Teaching Primary Programming with Scratch

Pupil Book – Year 4

PHIL BAGGE

A research informed scheme of work by Phil Bagge HIAS Computing Inspector/Advisor
Part of the HIAS Teaching Primary Programming from Scratch Series

Published in 2023 by University of Buckingham Press,
an imprint of Legend Times Group
51 Gower Street
London WC1E 6HJ
info@unibuckinghampress.com
www.unibuckinghampress.com

Published by arrangement with Hampshire Inspection and Advisory Service (part of Hampshire County Council)

ISBN 978-1-91505-4-241

CONTENTS

If you are struggling for time, I would recommend you do **Toy Give Away** or **Regular 2D shapes** and either **Fish Tank** or **Helicopter Game.**

INTRODUCTION

What Does This Book Do?

This book is a complete scheme of work for teaching primary programming using Scratch in Year 4 for 8–9 year olds.

What Is Included?

It includes permission to photocopy the pupil worksheets for your class or school.

These are clearly marked on the top right of the page.

It includes links to example code, project templates and slides to improve how you teach primary programming.

Part of a Series

It is part of a five-book series. Three other books include projects for other year groups.

Teaching Primary Programming with Scratch, Year 3

Teaching Primary Programming with Scratch, Year 5

Teaching Primary Programming with Scratch, Year 6

Teaching Primary programming with Scratch – Research-Informed Approaches

The teacher book explores methodology and pedagogy in detail helping you to understand why an approach is useful.

Progression

There is a clear, research-informed progression through the series and the grey-backed code shows which programming concepts are introduced in this book.

My Pupils have not used Scratch before

If pupils have a knowledge of the Scratch programming environment then they can start here. If they don't have this essential information I recommend you start with a couple of projects from Book 1 first. Pupils cannot use PRIMM methodology correctly until they understand simple sequence and the programming environment.

Pedagogue in a Few Paragraphs

Introduction to Programming Concepts Away From Code

Pupils are taught key programming concepts away from programming to lower cognitive load and make it easier to transfer these ideas from one programming language to another. They can record their algorithms on the knowledge organiser.

Paired Programming

Pupils are encouraged to work in same ability pairs for some parts of the projects because this has shown to be particularly helpful for pupils working within or below the expected outcomes.

PRIMM

Pupils are encouraged to read and understand code before they create their own code. We use the PRIMM method in this book.

Predict

Run

Investigate

Modify (change)

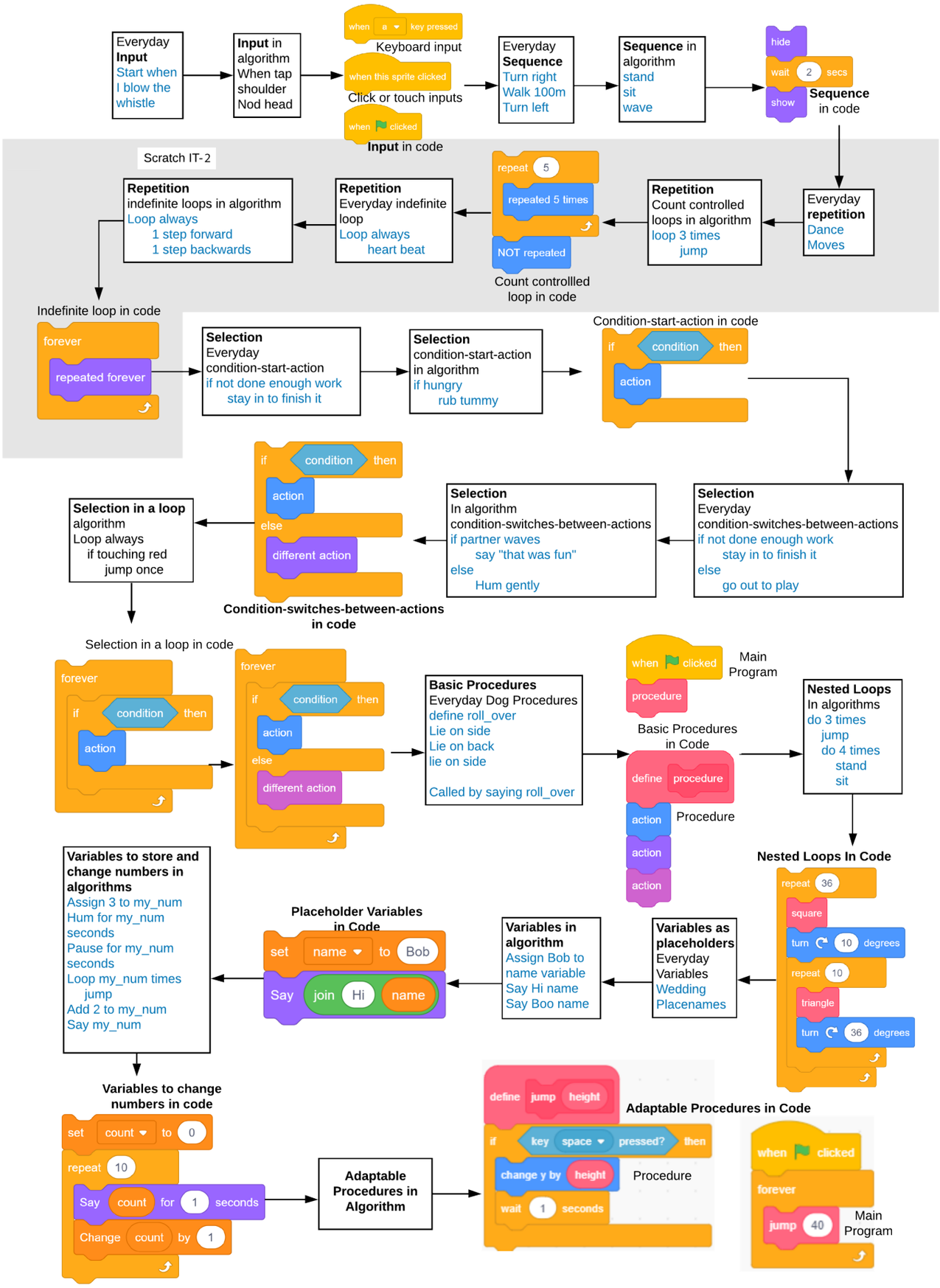
Make

Parsons

Four out of the five modules include a Parsons exercise to build code from a plan and pre-selected code. This can be useful for SEN pupils.

Creative

Each project provides time and stimulus to be creative in code within the zone of proximal development provided by the taught concepts and explored projects. In other words, it has reasonable projects that can be created independently or with minimum teacher support.


Everyday
Input
Start when
I blow the
whistle
Input in
algorithm
When tap
shoulder
Nod head
when a key pressed
Keyboard input
when this sprite clicked
Click or touch inputs
when clicked
Input in code
Everyday
Sequence
Turn right
Walk 100m
Turn left
Sequence in
algorithm
stand
sit
wave
hide
wait 2 secs
show
Sequence
in code
Scratch IT-2
Repetition
indefinite loops in algorithm
Loop always
1 step forward
1 step backwards
Repetition
Everyday indefinite
loop
Loop always
heart beat
repeat 5
repeated 5 times
NOT repeated
Count controllled
loop in code
Repetition
Count controlled
loops in algorithm
loop 3 times
jump
Everyday
repetition
Dance
Moves
Indefinite loop in code
forever
repeated forever
Selection
Everyday
condition-start-action
if not done enough work
stay in to finish it
Selection
condition-start-action
in algorithm
if hungry
rub tummy
Condition-start-action in code
if condition then
action
Selection in a loop
algorithm
Loop always
if touching red
jump once
if condition then
action
else
different action
Condition-switches-between-actions
in code
Selection
In algorithm
condition-switches-between-actions
if partner waves
say "that was fun"
else
Hum gently
Selection
Everyday
condition-switches-between-actions
if not done enough work
stay in to finish it
else
go out to play
Selection in a loop in code
forever
if condition then
action
else
different action
Basic Procedures
Everyday Dog Procedures
define roll_over
Lie on side
Lie on back
lie on side
Called by saying roll_over
when clicked
procedure
Main
Program
Basic Procedures
in Code
define procedure
action
action
action
Procedure
Nested Loops
In algorithms
do 3 times
jump
do 4 times
stand
sit
Nested Loops In Code
repeat 36
square
turn 10 degrees
repeat 10
triangle
turn 36 degrees
Variables to store and
change numbers in
algorithms
Assign 3 to my_num
Hum for my_num
seconds
Pause for my_num
seconds
Loop my_num times
jump
Add 2 to my_num
Say my_num
Placeholder Variables
in Code
set name to Bob
Say join Hi name
Variables in
algorithm
Assign Bob to
name variable
Say Hi name
Say Boo name
Variables as
placeholders
Everyday
Variables
Wedding
Placenames
Variables to change
numbers in code
set count to 0
repeat 10
Say count for 1 seconds
Change count by 1
Adaptable
Procedures in
Algorithm
define jump height
if key space pressed? then
change y by height
wait 1 seconds
Procedure
Adaptable Procedures in Code
when clicked
forever
jump 40
Main
Program

Knowledge

Key knowledge is introduced in the concept introductions and reinforced in each of the activities.

Revisiting Learning

It is important to revisit prior learning, so each module has questions and activities which revise learning from Year 3 on sequence and as we move from count-controlled loops to indefinite loops, prior loops are used and referenced to revisit learning.

Assessment

Summative Assessment

Summative assessment is baked into every stage of the PRIMM process, providing a wealth of data to determine progress.

Self–Assessment

Pupils self-mark to help them see how they have progressed, reducing teacher workload and enabling teachers to concentrate on pupils that need more support.

Hints & Tips

Every pupil's resource also includes a copy of the resource annotated with extra information to further teachers' programming knowledge, hints and formative assessment opportunities in case pupils are stuck and tips to adapt or support whole class teaching.

Many of these extra hints and tips will not be needed, but the more informed the teacher is, the better quality learning opportunity pupils will have.

Can We Start Here?

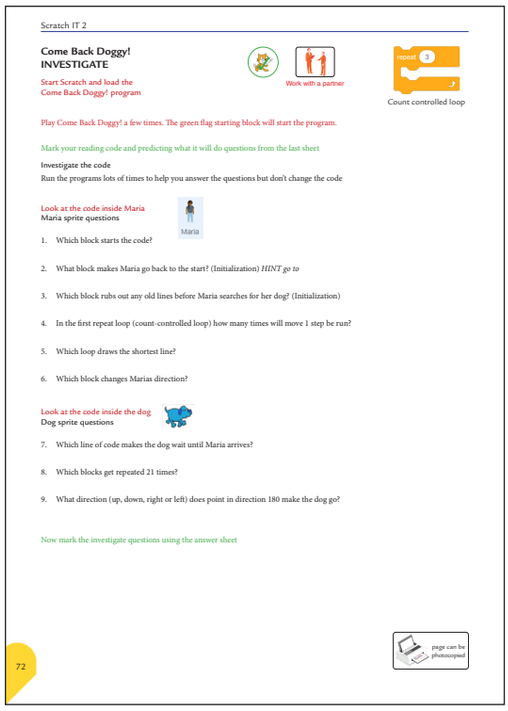
Pupils could start here if they have prior Scratch environmental knowledge. Agency over sprite, background and sound creation. The ability to snap blocks and delete code. If your pupils have not used Scratch before, I recommend starting with

Teaching Primary Programming with Scratch, Year 3

which uses different pedagogy appropriate to pupils prior knowledge.

Committed to Improvements

HIAS, Hampshire's Inspection & Advisory Service, is committed to developing and improving these resources. We recognize that primary programming is still its infancy in comparison with other subjects and that new research and primary practice will refine and improve teaching and learning in this area. All royalties earned from this series will be used to write more computing books and revise these resources as needed.



Scratch IT 2

Come Back Doggy!
INVESTIGATE

Start Scratch and load the Come Back Doggy! program

Work with a partner

repeat 3

Count controlled loop

Play Come Back Doggy! a few times. The green flag starting block will start the program.

Mark your reading code and predicting what it will do questions from the last sheet

Investigate the code
Run the programs lots of times to help you answer the questions but don't change the code

Look at the code inside Maria
Maria sprite questions

Maria

1. Which block starts the code?
2. What block makes Maria go back to the start? (Initialization) *HINT go to*
3. Which block rubs out any old lines before Maria searches for her dog? (Initialization)
4. In the first repeat loop (count-controlled loop) how many times will move 1 step be run?
5. Which loop draws the shortest line?
6. Which block changes Marias direction?

Look at the code inside the dog
Dog sprite questions

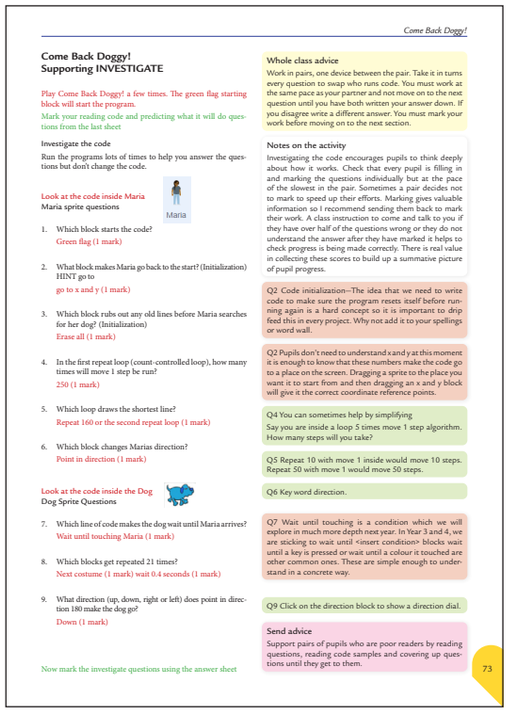
7. Which line of code makes the dog wait until Maria arrives?
8. Which blocks get repeated 21 times?
9. What direction (up, down, right or left) does point in direction 180 make the dog go?

Now mark the investigate questions using the answer sheet

page can be photocopied

72

Photocopiable resource for pupils



Come Back Doggy!

Come Back Doggy!
Supporting INVESTIGATE

Play Come Back Doggy! a few times. The green flag starting block will start the program.
Mark your reading code and predicting what it will do questions from the last sheet

Investigate the code
Run the programs lots of times to help you answer the questions but don't change the code.

Look at the code inside Maria
Maria sprite questions

Maria

1. Which block starts the code?
   Green flag (1 mark)
2. What block makes Maria go back to the start? (Initialization)
   HINT go to
   go to x and y (1 mark)
3. Which block rubs out any old lines before Maria searches for her dog? (Initialization)
   Erase all (1 mark)
4. In the first repeat loop (count-controlled loop), how many times will move 1 step be run?
   250 (1 mark)
5. Which loop draws the shortest line?
   Repeat 160 or the second repeat loop (1 mark)
6. Which block changes Marias direction?
   Point in direction (1 mark)

Look at the code inside the Dog
Dog Sprite Questions

7. Which line of code makes the dog wait until Maria arrives?
   Wait until touching Maria (1 mark)
8. Which blocks get repeated 21 times?
   Next costume (1 mark) wait 0.4 seconds (1 mark)
9. What direction (up, down, right or left) does point in direction 180 make the dog go?
   Down (1 mark)

Now mark the investigate questions using the answer sheet

Whole class advice
Work in pairs, one device between the pair. Take it in turns every question to swap who runs code. You must work at the same pace as your partner and not move on to the next question until you have both written your answer down. If you disagree write a different answer. You must mark your work before moving on to the next section.

Notes on the activity
Investigating the code encourages pupils to think deeply about how it works. Check that every pupil is filling in and marking the questions individually but at the pace of the slowest in the pair. Sometimes a pair decides not to mark to speed up their efforts. Marking gives valuable information so I recommend sending them back to mark their work. A class instruction to come and talk to you if they have over half of the questions wrong or they do not understand the answer after they have marked it helps to check progress is being made correctly. There is real value in collecting these scores to build up a summative picture of pupil progress.

Q2 Code initialization–The idea that we need to write code to make sure the program resets itself before running again is a hard concept so it is important to drip feed this in every project. Why not add it to your spellings or word wall.

Q2 Pupils don't need to understand x and y at this moment it is enough to know that these numbers make the code go to a place on the screen. Dragging a sprite to the place you want it to start from and then dragging an x and y block will give it the correct coordinate reference points.

Q4 You can sometimes help by simplifying
Say you are inside a loop 5 times move 1 step algorithm. How many steps will you take?

Q5 Repeat 10 with move 1 inside would move 10 steps. Repeat 50 with move 1 would move 50 steps.

Q6 Key word direction.

Q7 Wait until touching is a condition which we will explore in much more depth next year. In Year 3 and 4, we are sticking to wait until <insert condition> blocks wait until a key is pressed or wait until a colour it touched are other common ones. These are simple enough to understand in a concrete way.

Q9 Click on the direction block to show a direction dial.

Send advice
Support pairs of pupils who are poor readers by reading questions, reading code samples and covering up questions until they get to them.

73

Teacher Hints & Tips on the same photocopiable resource

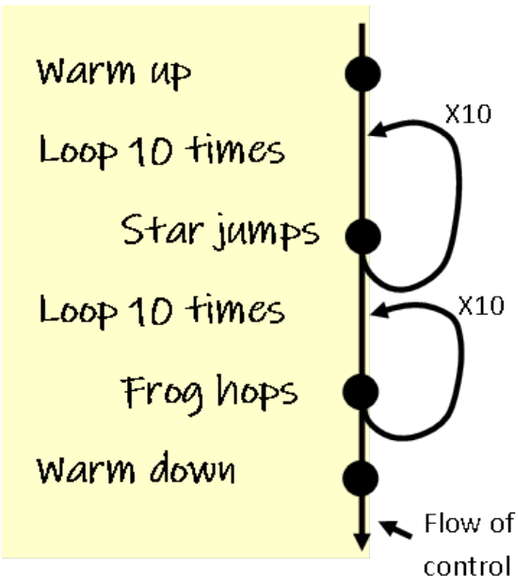
WE ARE LEARNING ABOUT LOOPS IN ALGORITHMS AND PROGRAMMING

Count-controlled loop algorithm

Algorithms

A set of instructions or rules to do something

Algorithms can be used to plan non-programming events such as exercise



We are

Indenting to show what instructions are inside a loop when writing planning algorithms.

Loop Vocabulary
repeat, loop, iterate, do so many times

An infinite indefinite loop

Can replace a sequence where there is a pattern.

Only ends when the digital device is turned off

Is called a forever loop in Scratch programming

Has a flow of control or order; the instructions are carried out

Is called an indefinite loop because we do not know how many times it will repeat

A loop is any set of instructions that are repeated

A count-controlled loop

Can replace a sequence where there is a pattern.

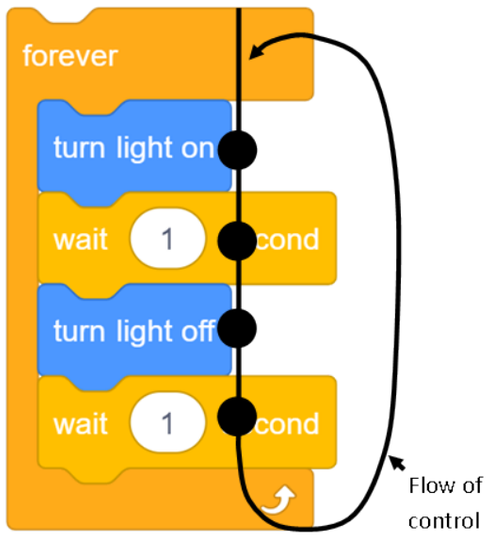
Is controlled by the number

Ends after the number of repeats are complete

Is called a repeat loop in Scratch programming

Has a flow of control

A flashing light is often programmed using an indefinite loop as shown below



Indefinite loop algorithm

photocopiable page

CHAPTER 1 Count-Controlled Loops

Introducing Count–Controlled Loops

These slides can be downloaded from the HIAS website https://computing.hias.hants.gov.uk/course/view.php?id=51.

Delivery

They are designed to be delivered to the whole class before pupils move on to using a count-controlled loop module of work such as

Toy Give Away

Regular 2d shapes

Dog Chase

They can also be delivered to a small group of pupils if they are working independently through resources in pairs.

Revision

If more than one count-controlled loop module is used, the slides could be used as a quick revision rather than an introduction

Format

Slides are provided in PDF and PowerPoint formats, and teachers who purchased the book are authorized to adapt the resources within their school or on closed learning platforms such as Seesaw, Google Classroom or Teams, as long as they are not shared outside the school community.

Hints

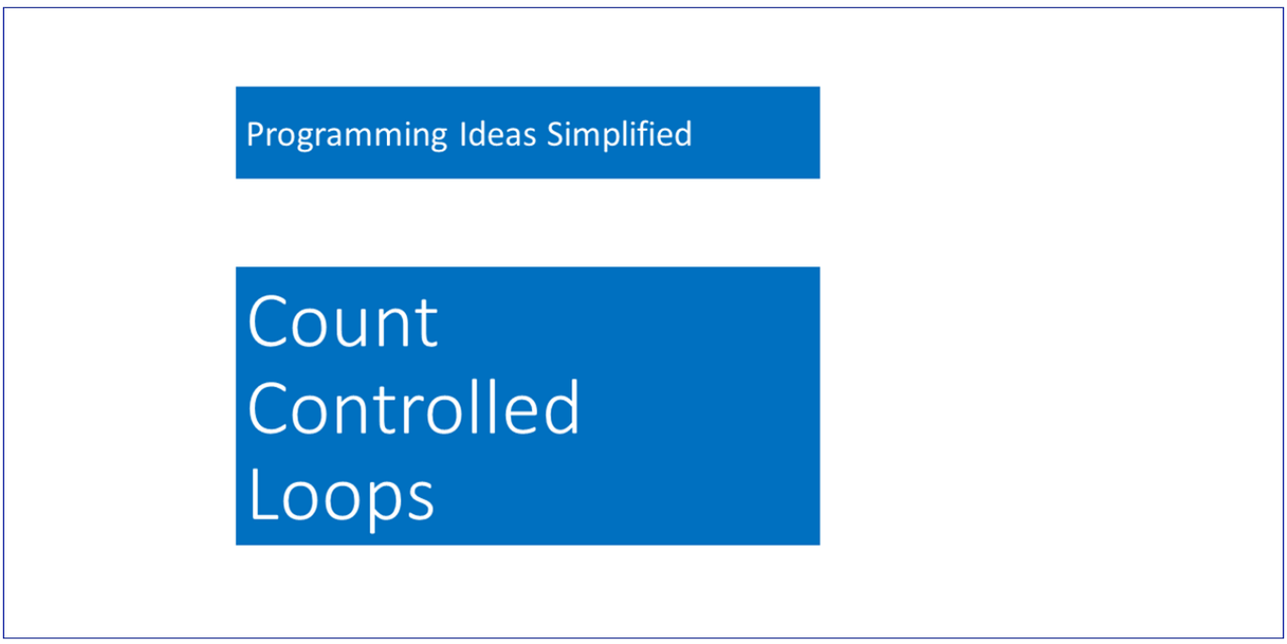
Extra hints and tips on usage are provided alongside each slide on the following pages.

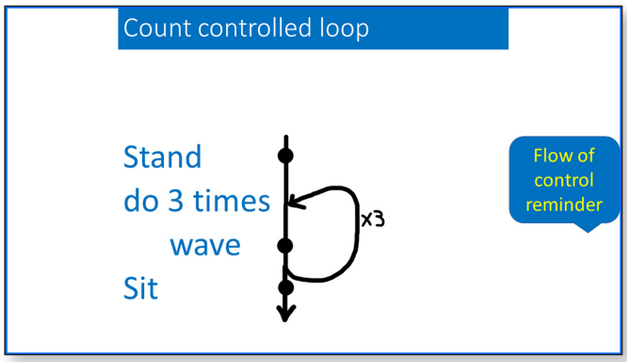
Resources

Pupils will need whiteboards and pens or paper and pencils

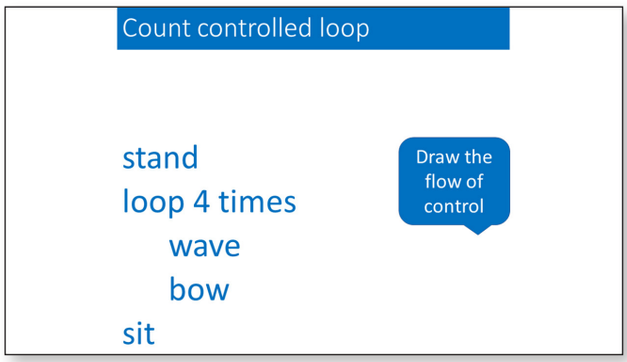
Knowledge Sheet

These is a knowledge sheet on page 11 that pupils can use to write their algorithms on and be reminded about key ideas.

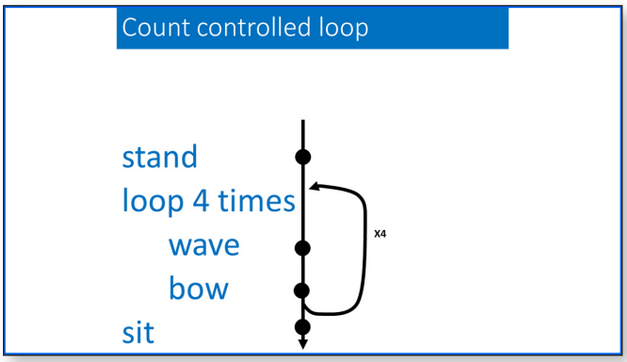




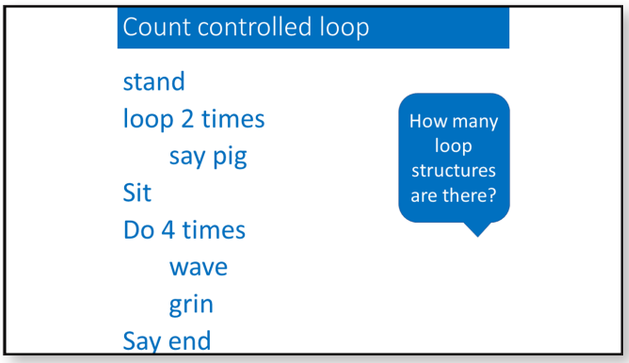
Point out that this is just an example to help them draw the next one themselves.



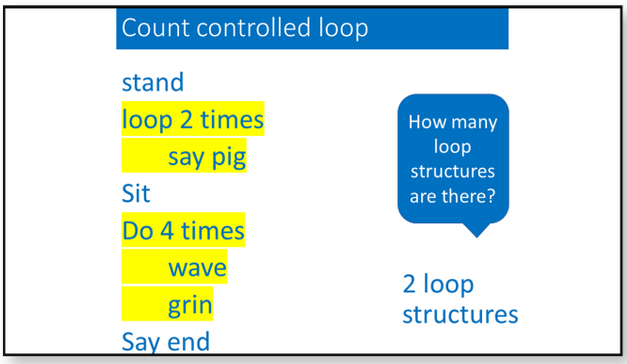
Remind pupils that they will need a dot for every action. Some pupils will benefit from this slide being printed out beforehand to be drawn on.



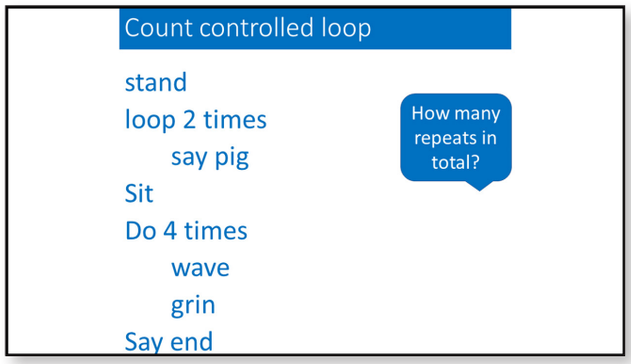
Break the flow diagram into parts and give marks for the sit dot, marks for the loop line, marks for wave and bow on the count-controlled loop and marks for stand at the top.

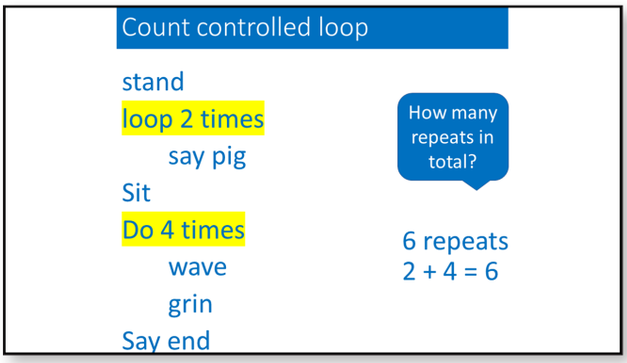


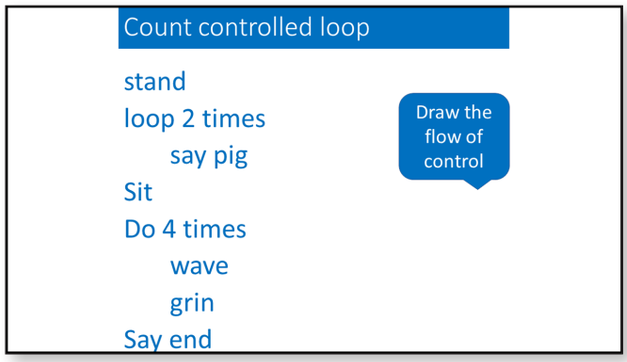
The loop structure is the part that tells you it is a loop.



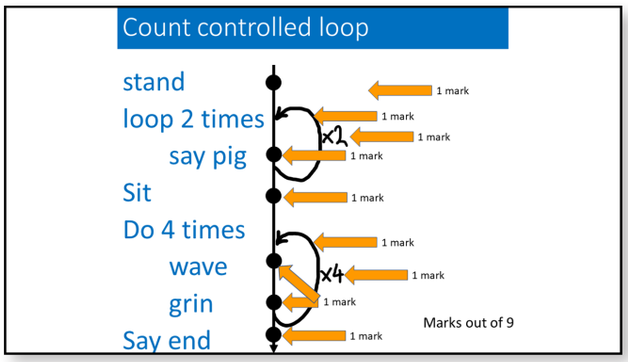
There are two loop structures, loop 2 times and do 4 times.

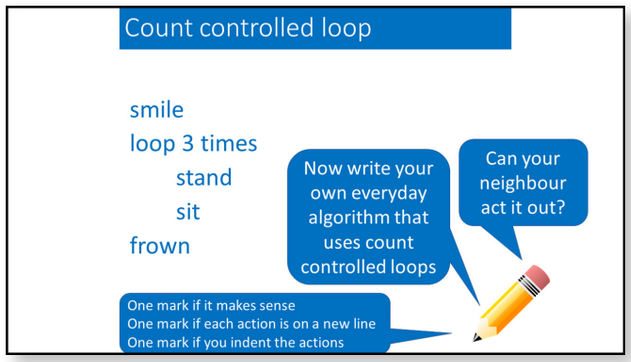




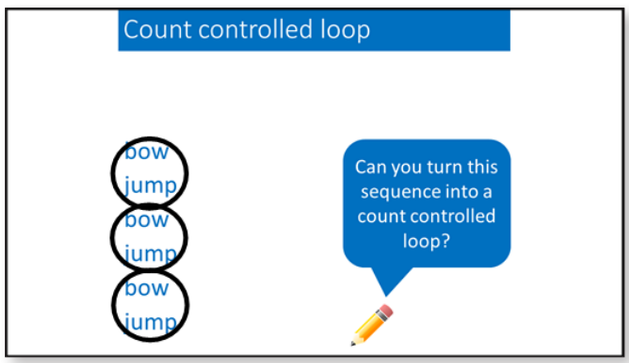


Some pupils will benefit from having this slide printed out to draw the flow directly on.

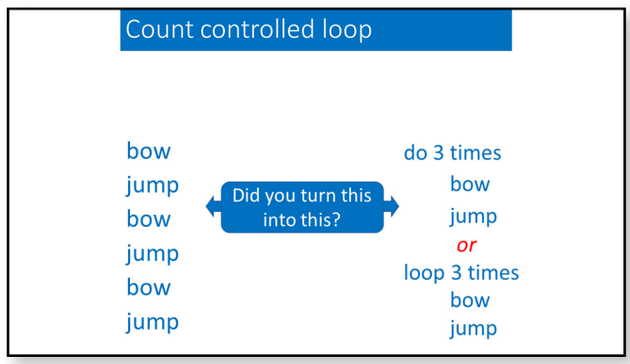




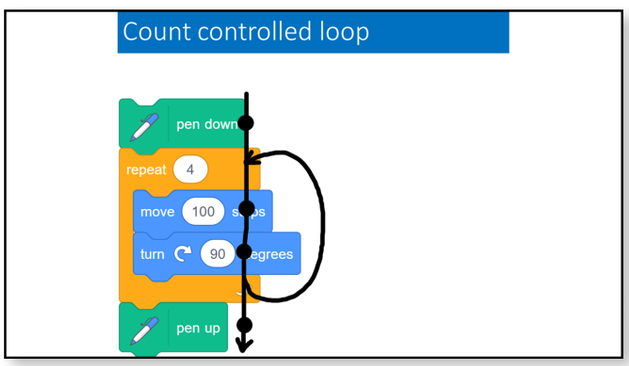
Pupils writing their own count-controlled loop algorithms that their partner can act out gives you time to formatively assess those who are struggling or provide writing or scribing support for those who you spotted in the earlier parts of the introduction.



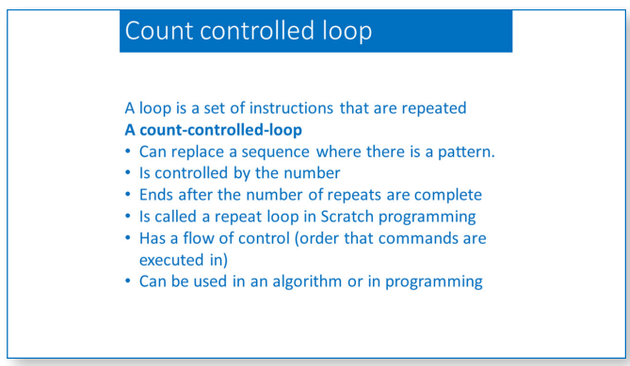
If pupils are struggling with this, ask them if they can spot the pattern in the sequence? Put circles around the pairs as shown. Ask pupils how many times the pattern is repeated.



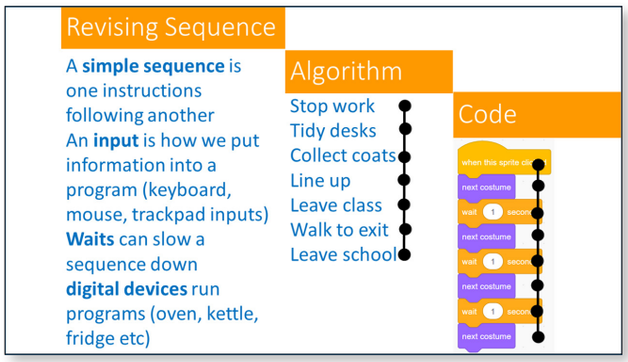
Point out that an algorithm can be written in any way as long as it can be understood by another human.
They could have used repeat 3 times.



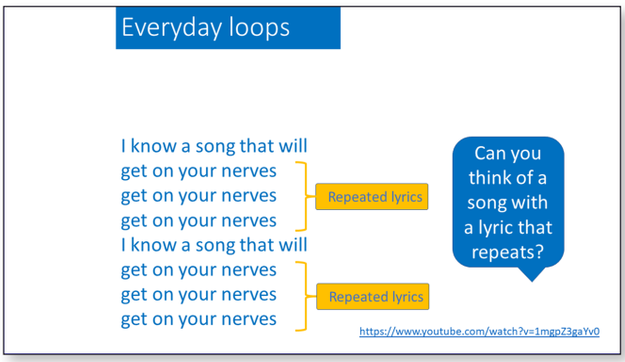
It is important to now point out what a count-controlled loop looks like in code. Identify that is called a repeat loop and point out where the number is.
Point out that the flow of control works on code count-controlled loops as it does on everyday algorithm count-controlled loops.



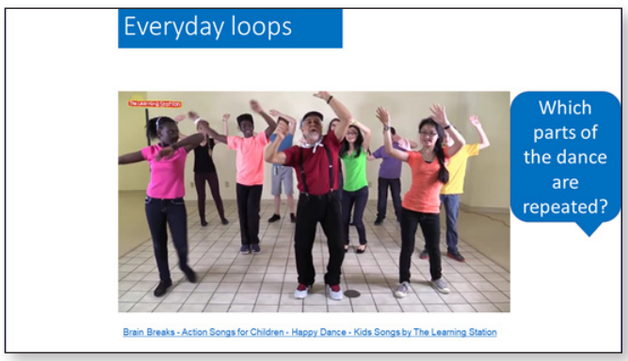
Read out this summary of the main points learnt from these slides.



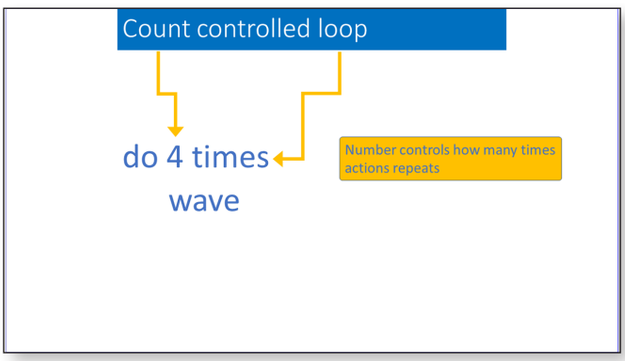
This slide reminds pupils of what they learnt about sequences in Year 3. Read the main points and point out sequence programming and sequence algorithms.



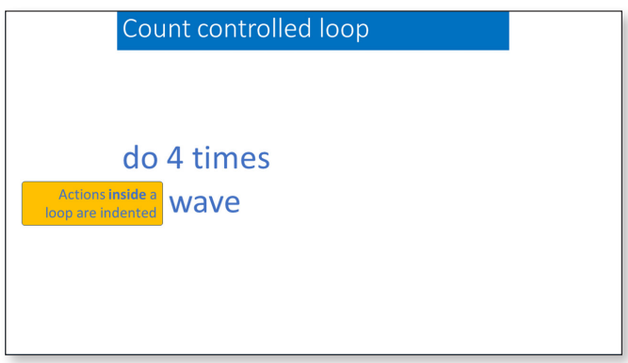
Ask a few children to sing or say popular repeated song lyrics.



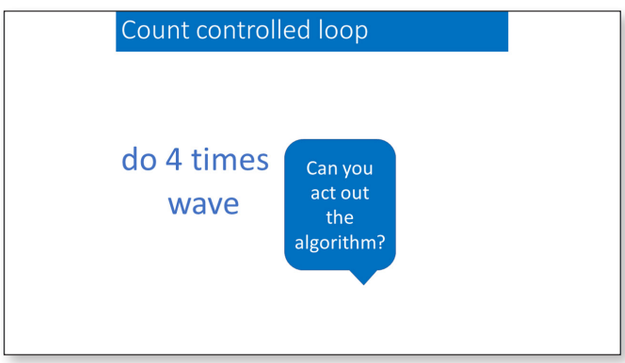
Ask pupils to watch the dance video and dance any move that includes repetition until you spot them doing it, at which time they can look for another example of repetition.



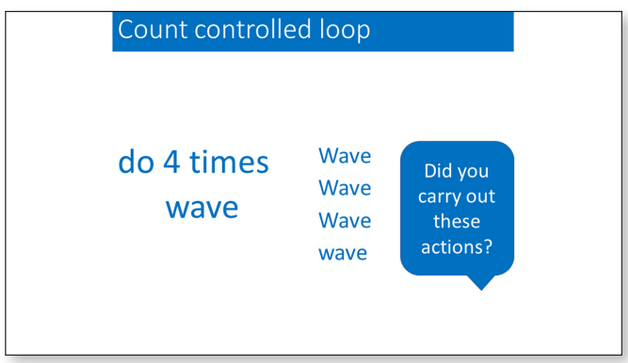
Explain that the number controls how many times wave repeats.



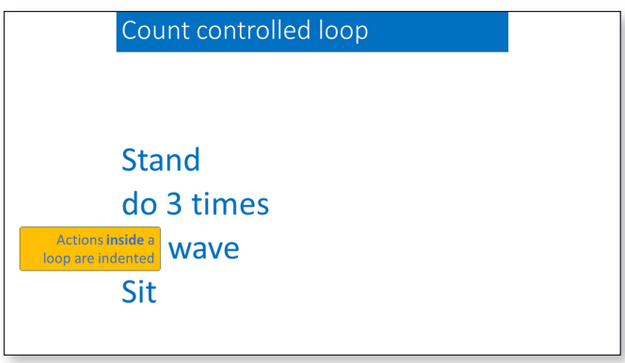
Point out that the wave is indented to show that it is inside the loop. All actions inside the loop are indented.



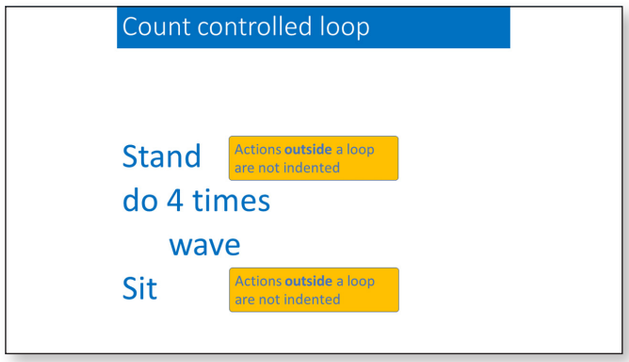
Point out that a count-controlled loop is a sequence of actions like wave, wave, wave, wave written in a different way.



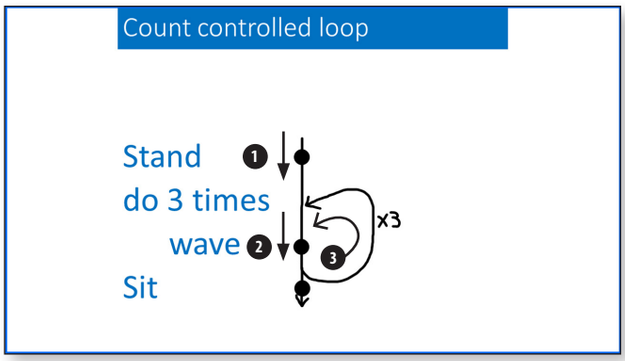
Watch your pupils carefully to see which ones are copying other children rather than following the instructions. When pupils are writing their own algorithms later, test these pupils with your own simple algorithm.



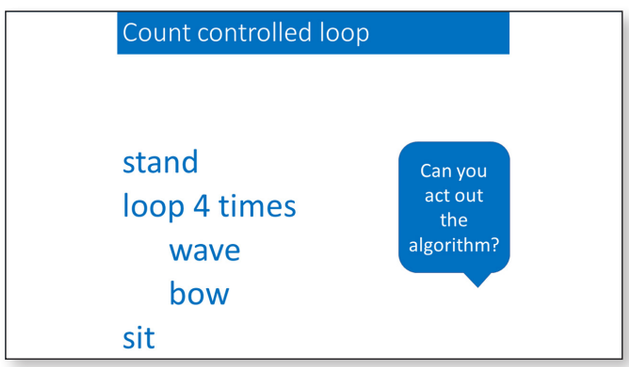
Point out that sit is not in the loop, as it is not indented.



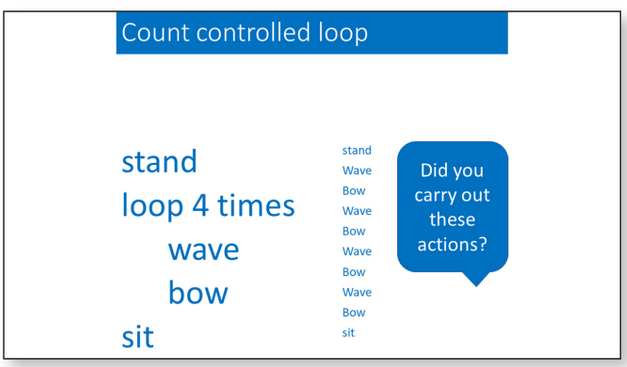
Point out that stand and sit are not in the loop, as they are not indented.



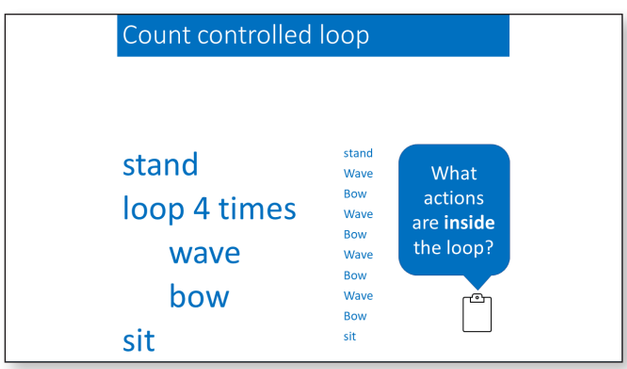
Trace your finger over the flow of control line while saying the actions. Now ask how we know that the loop repeats three times? Answer 3x symbol.



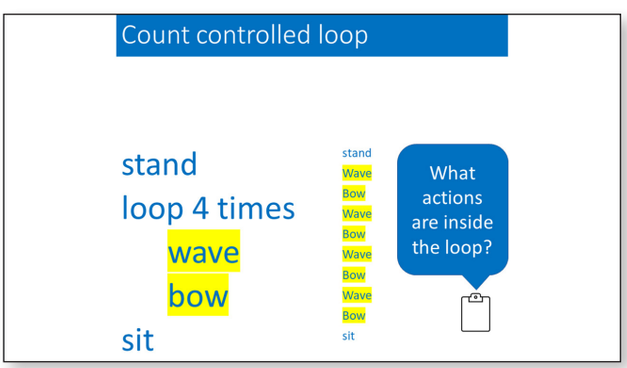
Watch carefully to see which pupils include sit inside the loop. If any pupils include sit inside the loop, point out that it is not indented.



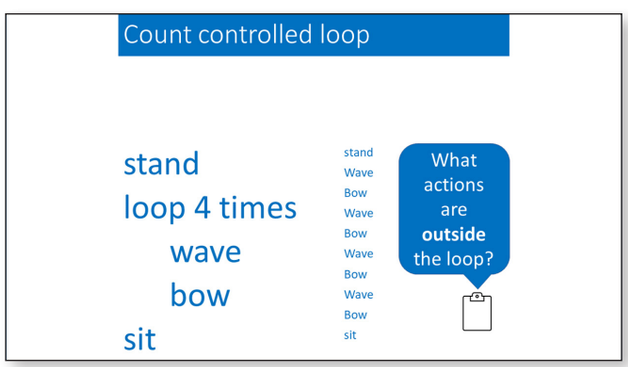
(The list helps pupils to see that a count controlled loop can be converted into a simple sequence very easily.)



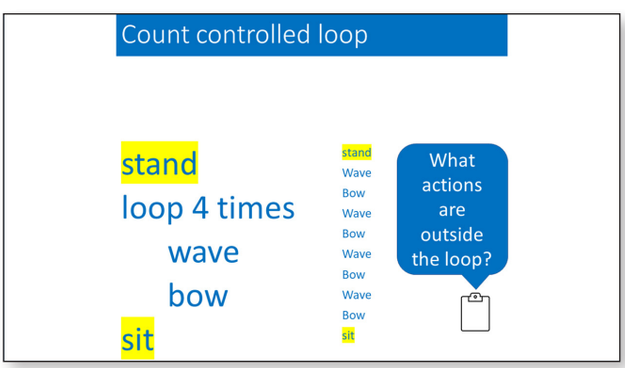
Ask pupils to answer this on a whiteboard.



Point to both the actions inside the loop and those in the sequence highlighted in yellow.



Ask pupils to answer this on a whiteboard.



Point to both the actions inside the loop and those in the sequence highlighted in yellow.