

CHAPTER 5

Making choices

Overview

Pupils identify where conditional selection can be found in their everyday lives before roleplaying and writing everyday conditions. They then examine code that uses conditions that start actions before building their own programming that uses conditions.

To do before the session

1. Look at the grid below and decide which optional and SEN activities you are going to include and exclude.
2. Print pupil worksheets for each activity chosen and staple into a booklet, one for each pupil.
3. Print marksheets for activities chosen to be placed where pupils can access them.
4. Download the code needed and place in a templates folder on your school network or add to a Scratch Studio or link on your learning platform.
5. Download the slides that go with the concept introduction.
6. Study the notes that go with the slides.
7. Examine the teacher help notes that are provided alongside every activity.
8. Photocopy some support cards for OR, AND & NOT for more ABLE.

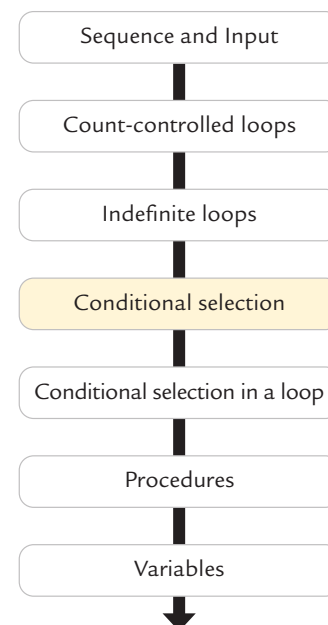
To do at the start of the session

If you have not introduced condition-starts-action with this class before, do this first as a whole class activity.

To do after the concept has been introduced

Each activity has whole class notes to help you explain what is needed if it is the first time pupils have carried out this type of activity. There are also core instructions underneath in case you are sticking to the core activities only.

How this module fits into a programming progression



Vocabulary

Condition, selection, true, false, condition met, pathway, decomposition, , input ask block, answer block, equals means same as, choice

Resource Name	Core Optional SEN	Teacher	Pupil Grouping	How Assessed	SCRATCH ACCESS
CONCEPT condition-starts-action	CORE	Leads Session	Solo Whole Class Activity	Formative	NO
PARSONS	OPTIONAL SEN OPTIONAL ALL (predict or parsons not both)	Support Poor Readers	Solo or Paired (Teacher choice)	Pupil Marked Marksheet Provided	YES Making Choices Parsons
PREDICT	OPTIONAL ALL (predict or parsons not both)	Support Poor Readers	Paired	Pupil Marked Marksheet Provided	NO
FLOW	OPTIONAL ALL	Support Poor Readers	Solo or Paired (Teacher Choice)	Pupil Marked Marksheet Provided	NO
INVESTIGATE	CORE	Support Poor Readers	Paired	Pupil Marked Marksheet Provided	YES Making Choices
CHANGE	CORE	Support Poor Readers	Paired	Pupil Marked Marksheet Provided	YES Making Choices
CREATE	CORE	Assesses Pupil Work and Checks Pupil Self-Assessment	Solo	Pupil Assessed & Teacher Assessed	YES Making Choices

Core activities general instructions

1. Group pupils in roughly same ability pairs. For **investigate** and **change** worksheets, pupils will work in pairs, for **create** they will work separately.
2. Give out the pupil booklets and explain that pupils need to follow the instructions on the sheets to explore how **count-controlled loops** work.
3. Explain that each pupil will record separately whilst working alongside their partner and keeping to the same pace as their partner.
4. Demonstrate where they can find the template code and explain that pupils will share one device for investigate and change.
5. Explain that during each question, only one person should touch the shared device and they should swap who that person is when there is a new question.
6. Encourage them to discuss their answers with their partner. If they disagree with their partner, they can record a different answer in their own booklet.
7. Show pupils where it says they should mark their work on the sheet where the answer sheets are in the classroom.

8. Remind pupils to return marksheets after marking, because there are not enough for every pair to have their own.

Key programming knowledge

A condition is a state we can check to see if it is true or false

Conditions starts with an if

Conditions are only checked once unless they are in a loop

Conditions lead to two possible pathways True and False

Conditions are only checked when reached in the flow of control

An algorithm is any set of instructions to carry out a task that can be understood by another human

Decomposition is breaking up a project into parts to solve separately



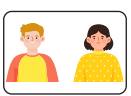

Resources

Making Choices

<https://scratch.mit.edu/projects/343948560/editor/>

Making choices PARSONS

<https://scratch.mit.edu/projects/329426702/editor/>

	On the sheet, if it says no Scratch, they must work only on the sheet.
	If it says Scratch with a green tick, they can use one device between the pair.
	If it says work with a partner, they must work at the same speed as their partner.
	If it says work on their own, they must do this using a separate device each working alone.

Scottish Curriculum for Excellence Technologies

I understand the instructions of a visual programming language and can predict the outcome of a program written using the language. TCH 1-14a

I can explain core programming language concepts in appropriate technical language TCH 2-14a

I can demonstrate a range of basic problem solving skills by building simple programs to carry out a given task, using an appropriate language. TCH 1-15a

I can create, develop and evaluate computing solutions in response to a design challenge. TCH 2-15a

English Computing National Curriculum Programs of Study Pupils should be taught to:

- **design, write and debug programs that accomplish specific goals**, including controlling or simulating physical systems; **solve problems by decomposing them into smaller parts**.
- **use sequence, selection and repetition in programs**; work with variables and various forms of input and output.
- **use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs**.

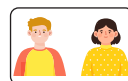
Welsh National Curriculum Relevant Strands

Progression Step 3.

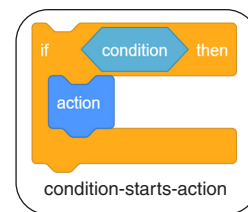
- I can use conditional statements to add control and decision-making to algorithms.
- I can explain and debug algorithms.

MAKING CHOICES PARSONS

Start Scratch and Load
Making Choices PARSONS



Work with a partner



Use the algorithm below to help you connect the Scratch blocks in the correct places in Sprite A.

Start the quiz when the a key is pressed

Show the sprite on the screen

Say a welcome

Ask the question $20+10=?$

If the answer is the same as 30 then

 Say correct

Ask the question $13-7=?$

If the answer is the same as 6 then

 Say well done etc

Ask the question $50\times 3=?$

If the answer is the same as 150 then

 Say Great answer

Pause for 3 seconds

Hide the sprite

Run the code and check your answer using the PARSONS ANSWERS sheet

PARSONS teacher help notes

Making Choices

Notes on the activity

Parsons problems were originally designed for university students to make sense of text-based programming by ordering chunks of code rather than writing all the code out. It was designed as a scaffold to reduce the time spent in the minute detail of code and concentrate on the logical order needed to solve a problem.

This example makes pupils think about how an algorithm can be written differently than the code, as it uses similar language to the code but not always the same. An algorithm is designed for another human to understand and can be created with a wide variety of commands and symbols. A machine can only follow the precise code language that the blocks are written in.

If pupils still need time to develop their code-connecting skills, it can be a good activity to start with. It is popular with pupils as it is a hands-on activity. If this is the case, you might ask pupils to complete this in pairs, but both having a device to do this individually.

Alternatively, it could be an option you only give to pupils who have struggled in other modules to help them familiarize themselves with the code first before moving on to investigating and modifying code.

Whole class advice

Remember an algorithm plan does not have to be written in code so the algorithm will not be the same as the code in every line.

Start the quiz when the a key is pressed

Show the sprite on the screen

Say a welcome

Ask the question $20+10=?$

If the answer is the same as 30 then

Say correct

Ask the question $13-7=?$

If the answer is the same as 6 then

Say well done etc

Ask the question $50 \times 3=?$

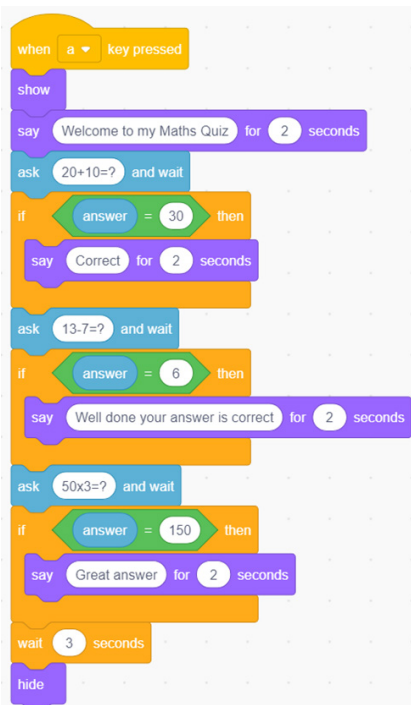
If the answer is the same as 150 then

Say Great answer

Pause for 3 seconds

Hide the sprite

Run the code and check your answer using the answers sheet



Send able advice

Use something to block all of the algorithm but the line pupils are working on, thus revealing one line of the algorithm at a time.

This makes it easier to concentrate on the immediate task rather than be overwhelmed by all of the algorithm.

If supporting pupils one to one, you could also get them to work with just three bits of code, one of which must be the code they need.

You could also part-build some of the code, so there are even fewer sections, as shown below.

You can use this version
<https://scratch.mit.edu/projects/578541554/editor/>

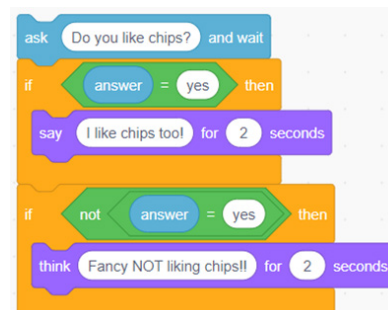
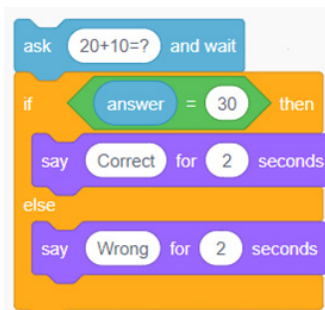
Individual advice

The first word in every line is the most important and will often give you a good clue as to which block you need.

Don't forget to run your code to check that it works before moving on.

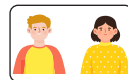
Understanding programming

The downside of Parsons problems is that if they are the only activity that pupils use in coding, they can encourage pupils to believe that there is only one right way to program a quiz question, whereas there are many ways to ask and check a quiz question. A couple of these have been coded on the right.

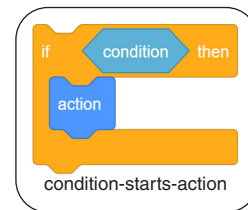


Making choices

PREDICT



Work with a partner



Draw a line to connect the code description to the correct code

```
ask 50x3=? and wait
if answer = 150 then
  say Great answer for 2 seconds
```

```
ask How old are you? Type years only and wait
if answer = 8 then
  say I am older than you! for 4 seconds
if answer = 9 then
  say Hey you are 9 like me. for 4 seconds
if answer = 10 then
  say You are older than me. for 4 seconds
if answer = 11 then
  say My brother is 11. for 4 seconds
```

The user is asked a question, and if their answer is a yes or a no the sprite changes costumes as well as talking to them.

The user is asked a question and if their answer is the same as 150, they are told, great answer.

The user types a number in and that number is checked four times to see if it is the same as an 8, 9, 10 or 11. The program talks to the user if you type in one of these numbers only.

Code Description

```
ask Do you like my blue hair? Type Yes or No? and wait
if answer = Yes then
  switch costume to laugh
  say We can be best friends :) for 2 seconds
if answer = No then
  switch costume to cross
  say You have made me sad! for 2 seconds
```

PREDICT TEACHER HELP NOTES MAKING CHOICES

Whole class advice

Make sure you work with your partner on this sheet. Take it in turns to read the code and definitions out loud to each other.

Notes on the activity

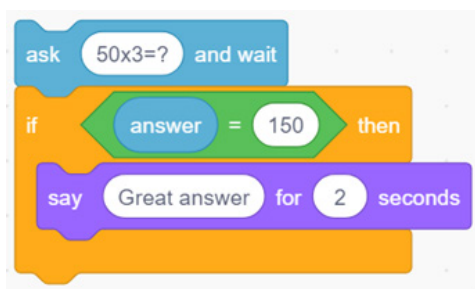
Prediction helps pupils to think about the bigger purpose of the code before they run the code and experience what it actually does. It is carried out away from their digital devices.

Individual advice

Instead of reading equals read the same as.
Instead of reading answer read the answer the user gave.
Demonstrate this by reading one with the pupil
For example,
Ask $50 \times 3 = ?$
If the answer the user typed in is the same as 150,
Say, great answer.

Send able advice

Cover up the two longer bits of code with an exercise book or paper. Read the shortest code and circle the key facts. Now ask your pupil if they can spot those facts in the definitions as you read them one by one. Now cover up two more sections of code and repeat the process. In doing so, you are reducing the amount of visual information the pupil has to process.



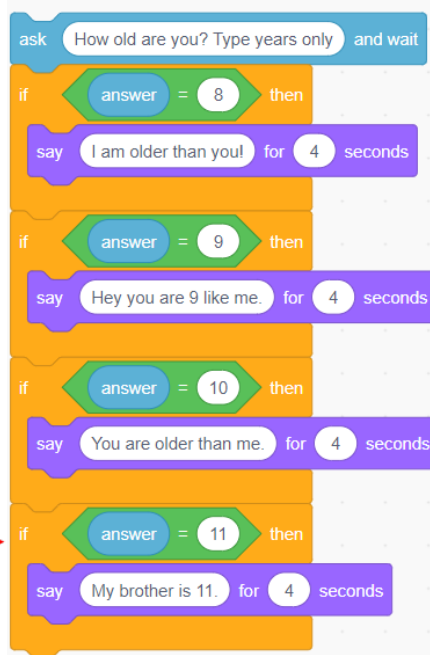
The user is asked a question, and if their answer is a yes or a no the sprite changes costumes as well as talking to them.

The user is asked a question, and if their answer is the same as 150 they are told, great answer

The user types a number in and that number is checked four times to see if it is the same as an 8, 9, 10 or 11. The program talks to the user if you type in one of these numbers only.

Individual advice

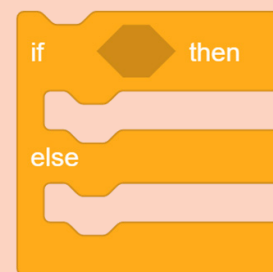
Read the definitions one by one slowly.
Underline any key words.
Are those same key words in the code?



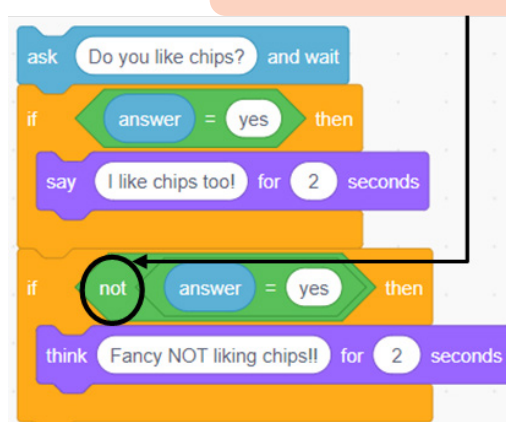
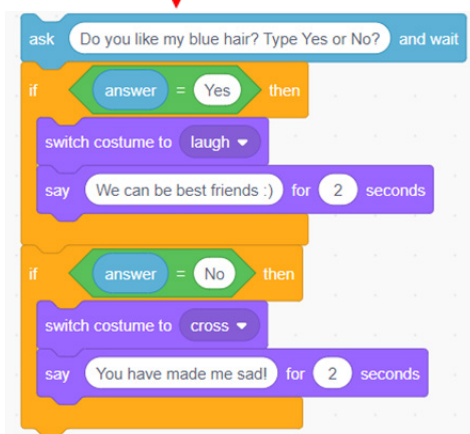
Understanding programming

While these examples are appropriate for our upper KS2 pupil, they are all incomplete, as they don't have any method of catching answers that are outside the conditions.

One simple way to do this would be with a condition-switches-between-actions block. Any code within the second else section would be run if the condition is NOT true.



Another way would be the use of the NOT block as shown below.



Teaching Primary Programming with Scratch

Pupil Book – Year 5

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

Copyright © Phil Bagge 2023

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

All rights reserved. No reproduction, copy or transmission of this publication may be made without written permission.

Except for the quotation of short passages for the purposes of research or private study, or criticism and review, no part of this publication may be reproduced, stored in a retrieval system, copied or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, now known or hereafter invented, save with written permission or in accordance with the provisions of the Copyright, Design and Patents Act 1988, or under terms of any licence permitting limited copying issued by the publisher.

This book is sold subject to the condition that it shall not, by way of trade or otherwise, be lent, resold, hired out, or otherwise circulated without the publisher's prior consent in any form of binding or cover other than that in which it is published and without a similar condition including this condition being imposed on the subsequent purchaser.

Any person who does any unauthorised act in relation to this publication may be liable to criminal prosecution and civil claims for damages.

ISBN 978-1-91505-4-265

CONTENTS

Introduction & Progression

Introduction	6
--------------	---

Introducing New Concepts

1 Condition-Starts-Actions	12
2 Condition-Starts-Action-In-a-Loop	16
3 Condition-Switches-Between-Actions	20
4 Condition-Ends-Loop	24

Programming Modules that use Condition-Starts-Action

5 Making Choices	28
------------------	----

Programming Modules that use Condition-Switches-Between-Actions

6 Wizards Choice Two	54
----------------------	----

Programming Modules that use Condition Checked in a loop

7 Butterfly Fun	74
8 Ocean Pollution	94

If you can only pick two modules due to time constraints, I would choose either **Making Choices** or **Wizards Choice Two** and either **Butterfly Fun** or **Ocean Pollution**.

INTRODUCTION

Scheme

This book is a complete scheme of work for teaching primary programming using Scratch in Year 5 for 9–10 year olds.

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 4

Teaching Primary Programming with Scratch, Year 4

If you are interested in the methodology and research-informed practice behind this series, as well as a wealth of other insights gained from teaching block-based programming for thousands of hours then

Teaching Primary Programming with Scratch – Research-Informed Approaches will be an informative read.

Permissions

It includes permission to photocopy the pupil and teacher help sheets for your class and school.

It includes links to example code, project templates and slides to introduce new programming concepts.

Progression

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

Pedagogy 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.

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.

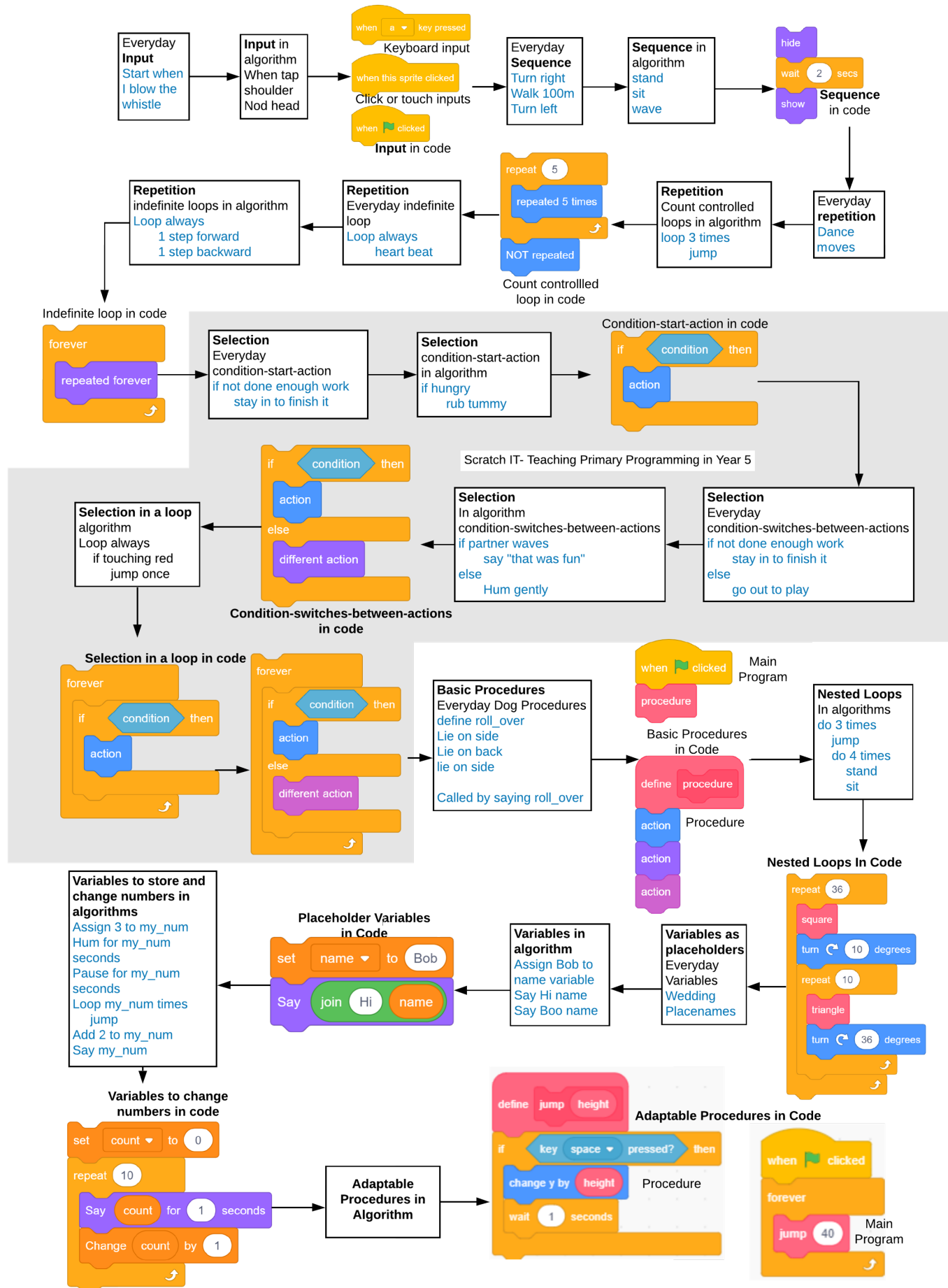
Predict

Run

Investigate

Modify (change)

Make



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.

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 4 on loops and prior modules in Year 5.

Assessment

Summative Assessment

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

If you have used earlier versions of these resources on the code-it website, then you will enjoy the new project assessment grid that combines pupil's self-assessment and quick teacher assessment ideally within the lesson.

Self-Assessment

Pupils self-mark to help them see how they have progressed, reducing teachers' workload and enabling teachers to concentrate on the pupils that might 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.

Yellow highlighted hints and tips are whole class suggestions

Lilac highlighted hints and tips are information to help teachers support SEN pupils.

Green highlighted hints and tips are suggestions to help the teacher support individual pupils stuck on a specific question.

Can We Start Here?

If pupils have never programmed with Scratch before a basic introduction project such as that provided in *Teaching Primary Programming with Scratch, Year 3* is a must.

I would also recommend a single module of count-controlled loops and one on indefinite loops found in

Teaching Primary Programming with Scratch, Year 4

Many of the projects in the book build on prior learning.

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 – Teaching Primary Programming in Year 4

Come Back Doggy!
INVESTIGATE

Start Scratch and load the **Come Back Doggy!** program

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

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

Look at the code inside the **Dog**
Dog sprite questions

- Which line of code makes the dog wait until Maria arrives?
- Which blocks get repeated 21 times?
- 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

70

photocopiable page

Photocopiable resource for pupils

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

- Which block starts the code?
Green flag (1 mark)
- What block makes Maria go back to the start? (Initialization)
HINT go to
go to x and y (1 mark)
- Which block rubs out any old lines before Maria searches for her dog? (Initialization)
Erase all (1 mark)
- In the first repeat loop (count-controlled loop), how many times will move 1 step be run?
250 (1 mark)
- Which loop draws the shortest line?
Repeat 100 or the second repeat loop (1 mark)
- Which block changes Maria's direction?
Point in direction (1 mark)

Look at the code inside the **Dog**
Dog Sprite Questions

- Which line of code makes the dog wait until Maria arrives?
Wait until touching Maria (1 mark)
- Which blocks get repeated 21 times?
Next costume (1 mark) wait 0.4 seconds (1 mark)
- 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 is 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.

71

Teacher Hints & Tips on the same photocopiable resource

WE ARE LEARNING ABOUT CONDITIONAL SELECTION IN ALGORITHMS AND PROGRAMMING

Condition-starts-action algorithm

A condition is a state we can check to see if it is true or false

Conditional Selection Vocabulary

condition, true false, selection, choice, pathways, equal

Conditions

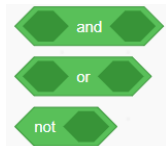
Start with an if

Only checked once unless they are in a loop

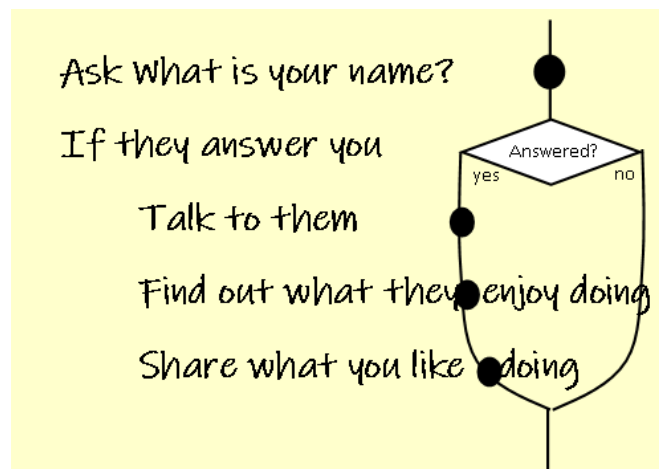
Two possible pathways: True and False

Are only checked when reached in flow of control

Extension Conditions can be combined with AND & OR
Reversed with NOT



Making friends Algorithm



Condition-within-infinite-loop

Decomposition

Breaking up a project into parts to solve separately.

Algorithms

A set of instructions or rules to do something.

We are indenting to show what actions are started if a condition is true

