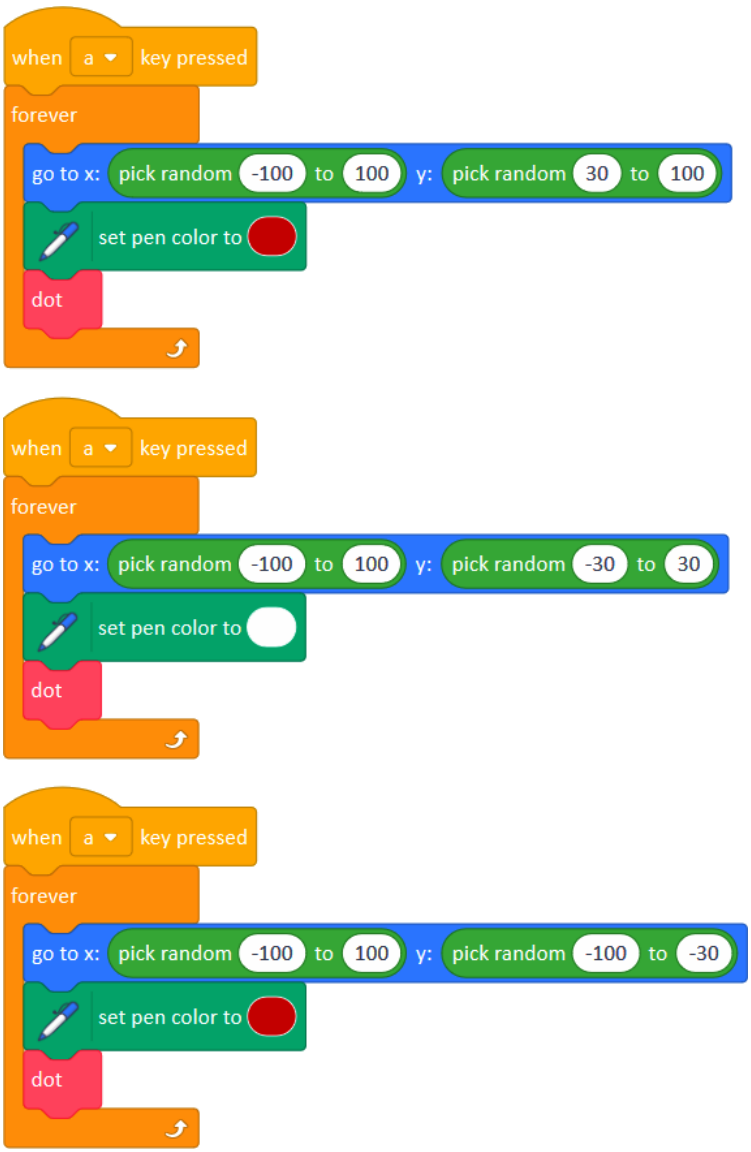


What are the differences between the three algorithms (the first one is from the previous page)?

Why are there red dots in the 'white' area in the second solution?

Why did we have to move **set pen color to …** *blocks inside the control structure* **forever***?*

c



Could you think of the fourth strategy, a version of the third script, which would use **repeat** *blocks instead of* **forever***?*

MODULE 6 • INVESTIGATION 1 • ACTIVITY 6.1.2

Unplugged and Hands-on: Envisage and Explore

LEARNING OBJECTIVES

Envisage how to create emergent shapes in different quadrants of the coordinate grid.
bridgE to full coordinate grid, reflections and translations.

ACTIVITY INSTRUCTIONS

This activity consists of three worksheets. Print and distribute them.

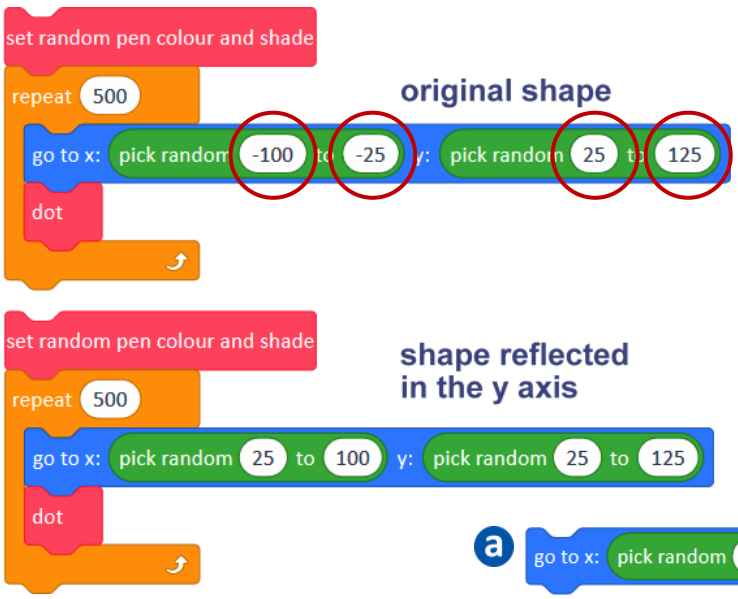
Pupils are asked to (a) fill in missing coordinates to produce the shape in the picture, then (b) add another script which will produce identical shape but reflected or translated – according to the instructions in 1 reflected in the y axis, in 2 reflected in the x axis, and in 3 translated.

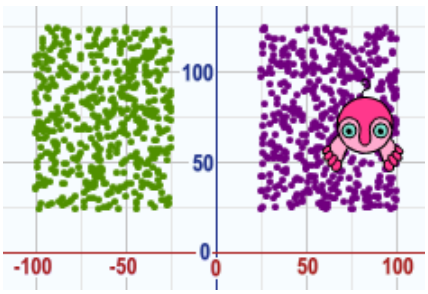
MATHEMATICS CONNECTIONS

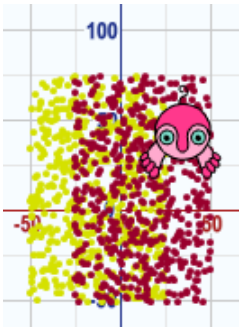
Encourage pupils to explain the connection between the numbers used in the ***go to x: y:*** *block* and the numbers used after the transformation. [e.g. a reflection in the y axis has the effect of transforming the x-coordinate into its opposite]

WORKSHEETS SOLUTIONS

1







a

go to x: pick random -150 to -25 y: pick random -75 to -25

go to x: pick random 25 to 150 y: pick random -75 to -25

b

go to x: pick random -75 to 0 y: pick random -25 to 100

go to x: pick random 0 to 75 y: pick random -25 to 100

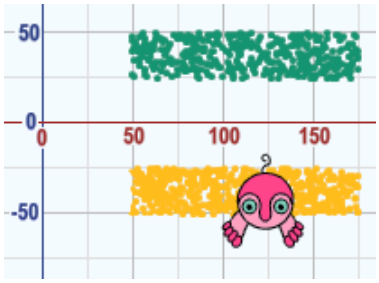
c

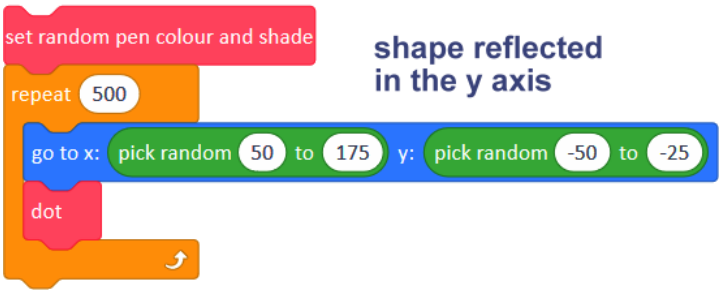
go to x: pick random -50 to 25 y: pick random -50 to 75

go to x: pick random -25 to 50 y: pick random -50 to 75

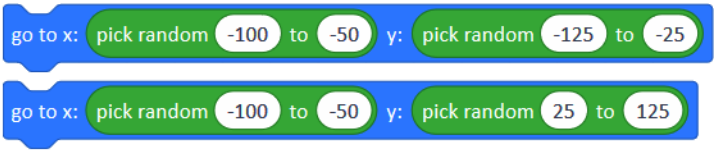
WORKSHEETS SOLUTIONS

2

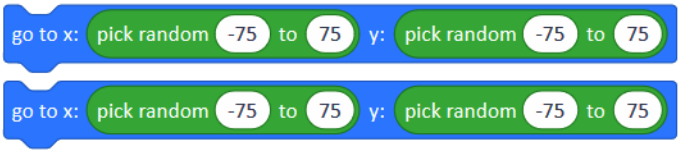




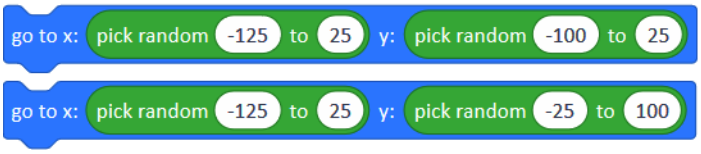
a

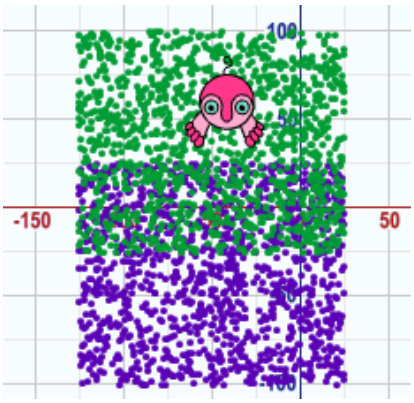


b

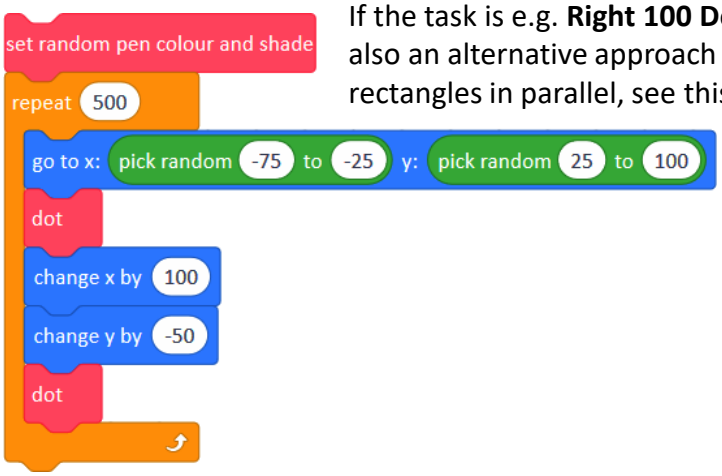


c

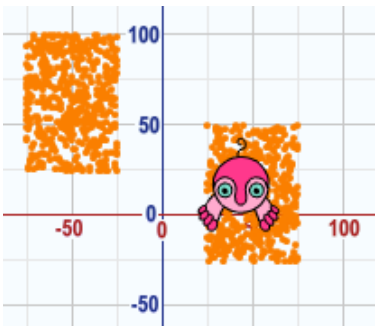




3



If the task is e.g. **Right 100 Down 50** (as is the first task), there is also an alternative approach to draw both original and translated rectangles in parallel, see this script:



1

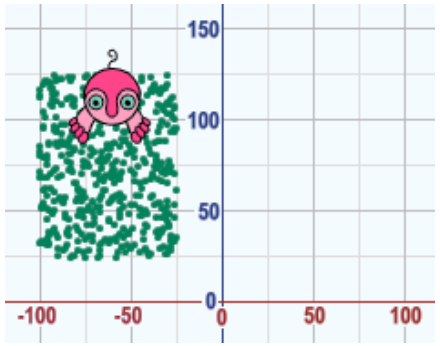
NAME	

PRINT

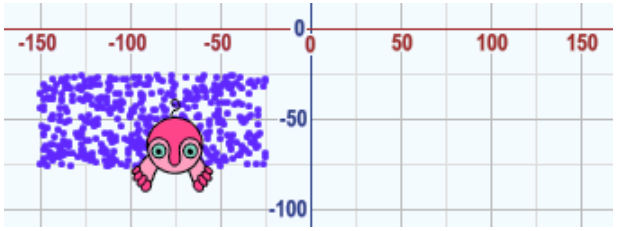
INVESTIGATION 1
Activity 6.1.2

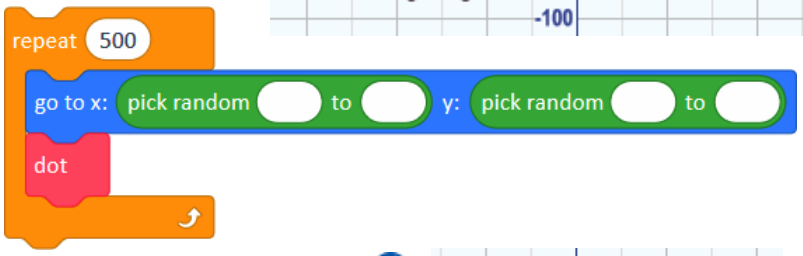
WHAT TO DO	Fill in the missing coordinates so that the script produces the emerging rectangle in the picture. Then build the second script in Scratch (similar to the first one) to produce the same shape but **reflected in the y axis**.

a

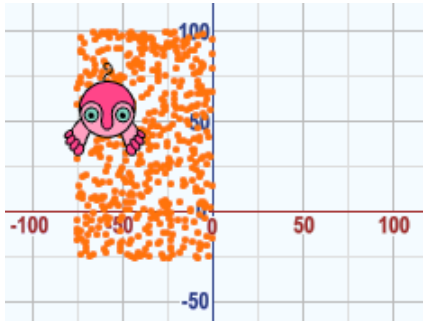


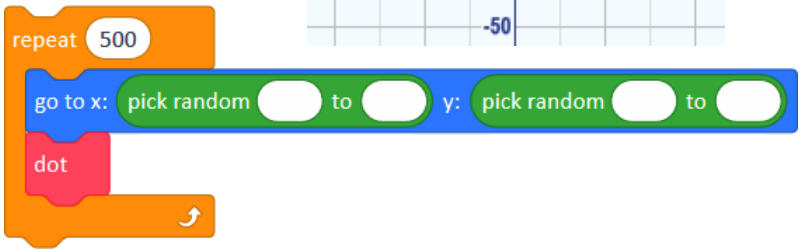
b



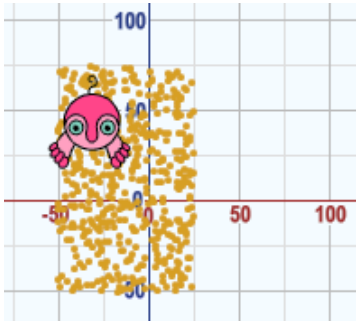


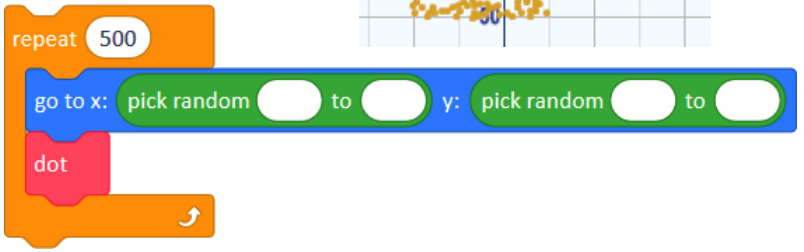
c





d





2

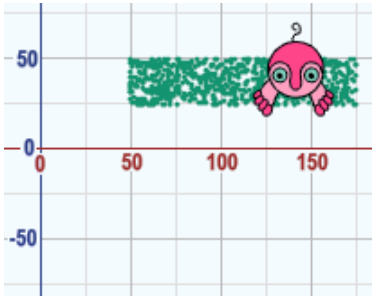
NAME	

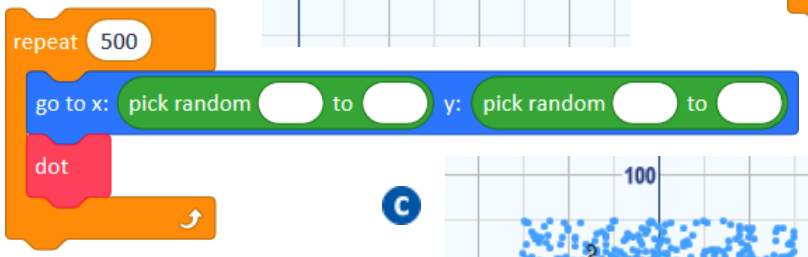
PRINT

INVESTIGATION 1
Activity 6.1.2

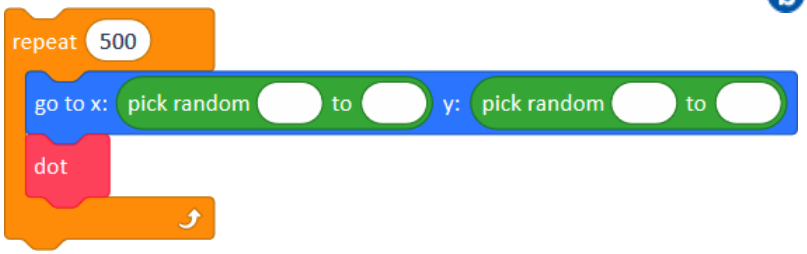
WHAT TO DO	Fill in the missing coordinates so that the script produces the emerging rectangle in the picture. Then build the second script in Scratch (similar to the first one) to produce the same shape **reflected in the x axis**.

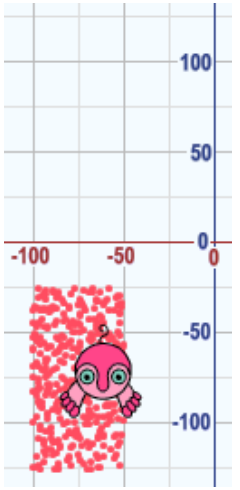
a



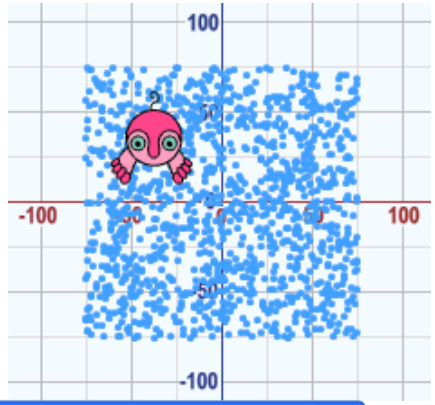


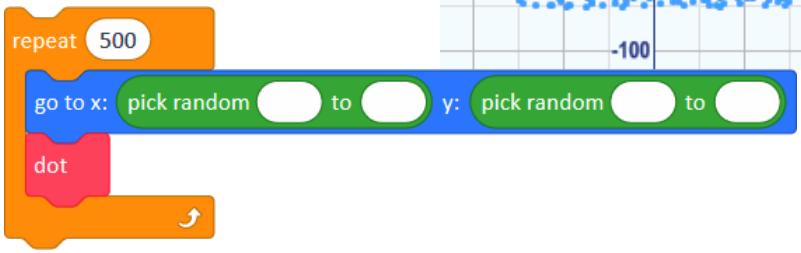
b



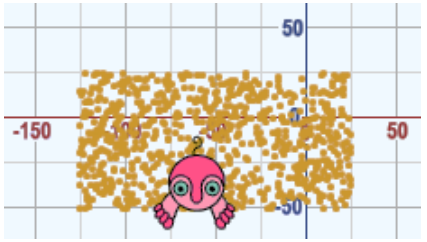


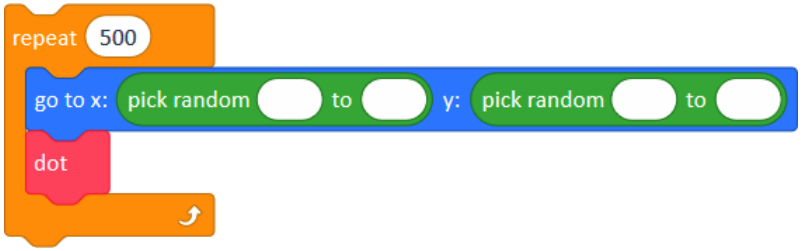
c





d





3

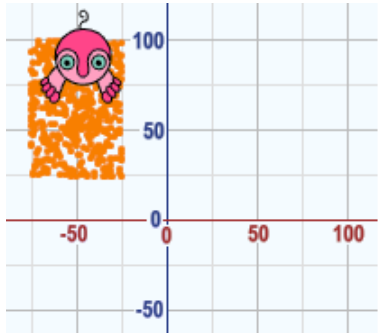
NAME	

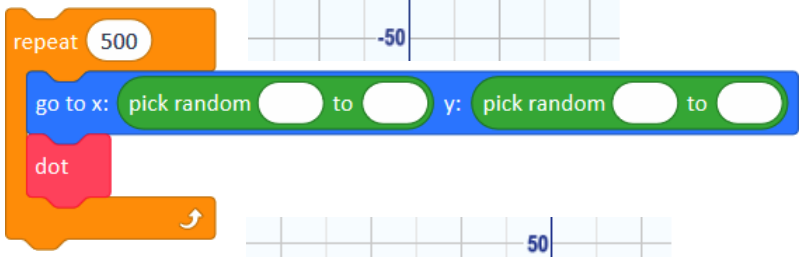
PRINT

INVESTIGATION 1
Activity 6.1.2

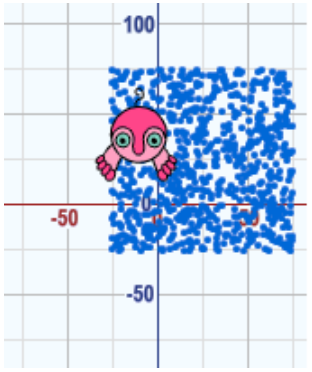
WHAT TO DO	Fill in the missing coordinates so that the script produces the emerging shape in the picture. Then build the second script to produce the same shape **translated right, left, up or down – based on the instruction**.

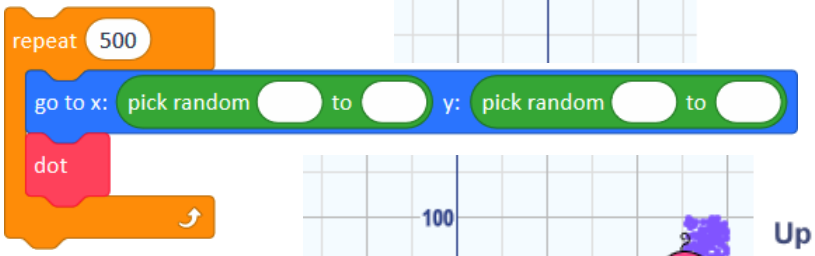
a Right 100 Down 50



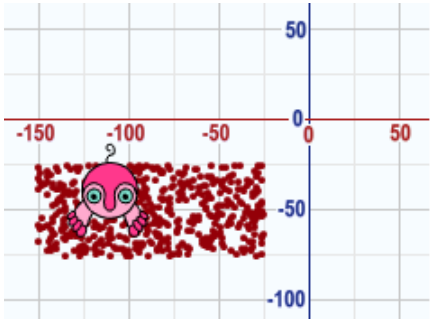


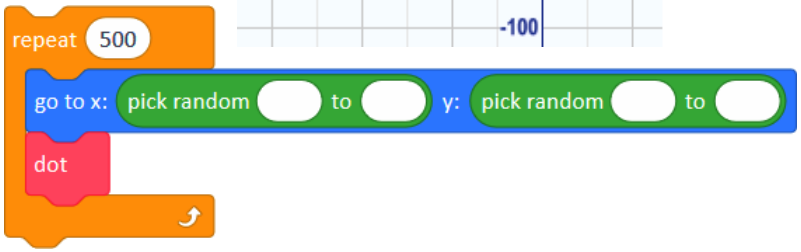
b Left 50 Down 50



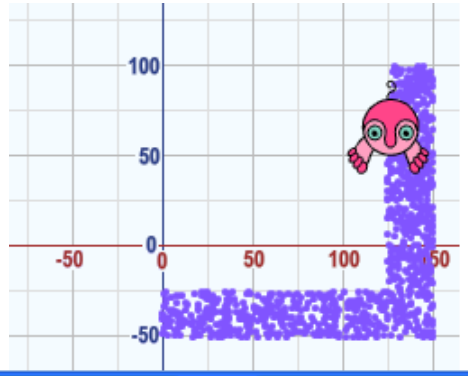


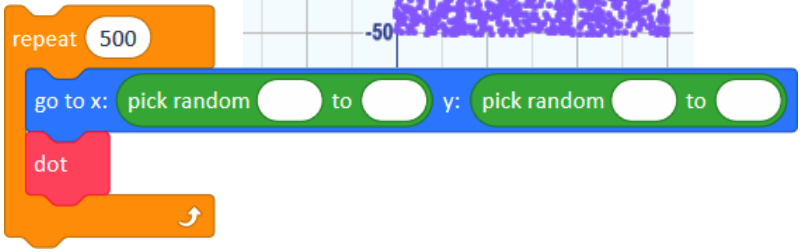
c Up 75 Right 50 Down 100





d Up 25 Left 50





MODULE 6 • INVESTIGATION 1 • ACTIVITY 6.1.3
Introducing Scale

LEARNING OBJECTIVES

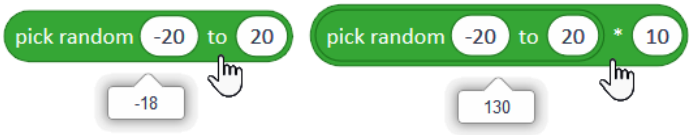
Explore how to create emerging shapes of different scale factors.
Explain how to change the scale of the coordinates grid.

ACTIVITY INSTRUCTIONS

Pupils continue in their project **61-Fleeeee Dots**. The final version of this project at the end of activity 6.1.3 will be **61-Fleeeee Dots FINAL**.

1. Pupils build the script to produce a **small** emerging square centred at **0, 0** (see right). They then pull out the **pick random …** block from the **y: …** hole of the **go to …** and run the script again. [The **y: …** hole will reset to the default value of 10.] They change the **y: …** value to other constant values and explore. [One of the values they try should be 0. Explore and discuss]

2. Pupils pull out the **pick random -25 to 25** from the **x: …** hole as well, but keep it isolated and explore which values it produces when clicked. They build a composite reporter block – for multiplying the same random value by 10…



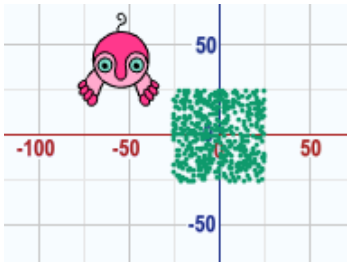
…and explore which values it produces **when clicked**. They explore it with different multipliers: 2, 5, 100…. When discussed, they make a variable **Scale** and use it instead the multiplier with different values.

3. Pupils insert the whole composite block back into the **x: …** hole and run the script. They duplicate it into the **y: …** hole as well and run the script. They increase the pen size in the *setup script* (to 8 or 10 or even more – up to the **Scale** value itself) and experiment with different **small numbers** for **pick random …** blocks.

4. Pupils set **Scale** to 25 and run the script, then switch backdrop to *grid 25* without clearing the stage. Discuss the dual way of referring to the points (dots) here: **standard Cartesian coordinates** and our new **'grid 25 coordinates'** (identical to the grid points of The Grid World in Module 5). Rename your variable **Scale** to **grid**, which is now more intuitive.

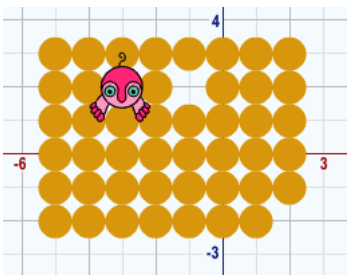
5. Pupils build scripts that produce emerging rectangles similar to those in the classroom presentation.

MATHEMATICS CONNECTIONS



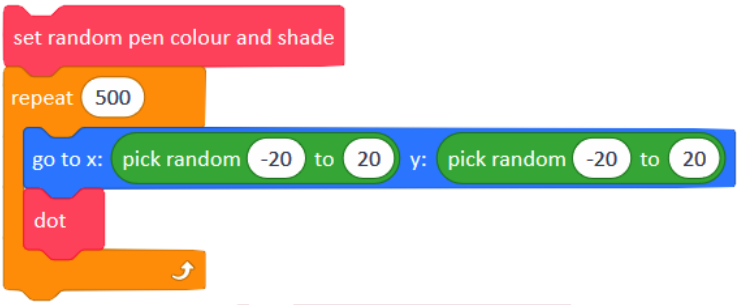
Discuss: What role does this multiplier play here? What if we replace it by 1, or 5, or 20? [It is a **scale factor** for numbers randomly picked from the specified interval]

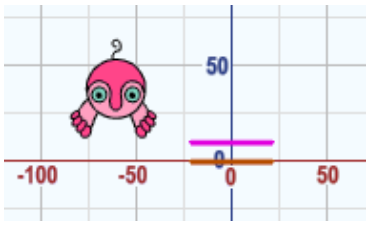
Discuss: If we set **Scale** to 10, which values you can use in both **pick random …** blocks so that the emerging shape fits within the stage? What if we set **Scale** to 20?



ADDITIONAL SUPPORT

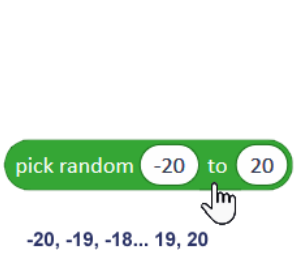
1 For the initial small square use inputs for the **pick random …** blocks like **-15** to **15** or **-20** to **20**…, so that even after multiplying by 10 iy will fit (as x coordinates) into the stage:



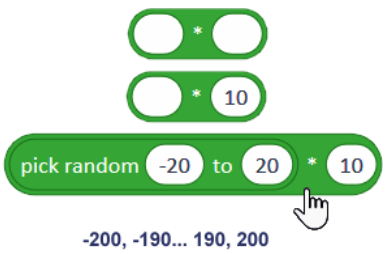


Note - when a reporter block is pulled out from a block, the value restored as an input in that block is typically 10.

2

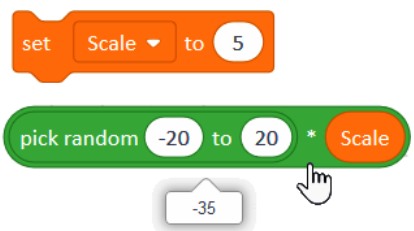


-20, -19, -18... 19, 20



-200, -190... 190, 200

-200, -190... 190, 200



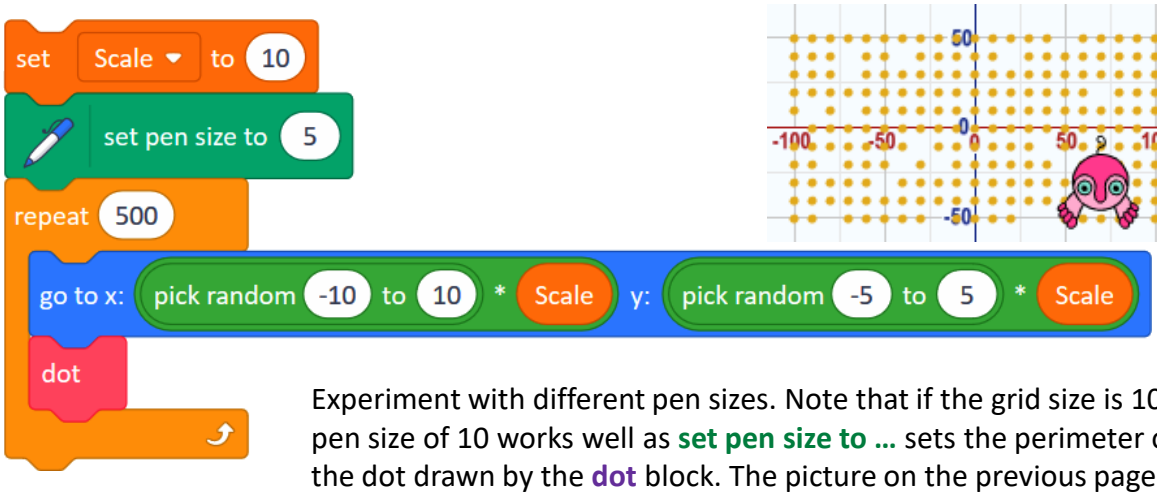
-100, -95, ... 90, 95, 100

When you make a new variable and build a block to set its value (to 10 or any other value), **do not forget to run** the **set** block. Otherwise the value of the variable will be 0, its initial value pre-set by Scratch.

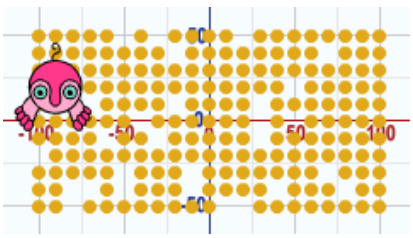
continued ▶

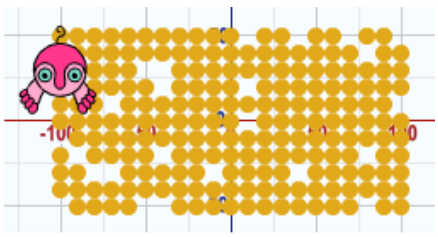
ADDITIONAL SUPPORT CONTINUED

3



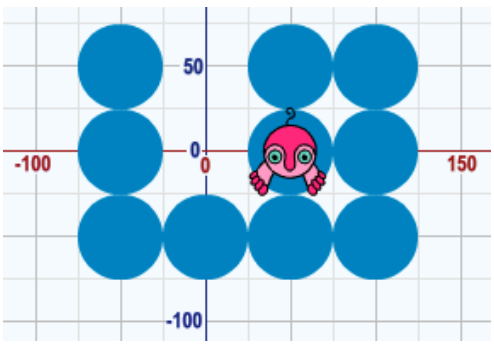
Experiment with different pen sizes. Note that if the grid size is 10, a pen size of 10 works well as **set pen size to …** sets the perimeter of the dot drawn by the **dot** block. The picture on the previous page illustrates jumping with the pen size of 5, below left is 8 and the right is 10. Therefore, instead of **set pen size to 5** or **set pen size to 8** we will use more a general command **set pen size to Scale**:

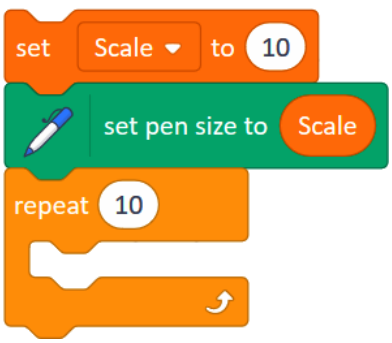




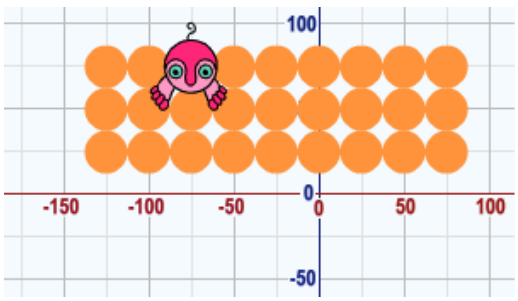
Try the one (with the pen size set to 50) on the right as well.

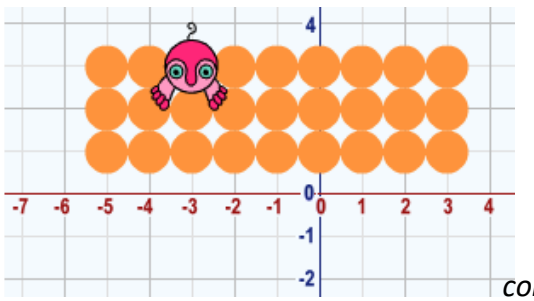
Note also that the bigger the dots get, the smaller **repeat** number suffices to cover the whole (or almost the whole) emerging rectangle.





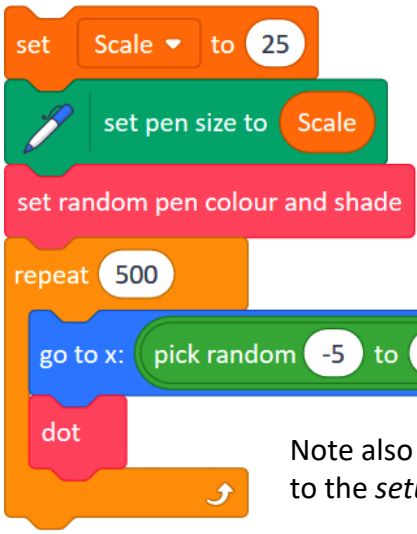
4



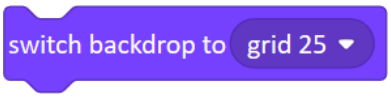


continued ▶

ADDITIONAL SUPPORT CONTINUED



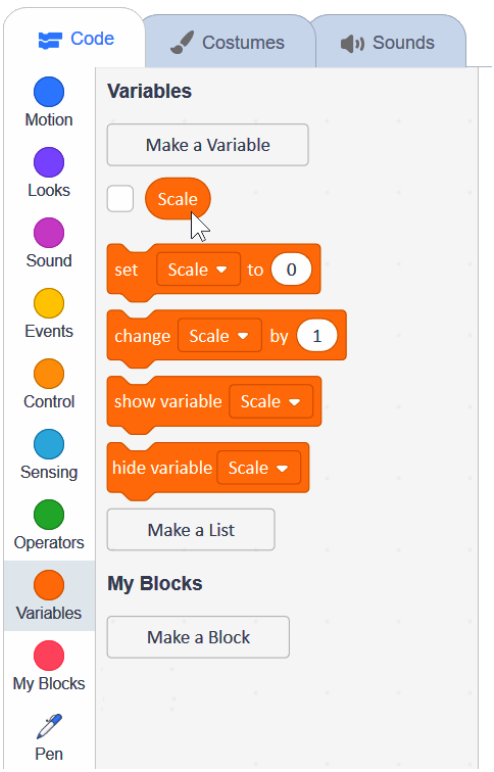
Note how straightforward it is to see the connection between the **pick random … to … input values** and the grid coordinates, when using the **Scale** set to 25 and *grid 25* backdrop.

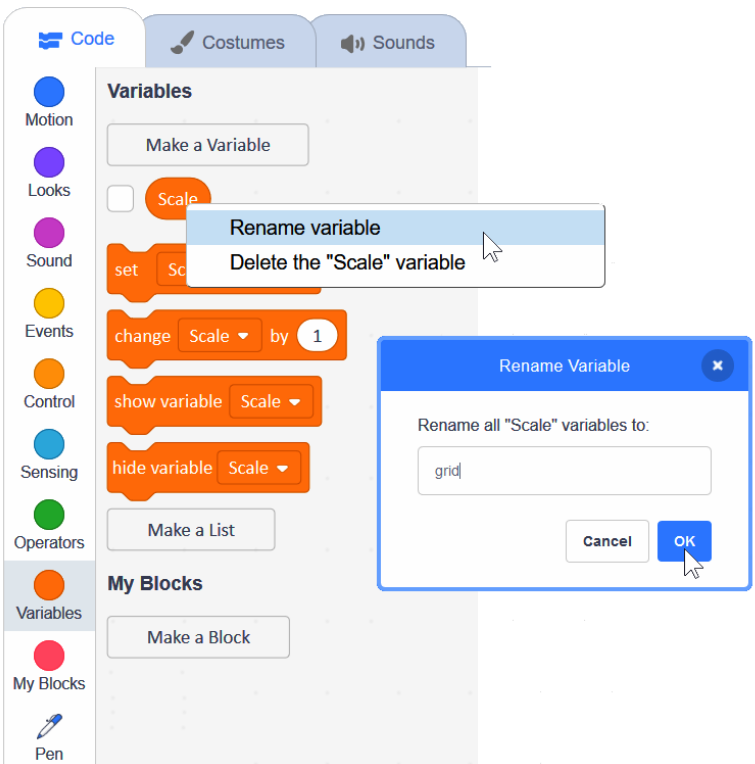


Note also that **set *Scale* to …** and **set pen size to Scale** blocks may be moved to the *setup script*.

Since we are now using *grid 25* backdrop (and later also *grid 10*) and drawing big dots – the same size as the grid, it might by more intuitive to rename our variable **Scale** to **grid** – as we did in Module 5 - Investigation 3, which is straightforward in Scratch.

To rename a variable, go to the **Data** group, right click (offline) or **Shift + click** (online and offline) the variable block, select **rename variable** and in the dialogue box change its name. All blocks with the old name in your project will be automatically modified. In a similar way, you can rename your own blocks or any broadcast message.

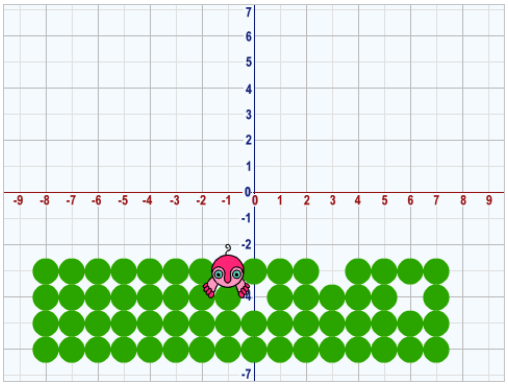
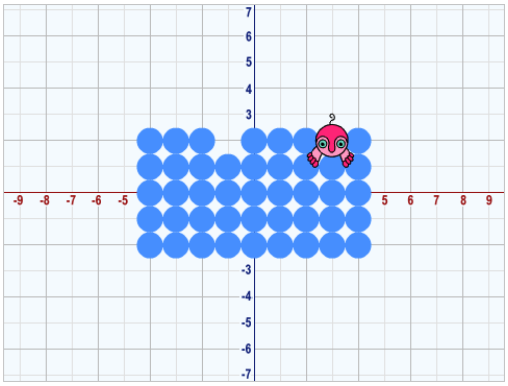
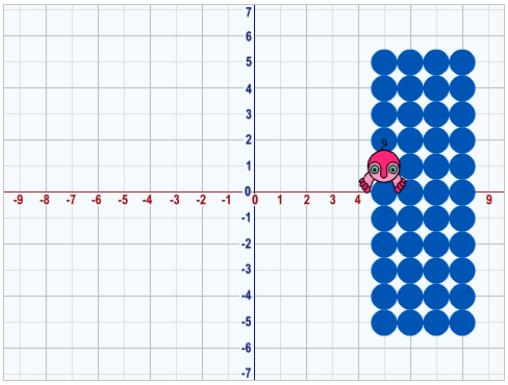
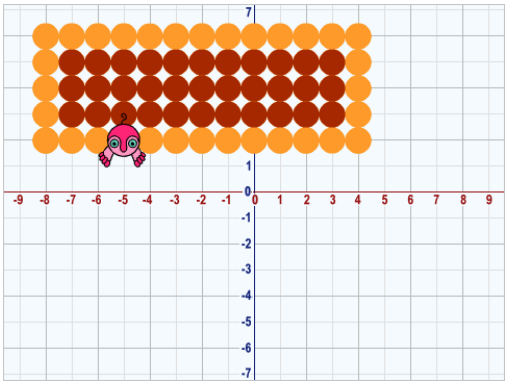
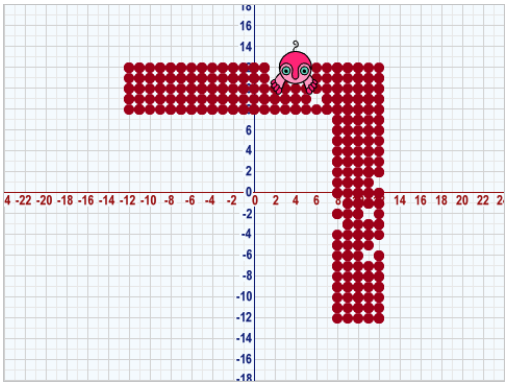
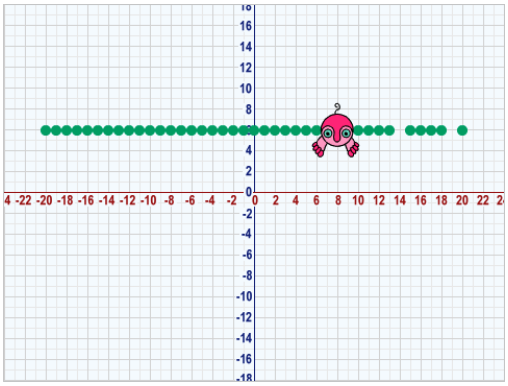
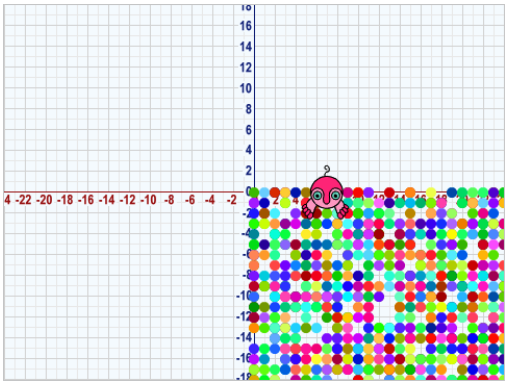
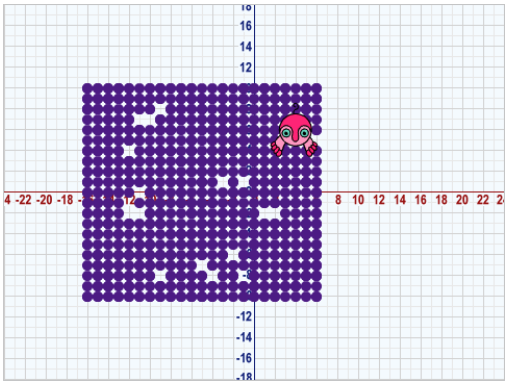




continued ▶

ADDITIONAL SUPPORT CONTINUED

5 Pupils keep the **grid** variable as 25 and try to create some of these emerging rectangles and shapes, or similar (continued on next page). They may also set **grid** to 10, switch the backdrop to *grid 10* and try some other shapes. *Discuss: which of theses shapes have what kind of symmetries?*

Module 6 • Investigation 1 • Activity 6.1.4
[Extension] Dotty Patterns

Learning Objectives

Explore how to simplify the jumping script to be a single line.
Explain how to set the left and right most positions that the sprite will jump to.

Activity Instructions

Pupils continue in their project **61-Fleeeee Dots**. The final version of this project at the end of Activity 6.1.4 will be **61-Fleeeee Dots Extension FINAL**.

Let **Fleeeee** now jump only within one line.

1. Pupils try to restrict the **pick random …** values for the **y: …** position to be the same number, e.g. **pick random -3 to -3** – the jumping behaviour now happens within one line. For this we may simplify the whole **repeat** by replacing the **go to …** block by **set x to …** and use

before **repeat** only once for the whole jumping. Pupils build this simplified jumping behaviour, making their own block for the **repeat** part of it – named for example **one line jumping**.

2. To make **Fleeeee** move to the upper 'line' (and possibly run **one line jumping** there again) pupils build and explore…

Pupils build a script using **one line jumping** inside **repeat** - making **Fleeeee** jump one line higher each time. They may use different random pen colours for dots in each line.

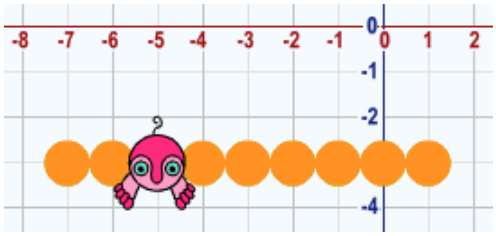
3. Pupils explore the **one line jumping** definition: what sets the left most position within a line where **Fleeeee** may jump? They turn that value into a variable, named e.g. **Left**. They use **Left** in the definition and explore different initial values of **Left**.

4. Pupils insert **change *Left* by 1** block inside the **repeat** and explore different initial values and resulting patterns. They may also try changing **Left** in the **repeat** loop by 2, by -1 etc.

5. So far in our examples the right most position within each line was 1. Pupils try it as **Left + 5** or **Left + 2**.

6. The right most position may be specified by another variable and change itself in the **repeat** loop as well.

Mathematics Connections

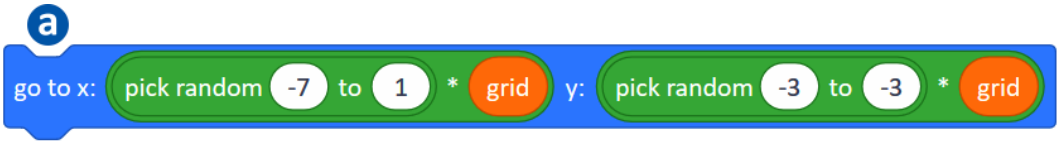


Note that **-3** in the **set y to …** block on the left represents the *grid coordinate* of all dots in that line of the grid, while **-3 × grid** is their Cartesian y co-ordinate.

ADDITIONAL SUPPORT

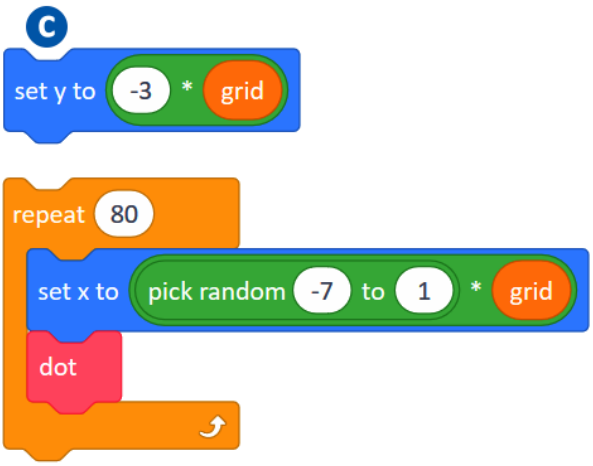
1 These are the steps of the transition from previous jumping into the new **one line jumping** - **pick random …** for **y:** can be turned into **(b)** and used separately in **(c)** before the **repeat** block, which will now use **set x to …** and can be turned into a new block:

a

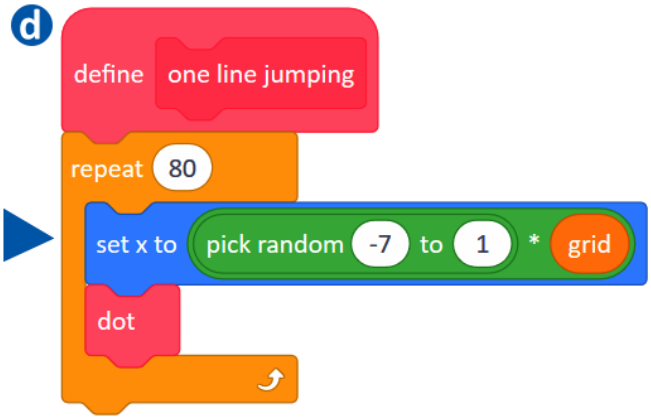


b

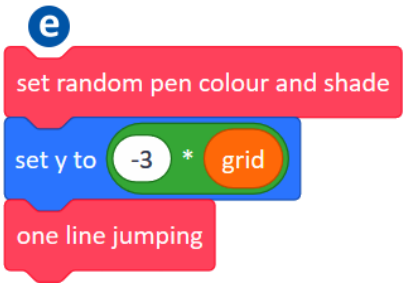
c



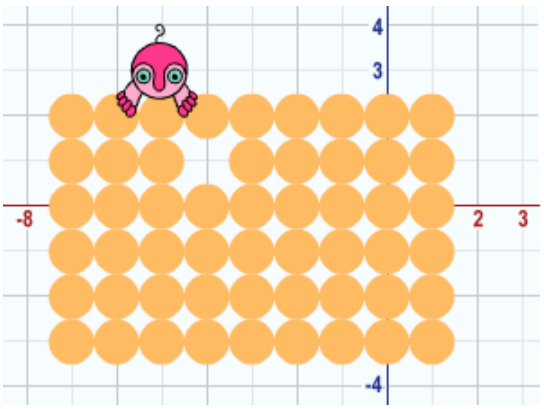
d

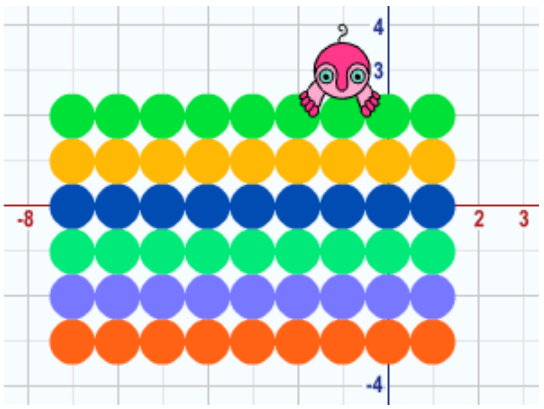


e



2

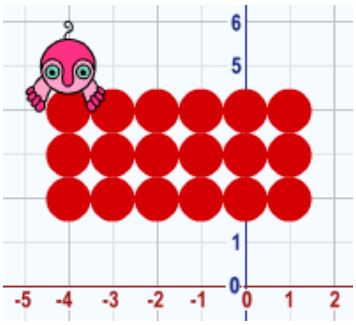


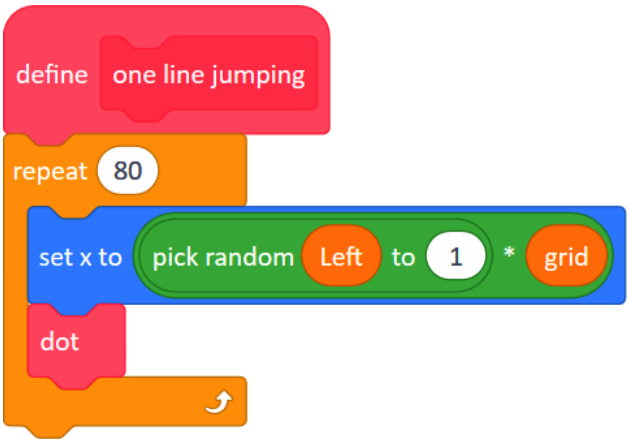


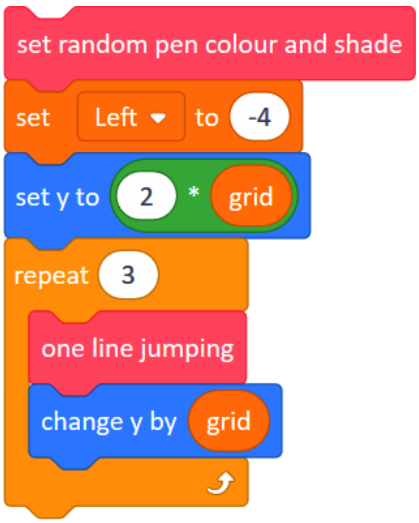
continued ▶

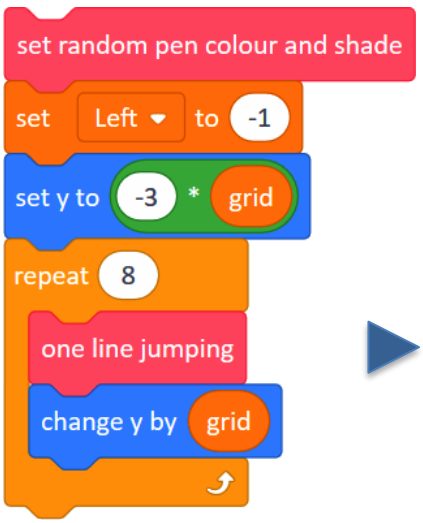
*Why does **Fleeeee** always ends up one line above the last line drawn? [because the last line in the repeat is to change y by grid]*

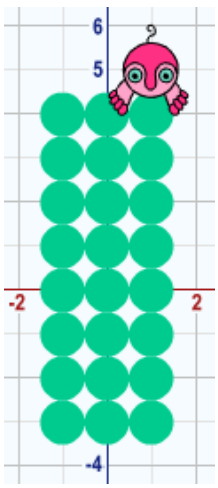
3



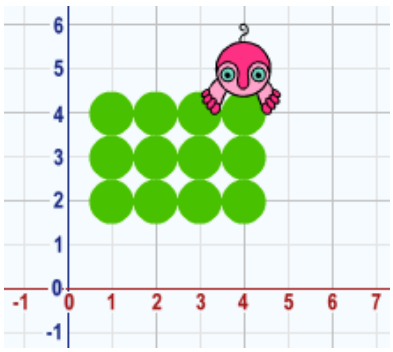








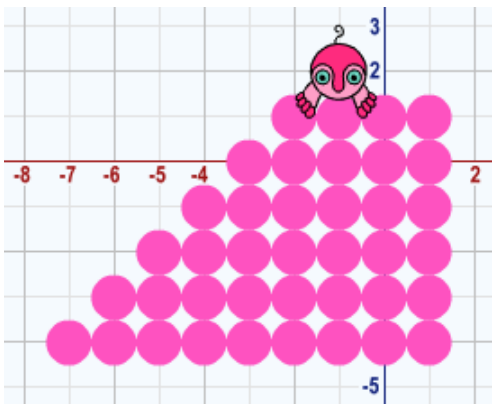
continued ▶



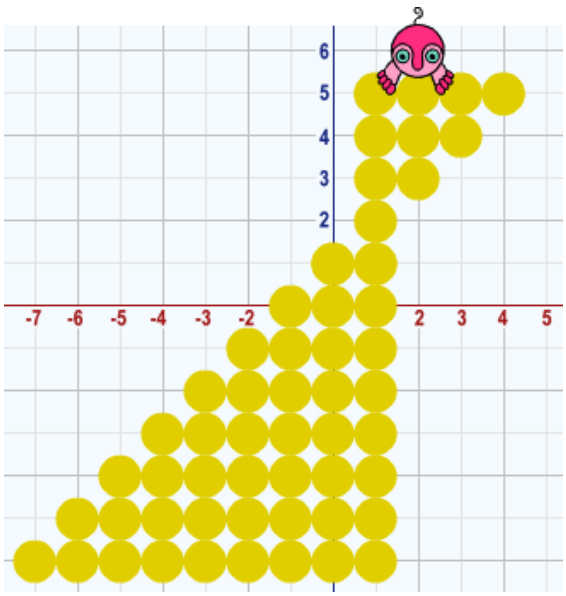
```
set random pen colour and shade
set Left to 4
set y to 2 * grid
repeat 3
    one line jumping
    change y by grid
```

Due to the fact that **pick random 4 to 1** produces the same set of values as **pick random 1 to 4**, we can set **Left** to a value which is in fact to the right of the second input, see the example above.

4



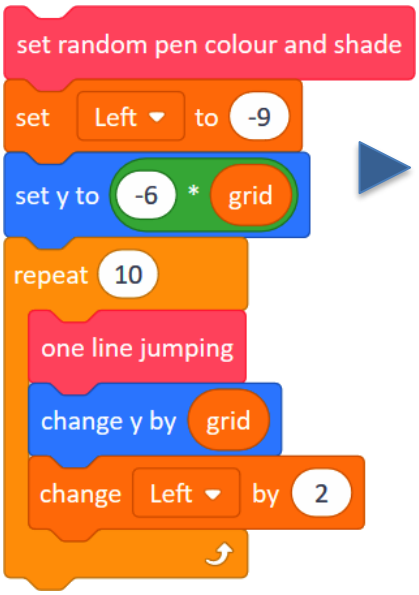
```
set random pen colour and shade
set Left to -7
set y to -4 * grid
repeat 6
    one line jumping
    change y by grid
    change Left by 1
```

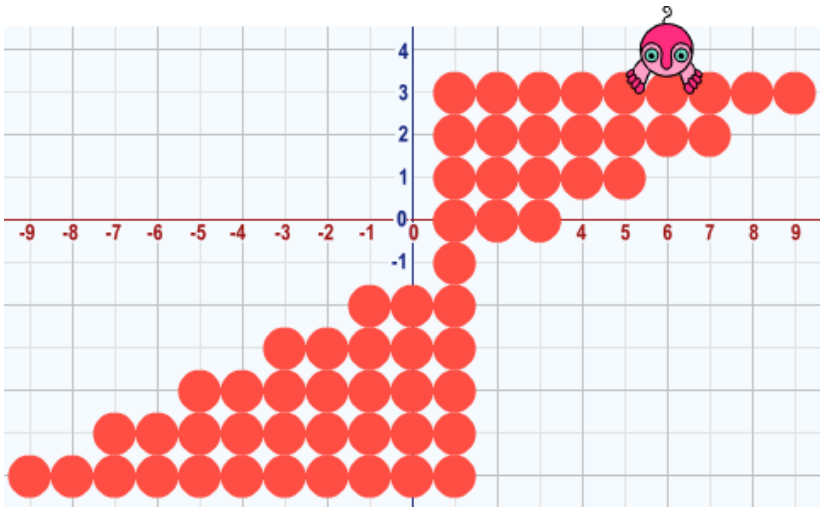


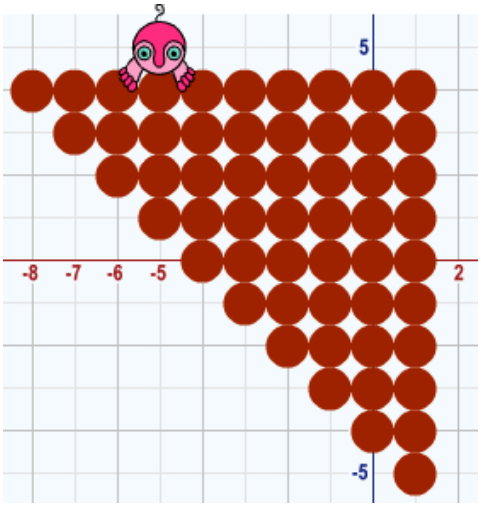
```
set random pen colour and shade
set Left to -7
set y to -6 * grid
repeat 12
    one line jumping
    change y by grid
    change Left by 1
```

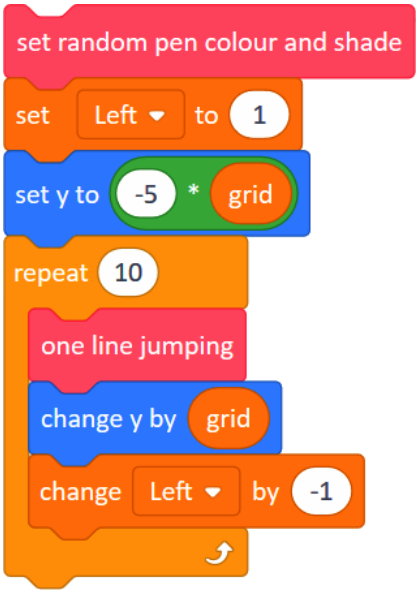
continued ▶

ADDITIONAL SUPPORT CONTINUED



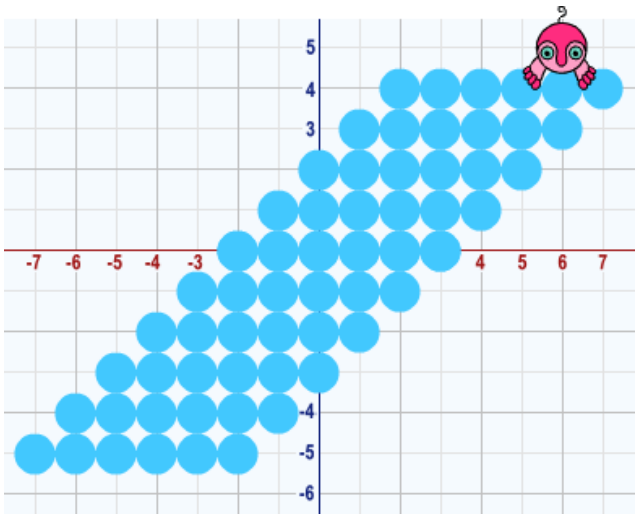






ADDITIONAL SUPPORT CONTINUED

5

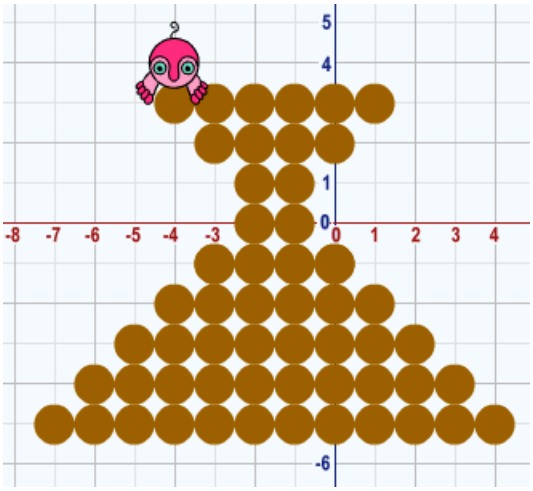


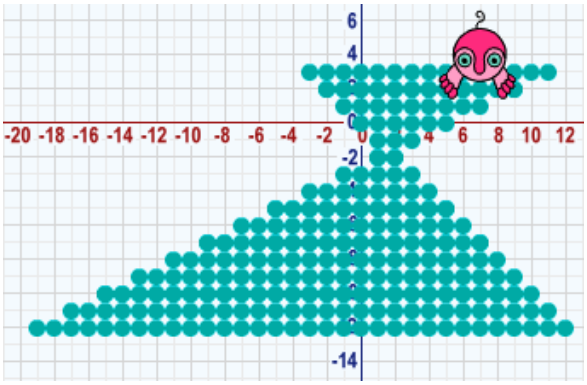

```
set random pen colour and shade
set Left to -7
set y to -5 * grid
repeat 10
    one line jumping
    change y by grid
    change Left by 1
```

```
define one line jumping
repeat 80
    set x to pick random Left to Left + 5 * grid
    dot
```

6

```
define one line jumping
repeat 80
    set x to pick random Left to Right * grid
    dot
```

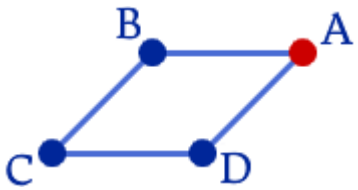




MODULE 6: INVESTIGATION 2

Coordinate Shapes

This investigation develops the use of coordinates in the context of vertices of shapes which have geometric properties. The activities encourage the reading and writing of coordinate notation through the use of the letter of the alphabet on a grid. Scaling is revisited to transform pixel coordinates to grid coordinates. Connections can be made to the geometric properties of shapes by creating problems and encouraging pupils to exchange their own creations using the **62-Coordinates FINAL** project which is built during the investigation.



- **Activity 6.2.1** – Letters and Co-ordinates
- **Activity 6.2.2** – Busy Fleeeee And Clever Points
- **Activity 6.2.3** – Tricky Triangles
- **Activity 6.2.4** – Quirky Quadrilaterals

Scratch projects

62-Grid Letters
62-Grid Letters FINAL

62-Coordinates
62-Coordinates INT1
62-Coordinates INT2
62-Coordinates FINAL

LINKS TO PRIMARY NATIONAL CURRICULUM

CURRICULUM OBJECTIVES	LINK WITH SCRATCHMATHS
Mathematics Describe positions on the full coordinates grid. Interpret remainders as whole number remainders, fractions, or by rounding, as appropriate for the context. Solve problems involving similar shapes where the scale factor is known or can be found. Compare and classify geometric shapes, including quadrilaterals and triangles, based on their properties and sizes. Plot specified points and draw sides to complete a given polygon.	▶ Pupils are required to code and decode messages using the positions of letters on the coordinates grid. ▶ [Extension] Pupils encounter the need to round the result of division operations in order to get sprites to snap to the nearest grid point. ▶ Pupils are required to use the grid variable (in a similar way to the previous investigation) to adapt their project to different grid intervals. ▶ Pupils create and exchange coordinate problems whose solutions require coordinate and properties of shape knowledge.

Module 6 • Investigation 2 • Activity 6.2.1
Letters and Coordinates

Learning Objectives

Explore how to create a coordinate coding scheme.
Exchange messages/tasks with a partner to decode using the coordinate coding scheme.

Activity Instructions

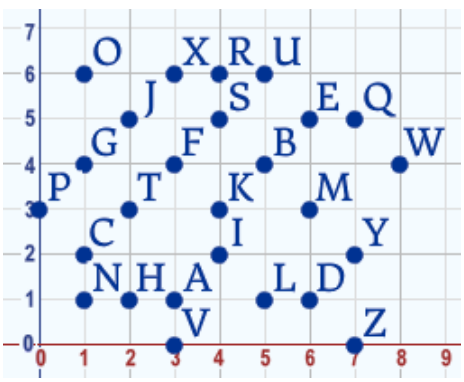
Pupils open project **62-Grid Letters**, **Save as a copy** (online) or **Save as** (offline) and rename. The final version of this project at the end of Activity 6.2.1 will be **62-Grid Letters FINAL**.

1. Pupils explore the project, the **red point** sprite, its *setup script* and its costumes.

2. Similar to Activity 5.3.2, we want to control the position of the **red point** sprite by four arrow keys. Pupils build four scripts: e.g. **when** *up arrow* **key pressed**, the sprite will **point in direction 0 (up)** and **move** forward by **grid** steps. Similarly, the other three arrow keys will make it point in the corresponding direction and **move** to the next **grid point**.

3. Pupils choose a letter (e.g. the first letter of their name) and build a script: if that letter is pressed, the **red point** will switch its costume to that letter (i.e. the costume with that name), **stamp** the letter and switch back to its *red point* costume.

4. They build (by duplicating) several similar scripts for some more letters and in pairs play the activities on the following page.



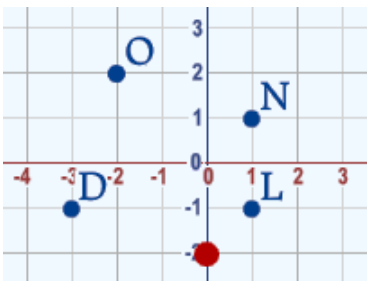
Mathematics Connections

Discuss: What are the names of the costumes? Are there any variables in the project?

Discuss: What is the value of the **grid** variable? How can we find out? How does it correspond with the backdrop? [Click the **grid** variable block in the palette, or drag out onto the stage, it will report 25. Each square on the backdrop corresponds to 25 pixels. We can check this by dragging the red point sprite to corners of the grid on the backdrop.]

Remember how the costumes of the **ones** sprite in Module 4 were named? They were 1, 2, 3… Now they are *red point*, then *A*, *B*, *C*…

Pupils navigate the **red point** sprite by arrows and scatter the letters in the grid.



Use the following list of coordinates to read the word:
(1, -1), (-2, 2), (1, 1), (-3, -1), (-2, 2), (1, 1)
L O N D O N

ACTIVITY INSTRUCTIONS CONTINUED

Coordinate challenge – from nRich activity 5038 **(**https://nrich.maths.org/5038)

Individually

Can pupils position the ten letters in their correct places according to the eight clues below?

A	B	C	D	E	P	S	X	Y	Z

Clues:

1. *The letters at (1,1), (1,2) and (1,3) are all symmetrical about a vertical line.*
2. *The letter at (4,2) is not symmetrical in any way.*
3. *The letters at (1,1), (2,1) and (3,1) are symmetrical about a horizontal line.*
4. *The letters at (0,2), (2,0) have rotational symmetry.*
5. *The letter at (3,1) consists of just straight lines.*
6. *The letters at (3,3) and (2,0) consist of just curved lines.*
7. *The letters at (3,3), (3,2) and (3,1) are consecutive in the alphabet.*
8. *The letters at (0,2) and (1,2) are at the two ends of the alphabet.*

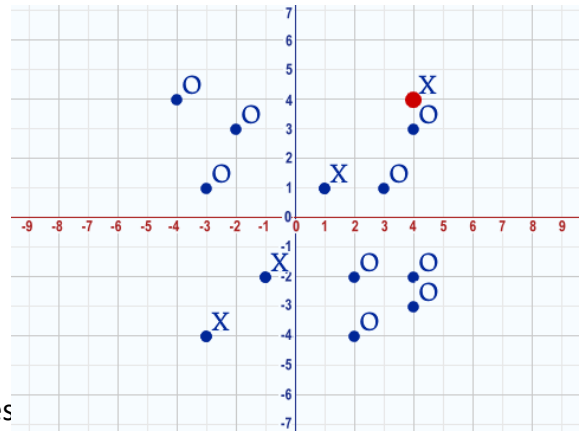
Code breaking

In pairs

1. Pupil 1 and 2 stamp the alphabet letters into the four quadrants.
2. Pupil 1 writes a secret question using coordinate notation for their alphabet grid.
3. Pupil 2 decodes the question and responds using coordinate notation for their alphabet grid.

Coordinate bingo

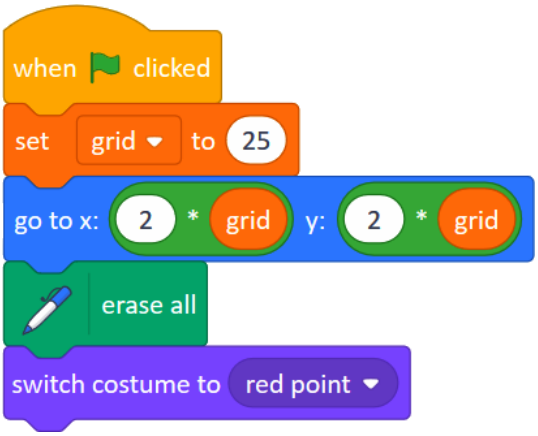
Individually, led by the teacher



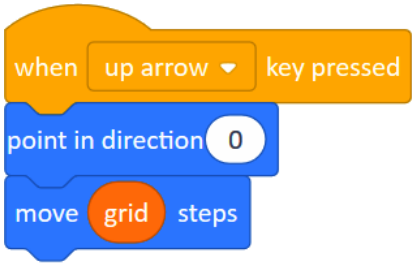
1. Pupils stamp 15 coordinates using the **letter O** on the -4 to 4 axes only (to speed up the game – e.g. see picture on right)
2. Teacher (aka the Bingo Caller) calls out coordinates like bingo numbers.
3. Pupils stamp the **letter X** over the O if their chosen coordinates are called out.
4. The first pupil to cross off all 15 of their choices is the winner and should shout out “Bingo!”.

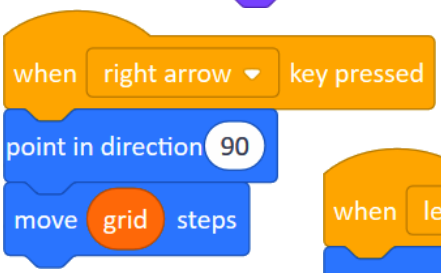
ADDITIONAL SUPPORT

1 This is the *setup script* of the **red point** sprite. Note that the first block could have been deleted as the initial value of the **grid** variable is saved in the project. However, the script is easier to read and understand with this.

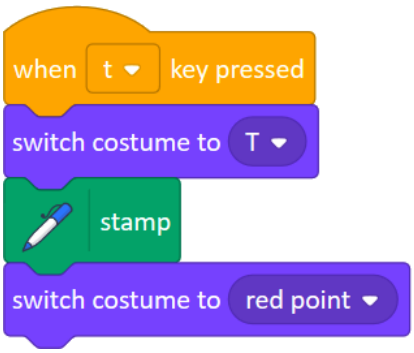


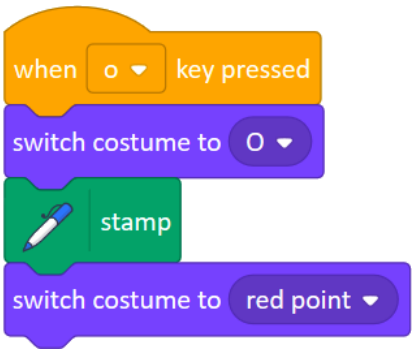
2



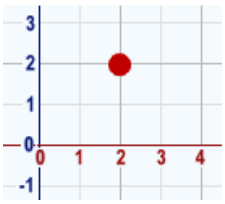


3 Here are the example scripts for two letters – T and O (note that they are almost identical and any other script can be created by duplicating and slightly modifying). The **red point** sprite sits in a grid point, see (a). Then we press T – the sprite will switch its costume to letter T with a small dark blue dot, stamp it, then switch back to its *red point* costume, see (b). We then press e.g. the right arrow key, see (c), then press O, see (d), then press down arrow, see (e).

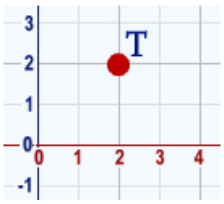




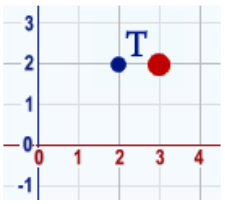
a



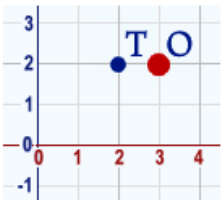
b



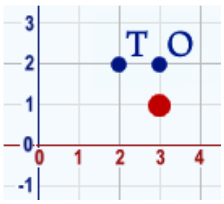
c



d



e



With this picture, the word TO can be alternatively represented as (2, 2), (3, 2). In the other way round, the list of coordinates (2, 2), (3, 2), (3, 2) can be read as TOO.

Module 6 • Investigation 2 • Activity 6.2.2
Busy Fleeeee and Clever Points

Learning Objectives

Explore how to draw a line that connects two points on the coordinates grid.
Explain how to reconnect the points when the position of one sprite is changed.

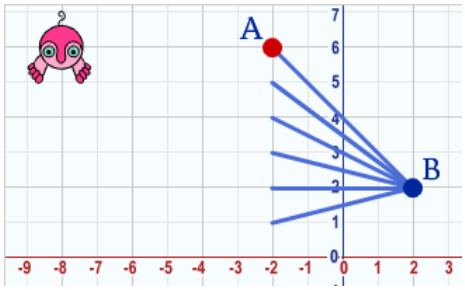
Activity Instructions

Pupils open project **62-Coordinates**, **Save as a copy** (online) or **Save as** (offline) and rename. The final version of this project at the end of Activity 6.2.2 will be **62-Coordinates INT1**.

1. Pupils explore the project, its sprites and their scripts. The **point A** has only one costume and can be moved within the grid by the arrow keys. The **point B** sprite has only its *setup script*. They have different colours to stress that they will behave differently.
2. Explore one small difference between **points A** and **B**: in the **developing mode** (which we use the most) both sprites can be dragged by mouse; **point A** can be moved by the arrow keys as well. However, in the **full screen mode** (also known as the **player mode:** to get there click the button in the upper right corner) **point A** cannot be dragged, while **point B** can (see Additional support and note the difference).
3. Although it may look like **Fleeeee** has not got much to do in this project, this is not the case: When asked, it will connect **points A** and **B** by a line and jump back to its 'home' position in the corner.
   Pupils build a script for **Fleeeee** to do that, debug it, then turn it into a definition of a new block **connect**.
4. Pupils build a **when *space* key pressed** script for **Fleeeee** that will **connect** the points. They use it, move **point A** by arrows and connect again. Pupils add the **erase all** block above the **connect** (**erase all** the stage before connecting by a new line). They explore the option with replacing **when *space* key pressed** by **when green flag clicked forever** … (**connect** the points again and again).
5. **[Extension]** Pupils use the **blink** and **nod** blocks (provided) and build the **forever** animations for **Fleeeee**: from time to time it will **blink** or it will **nod**.
6. **[Advanced Extension]** When **point B** is being dragged, we may or may not position it exactly into one of the grid points. Now pupils build a **forever** script which will make **point B** **'snap' to the nearest grid point**.

Mathematics Connections

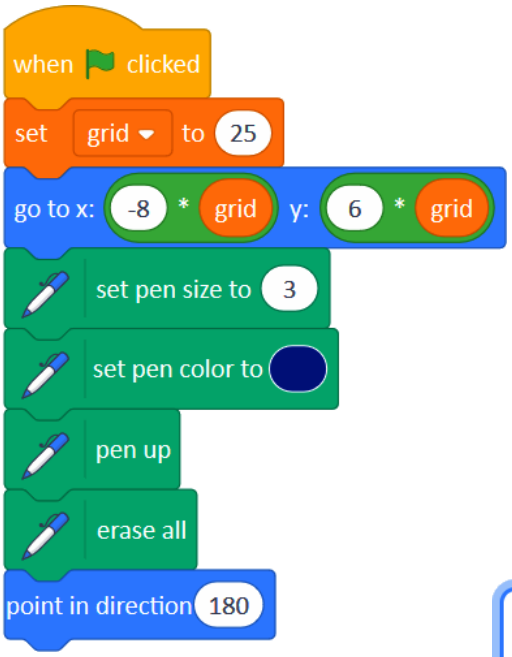
Discuss: What are the initial Scratch coordinates of each sprite? What are their grid coordinates? How do these correspond?

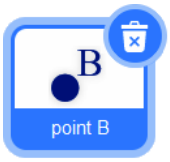


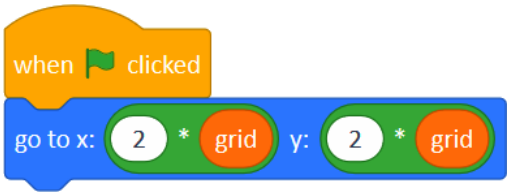
The same behaviour may then be copied to **point A** – which can be both dragged or moved by arrows in the developing mode.

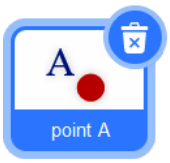
ADDITIONAL SUPPORT

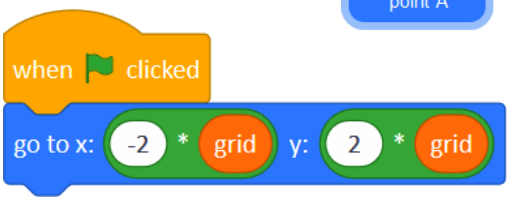
1

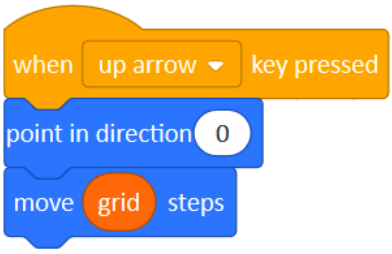


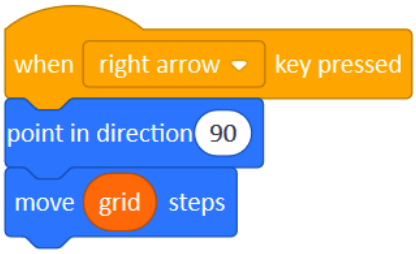






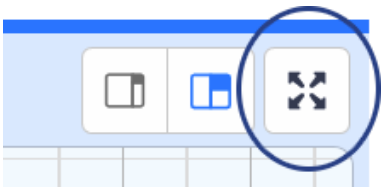


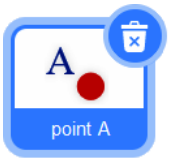


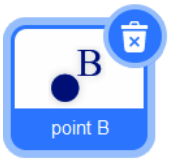


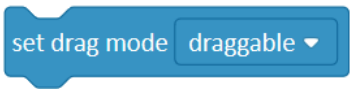
2

Full screen mode (or player mode) on/off.





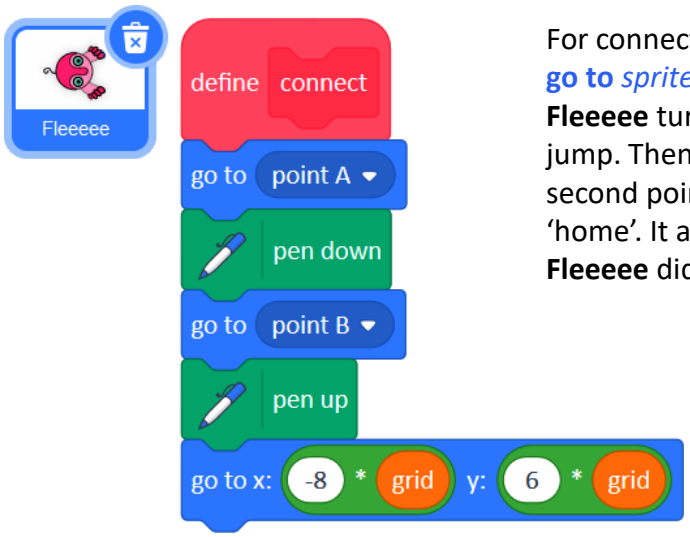




continued ▶

ADDITIONAL SUPPORT CONTINUED

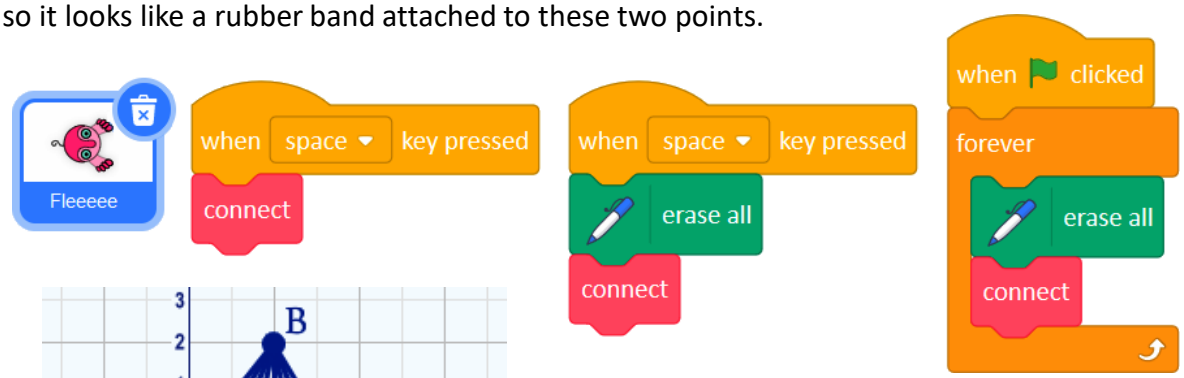
3

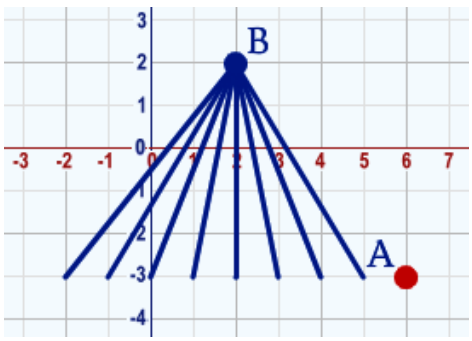


For connecting sprites by a line we will use the **go to** *sprite* block for 'jumping'. Note that **Fleeeee** turns its **pen down** only after its first jump. Then – with its **pen down** – jumps to the second point, turns its **pen up** and jumps back 'home'. It all happens so quickly that it looks as if **Fleeeee** did not move at all.

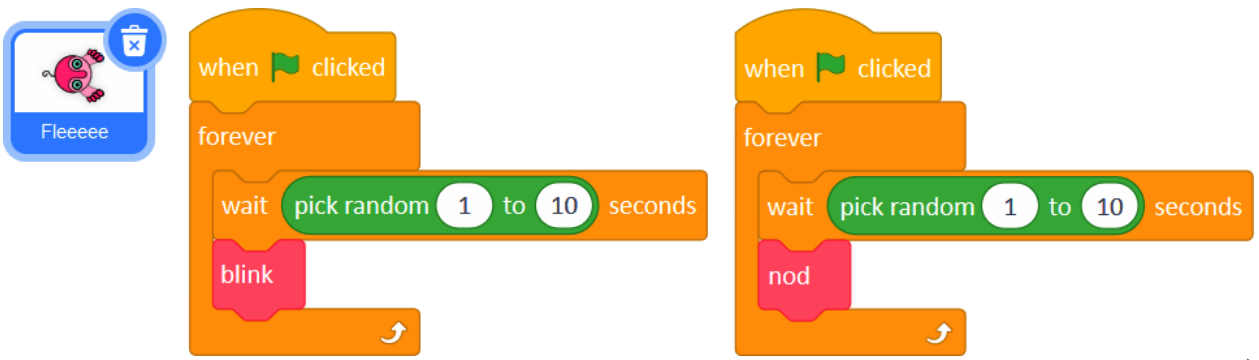
4

In (a) below **Fleeeee** connects the points A and B whenever we press the space key. As there is no **srase all** block, the previous lines stay on the stage as well. In (b) the previous line is first cleared away. In (c) the line between A and B is 'updated' again and again so it looks like a rubber band attached to these two points.



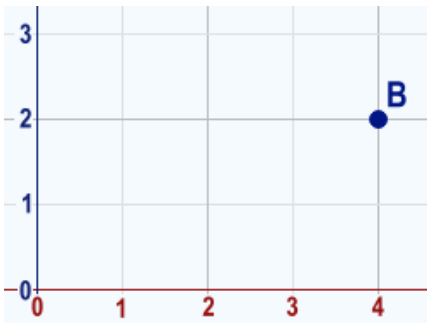


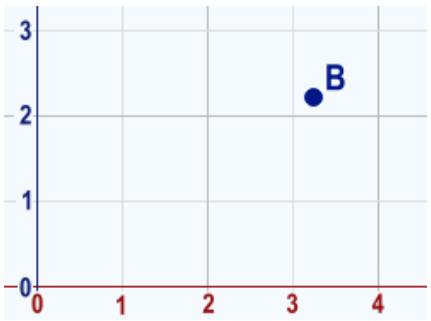
5 Each of the following **forever** scripts **waits** a random number of seconds (in our solution, between 1 and 10) and runs the animation.



continued ▶

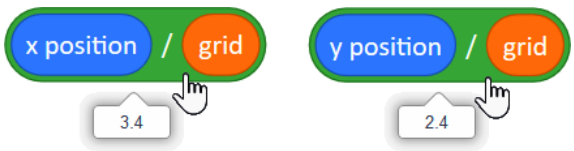
6





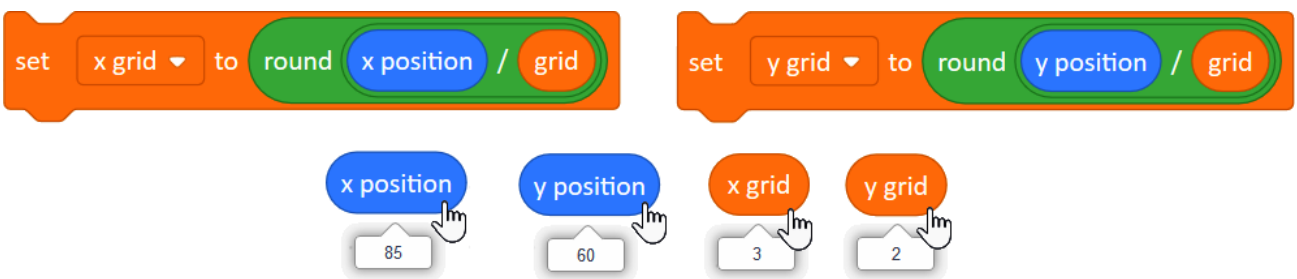
From Activity 6.1.3 we already know that for any **grid point** represented in the Scratch coordinates by **x position** and **y position** we can easily find the grid coordinates by dividing both numbers by the grid size:

However, if **B** would not sit exactly in a grid point, like (85, 60), these numbers would not be integers:

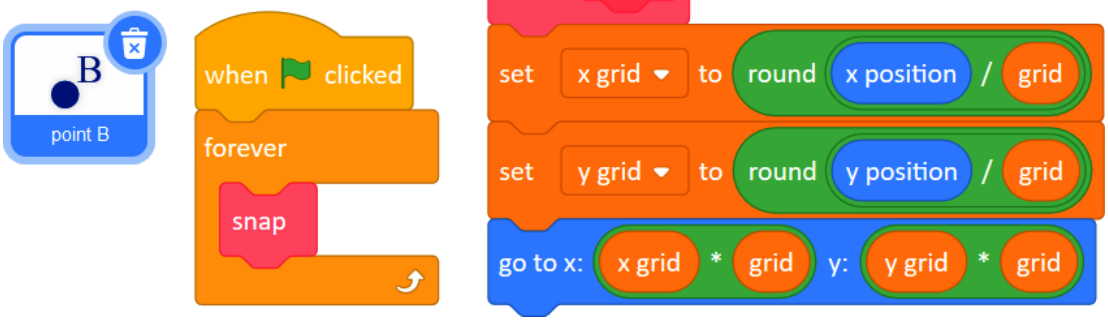


To turn these numbers into integers corresponding to grid points, we have to round them:

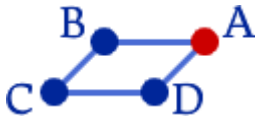
So, if **B** sits at (85, 60), the closest grid point is (3, 2) – see the picture top right. Thus we want **B** to snap to that point. To make the **snap** block more readable, let us give clear names to these two values – **x grid** and **y grid**, the grid coordinates of **B**.



To make **point B** 'snap' to that grid point we must multiply **x grid** and **y grid** by the grid size – to turn those grid coordinates back into standard Scratch coordinates:



MODULE 6 • INVESTIGATION 2 • ACTIVITY 6.2.3
Tricky Triangles



LEARNING OBJECTIVES

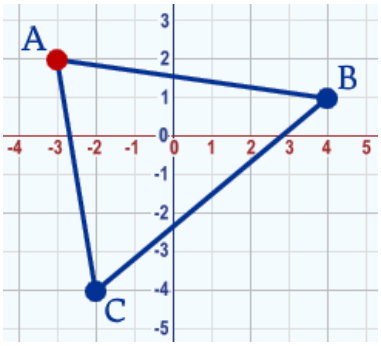
Explore and **explain** the coordinates of the vertices of different types of triangles
Exchange problems which involve triangles and coordinates.

ACTIVITY INSTRUCTIONS

MATHEMATICS CONNECTIONS

Pupils continue in their own version of project **62-Coordinates**, or open the **62-Coordinates INT1**, **Save as a copy** (online) or **Save as** (offline) and rename. The final version of this project at the end of Activity 6.2.4 is **62-Coordinates INT2**.

1. Pupils explore the costumes of the **point B** sprite.
2. Pupils duplicate the **point B** sprite, switch its costume to costume named *C* and rename the new sprite to **point C** – so that it will become our third point. They change the initial grid coordinates in this new sprite's *setup script*.
3. Pupils extent the **connect** script of the **Fleeeee** so that it connects all three points, thus drawing a triangle.
4. **[Optional]** Pupils **switch backdrop to** *grid 10* and modify the scripts so that all sprites react correspondingly.



Suggested Tasks

Interesting Isosceles *Whole class with teacher*

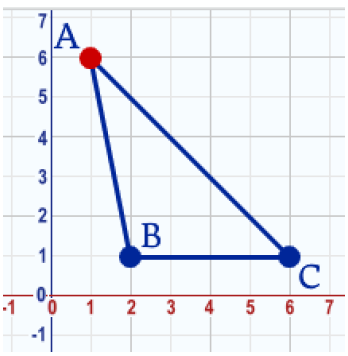
You can use the project to **ask** open questions to the whole class which encourage pupils to develop their envisaging and geometrical reasoning skills.

For example: set B and C to be the vertices of a horizontal edge of a triangle.

Ask what are the coordinates of A to create an isosceles triangle?

Is there another solution, another?

Can you explain your answer?



Tricky Triangles

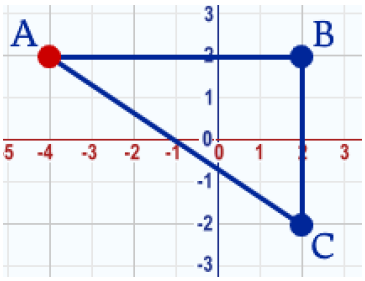
ACTIVITY INSTRUCTIONS

Types of triangles *In pairs*

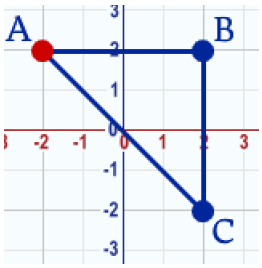
Move A, B, C to create an example of different types of triangles.

Save a picture of the stage for each example. (Right click the stage). Exchange your examples with another pupil, compare your images. What is the same, what is different?

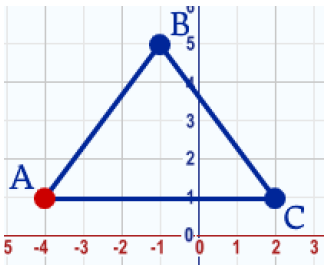
One solution



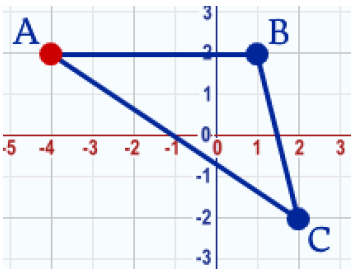
Right-angled triangle



Right-angled isosceles triangle



Isosceles triangle



Scalene triangle

Encourage pupils to **explain** why diagonal edges are equal in length. Pupils might need to be prompted to count how many squares up and across describe the diagonal edge. E.g. in the isosceles example above, we can see that the edge AB is 3 squares across and 4 up, this is the same as edge BC which is 3 squares across and 4 down.

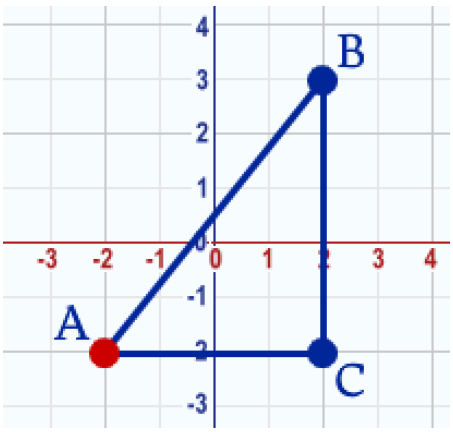
Note: It is impossible to create an equilateral triangle using whole number coordinates, the proof of this is degree level mathematics! Can you turn the triangle into three different isosceles triangles?

Tricky Triangles

ACTIVITY INSTRUCTIONS

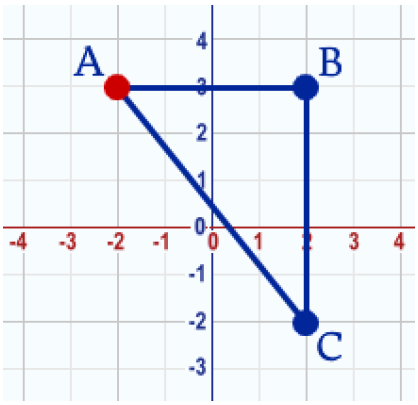
A Goes Up, A Comes Down *Individually*

Set up the investigation by dragging point B to (2, 3), C to (2, -2) and A to (-2, -2). Ask pupils the questions below.



Q1. What type of triangle have you made?

A1. A right-angled triangle



A is at (-2, 3)

*Q2. Using **only the up and down arrows** (e.g. the x-coordinate is always -2).*

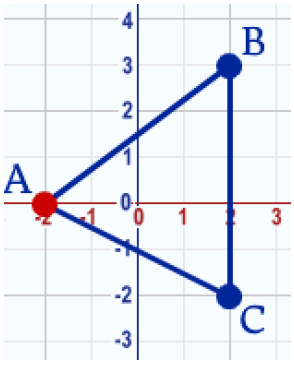
can you:

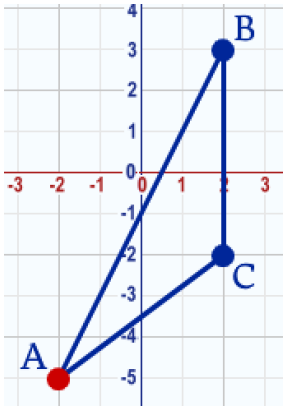
- make a different right-angle triangle?

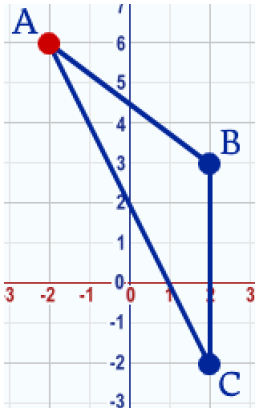
Extension

Can you:

- turn the triangle into **three** different isosceles triangles?
- explain how you know the triangles are isosceles?







A can be at positions (-2, 0), (-2, -5) and (-2, 6)

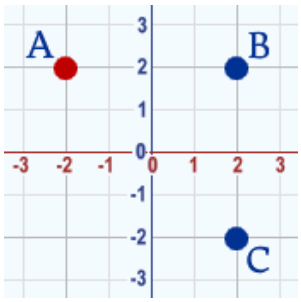
The explanation is based upon the special triangle which has sides which measure 3, 4 and 5. Pythagoras is beyond KS2 but pupils can justify by measuring on square cm paper and is an interesting phenomenon!

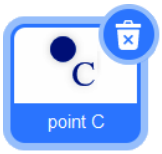
ADDITIONAL SUPPORT

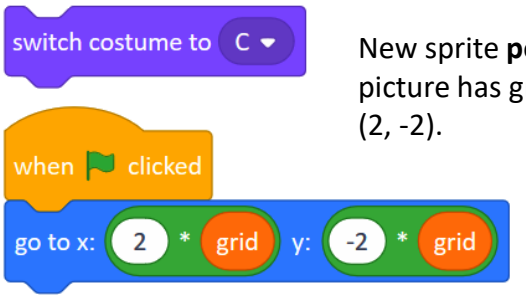
1

Point B has five costumes for more points to be added later. (You may want to change the position of the letters in the costumes and backdrops editor. However, keep the centre of each costume in the centre of the blue circle.)

2



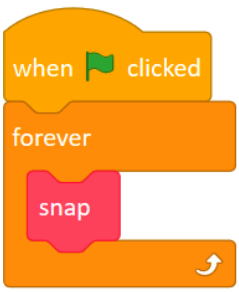


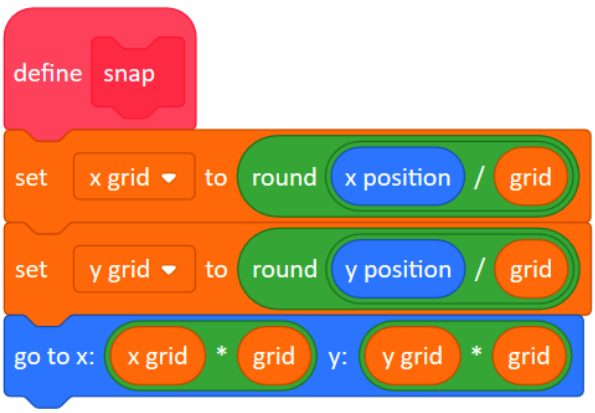


New sprite **point C** in this picture has grid coordinates (2, -2).

Note that:

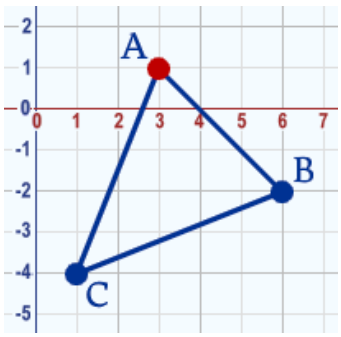
- You may either insert **switch costume to** *C* into the **point C** sprite's *setup script* or just run it once then delete it.
- When duplicating **point B**, all scripts are duplicated as well. Thus, if you have built the snapping behaviour for **point B**, new sprite **point C** will have it as well.





3

While in the previous activity **Fleeeee** connected only two points – by a line, now it has to make the 'round tour' from **point A** to **point B**, to **point C** and back to **point A**, then get back to its "home" position.



MODULE 6 • INVESTIGATION 2 • ACTIVITY 6.2.4
Quirky Quadrilaterals

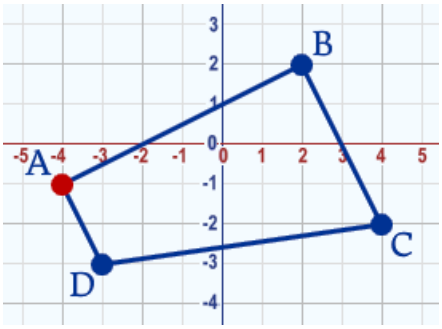
LEARNING OBJECTIVES

Explore and **explain** the coordinates of the vertices of different types of quadrilaterals.
Exchange problems which involve quadrilaterals and coordinates.

ACTIVITY INSTRUCTIONS

Pupils continue in their own version of project **62-Coordinates**, or open the **62-Coordinates INT2**, **Save as a copy** (online) or **Save as** (offline) and rename. The final version of this project at the end of Activity 6.2.4 is **62-Coordinates FINAL**.

1. Pupils duplicate the **point C** sprite, switch its costume to *D* and rename the new sprite to **point D**. They change the initial grid coordinates in the new sprite's *setup script*.
2. Pupils extend the **connect** script of the **Fleeeee** so that it connects all four points to draw a quadrilateral.
3. **[Extension]** Pupils **switch backdrop to** *grid 10* and modify the scripts so that all sprites react correspondingly.



MATHEMATICS CONNECTIONS

Suggested Tasks

Coordinate Challenge *Whole class with teacher*

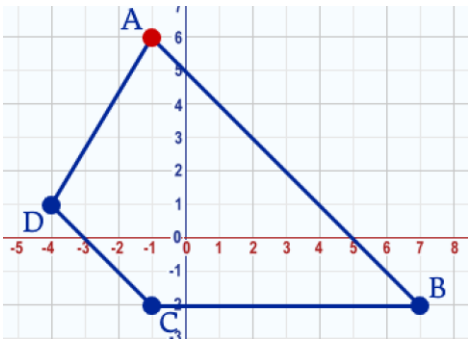
You can use the project to **ask** open and closed questions to the whole class which encourage pupils to develop their envisaging and geometrical reasoning skills, as in the previous activity. There are many examples of this type of question in previous KS2 testing materials freely available online.

Example 1

Set B and C to be the vertices of an edge of a parallelogram (parallel to the x-axis), and D to another vertex as in the image.

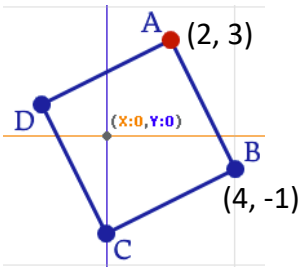
Ask what are the coordinates of A to create a parallelogram?

Can you explain your answer? How do you know the sides are parallel?



Example 2

A square with vertices (ABCD) has coordinates A (2, 3) and B (4, -1). What are the coordinates of C and D?



Quirky Quadrilaterals

ACTIVITY INSTRUCTIONS

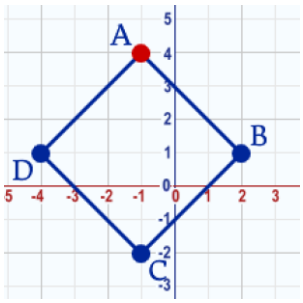
Types of Quadrilaterals *In pairs*

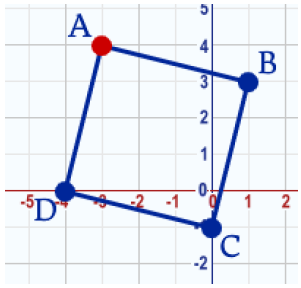
Move A, B, C, D to create an example of each of the different types of quadrilaterals.

Save a picture of the stage for each example. (Right click the stage). Exchange your examples with another pupil, compare your images. What's the same, what is different?

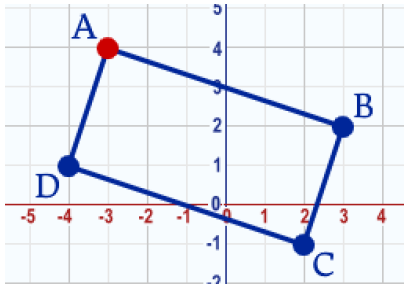
One solution

These are not all so *obvious*, encourage and challenge**!**

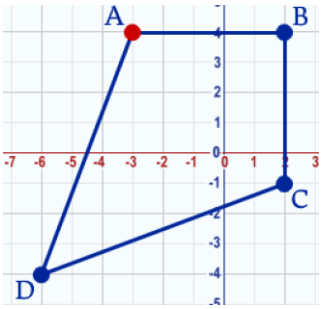




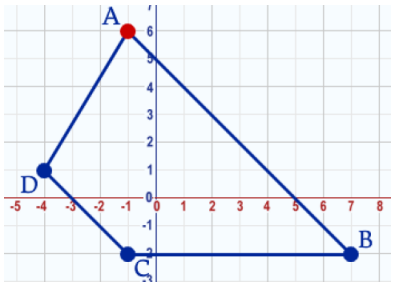
Squares



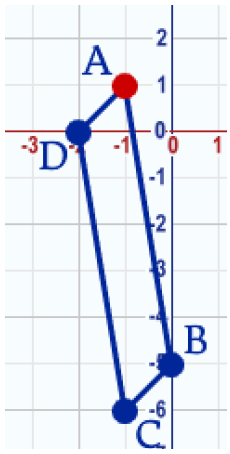
Rectangle



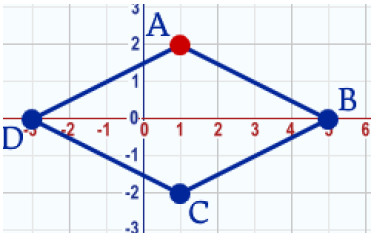
Kite



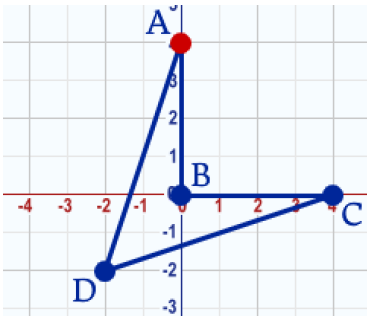
Trapezium



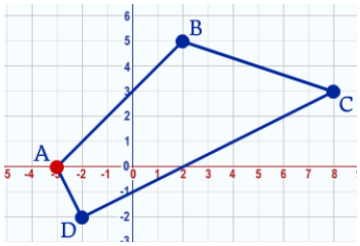
Parallelogram



Rhombus



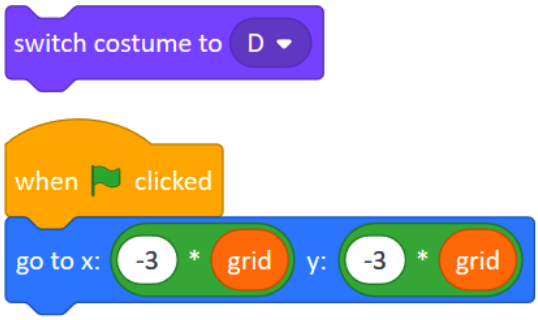
Arrowhead concave quadrilateral

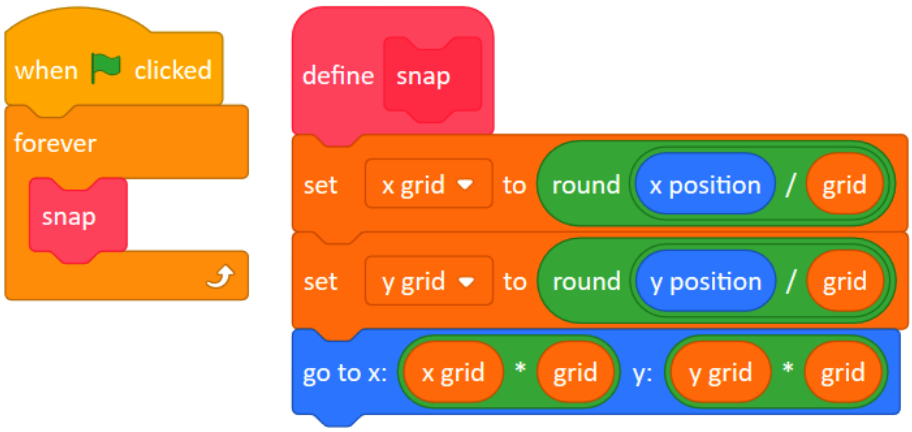


Convex quadrilateral

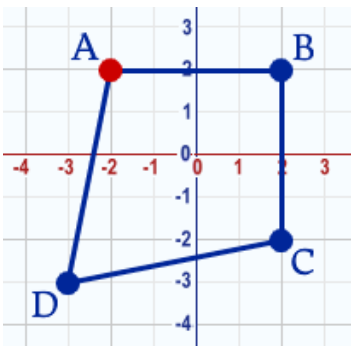
ADDITIONAL SUPPORT

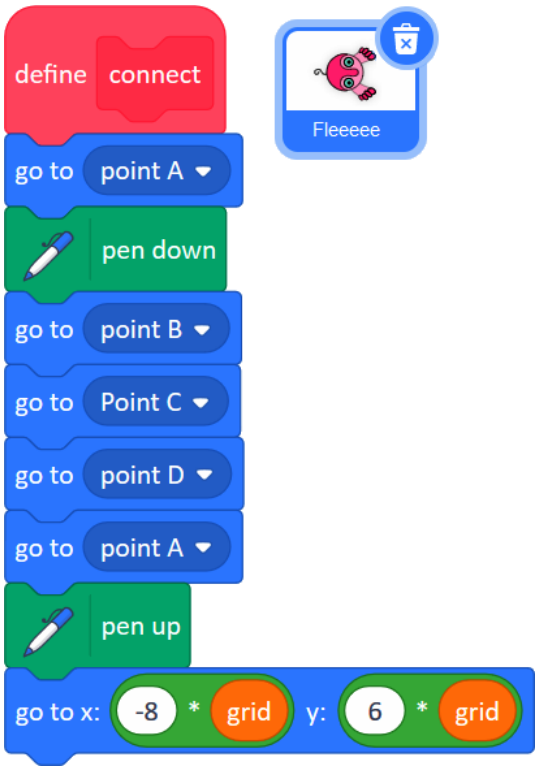
1





2



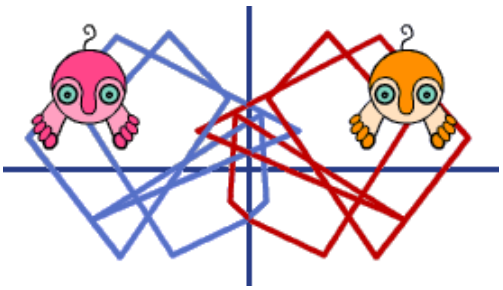


MODULE 6: INVESTIGATION 3

Transformations

This investigation develops pupils' understanding of the effect of reflection and translation on coordinate points in a very tactile and practical way. Pupils explore what happens to a sprite's position as they control the movement of one sprite and another sprite mimics its behaviour by following a rule. For example in the picture below, as the Fleeee sprite (pink) is dragged around the stage, the Meeeee sprite (orange) will mimic the Fleeee sprite's position **reflected** in the y axis.

Pupils should continue to explore and explain the effect of different transformations on the position (the coordinates) of an object after transformation.



- **Activity 6.3.1** – Mimic Meeeee
- **Activity 6.3.2** – Shadows, Translations and Reflections
- **Activity 6.3.3** – Through the Looking Glass

Scratch projects	**63-Mimic Meeeee**	**63-Looking Glass**
	63-Mimic Meeeee INT	**63-Looking Glass FINAL**
	63- Mimic Meeeee FINAL	

LINKS TO PRIMARY NATIONAL CURRICULUM

CURRICULUM OBJECTIVES	LINK WITH SCRATCHMATHS
Mathematics Describe positions on the full coordinates grid.	▶ Pupils are required to draw and then translate/reflect patterns and shapes in all four quadrants.
Solve problems involving similar shapes where the scale factor is known or can be found.	▶ **[Extension]** Pupils have the opportunity within an extension activity to explore different scale factors in their translated drawings.
Identify, describe and represent the position of a shape following a reflection or translation.	▶ Pupils are required to explore translations of patterns and shapes through using a second sprite that mimics the behaviour of the first.
Describe movements between positions as translations of a given unit to the left/right and up/down.	
Compare and classify geometric shapes. Plot specific points and draw sides to complete a given polygon.	▶ Pupils are required to revisit activities from Y5 involving drawing polygon pictures (such as houses made from squares and triangles), but to extend these to also incorporate a reflected version of the drawing.

Module 6 • Investigation 3 • Activity 6.3.1
Mimic Meeeee

Learning Objectives

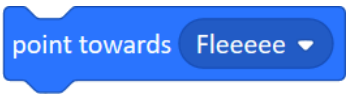
Explore how to make one sprite mimic the behaviour of another sprite.
Explain how this mimicking behaviour works.

Activity Instructions

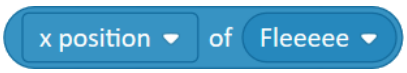
Pupils open project **63-Mimic Meeeee**, **Save as a copy** (online) or **Save as** (offline) and rename. The final version of this project at the end of Activity 6.3.1 will be **63-Mimic Meeeee INT**.

1. Pupils explore the project, its sprites, their costumes and scripts. While **Meeeee** has only a simple setup script, **Fleeeee** also has two animation blocks **nod** and **blink** and two **forever** scripts to use them – pupils are familiar with these from Investigation 1.

2. **Fleeeee** is a 'hero' for **Meeeee** – it wants to be just like **Fleeeee**, it wants to observe **Fleeeee** and learn from it. Pupils will help **Meeeee** by building a **when green flag clicked** script, which will make **Meeeee** **forever** look at **Fleeeee**, using:



3. Pupils modify this 'spying' script of **Meeeee**: it will always **say** the current x position of **Fleeeee**, using:



4. Pupils make **Meeeee** imitate **Fleeeee** by another script which will **forever** switch its costume to the current *costume #* **of** *Fleeeee*.

5. Pupils replace **say** in the script by **set x to …** block so that when they drag **Fleeeee**, **Meeeee** will mimic its actual x position.

6. Pupils replace **set x to …** and *x position* **of** *Fleeeee* by **set y to …** and *y position* **of** *Fleeeee*.

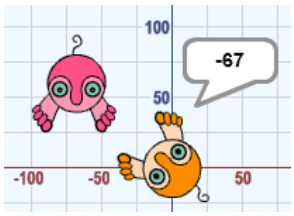
7. Pupils change the initial position of **Fleeeee** to be **0, 0** , set **Meeeee**'s **pen down** and use **go to x: … y: …** for x and y positions of **Fleeeee** instead of mimicking just one of *x position* or *y position* **of** *Fleeeee*. [Remove the **point towards** block and observe the effect.]

   Pupils drag **Fleeeee** around the stage, and **Meeeee** will mimic its position and draw with its pen.

Mathematics Connections

There are two backdrops in the project – *grid 50* or *axes*. Choose whichever works best for your pupils for these activities.

Drag **Fleeeee** around the stage. **Discuss:** What is happening with **Meeeee**? Why does **Fleeeee** not leave a line when we drag it?

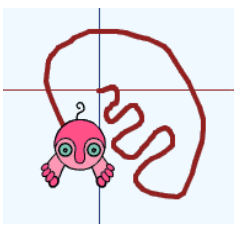


Drag **Fleeeee** after adding the **set x to** block. What do you notice? [Meeeee's y coordinate doesn't change.]

Replace the **set x** block with **set y.** What do you notice when you drag **Fleeeee** [Meeeee's x coordinate doesn't change.]

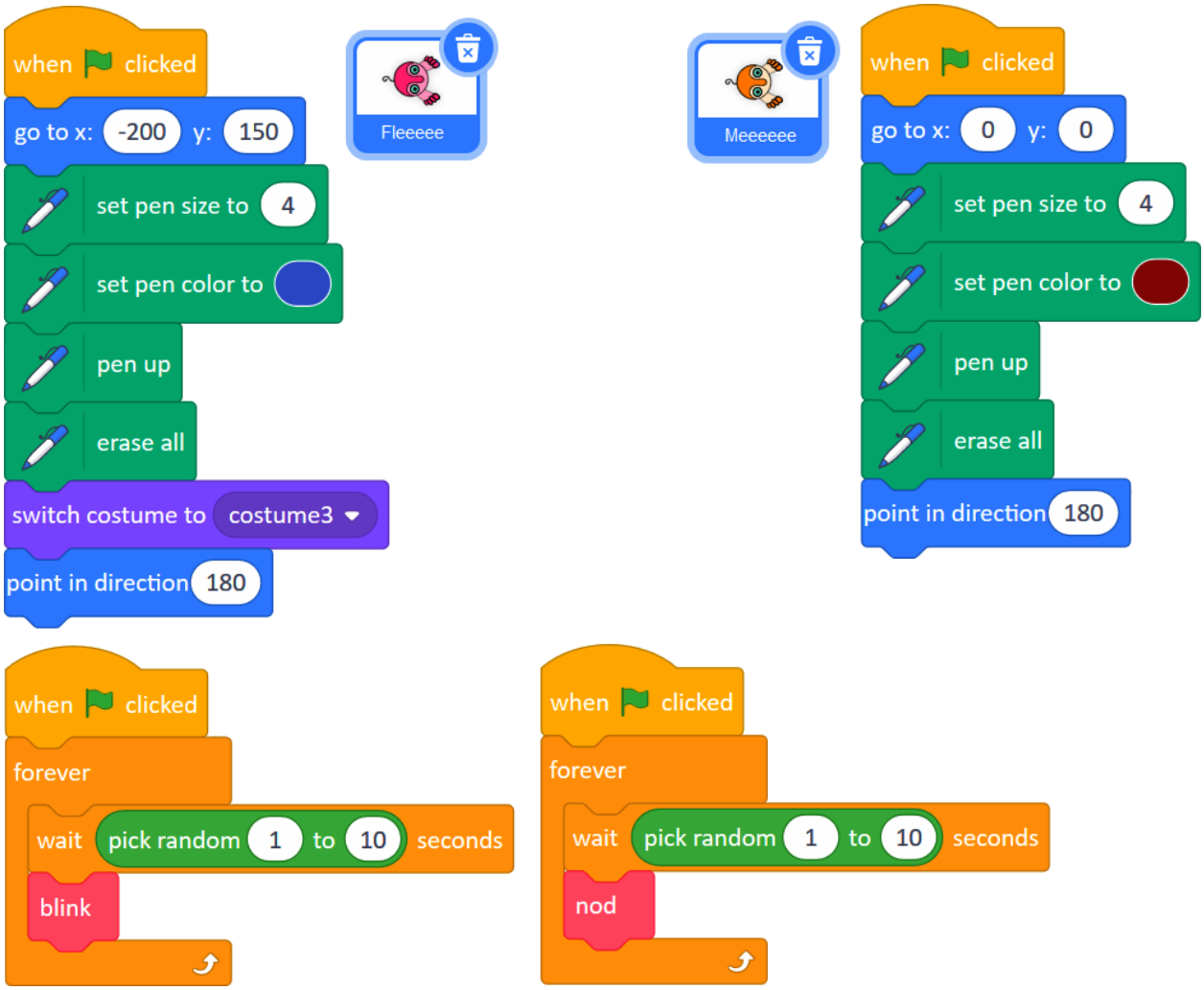
Now when we click green flag and start dragging **Fleeeee**, it seems to be drawing a line.

Discuss: Why? Who in fact draws the line? Where is **Meeeee**?



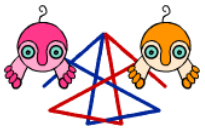
ADDITIONAL SUPPORT

1 *Setup scripts* of both sprites are below. Fleeeee also has two definitions of **blink** and **nod** (not shown here) and two additional **forever** loops to periodically run the animations.

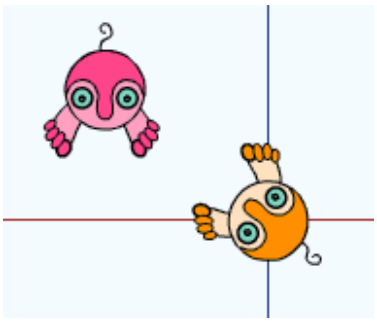


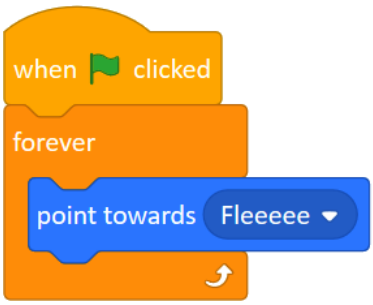
2 Here we are using a simple **point towards …** **Motion** block, which has a small drop down menu in it. We select the name of **Fleeeee** and add **when green flag clicked** hat block with **forever**, so that **Meeeee** points towards **Fleeeee** wherever it moves – or is being dragged. To see the result, drag **Fleeeee** around the stage.

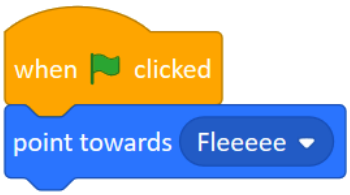
continued ▶



Compare the two possible solutions, see below. Make sure pupils understand why the solution on the right would not work correctly *[It is because **Meeeee** would point – i.e. look towards **Fleeeee** only once when the green flag is pressed, instead of turning and pointing towards **Fleeeee** wherever it moves or is dragged.]*

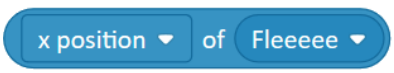


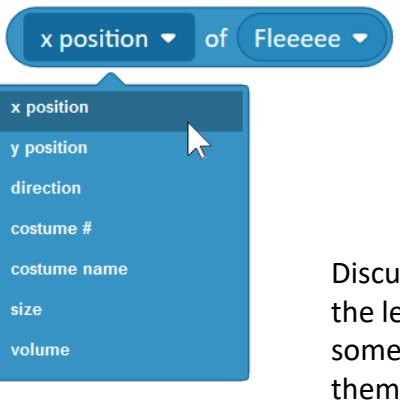


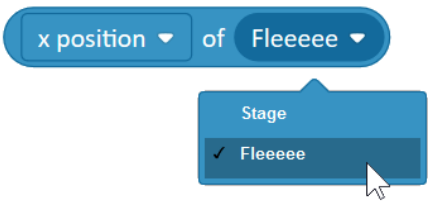


3 For "spying" i.e. learning about some settings of another sprite we use a highly general and powerful **reporter block** from the **Sensing** group. It is one of very few Scratch blocks which has two drop down menus – both on the left side and on the right side. We have already used it in extension activity 4.3.4.

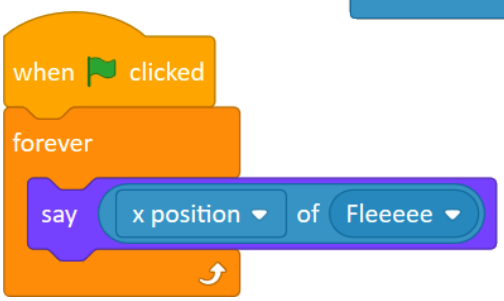
Note that if you opened the right drop down menu with **Fleeeee** being selected in the sprites area, the list of options would contain Stage and all other sprites (**Fleeeee** in our current situation).







Discuss together the list of settings from the left drop down menu – to explore some other items here as well. One of them will be used in the next step.

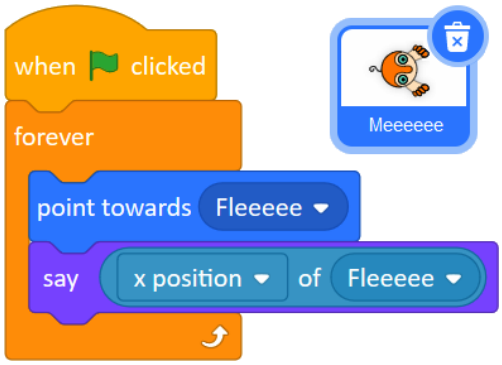


Encourage pupils to explore and explain why (as in previous step) this script would not work properly without the **forever** block. To demonstrate this, they drag **Fleeeee** around the stage.

continued ▶

ADDITIONAL SUPPORT CONTINUED

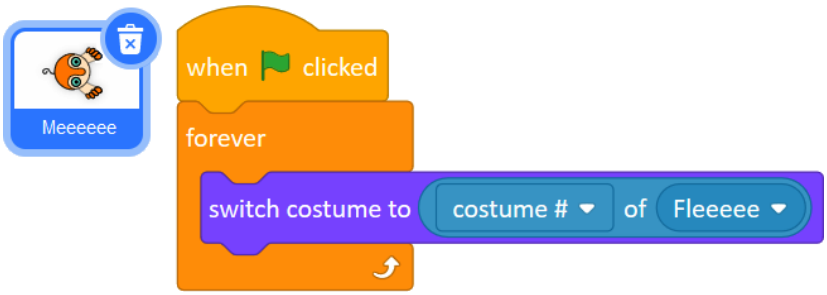
An alternative solution is to keep the **point towards** *Fleeeee* block and add the **say …** block inside the same **forever**, see right.



You may want to delete both of these **point towards** *Fleeeee* and **say …** scripts later in the activity as the **say …** bubble could become distracting.

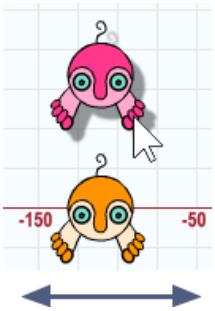
To test this mimic behaviour of **Meeeee**, drag **Fleeeee** along the stage. Why does it not leave a line – even if we turn its **pen down**? It is because dragging a sprite is a **direct manipulation operation**, not **computational – programmed operation** like making it **move … steps**.

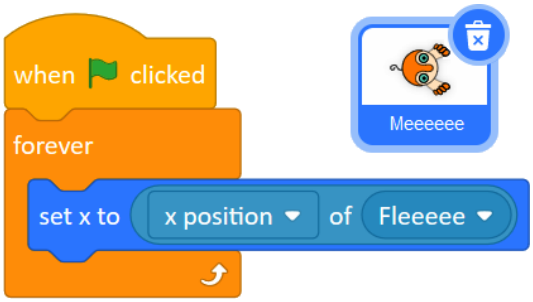
4



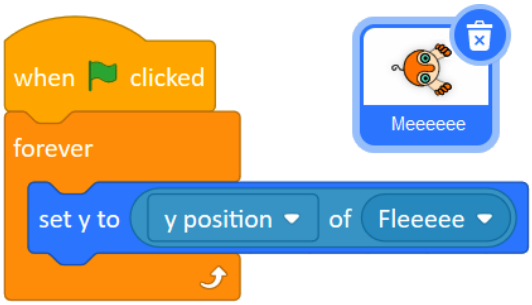
Note that if we run this "spying" script, it will look as if **Meeeee** knew how to **blink** and **nod** – but in fact it does not, it only mimics the costumes of **Fleeeee** all the time.

5





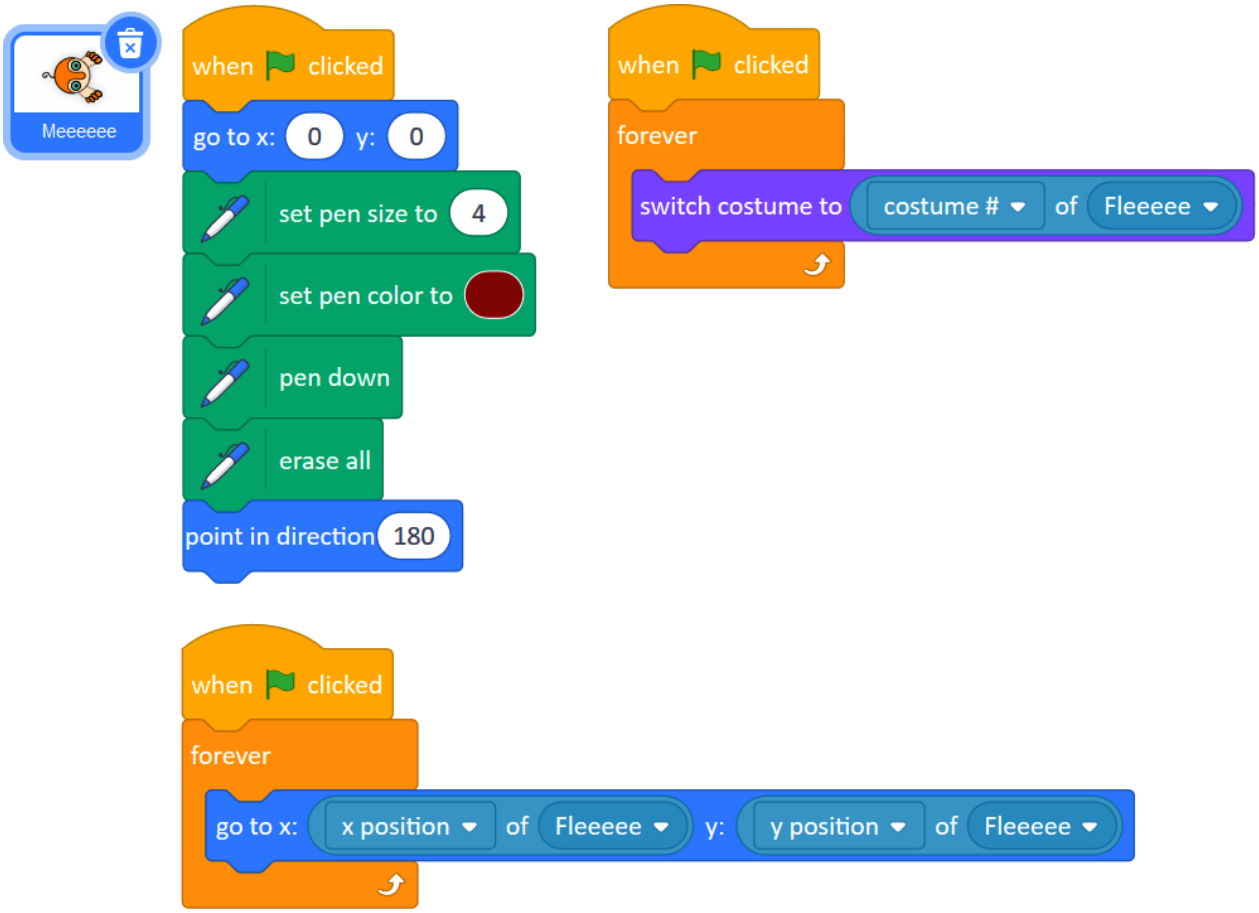
6



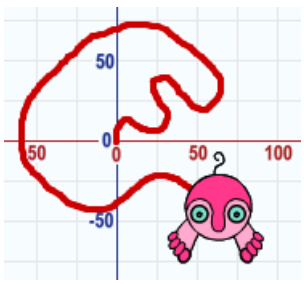
continued ▶

ADDITIONAL SUPPORT CONTINUED

7



Do not forget to modify the initial position of Fleeeee to be go to x: 0 y: 0.



An alternative solution (and for this particular task even simpler) would be to use the **go to** *Fleeeee* **Move** block, see below. However, in the following activities we will need the solution with **go to x: … y: …** block as we will insert different expressions in it.

MODULE 6 • INVESTIGATION 3 • ACTIVITY 6.3.2
Shadows, Translations and Reflections

LEARNING OBJECTIVES

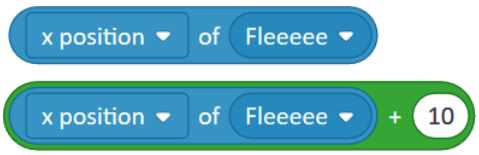
Explore how to make one sprite translate the drawing behaviour of another sprite.
Explain how to restrict the movement of a sprite to a single quadrant of the grid.

ACTIVITY INSTRUCTIONS

Pupils continue in their own version of project **63-Mimic Meeeee**, or open the **63-Mimic Meeeee INT**, **Save as a copy** (online) or **Save as** (offline) and rename. The final version of this project at the end of Activity 6.3.2 is **63-Mimic Meeeee FINAL**.

So far, we have been dragging **Fleeeee** around the stage. In this activity we will build a script for it to **glide. Meeeee** will still mimic **Fleeeee**, however the mimicking effect will be considerably improved when clicked.

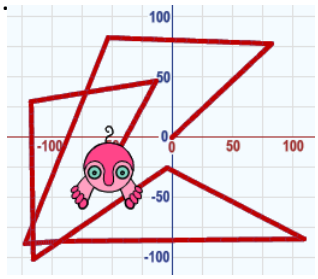
1. Pupils build an isolated **glide** block for **Fleeeee** – to fly at random within the whole stage or on the same side of the y axis. They click the green flag so that **Meeeee** is mimicking **Fleeeee.** They add a **repeat 10** around the **glide** and also a **when this sprite clicked** hat block.
2. Pupils replace **pen up** in the *setup script* of **Fleeeee** with **pen down**. Note: **Meeeee** still draws red lines. To see both red and blue lines, pupils modify the mimicking script of **Meeeee** by moving the its position 10 pixels horizontally e.g.



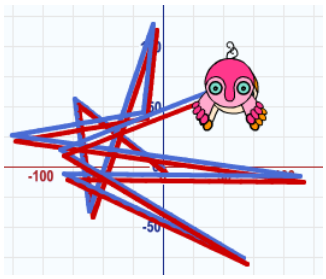
3. Pupils use an **Operator** block to change the *y position* **of** *Fleeeee* and explore using small values of translation (e.g. between – 10 to + 10). Use different pen sizes and pen colours of **Fleeeee** and **Meeeee** to create a "line with a shadow" effect.
4. Pupils restrict **Fleeeee**'s random gliding within the upper left quadrant and make **Meeeee** draw identical **glide** doodles translated (a) **Right 200**, (b) **Down 150**, (c) **Right 50 Down 50** or similar.
5. Pupils replace the operators with multiply x or y position (or both) of **Fleeeee** by -1 – creating **reflected doodles** in the x or y axis or both.
6. **[Extension]** Pupils multiply x or y position (or both) by 0.5 or -0.5 and explore the outcome doodle.

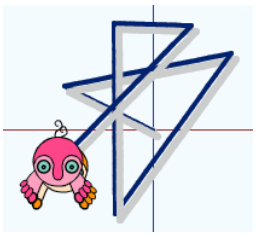
MATHEMATICS CONNECTIONS

Use either *grid 50* backdrop or *axes*, whichever works best for your pupils.



Ask: When **Fleeeee** has its **pen down** why are all the lines red? [The blue lines of **Fleeeee** are immediately covered by red lines of **Meeeee.**]

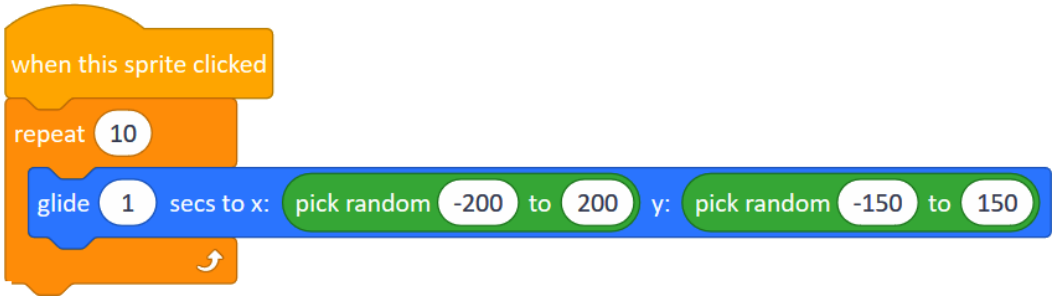




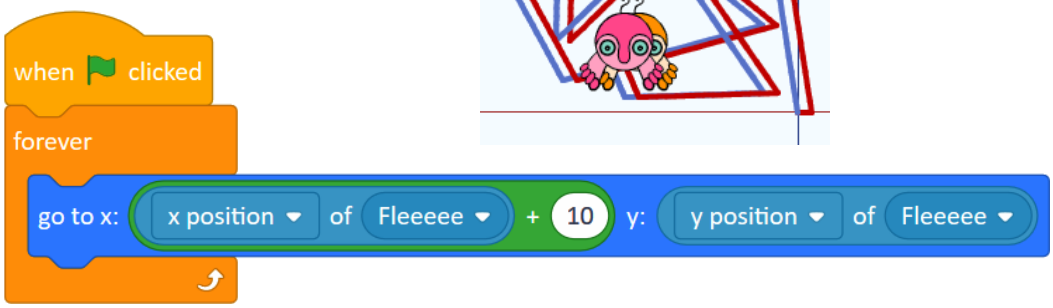
ADDITIONAL SUPPORT

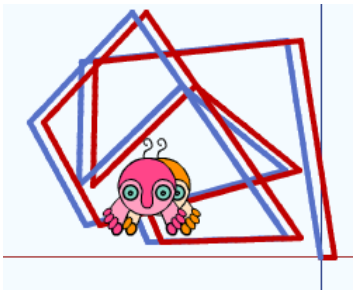
1

Pupils may experiment with different times in the **glide** block.



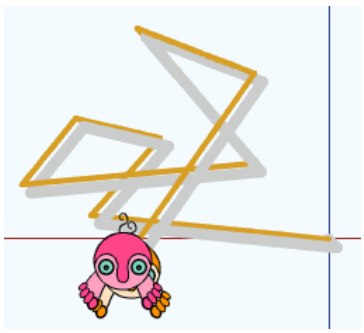
2

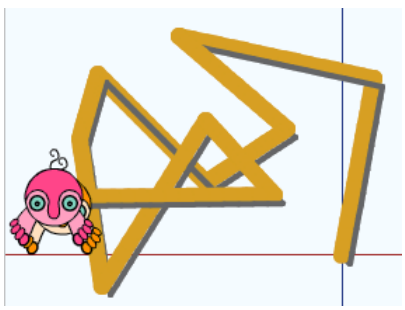




3

Adding positive number to *x position* **of** *Fleeeee* means translating **Meeeee**'s drawing to the right or horizontally. Subtracting a number from *y position* **of** *Fleeeee* means translating **Meeeee**'s drawing downwards or vertically.

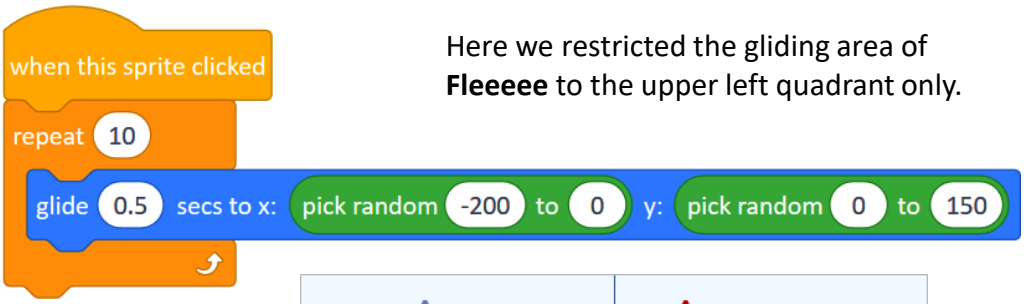




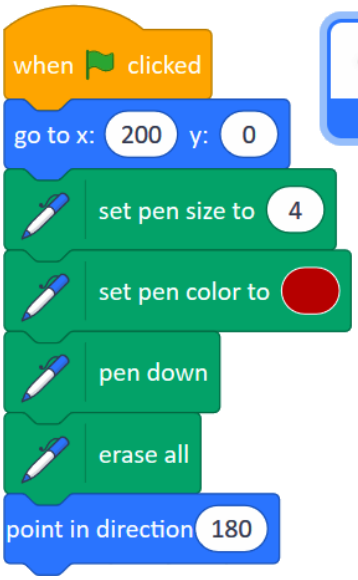
continued ▶

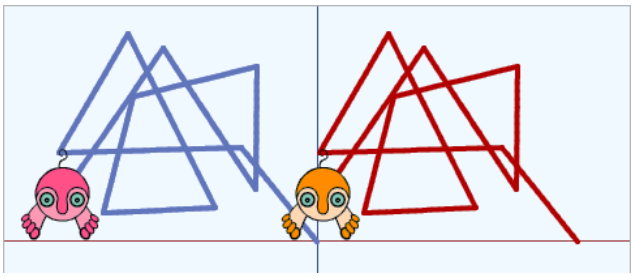
ADDITIONAL SUPPORT CONTINUED

4

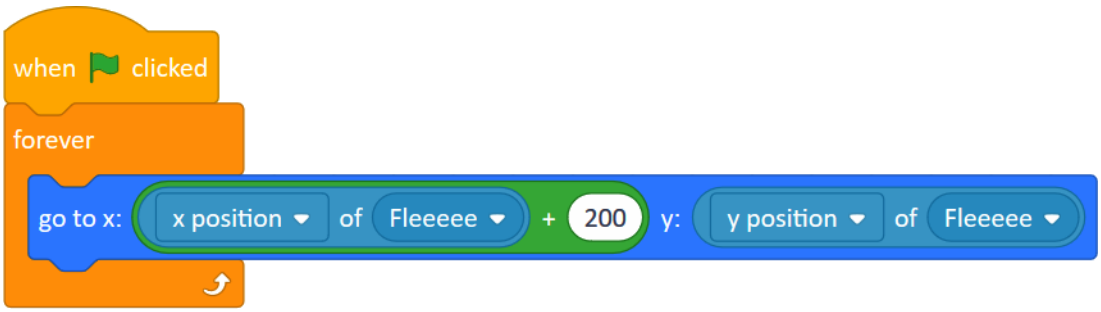


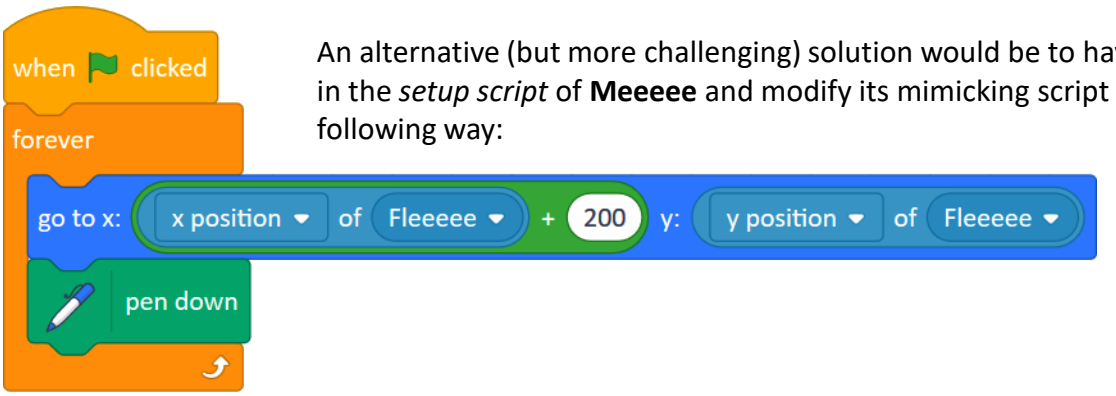
Here we restricted the gliding area of **Fleeeee** to the upper left quadrant only.





Here, for the translation **Right 200**, we modified the initial position of **Meeeee** to also be a translated initial position of **Fleeeee**. Otherwise, there would be a line from **Meeeee** connecting its initial position **0, 0** to the first translated point at **200, 0**.



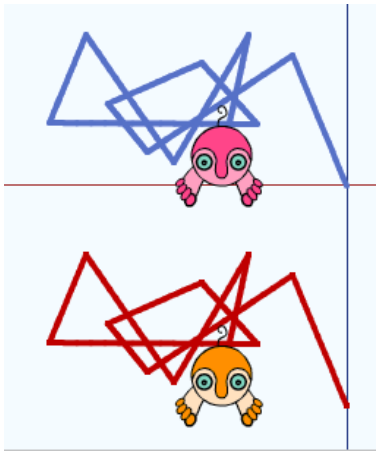


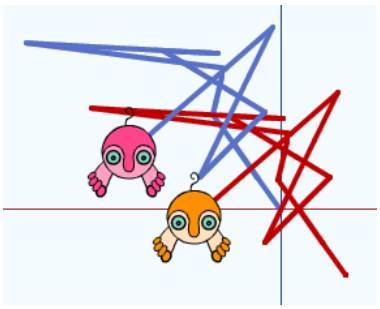
An alternative (but more challenging) solution would be to have **pen up** in the *setup script* of **Meeeee** and modify its mimicking script in the following way:

continued ▶

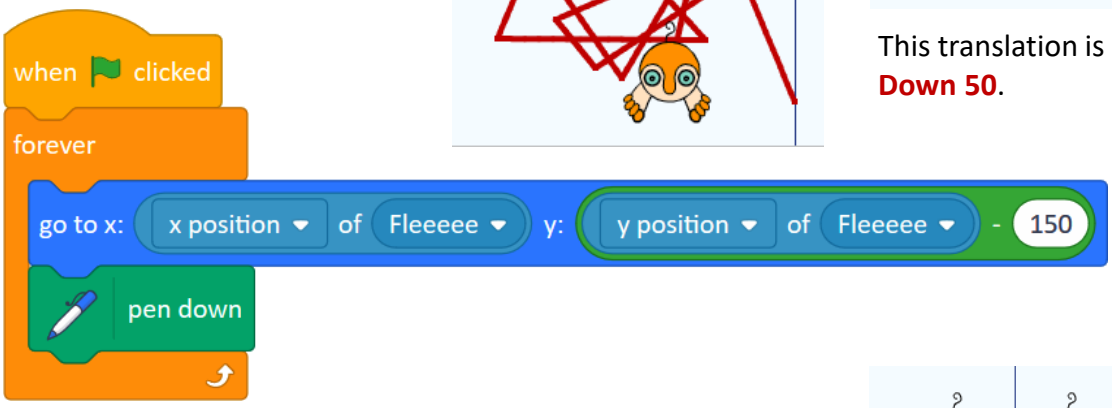
ADDITIONAL SUPPORT CONTINUED

This translation is **Down 150**.

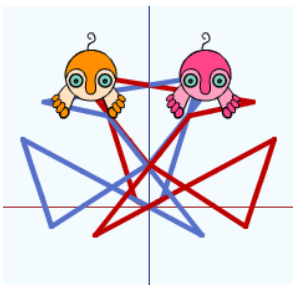


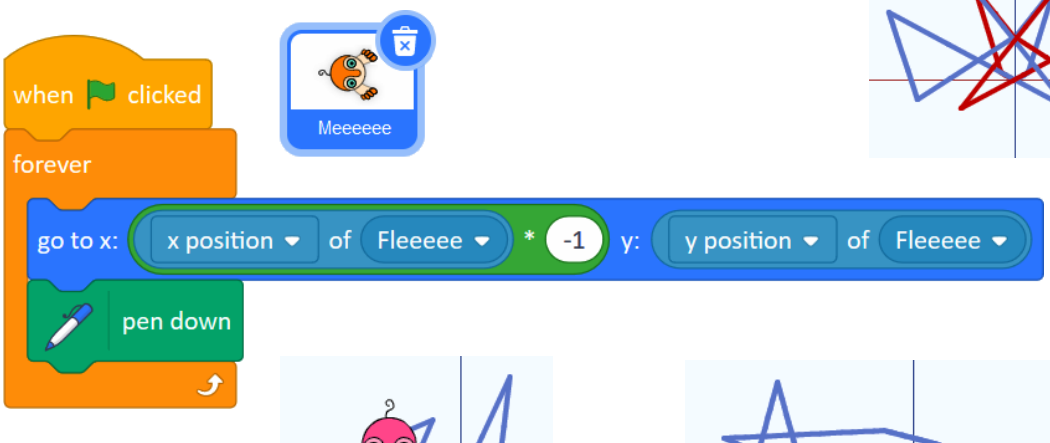


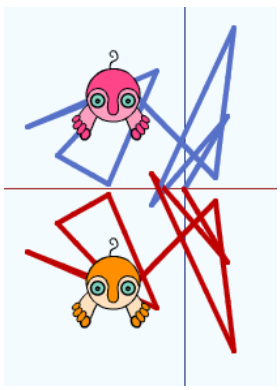
This translation is **Right 50 Down 50**.

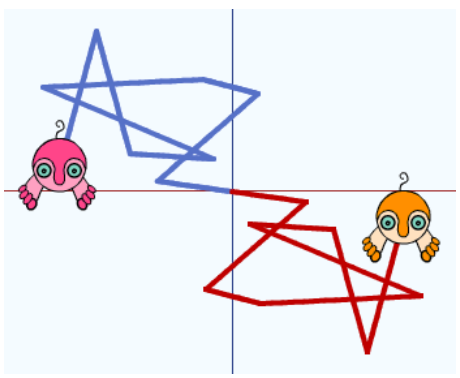


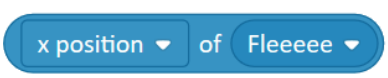
5

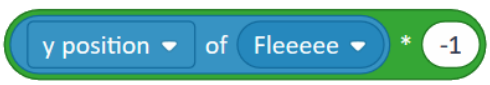


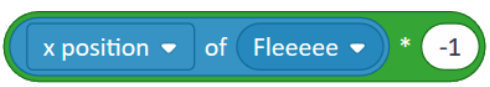


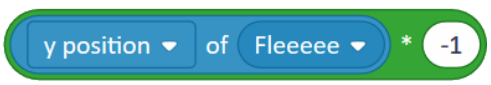




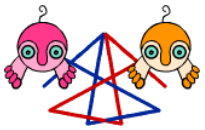






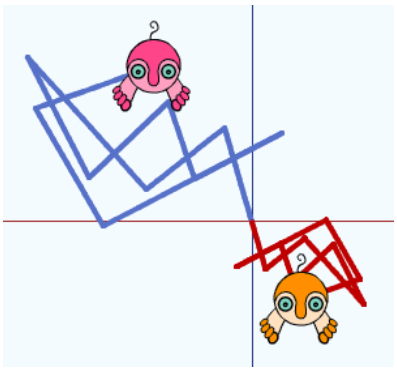


continued ▶

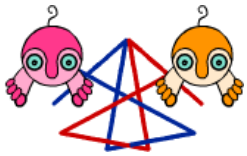


ADDITIONAL SUPPORT CONTINUED

6 If we multiply x and/or y position of **Fleeeee** by 0.5, the resulting doodle will be half the size (a reduction). Explore multiplying by -0.5. Explore multiplying x by one number and y by a different number, e.g. 1 and -0.5 or -0.5 and -0.5



MODULE 6 • INVESTIGATION 3 • ACTIVITY 6.3.3
Through the Looking Glass



LEARNING OBJECTIVES

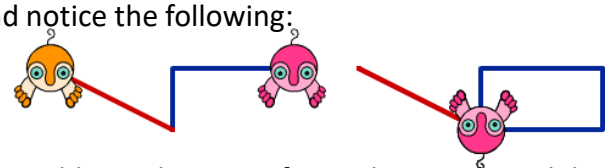
Explore how to create reflected polygon drawings.
Explain the differences between creating a translated and reflected drawing.

ACTIVITY INSTRUCTIONS

Pupils open project **63-Looking Glass**, **Save as a copy** (online) or **Save as** (offline) and rename. The final version of this project at the end of Activity 6.3.3 will be **63-Looking Glass Final**.

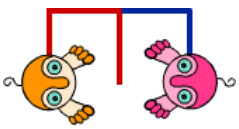
In activity 6.3.1 we dragged **Fleeeee**, and **Meeeee** mimicked its x and y positions, as well as its **costume #**. In activity 6.3.2 we made **Fleeeee** **glide** along the stage. Now we are going to program **Fleeeee** as we did earlier with **Beetle**. **Meeeee** will imitate all movements of **Fleeeee** as if in a looking glass (mirror).

1. Pupils open the project, click the green flag and explore how **Meeeee** imitates **Fleeeee**.
2. Pupils click the two block script for **Fleeeee** with a **move …** and a **turn …** block. They click it several times – it seems to work well for **programming** **Fleeeee** and **Meeeee** **imitating**. But there is an issue: try to build a script for **Fleeeee** to draw a rectangle e.g. 50 by 100 and notice the following:



(See additional support for explanation). To debug this problem instead of **move … steps** pupils use a new move block we've defined as **stroll … steps** (to behave as a 'slow' move).
3. Pupils make variable **side length** and use it with the **stroll …** block they define their own **square**. They use **square** to create more complex pictures (see additional support and pupil presentation).
4. **[Extension]** Pupils modify the mimicking script of **Meeeee** to (a) instead of reflecting in the x axis, it will multiply *x position* ***of*** *Fleeeee* by 0.5,
(b) reflect the drawing of **Fleeeee** in the y axis.
5. **[Extension]** Pupils define different polygons using **side length** in **stroll …** and use them to create complex drawings.
6. **[Extension]** Using **square** and **triangle**, pupils define their own **house** and **row of houses** of random sizes, with **Meeeee** drawing its reflection in the y axis.

MATHEMATICS CONNECTIONS

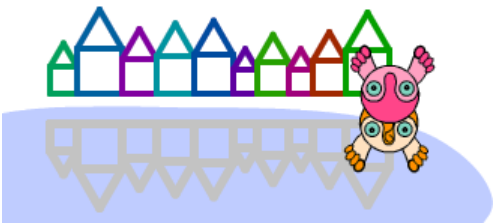


Ask: *why is the first line red?*

Discuss: *who moved first and who second?*

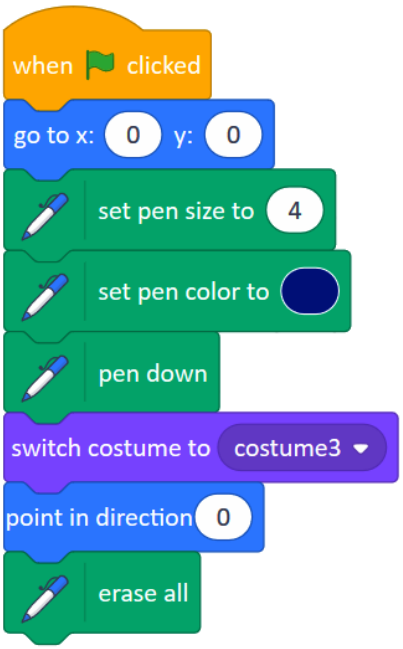
Build regular polygons, squares, triangles, hexagons, octagons to utilize the knowledge from earlier modules.

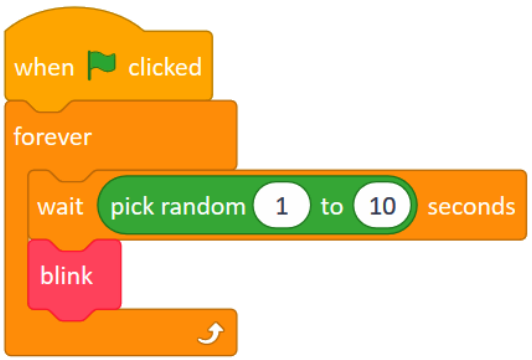
Ask *how do we calculate the angle to turn?* [360/number of sides. Remember the sprite turns through the exterior angle. A sprite will turn through 360° as it traces the outline of any closed shape.]

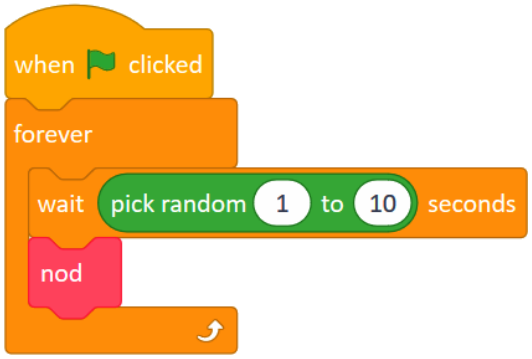


ADDITIONAL SUPPORT

1

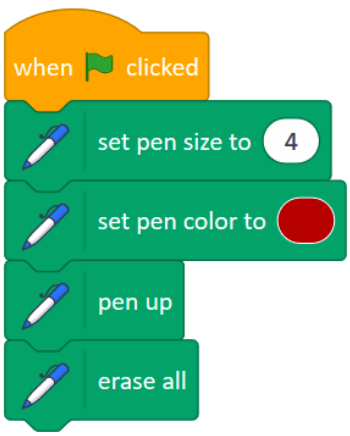


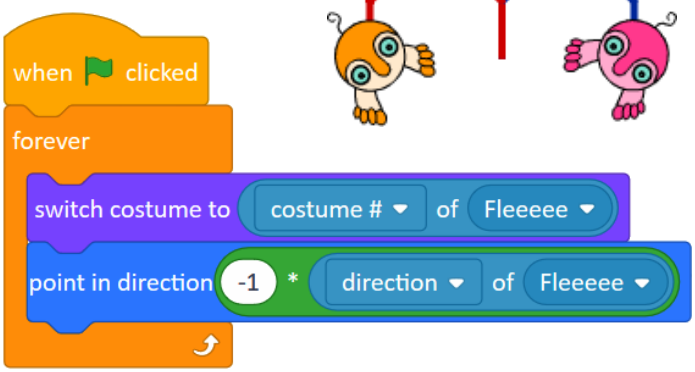


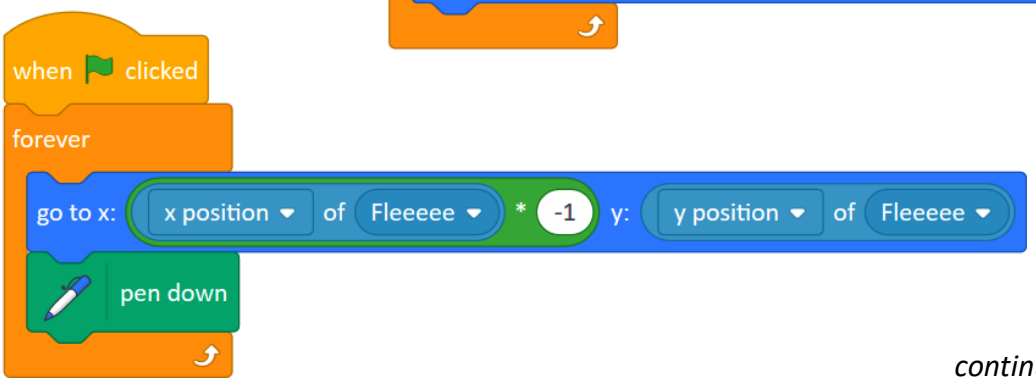


Pupils are already familiar with all of these scripts from the previous projects. Isolated short scripts here can inspire them to test **Meeeee** and the way it imitates **Fleeeee**. Pupils click the script repeatedly.

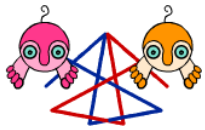
Do not forget to click the green flag first – to make all *setup scripts* and **forever** scripts run.







continued ▶

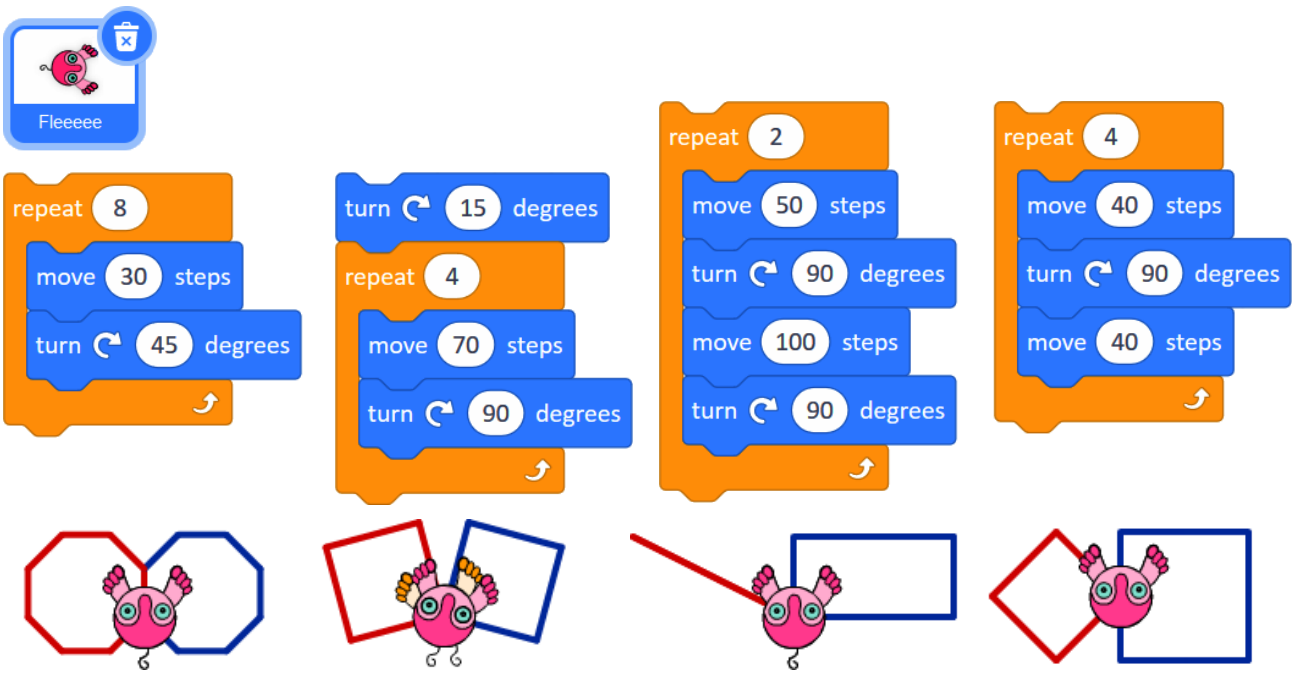


The *setup scripts* for **Meeeee** puts the pen down after jumping to its initial position. In the first **forever** script there is a sophisticated block for making **Meeeee** point as if 'in the mirror', see (a). The effect will be for Meeeee to face in the opposite direction of Fleeeee.

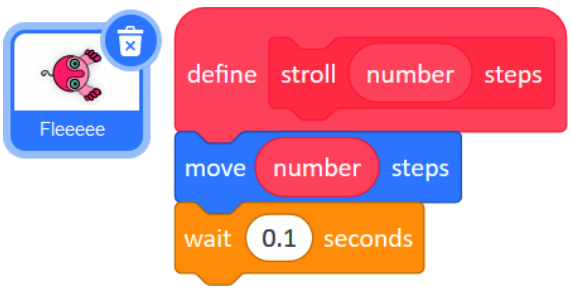
The same operator block works well for reflections in the y axis when mimicking starts, see (b).

2 While scripts like the two below left below work well for both **Fleeeee** and imitating **Meeeee**, scripts on the right do not. A sequence of **move** and **turn** blocks inside one **repeat** runs so quickly that the mimicking script of **Meeeee** 'misses' the intermediate steps. For example, when drawing rectangles, **Meeeee** jumps directly to the opposite corner of the rectangle.

If we put a **wait 0.1 secs** block after **move** and **turn**, everything would work well.

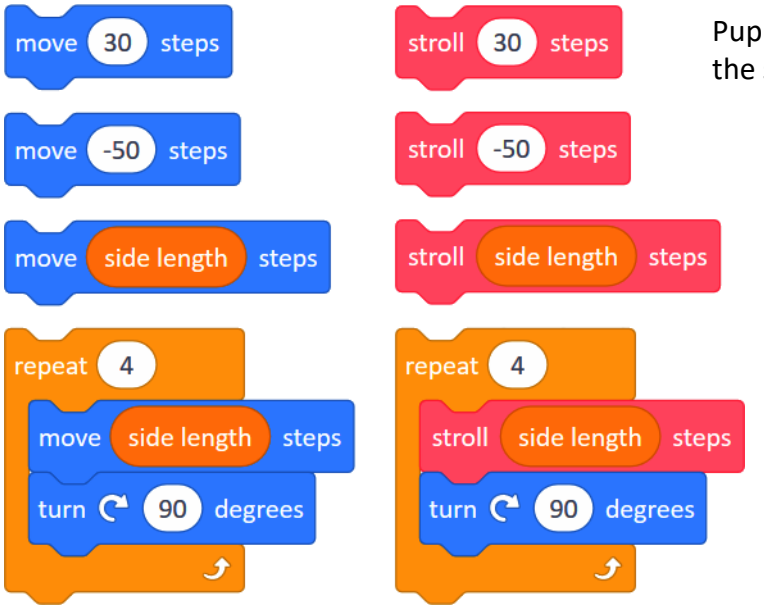


However, the *safest* solution is to make **Fleeeee** move "slower". We do not want pupils to insert a **wait** block after each **move,** so we define a new version of the **move** block with the short wait built in. Although such a definition exceeds the computational skills of Y6 pupils, we consider it to be an appropriate open 'window' to secondary computing – **our own new blocks with inputs** illustrate one of many computing concepts to learn in KS3 computing.



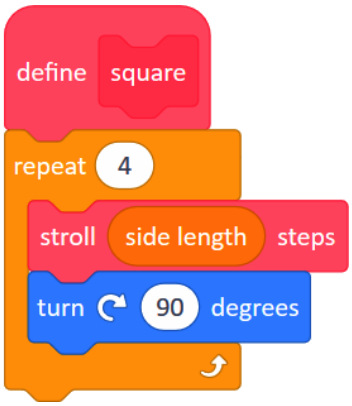
continued ▶

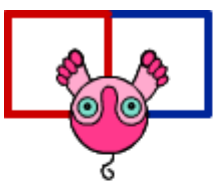
ADDITIONAL SUPPORT CONTINUED

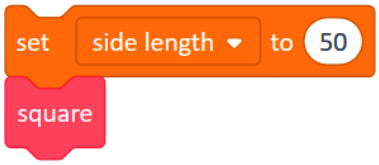


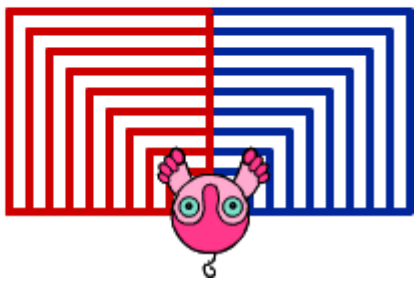
Pupils will use **stroll … steps** in exactly the same way as **move … steps** block.

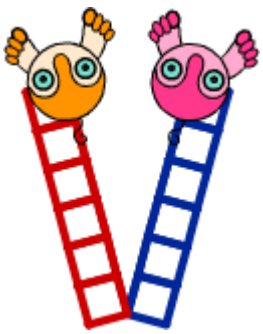
3

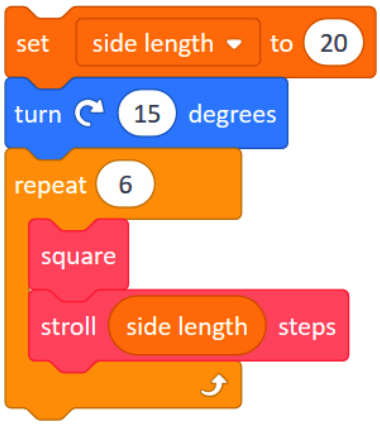


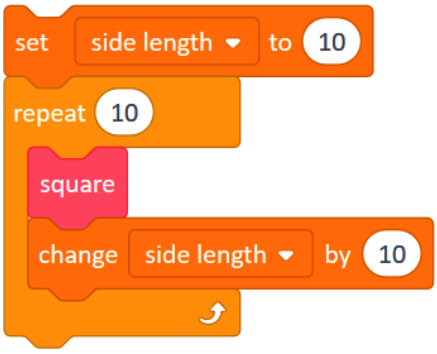






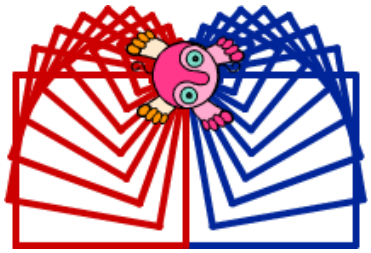


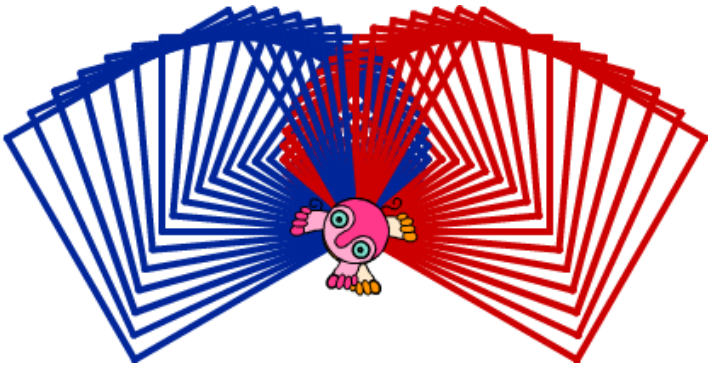




continued ▶

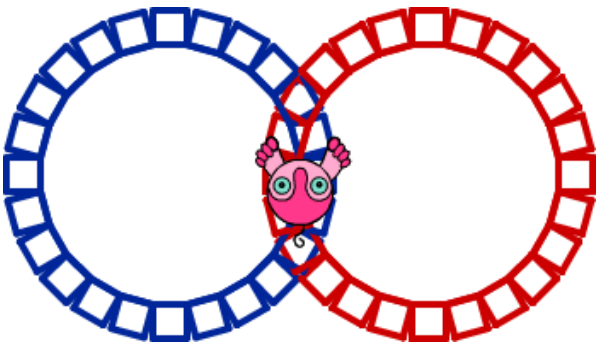
ADDITIONAL SUPPORT CONTINUED

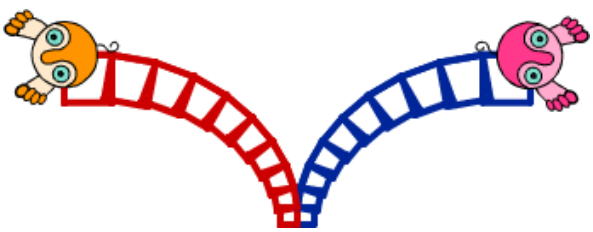




```
set side length to 10
repeat 10
  square
  change side length by 10
  turn ↻ 10 degrees
```

```
set side length to 6
repeat 25
  square
  change side length by 6
  turn ↺ 5 degrees
```





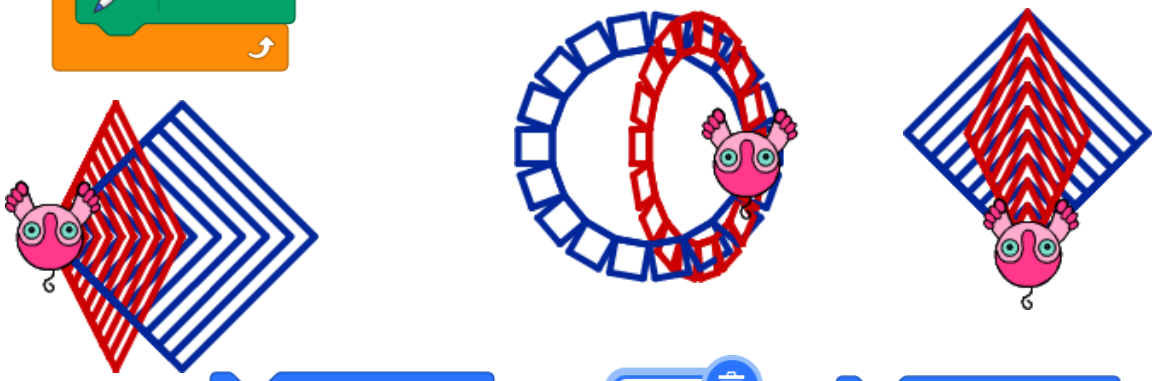
```
set side length to 20
repeat 24
  square
  stroll side length steps
  turn ↺ 15 degrees
```

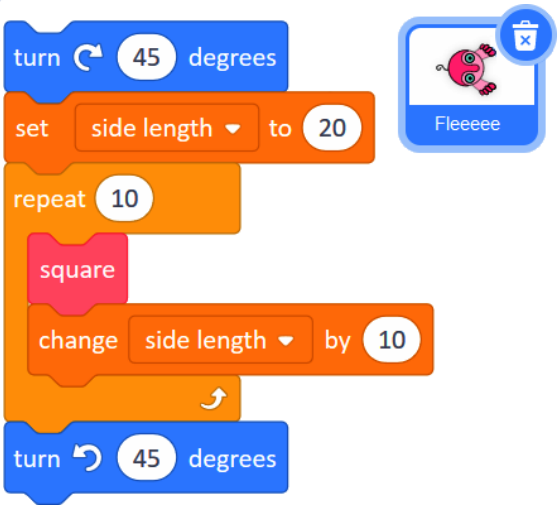
```
set side length to 10
repeat 10
  square
  stroll side length steps
  change side length by 2
  turn ↻ 10 degrees
```

continued ▶

ADDITIONAL SUPPORT CONTINUED

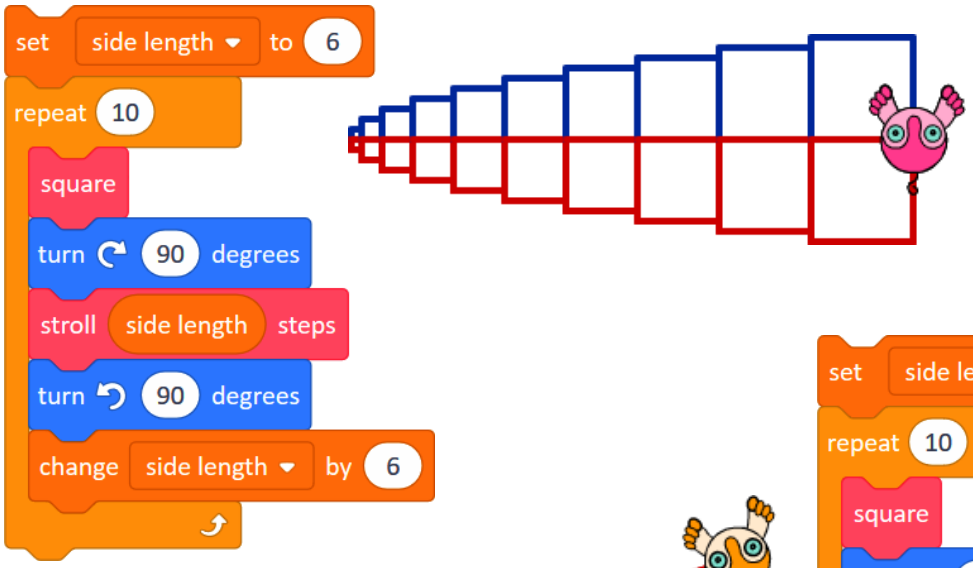
4

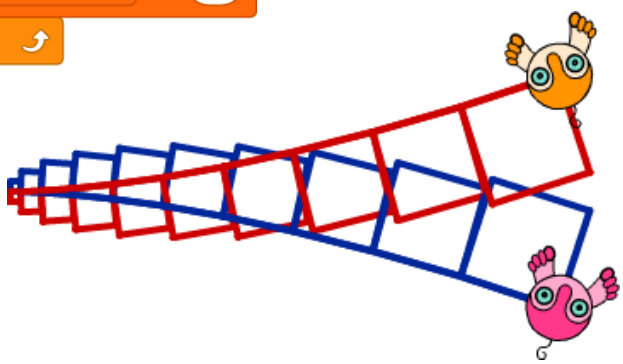


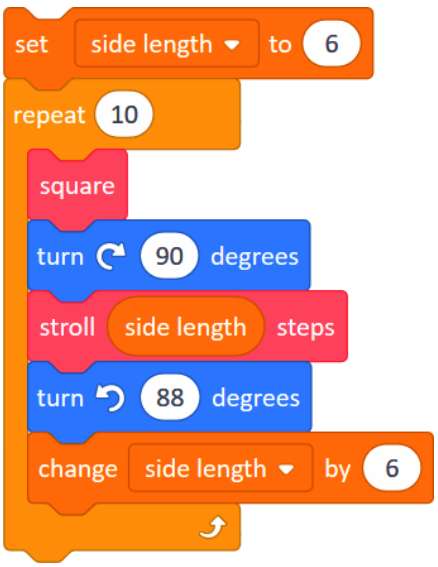


continued ▶

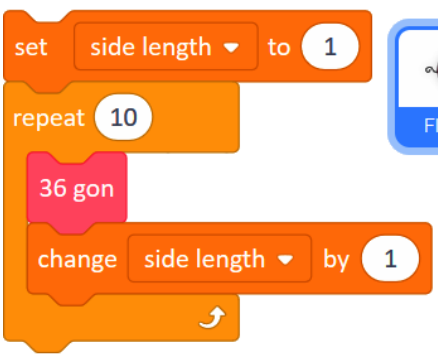
ADDITIONAL SUPPORT CONTINUED

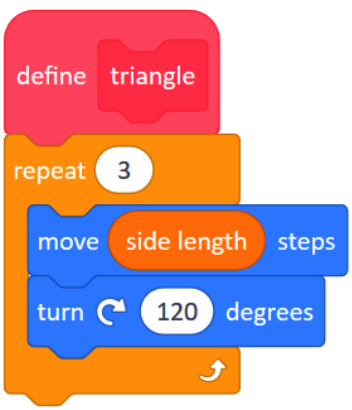


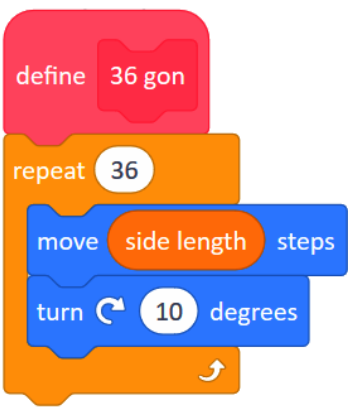


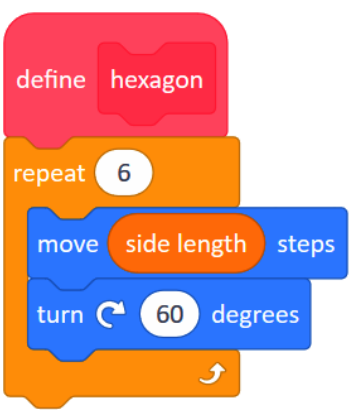


5







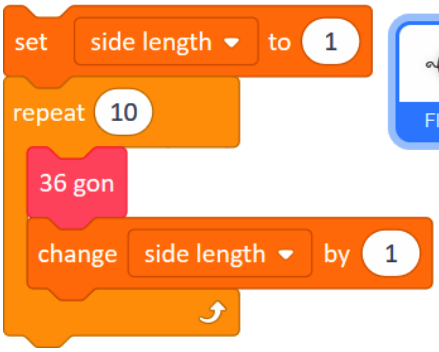


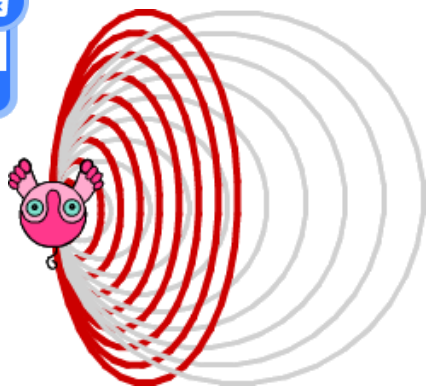
Meeeee reflects the drawing of **Fleeeee** in the y axis.

continued ▶

ADDITIONAL SUPPORT CONTINUED

In this one **Meeeee** multiplies *x position* **of** ***Fleeeee*** by 0.5.





6 Switch backdrop to the *in the day* or *in the night*.