Barefoot

Recommended for
ages 7-11

Shapes and Crystal Flowers

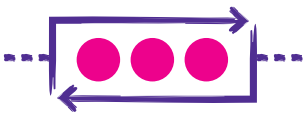
An introduction to repetition using Scratch

Activity Duration: **1 hour**

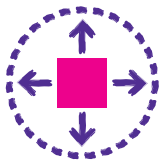
Principal partners

BT Computing at School

Concepts and approaches covered



Repetition



Programming

Overview

In this activity, pupils design **algorithms** to draw patterns made of simple shapes before writing a Scratch program to draw their shapes. In doing so, they learn about **repetition**.

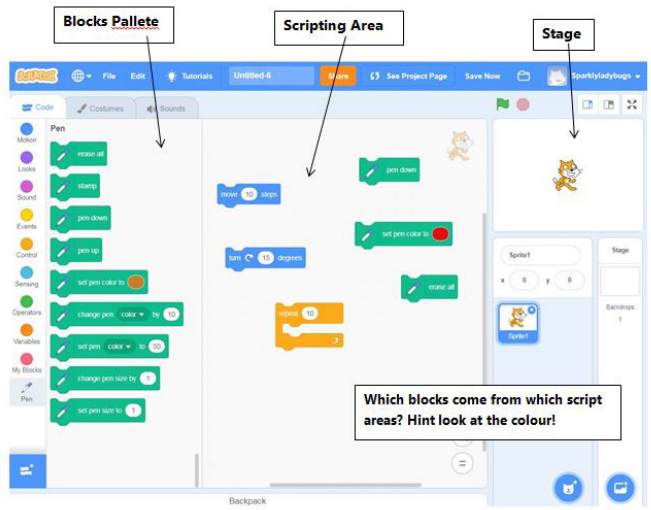
Pupil objectives

- ☑ **I can write a program that uses a repeat command**
- ☑ **I can explain what the repeats in my program do**

Introduction (15 mins)

Challenge: draw a square (10 mins)

- Explain that you would like pupils to complete a programming challenge: draw a square using Scratch
- Ask pairs to work out the algorithm that they think they will need and discuss as a class
- Open Scratch and if necessary, review the main areas of Scratch (slide 2, shown below)



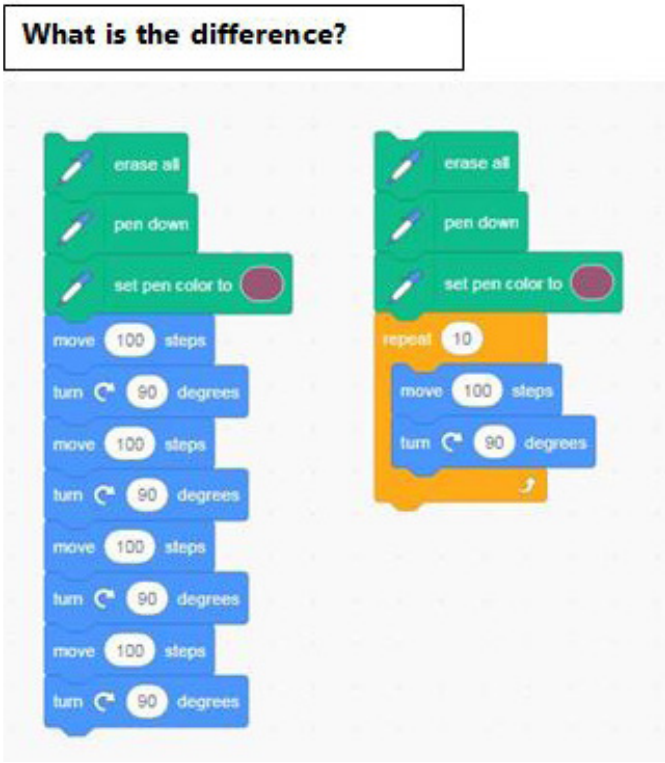
This page could be printed off as a help sheet as it has the basic blocks that pupils will need to create their shapes, as well as the block categories as shown circled.

- Ask pupils to open Scratch and create a new project with an appropriate file name saved to the school network, or the Scratch website if they have their own accounts
- Give pupils five minutes or so to use their algorithm to write their code that might have been better. Ensure pupils swap roles during this time

- If required, support pupils by showing them the relevant blocks such as 'pen', 'move', 'turn' and 'clear'. Once all pupils have attempted this challenge, share and discuss their solutions

Introduce repetition (5 mins)

- Share a program to draw a square that uses only a sequence of instructions and one that uses the repeat command (similar to the ones below). If no pupils have used repetition, an example is shown on slide 4. Invite pupils to discuss the differences
- Ensure pupils understand that some steps are repeated in the sequence-based program and that these steps have been moved inside the repeat command. Ask pupils what the number next to the repeat is to ensure they understand this is the number of times the loop will run



Drawing a square as a linear sequence of steps and with a repeat (slide 4).

- Set the pupils a challenge: by the end of the lesson they should be able to tell you how repetition loops are useful!

Main task (35 mins)

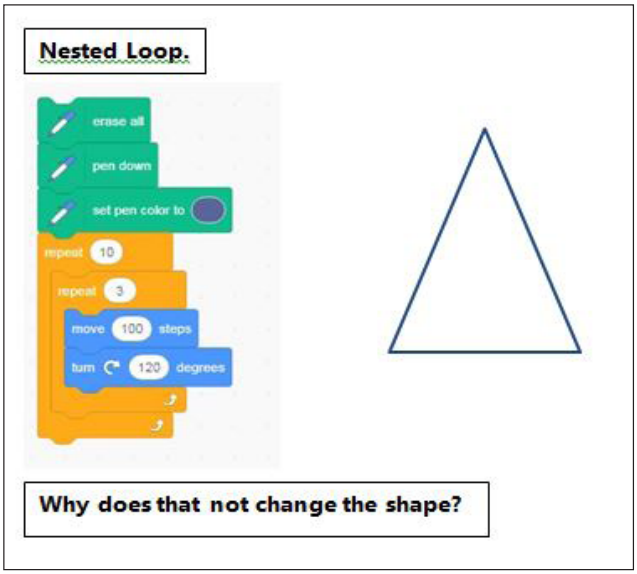
Use repetition to draw various regular polygons (10 mins)

- Ask pupils to write a repeat-based version of their own 'draw a square' program. Don't model this, instead allow pupils to find out independently how to move blocks into the repeat command
- As pupils complete the task of coding a square using a repeat command, challenge them to write new programs to draw other rectangular polygons using repetition. E.g. an equilateral triangle, a regular pentagon, a regular octagon

- Show pupils how to copy their earlier code to do this (the code can be duplicated by right-clicking and selecting duplicate, or it can be copied across sprites by dragging the code from the current scripting area onto another sprite's image)
- Encourage pupils to use trial and improvement (**debugging**) and **logical reasoning** to work out the angle they need for new shapes
- Ask pupils to show the shape they have drawn to their peers and ask them what it is, and what repeat has been used to create the shape

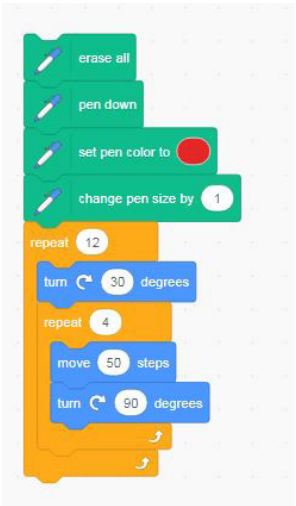
Nested loops (10 mins)

- Show pupils how to create a nested repeat (a repeat within a repeat, often called a nested loop). Explain that these have sometimes been called 'crystal flowers'. You could at this point discuss the role of repeating pattern in nature or art, perhaps by looking at a small selection of flower photographs (slides 7 and 8 have some examples)
- Using one of your pupil's programs that draws a regular polygon, add an outer loop. Ask pupils to predict what will happen. Model testing it. (The pattern does not change we just drew over the top as many times as the outer loop, see example below)

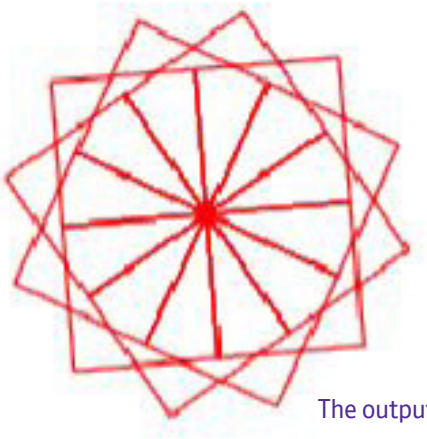


Drawing a square as a linear sequence of steps and with a repeat (slide 4).

- Add a turn within that outer repeat (loop). Ask pupils to predict what will happen. Model testing it

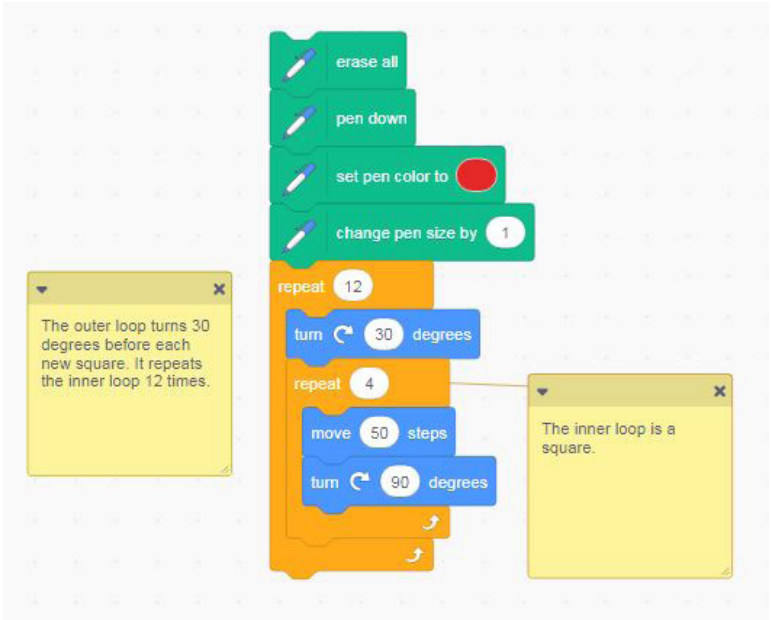


A nested loop. The extra repeat has been snapped around the repeat code that draws a square and a turn command added within the outer loop.



The output from the nested loop code.

- Model how to add comments, describing what shape the inner loop is and how the repeat works (right-click on a block in the scripting area and click on 'add comment')

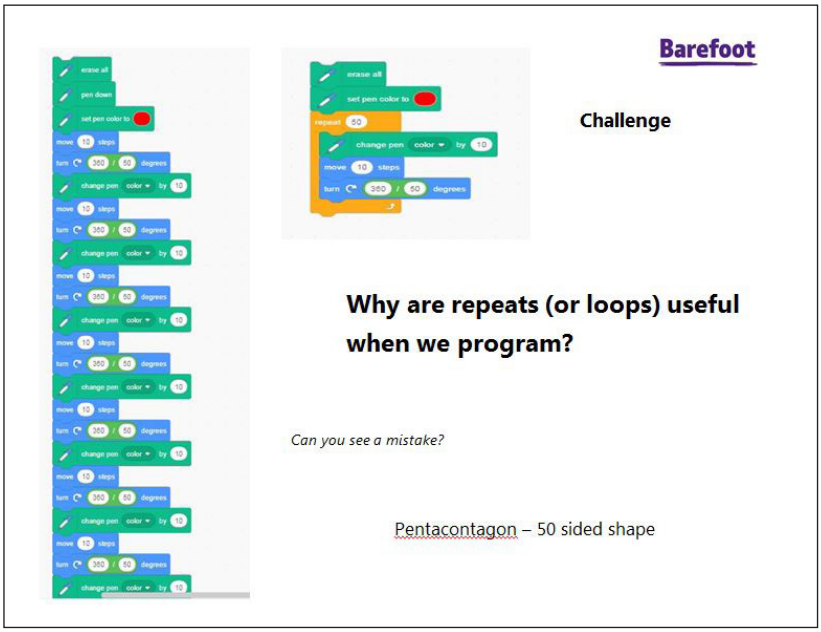


The output from the nested loop code.

- Challenge pupils to use nested loops to create new patterns, copying existing code to create new programs and adding comments to their work. Encourage pupils to change the angle and number of repetitions in the outer loop to explore the emerging patterns and to predict what the effect their changes to their code will have on their patterns
- Further challenges could include:
  - use different pen colours or pen sizes for parts of the shape (e.g. by using the pen block 'change pen colour by')
  - change the inner repeat to draw a triangle, pentagon or other regular shape
  - add a 'move' block within the outer repeat

Plenary (5 mins)

- Ask pupils to show another pair one of their shapes and ask them to work out what the loops are that have been used. Then they should share the program and see if they can make any suggestions to improve the code
- Look at one or two interesting examples as a class. Ask pupils to discuss why repetition is useful when we program. Guide the discussion to the idea that you reduce the number of lines of code, and so it is quicker to code and easier to fix. An ideal example to help this thinking is ask if you created a **pentacontagon** (50 sided shape) or **chiliagon** (1000 sides). How long would it take to write the program with just a sequence?



The output from the nested loop code.

Differentiation

Support:

- Pupils who find the maths difficult could be given the angles of the shapes they are using and the values to use for the outer nested loops, although all pupils can experiment with changing these values in the Scratch programs
- When programming, pupils who find it difficult to remember which blocks to use can be given sample code that does not give the answer, but is for a different shape
- A **template Scratch file** has been provided for those pupils who need it, this file has the sequence based square program already coded and a repeat block already on the scripting area

Stretch & Challenge:

- Pupils could use rectangles, rhombuses, parallelograms or other quadrilaterals as the base shape for their repeating patterns
- Pupils could be asked to add a 'move' block to the outer loop as well as a turn, and encouraged to explain the effect
- Pupils should be asked to explain how their loops work, perhaps using comments in Scratch. Pupils can be asked to add commands to change the colour or thickness of lines, at different points in their shape design, explaining their reasoning of what they changed and what the impact was

Assessment opportunities

- Informal teacher assessment of progress during the lesson. Key pupil knowledge and skills to identify include:
  - Do pupils understand the concept of repetition?
  - Do pupils see the benefit of using repetition commands?
  - Can pupils use repetition commands to draw shapes?
  - Can pupils nest repetition commands to create crystal flowers?
  - Can pupils debug their programs when required?
- Formal, summative assessment of Scratch projects (pairs)
- Peer assessment and feedback of crystal flower programs during the plenary

Teaching notes

Concepts and approaches:

- Pupils use **repetition** commands to write efficient code to draw regular polygons. They then use nested repetition commands to create patterns
- Pupils create their crystal flower **programs** in Scratch
- If required pupils **debug** their code to remove errors
- Pupils write out the **algorithm** for drawing regular polygons and use these to help write the code for their programs

Curriculum links

Computing

- KS2: use repetition in programs

Maths

- **Year 3:** draw 2D shapes; recognise angles as a property of a shape or a description of a turn
- **Year 4:** compare and classify geometric shapes, including quadrilaterals and triangles, based on their properties and sizes
- **Year 5:** know angles are measured in degrees: estimate and compare acute, obtuse and reflex angles; draw given angles, and measure them in degrees (°); identify: angles at a point and one whole turn (total 360°)
- **Year 6:** draw 2-D shapes using given dimensions and angles; compare and classify geometric shapes based on their properties and sizes and find unknown angles in any triangles, quadrilaterals, and regular polygons

Art and Design

- To develop their techniques, including their control and their use of materials, with creativity, experimentation and an increasing awareness of different kinds of art, craft and design

Resources

- **MIT's Scratch 3.0** (see **this guide**)
- Shapes and crystal flowers teachers' presentation - you can access these resources from the downloads folder
- Word file for editing the house algorithm using track changes, or for printing for use under a visualiser
- Some pupils may benefit from using a preloaded Scratch project. This contains the code to draw a square using sequence rather than repeat and a repeat block ready to use in the scripting area (You can also access the Scratch resource using this link: **http://Scratch.mit.edu/projects/26526384/**)

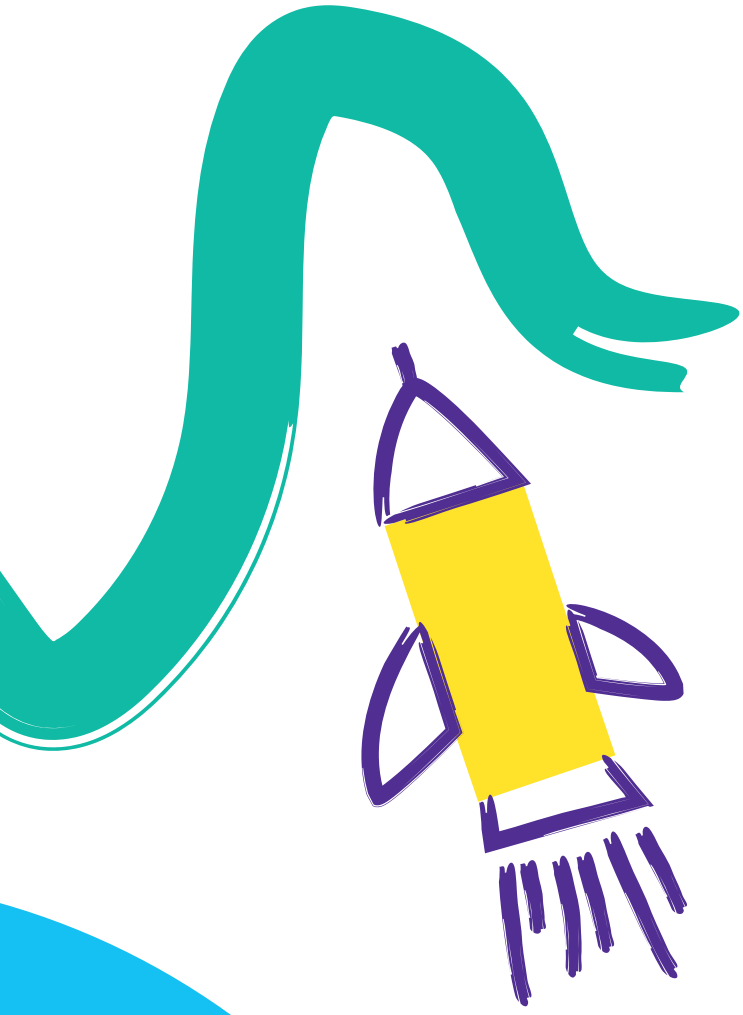
Taking this further

- Scratch workbook exploring shape: **http://scratch.ie/sites/all/themes/scratch_theme/resources/newworkbook/Module2.pdf**
- **Creating snowflakes in Scratch**

Related activities

Lower KS2 Fossil formation sequencing activity
Upper KS2 Viking raid animation sequence activity
KS2 Maths Quiz - selection activity
KS2 Maths Quiz - variables activity

Get more **Barefoot**



Have you had a great Barefoot workshop, or delivered a fun computer science lesson? Send us your comments and pictures via our social channels to help get more teachers involved!

@BarefootComp

/barefootcomputing

barefootcomputing.org

You can start your journey here

Principal partners

Supported by

Version 1118 Expires Dec 2021