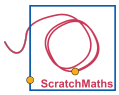
UCL ScratchMaths Curriculum

Module 4:
Building with Numbers

Teacher Materials

Developed by the ScratchMaths team at the UCL Knowledge Lab, London, England

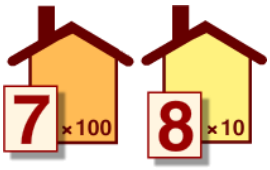


Module 4: Building with Numbers

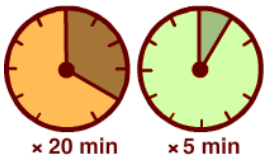
Investigation 1
Place Value Models

Investigation 2
Timers and Stopwatches

Investigation 3
The Conversion Game



Investigation 4
Exploring Conversions

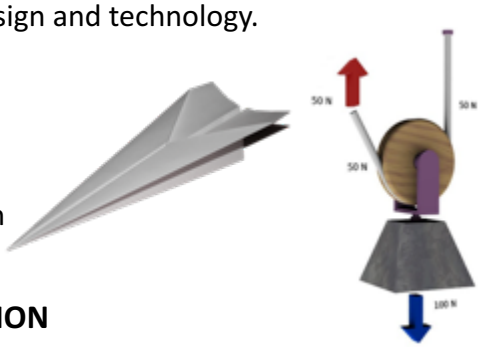


Introduction to Module 4

Module 4 is focused around exploring place value and requires pupils to use their knowledge of broadcasting to build place value models within several different contexts. This module could potentially be linked with several different areas of the Key Stage 2 curriculum beyond mathematics and computing such as science as well as design and technology.

Science: Forces

The last investigation of this module explores conversions which include distance, mass and time. This could be linked to the measurements within pupil experiments with different forces as part of the science curriculum.



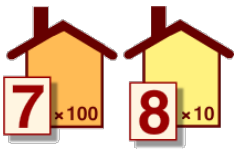
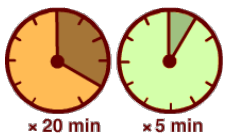
Design and Technology: Cooking and Nutrition

The weight conversion game in investigation 4 could also be linked to the cooking and nutrition aspect of the design and technology curriculum. The game could be further adapted to allow pupils to convert between imperial and metric weight measures within recipes.

Key Vocabulary and Concepts Covered by Module 4

Scratch	Computing	Mathematics
▶ Costume # block ▶ … + …, … - …, … = … blocks ▶ Stop all block ▶ … of … block	▶ Broadcasting ▶ Conditions ▶ Repetition, Sequence and Selection ▶ Parallel behaviours ▶ Events ▶ Expressions ▶ Definitions ▶ Logical Reasoning ▶ Decomposition	▶ Place value ▶ Addition, subtraction ▶ Random numbers ▶ Coordinates ▶ Sequences ▶ Time, Weight, Length ▶ Angles ▶ Estimations ▶ Conversions ▶ Fractions

Map Of Module 4

	Activity 1	Activity 2	Activity 3	Activity 4
Investigation 1 **Place Value Models** 	**Digits Up, Digits Down** Starter project: **4-Dials Up**	**Unplugged: Flip Flip Nudge Nudge**	**Playing with Place Value** Continue with: **4-Digits Up**	**Sequences** Starter project: **4-Sequences**
Investigation 2 **Timers and Stopwatches** 	**Build a Stopwatch** Starter project: **4-Stopwatch**	**Unplugged: Nudge Nudge Get Get**	**Countdown Conundrum** Starter project: **4-Timer**	**Dizzy Dials** Starter project: **4-Dials**
Investigation 3 **The Conversion Game** 	**Unplugged: Playing the Conversion Game**	**Building Conv. Game: The Display** Continue with: **4-Conversion Game**	**Building Conv. Game: Conversion Houses** Continue with: **4-Conversion Game** or start with: **4-Conversion Game INT1**	**Building Conv. Game: Record Keeping** Continue with: **4-Conversion Game** or start with: **4-Conversion Game INT2**
Investigation 4 **Exploring Conversions** 	**Converting Length** Starter project: **4-Converting Length**	**Converting Mass** Starter project: **4-Converting Mass**	**Converting Time** Starter project: **4-Converting Time**	

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

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 4 Connections to KS2 Computing Curriculum

Curriculum Objectives	Link with ScratchMaths
Design, write and debug programs that accomplish specific goals	▶ Pupils are required to design, build and debug programs within Scratch that perform goals such as representing a place value model, counting up and counting down in hours/minutes/seconds as well as converting between different units of measure.
Solve problems by decomposing them into smaller parts	▶ Pupils are encouraged to solve the problem of building a place value model within different contexts by breaking down the model into smaller parts and thinking about, for instance, the relationship between the **ones** digit and the **tens** digit, then the **tens** digit and the **hundreds** digit etc. They then apply a similar approach to a different context such as time and identifying where modifications need to be made.
Use sequence, selection, and repetition in programs	▶ Pupils are encouraged to consider the sequence of blocks within their scripts when moving from building a stopwatch to a countdown timer. When counting down it is necessary for a digit to checks its own value is not 0 before "borrowing" from the next digit to ensure it is possible to continue counting down and therefore the sequence of blocks is crucial at this stage. ▶ Pupils use selection to determine at which point to "carry" or "'borrow" within their place value models and within the later extension activities to select different behaviours based on the condition of touching a specific colour. ▶ Pupils use different types of repetitions, including repeating a specific number of times to change the value of their place value models and also in the later extension activities using forever to continuously check wherever a specific condition is true.
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 outcome of their place value models, for instance when working with a single digit reasoning what will happen when 7 is added to 9. Also in later extension activities pupils are required to logically reason the input number based a given the digits and colour (representing x1, x10 or x100) record output.

MODULE 4: INVESTIGATION 1

Place Value Models

"Place Value Models" provides the foundation to the module as computing building blocks are used in a variety of contexts to explain and explore the mathematical concept of place value. Pupils build a two digit Scratch counting model using a **ones** and **tens** sprites which represent place value digits. Each sprite wears 0-9 as its costumes. Pupils begin to develop their understanding of variable, by using a reporter block ('indirect variable') **costume #** to trigger the broadcast "carry" behaviour.

Pupils build simple linear sequences and explore, explain and envisage the impact on the place value digits as the sequence increases. Reinforcing the "carry" or "exchange" process practically through pupils modelling using costume cards (0-9) and broadcasting to each other is encouraged throughout the module. The model can be extended to include additional place values and decimals.

Note that the costumes are intentionally ordered as 1, 2, 3… 9, 0, so that 1 (the costume which looks like 1 and is named 1) is the first costume; 2 (the costume which looks like 2 and is named 2) is the second costume… etc. and 0 (the costume which looks like 0 and is named 0) is the tenth costume. When the digit sprite wears its tenth costume, i.e. it looks like 0, the reporter block **costume #** will report (output) 10 as its value. However, when switching costumes using the **switch costume to …** block, the names of the costumes are used. Thanks to the order of the costumes and their names it is natural and intuitive to say **switch costume to 1** (to make the sprite look like 1), **switch costume to 2** (to make the sprite look like 2)… and **switch costume to 0** (to make the sprite look like 0).

- **Activity 4.1.1** – Digits Up, Digits Down
- **Activity 4.1.2** – Unplugged: Flip Flip Nudge Nudge
- **Activity 4.1.3** – Playing with Place Value
- **Ext. Activity 4.1.4** – Sequences

Scratch projects **4-Digits Up** **4-Digits Up FINAL** **4-Sequences** **4-Sequences FINAL; 4-Sequences Extensions FINAL**

LINKS TO PRIMARY NATIONAL CURRICULUM

CURRICULUM OBJECTIVES	LINK WITH SCRATCHMATHS
Mathematics Solve number and practical problems involving place value Perform mental calculations, including with mixed operations Generate and describe linear number sequences (KS3) Work with experiments that involve randomness	▶ Pupils use their knowledge of place value to build their own counting model up to 4 digits ▶ Pupils perform mental calculations to envisage the next number in sequences involving different operations ▶ Pupils build scripts to that generate different sequences of numbers and to envisage the next number in the sequence ▶ Pupils build scripts that generate a sequence of random numbers within a set range

MODULE 4 • INVESTIGATION 1 • ACTIVITY 4.1.1

Digits Up, Digits Down

LEARNING OBJECTIVES

Explore how to increase and decrease the sprite costume number to display different values.
Exchange ideas for changing the costume number in different positive and negative increments

ACTIVITY INSTRUCTIONS

Pupils open project **4-Digits Up**, **Save as a copy** (online) or **Save as** (offline) and rename.

1. Pupils explore the project, its **ones** sprite and its *setup script*. They explore and reset its costumes in the **Costumes tab**.
2. Pupils drag into the scripts area (keeping isolated) the **next costume** block and click it repeatedly. In the **Looks** group they click the check box of the **costume #** **monitor** and observe the **reporter window** with the costume's number in the stage.
3. Pupils add the **when this sprite clicked** hat block and explore the script.
4. Pupils choose one of the following tasks to explore and then report back to the class. They modify the previous script so that when **ones** is clicked it will:
   - increase its value by 3
   - increase its value by 7 (use **repeat** to increase in steps with short **wait**)
   - set its value to 5
   - set its value to a random value

continues on the next page

MATHEMATICS CONNECTIONS

Show images or objects of digital and analogue clocks or timers/ counters. E.g. the police might use a small device to count people at an event. Discuss: *Where else might you see or use a counter like this?*

The sprite's costumes represent the digit cards for the decimal system or base 10.

Discuss: *What happens after digit 9? Can we show 10, why not?*

When pupils have created a different increment or behaviour for the **ones** sprite they should be encouraged to envisage what will happen before the sprite is clicked.

Pupils can play with a **simple envisaging game** in pairs or with the teacher. Set up a behaviour of the sprite so that when clicked it increases (or after step 5 on the following page also decreases) in steps, e.g. add 3. The starting costume is set to a number, e.g. 4, – a question posed: *What's next?* and the pupil responds and the sprite clicked. This algorithm is then repeated. i.e. 4, 7, 0….

The next one is harder – 8 etc. Speeding up the game if the teacher is playing and pupils chanting quickly becomes exciting.

...continued

5 Pupils start using **switch costume to ...** in combination with the **costume #** block. They choose one of the following tasks, then report back. They will use **Operators** blocks for adding or subtracting, so that when **ones** is clicked its value will:

- change by adding 2
- change by subtracting 1
- change by subtracting 7.

Discuss: the connection between costume # and the displayed digit. E.g. *What would* **costume #** *15 be*? *What about costume 21 or -1?* (This can be demonstrated or explored within the script.)

Unplugged: you can play the envisaging games on paper using the worksheet on page 9. Pupils can choose their own starting digit and then values to add and note down the predicted sequence in Scratch. Then encourage them to build this sequence in Scratch to check.

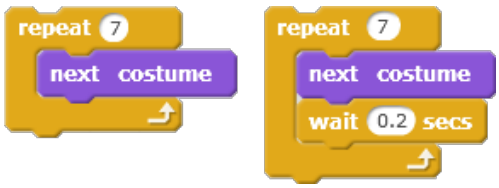
CONNECTIONS TO Y5 SCRATCHMATHS

*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

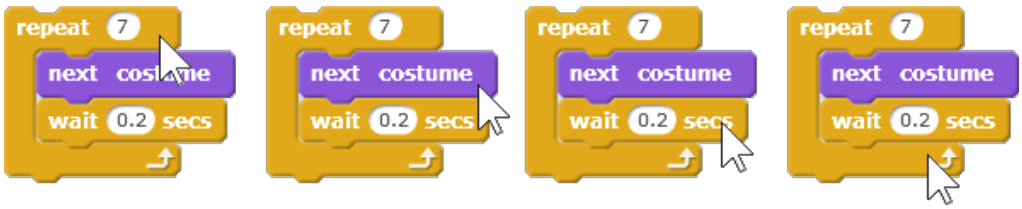
1 In all ScratchMaths *starter projects* there is a **when green flag clicked** script already built – called *setup script*. Pupils will use it to reset the stage and sprite(s) into the "initial state". We recommend to keep it as it is, or – when appropriate – modify some of its inputs etc.

2 We started using **multiple costumes** of sprites in Module 1. In activity 1.2.3 we used **next costume** block, in 1.3.3 also **switch costume to …** We used **next costume** for animations in Module 3.

4 We started using the **repeat** control block in Module 1, activity 1.2.1. In Module 3 we used it for making **Tera** jump and float slowly and smoothly, for making **Nano** disappear and reappear smoothly etc. Sometimes we use **wait** block inside **repeat** to make the change (process) slower.



To run any script (even if not completely finished yet) in the scripts area we simply click it. Whichever block we click in a script **the whole script is always run as a whole**, e.g. see the examples below, the whole script will be run for each case.



5

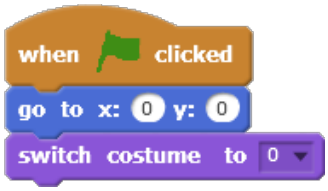
These are different **reporter blocks**. Each of them (a) **represents a value**, (b) **has rounded ends** and (c) can be inserted into a hole of another block **as its input**. We have already used three reporter blocks in Modules 2 and 3 (see upper three on the left).

In this module we are using three new reporters – to report the sprite's **costume** number, and operators **… + …** and **… - …**.

ADDITIONAL SUPPORT

*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

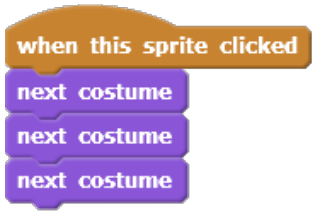
1 Sprite **ones** has 10 costumes: first of them (costume number 1) is a card with big red digit 1. The name of this costume is 1, etc. However, the last costume, i.e. the 10th costume looks like 0 and the name of that costume is 0.

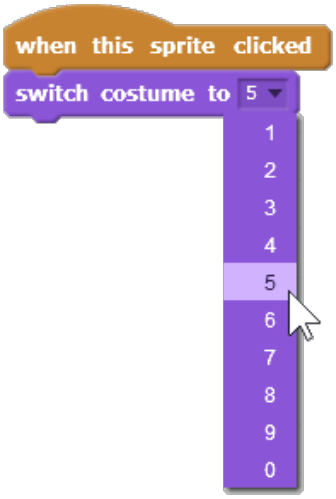


2 **Reporter blocks** like **costume #** or **x position** have check boxes next to them, click the one next to the **costume #** to get a small **monitor** displayed in the stage. However, using the reporter blocks directly in the scripts area – either isolated or inside other blocks – is essential for programming.

3

4 Small tasks exploring different **when this sprite clicked** behaviours:



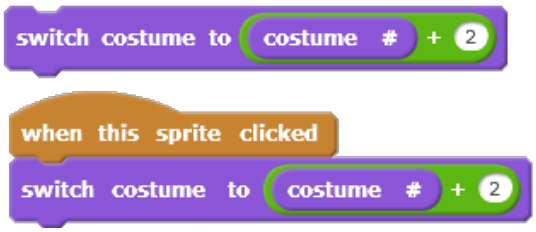


ADDITIONAL SUPPORT CONTINUED

5 **Operators** are used to compute values – either isolated for direct drive explorations in the scripts area, or inserted in other blocks to build more complex behaviours. This is how we now start using **costume #** in an expression and in **switch costume to …**

6

8



On the left, there is an example sequence of steps to take to solve the *change by adding 2* task.

On the right there are the example solutions for the *change by subtracting 1* and *change by subtracting 7* tasks. Note that the first of these scripts will be repeatedly used in later investigations – therefore we will later build a new block **previous costume** with exactly this script as its definition.

PRINT

INVESTIGATION 1
Activity 4.1.1

17

NAME

WHAT TO DO

Fill in the number that would be generated each time you add the **value to add** to the **initial value** and keep doing this until you get back to the same initial value . Try with your own values. Remember you only have one digit.

initial value: 2 | value to add: 3 | sequence:

initial value: | value to add: 7 | sequence:

initial value: | value to add: | sequence:

initial value: | value to add: | sequence:

initial value: | value to add: | sequence:

initial value: | value to add: | sequence:

Module 4 • Investigation 1 • Activity 4.1.2

Unplugged: Flip Flip Nudge Nudge

Learning Objectives

Envisage the process of "carrying" from ones to tens (as well as to hundreds and thousands).

Activity Instructions

- Use the number cards provided on the following pages to create four flip books of numbers (e.g. by printing out each number on a separate piece of card and spiral binding), each ranging from 0-9.
- Choose four pupils to stand at the front.
- Give each flip book to one of the pupils, representing **ones**, **tens**, **hundreds** and **thousands**.

  - All pupils at the front should hold their flip books out in front of them.
  - Set the initial number (for example 0977) by changing each of the pupils flip books.
  - Tell the three pupils who are representing the **tens**, **hundreds** and **thousands** to position themselves so they can't see the number of the other pupils' flip books (other pupils in the class can be responsible for making sure there is no cheating!).
  - Have another pupil choose a number (<10) e.g. by rolling a die and instructing the **ones** pupil to increase their flip book by 1 that many times.
  - If the **ones** pupil gets to 0 they should nudge the **tens** pupil who must increase their flip book by one for every nudge they receive.
  - If the **tens** pupil gets to 0 they should nudge the **hundreds** pupil to increase their flipbook and the same applies between the **hundreds** and **thousands** pupils.
  - Keep repeating this process until you reach a target number e.g. 1000.

Repeat this activity to give the pupils at the front the chance to observe what was happening.

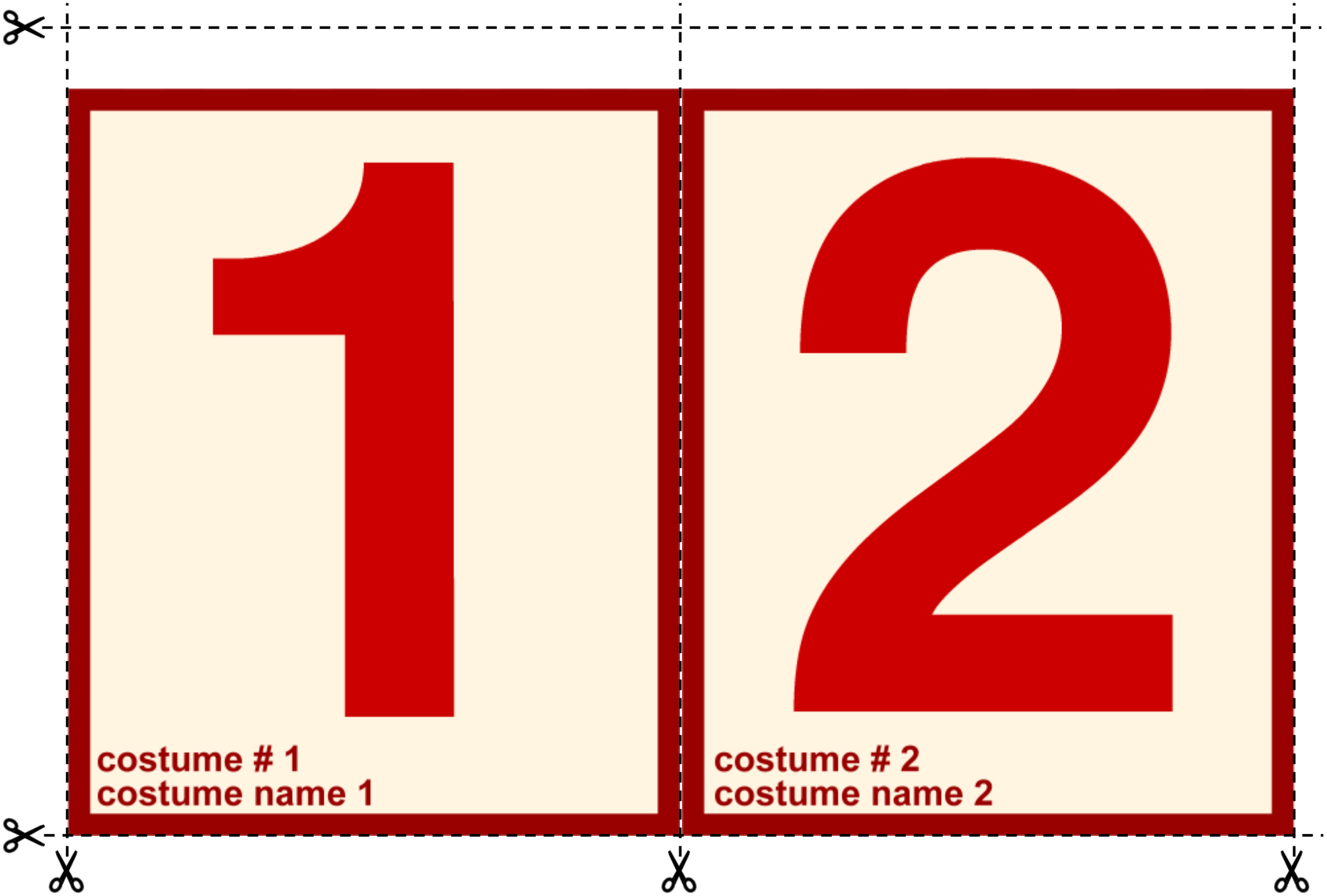
- **[Extension]** Once pupils have got the hang of it try adding larger numbers which require multiple nudges in one turn.

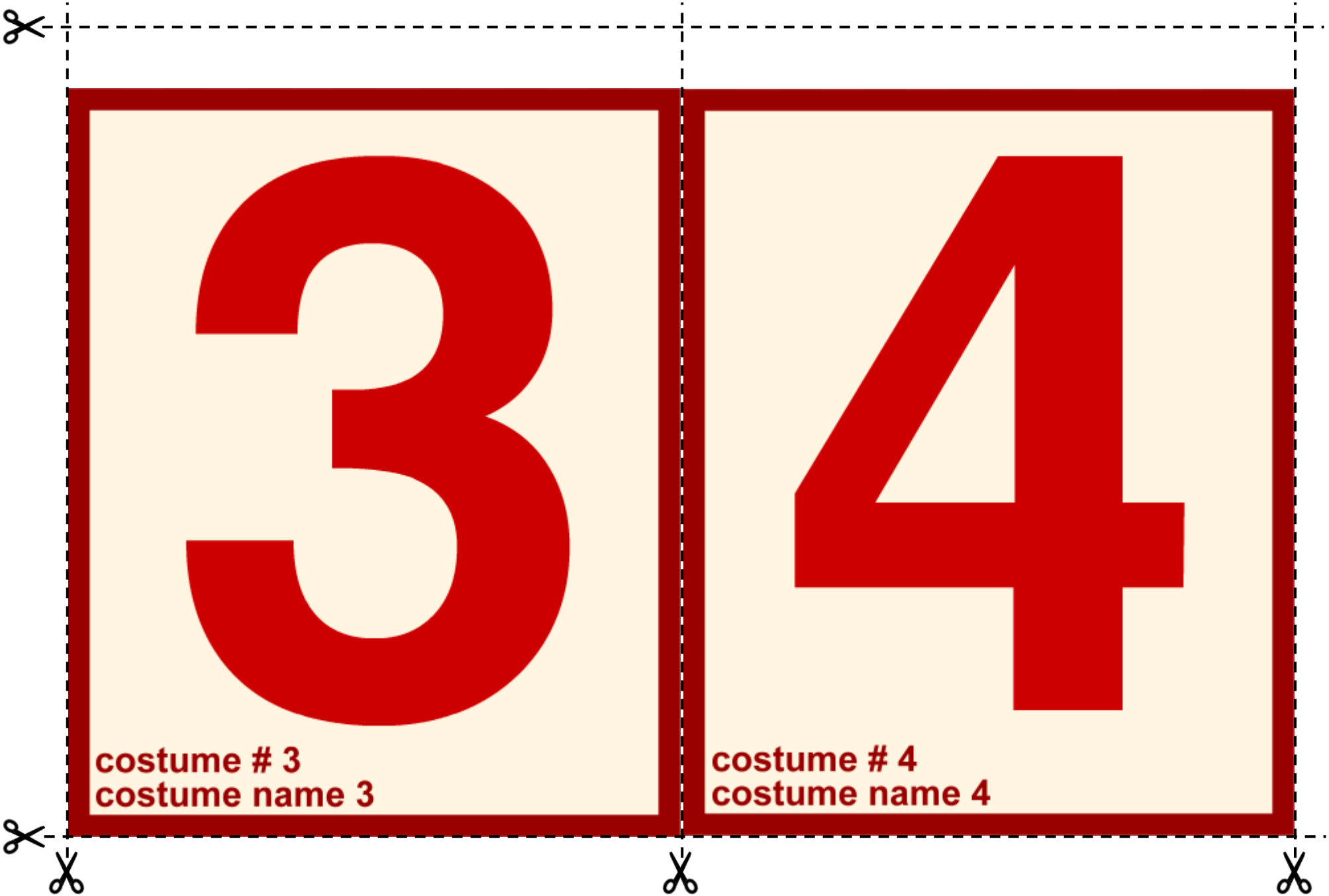
Mathematics Connections

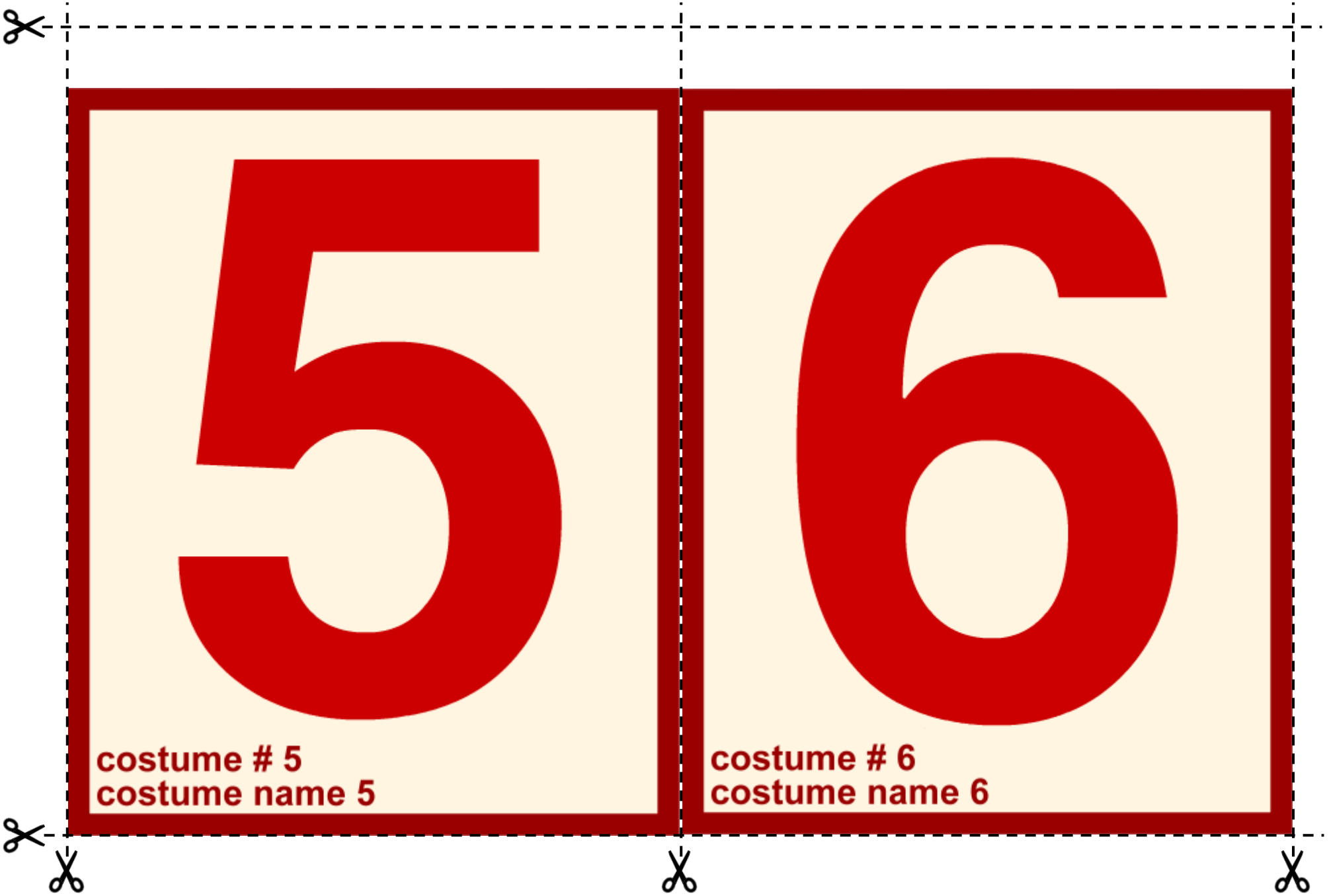
Connect the number card flip book with place value models seen during KS1 and KS2. Ask pupils what the value of each card is to connect with the place value model.

Discuss as a class. Ask pupils to explain what they have seen.

Discuss: *What's the smallest (or the largest) number of rolls of the die to get from 0977 to 1000? How can we calculate this?* [Largest will be the same as the subtraction 1000-0977]

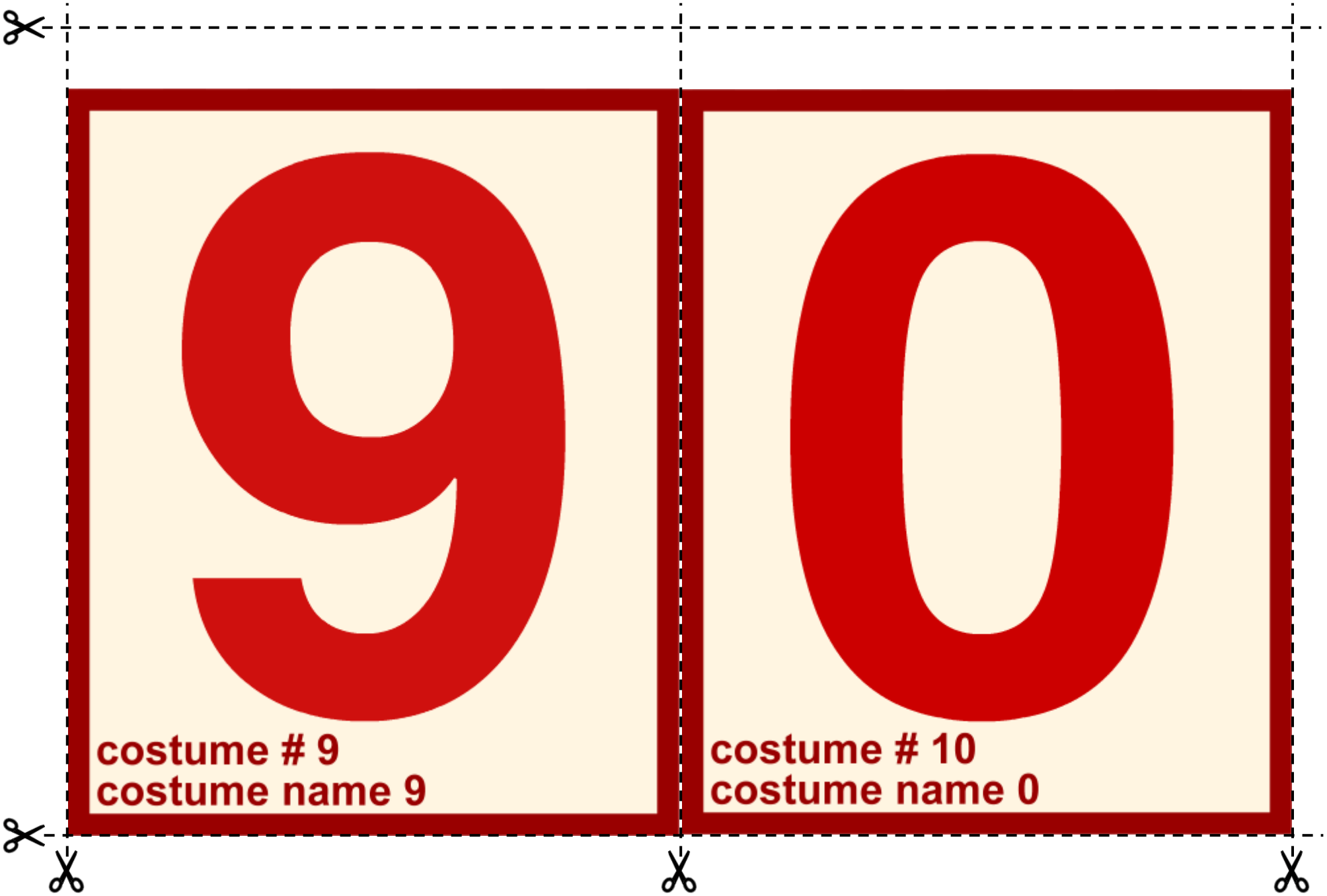

1
costume # 1
costume name 1
2
costume # 2
costume name 2

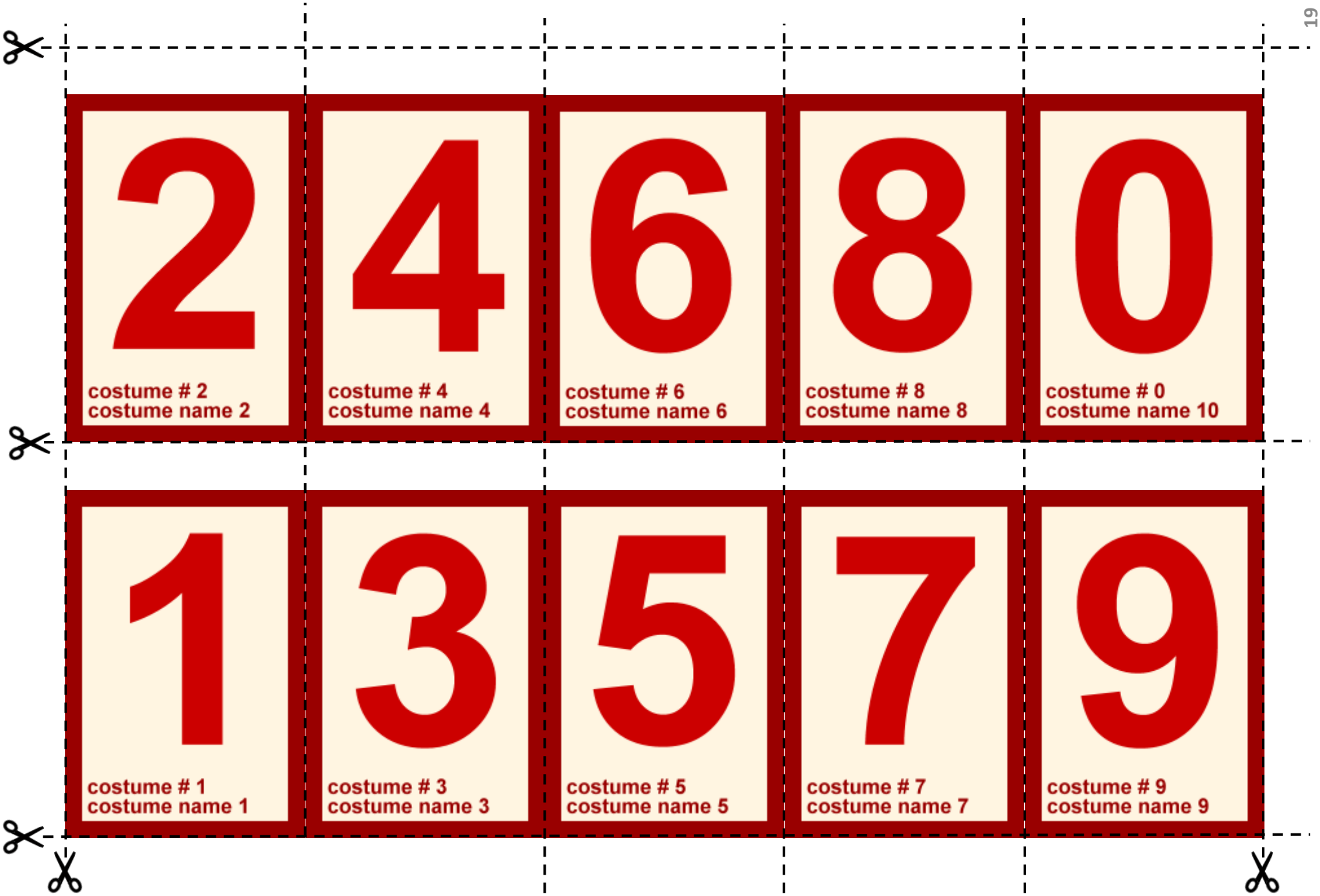

3
costume # 3
costume name 3
4
costume # 4
costume name 4


5
costume # 5
costume name 5
6
costume # 6
costume name 6


7
costume # 7
costume name 7


8
costume # 8
costume name 8


9
costume # 9
costume name 9
0
costume # 10
costume name 0


2
costume # 2
costume name 2
4
costume # 4
costume name 4
6
costume # 6
costume name 6
8
costume # 8
costume name 8
0
costume # 0
costume name 10
1
costume # 1
costume name 1
3
costume # 3
costume name 3
5
costume # 5
costume name 5
7
costume # 7
costume name 7
9
costume # 9
costume name 9

Module 4 • Investigation 1 • Activity 4.1.3

Playing with Place Value

Learning Objectives

Explore and build the place value model up to four digits.
Explain how to use four digit-sprites to count from 0 to 9999.

Activity Instructions

Pupils continue in their **4-Digits Up** project. *The final version of this project at the end of Activity 4.1.3 is* **4-Digits Up FINAL**.

1. Pupils switch to the second backdrop with four empty placeholders – called *4 digits*. They modify the initial position of **ones** in the *setup script* so that it sits atop the right most placeholder.
2. Pupils delete all scripts of **ones** other than its *setup script*. Then they duplicate the sprite and rename the new one to **tens**.
3. Pupils modify the *setup script* of **tens** so that it is positioned atop the appropriate placeholder. They repeat the same process and create the third and fourth sprites – renamed to **hundreds** and **thousands**, with similarly modified *setup scripts*.
4. For the **ones** sprite pupils build a simple **when this sprite clicked** **next costume** script. From the previous unplugged activity they already know that if **ones** runs **next costume** again and again and finally gets to 0 (its last costume), it must "nudge" the **tens**, e.g. by **broadcast** *add 10*.
5. **Tens** will react to this message by **next costume**. Pupils build **when I receive** *add 10* script for **tens**, then extend it to react by **broadcast** *add 100* when it gets to 0.
6. Pupils modify the initial costumes of the sprites so that they can explore situations like the following (by clicking **ones** only a few times):

7. Pupils build the proper scripts for **hundreds** and **thousands** sprites. They test their new **four digit 'number'**.

Mathematics Connections

Remind pupils that digits positioned in the **tens** place holder represent the **value** of the digit × 10.

Discuss: *Why do we need the second sprite if we want to count up to 99 instead of only up to 9?*

Discuss: *When we count up by clicking the **ones**, when exactly should **tens** increase its value, i.e. go to the* **next costume**?

Ask questions to connect number of 'clicks' with addition/subtraction:
How many clicks are needed if the sprites read 578 and we want 600?
The ones sprite was clicked 17 times to read 965, what was the starting number? Check your answer.

Discuss: *What would happen if the **hundreds** sprite reacted to add 10 message instead of reacting to add 100?*

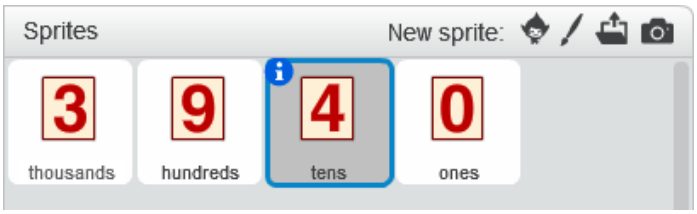
CONNECTIONS TO Y5 SCRATCHMATHS

*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

1 Completed scripts usually have **hat blocks** as their first block. Each hat block represents a certain **event**, like **when green flag clicked** or **when this sprite clicked**. All other blocks in a script define how to react if that particular event occurs.

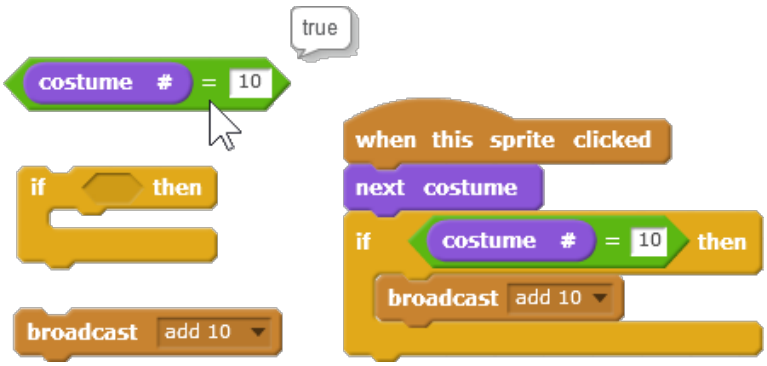
Note that a sprite may have more than one script with the same hat block. For example, when the sprite is clicked, all its **when this sprite clicked** scripts (i.e. all scripts with that hat block) are run in parallel.

2 In many projects there is more than one sprite. We started working with **multiple sprites** in Module 3. In Investigation 3.1 we started building individual behaviours for **Tera**, **Nano**, **Pico** and **Giga**. In Investigation 3.3 the sprites started to communicate and collaborate. To do so, they **broadcast and receive messages**, see below.



Note that each sprite has **its own scripts area**. If we want to read, modify or build a script or scripts of a sprite, we first have to select that sprite by clicking it in the sprites area. In the picture above, the **tens** sprite is currently selected.
Note also that when we click the green flag, all **when green flag clicked** scripts of all sprites are run in parallel.

4 In Module 3, Investigation 3.2 we started using **conditions**, special blocks reporting whether something is **True** or **False**. E.g. here we need to know whether the costume number equals 10.



In Activity 3.2.2 we started using condition as an input to the **if … then …** control block. It runs the block(s) inside it, if the condition is **True**. In this example, if the costume number is 10, the sprite will broadcast a message.

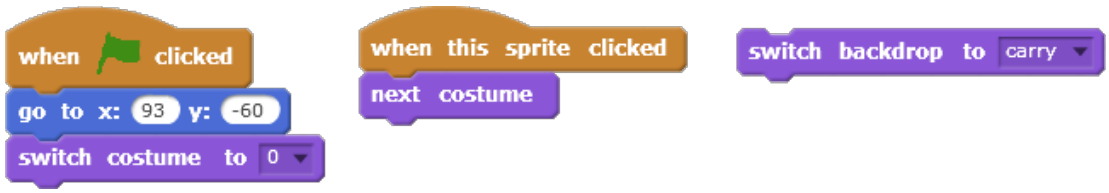
5

If there is a sprite that is ready to react to that particular *message* (i.e. "listening"), its **when I receive** *message* script will be run.

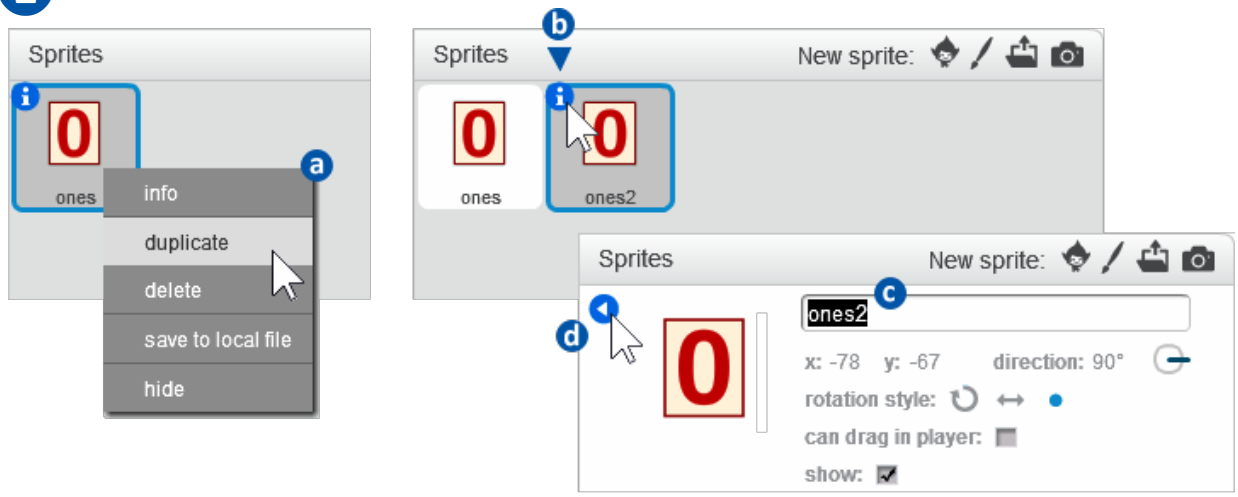
ADDITIONAL SUPPORT

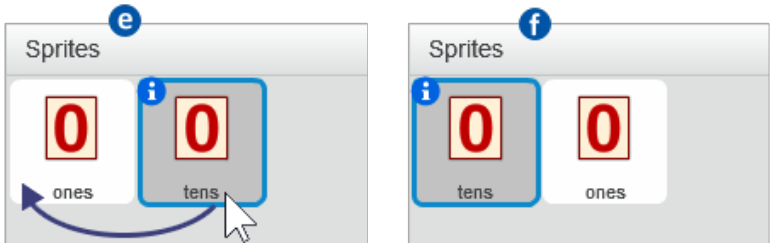
*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

1



2

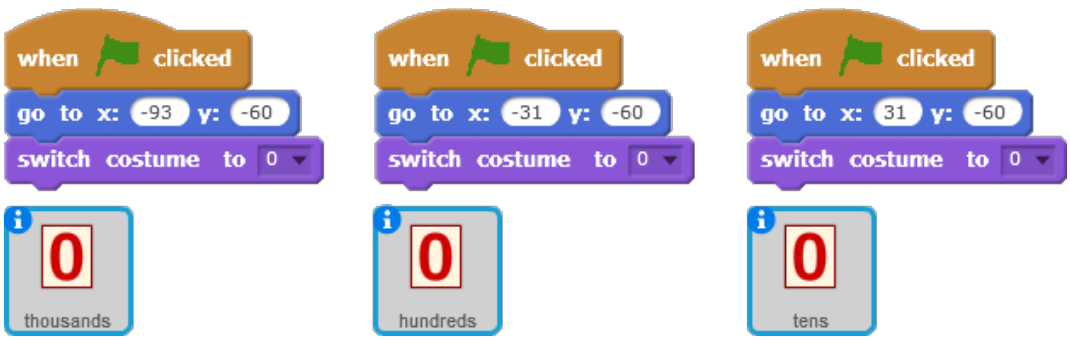




Note that at present (Scratch 2.0 V 447) swapping sprites by dragging in this way is possible only in the online version.

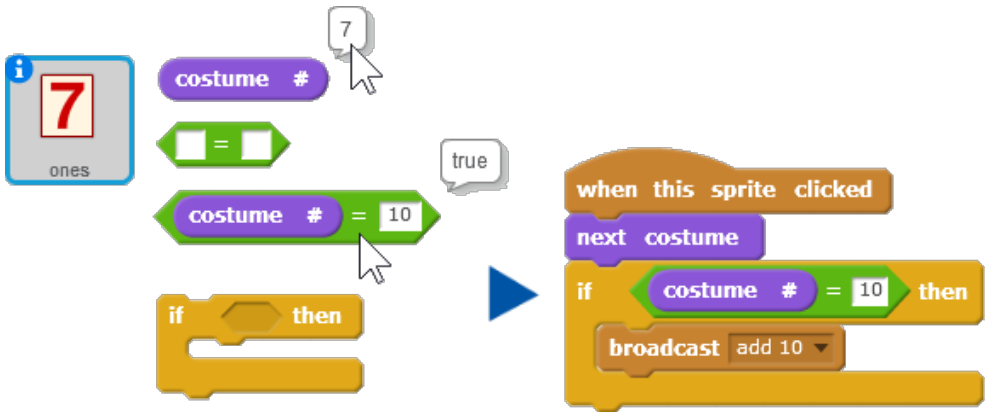
We duplicate sprites by **Shift + left click** then choosing the **duplicate** command from the menu (see **a**). Alternatively, there is a small **duplicate tool** above the green flag sign. Scratch will give its own name to the new sprite – **ones2**, see **b**. If we then click the small blue and white i (see **b**) a settings dialogue of that sprite is displayed. Here we can rename the sprite e.g. to **tens** (see **c**), then close this dialogue by clicking the small blue circle (see **d**). As far as in the stage **tens** will always sit to the left from **ones**, it will be better to swap their order (if possible) by dragging the one from right to left, see **e** and **f**.

3



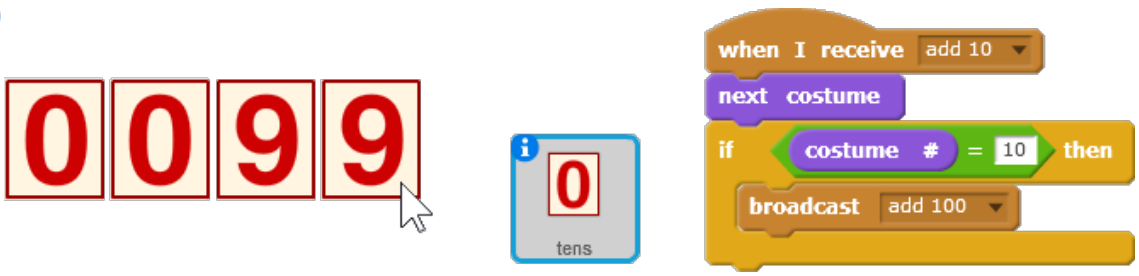
ADDITIONAL SUPPORT CONTINUED

4



Note that only the **ones** sprite reacts to **when this sprite clicked**. All other sprites will react only to **when I receive** messages.

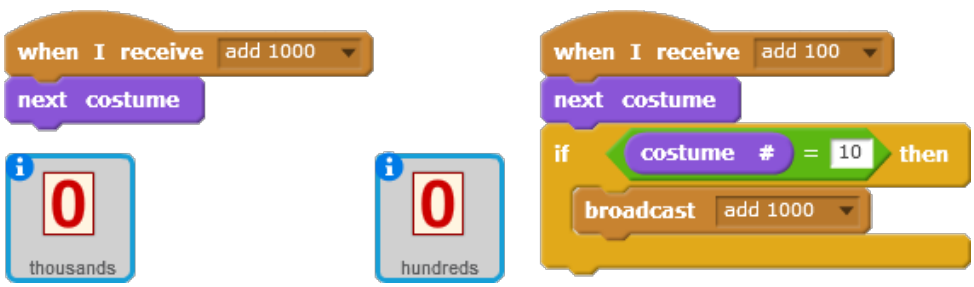
5



The **tens** sprite reacts to the *add 10* message by **next costume**. I.e. when **ones** shows 9 and we click it, it goes to 0 and broadcasts a message which will make **tens** go to its **next costume**.

However, when **tens** gets to 0 (which will happen when we click **ones** once more, see the picture above), it should broadcast *add 100*.

7



Discuss: *Why do we not need any if and broadcast blocks for the* **thousands** *sprite?*

MODULE 4 • INVESTIGATION 1 • ACTIVITY 4.1.4

[Extension] Sequences

LEARNING OBJECTIVES

Explore how to build different sequences of single digit numbers.
Explain how different starting numbers change the sequence i.e. for counting in 3s, 4s etc.

ACTIVITY INSTRUCTIONS

Pupils open project **4-Sequences**, **Save as a copy** (online) or **Save as** (offline) and rename. *The final versions of this project at the end of Activity 4.1.4 will be* **4-Sequences FINAL** *and* **4-Sequences Extensions FINAL**.

1. Pupils explore the behaviours of the two sprites in the project – **ones** and **input**.

We want to use the **input** sprite to add its actual value to **ones**, again and again. E.g. in the picture on the right, **input** will be increased by 8 to become 1. If we repeat that step, we will get a sequence of 3, 1, 9, 7, 5 and again 3…

2. For the **input** sprite pupils build an isolated block **broadcast** *add 1*. The **ones** sprite will react by going to the **next costume**.
3. Pupils add the **repeat** block around the **broadcast** with a short **wait** command, e.g. 0.2 sec. The **repeat** should run as many times as the actual **costume #** of **input** sprite currently is. Pupils then give a name to this script: they make a new block, e.g. **add input** and run it several times to generate the sequence of numbers (they may want to add **when** *space* **key pressed** hat block to it).
4. **[Extension]** Pupils build an alternative way to **add input** to **ones**: they will simply drag the **input** sprite onto **ones**. It will run **add input** then **glide** back 'home'.

Pupils build another **when green flag clicked** script for **input**: *whenever* (i.e. **forever** with the **if … then …** block inside) the sprite is **touching** *ones* it will **add input** and **glide** back.

5. **[Extension]** When the **input** sprite is 0, the **add input** block adds 1 to **ones** 10 times. *Why? How can we avoid this happening?*

MATHEMATICS CONNECTIONS

This activity is an extension. As an alternative pupils can explore the sequences using the worksheet from 4.1.1 along with the final version of the project.

A **term** is defined as any number in a sequence.

Pupils explore different sequences by clicking on both the **ones** and **input** sprite to select the starting term (number) and then dragging the **input** sprite over the **ones** sprite repeatedly.

Pupils can share sequences with each other or to the teacher.

Ask questions:
Write down the next three terms in this sequence 3, 7, 1, 5… ?
For the sequence 0, 3, 6, 9, 2, 5, 8, 1, 4, 7 what value was added, what was the initial value? What was your strategy for working out the next number when the sequence goes past 0?

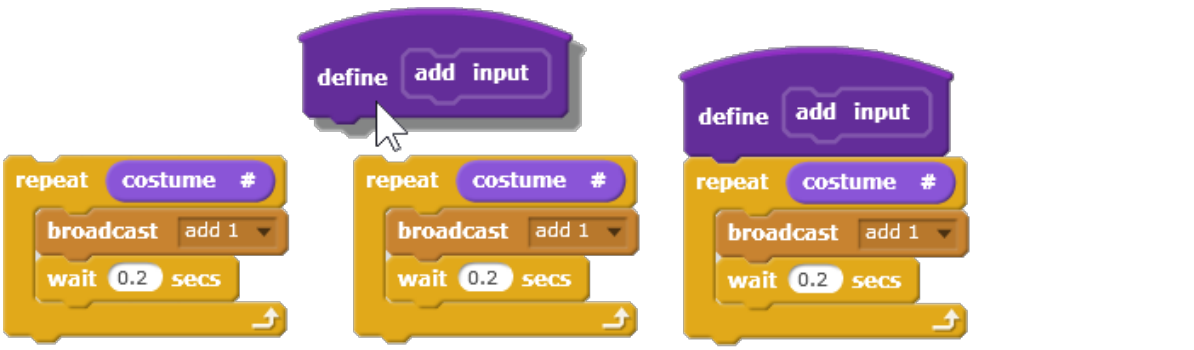
How long is your sequence? Is it always the same? What would be the longest sequence or the shortest one?

CONNECTIONS TO Y5 SCRATCHMATHS

*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

3 In Module 3 we often used **repeat** to run certain actions, a jump, a ghost effect etc., in several smaller steps instead of one quick change. Sometimes we also added the **wait** block with a short delay.
In this activity we want to **broadcast** a message *add 1* to another sprite as many times as the **input** sprite shows. In the situation below, it displays 3, that is, its costume number is 3. Thus this script will broadcast the message three times, with a short delay in between – so that we can see the **ones** sprite increasing its number by 1 three times in a quick loop.

Pupils started **making their own blocks** in Module 1, Activity 1.4.1, and continued throughout the Y5 materials. We always encourage pupils to (1) build and debug a script, (2) give it a name by making a new block and attaching the **define** *my new block* hat block to that script, (3) keep the definition as it is, but (4) use the new block as a shortcut to do the same as the whole previous script – see the **if … then …** script below.

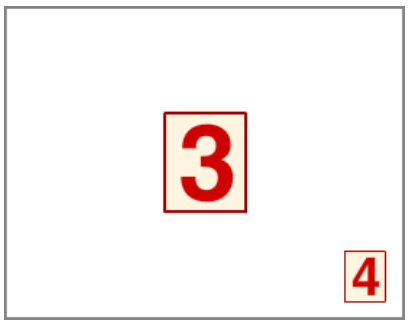


4 The **forever** block – a kind of infinite **repeat** – was introduced in Activity 3.1.4 to make **Pico** walk around forever. In Activity 3.2.2 we started using the **if … then …** block inside **forever** to monitor certain conditions again and again – and reacting as appropriate. It is intuitive and natural to refer to this **forever + if** structure as *whenever*.
In this activity we may read it as: *whenever* the **input** sprites touches **ones**, it will add its value to **ones** then glide back home.

ADDITIONAL SUPPORT

*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

1



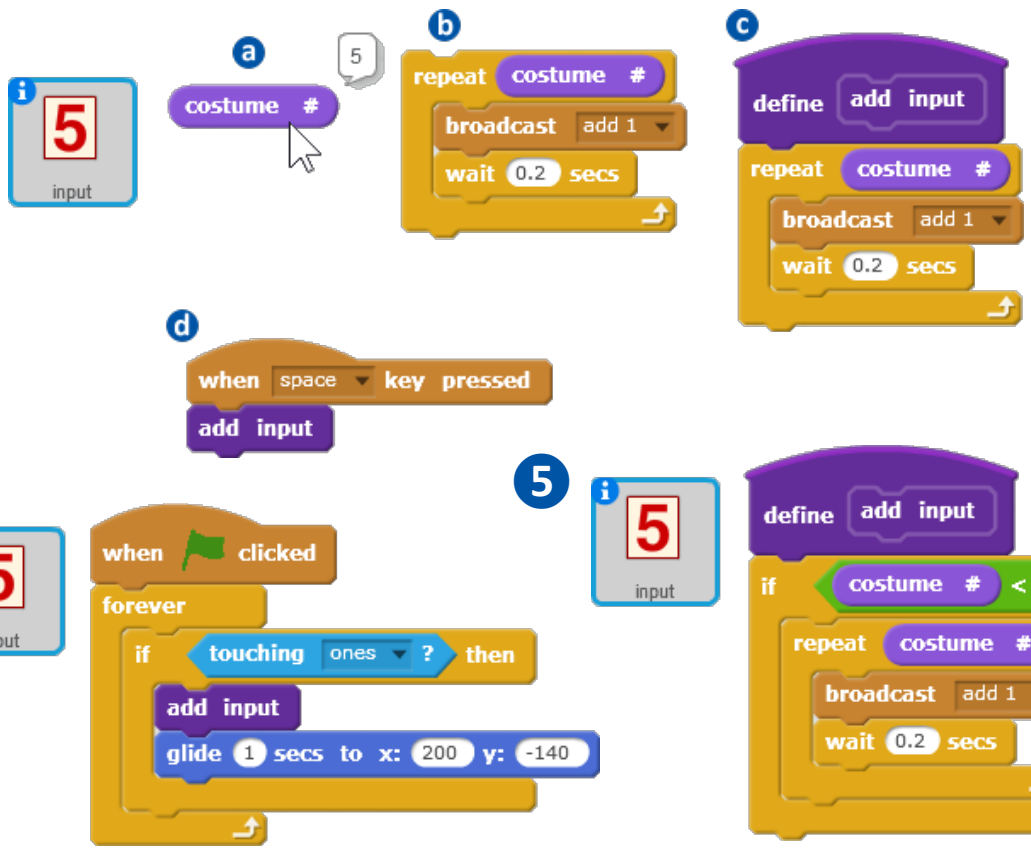
There are two sprites in the project:

- the **ones** sprite, as in the previous project, with a simple **when this sprite clicked next costume** behaviour,
- another digit – the **input** sprite (in the lower right corner) with a simple **when this sprite clicked next costume** behaviour.

This means, we can set any value to the **ones** and to the **input** sprite as well.

2

3 We first explore the isolated **costume #** block (see **a**), then use it as the **repeat** number: the message *add 1* will be broadcast exactly the **costume #** times (see **b**). We make a new block **add input** (see **c**) and use it in the **when** *space* **key pressed** script (see **d**).



4

5

We build a **when green flag clicked** script for the **input** sprite which will **forever** check whether certain condition is true – i.e. *Whenever* **input** touches **ones** it will **add input** to **ones** then fly back home.

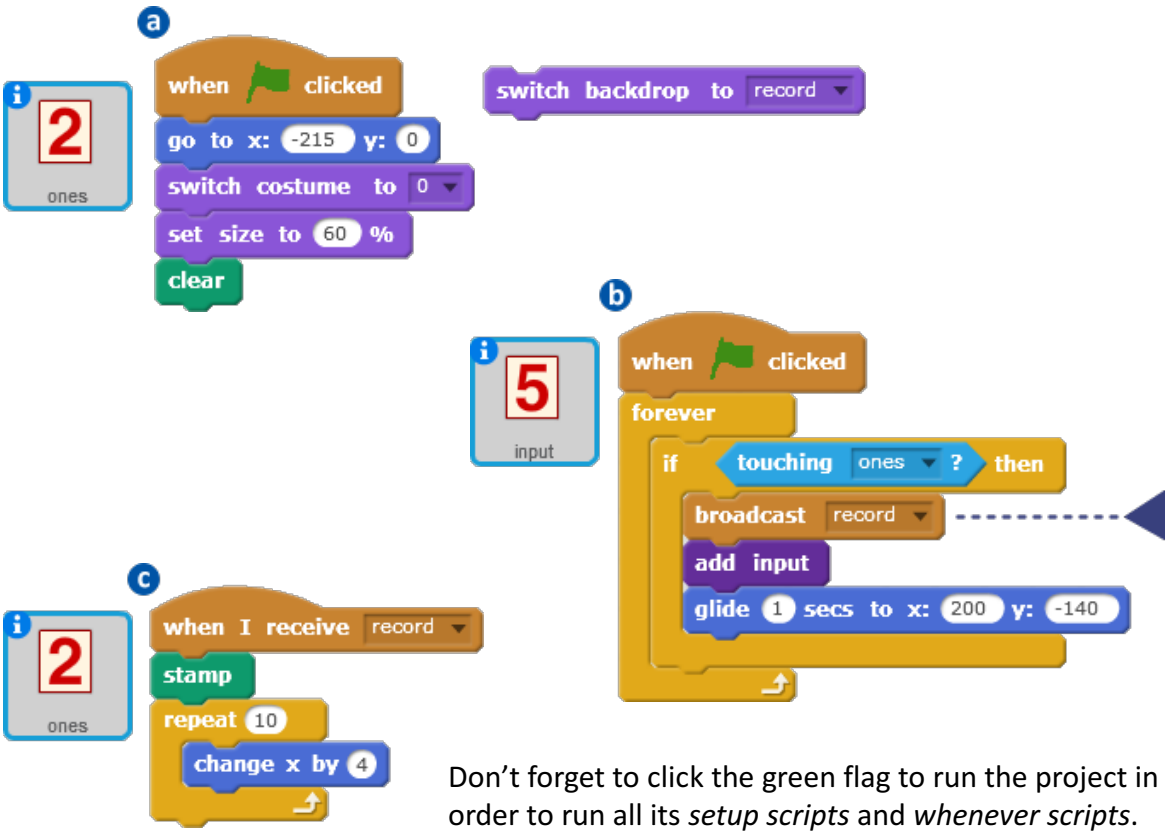
ADDITIONAL EXTENSION: RECORDING

In this extension we continue the development of the **4-Sequences** project, we will now build the recording behaviour that pupils simulated using the 4.1.1 worksheet. Assuming that Extension 4 was built it takes 3 simple steps to add a new functionality into the project: the sequence will be constructed (recorded) directly in the stage.

a Pupils switch the backdrop to the second one named *record* – it has a row of empty placeholders. Pupils modify the *setup script* of the **ones** sprite: they set the size of the sprite to 60 % and find out the correct position for it to start at the leftmost placeholder.

b Pupils add a **broadcast** block into the *whenever* script of the **input** sprite: *whenever* it touches the **ones**, it will first **broadcast** *record*, then proceed as before.

c The **ones** sprite will react to *record* by stamping itself and moving to the next placeholder.



Don't forget to click the green flag to run the project in order to run all its *setup scripts* and *whenever scripts*.

MODULE 4: INVESTIGATION 2
Timers and Stopwatches

"Timers and Stopwatches" develops and rebuilds the place value model built in Investigation 1 in the context of time. The Scratch place value model for time is particularly elegant since the sprite's costumes must be adapted to support the digits needed for counting in seconds.

Pupils may not initially see the connections of time and place value which are reinforced by building the model; the **tens of seconds** sprite changing from 5 to its next costume (the 0) trigger the "carry" behaviour. This is identical to the *add 10* behaviour in Investigation 1.

Counting down is computationally more difficult than counting up using the mathematical concept of "borrowing". Pupils develop their concept of variable using **costume #** in simple expressions. Pupils should bridge to pencil and paper calculations which include "borrowing" and "carrying" and be encouraged to think using the Scratch model.

- **Activity 4.2.1** – Build a Stopwatch
- **Activity 4.2.2** – Unplugged: Nudge Nudge Get Get
- **Activity 4.2.3** – Countdown Conundrum
- **Ext. Activity 4.2.4** – Dizzy Dials

Scratch projects

4-Stopwatch	4-Timer	4-Dials
4-Stopwatch FINAL	4-Timer FINAL	4-Dials FINAL
	4-Timer Extensions FINAL	4-Dials Extension FINAL

LINKS TO PRIMARY NATIONAL CURRICULUM

CURRICULUM OBJECTIVES	LINK WITH SCRATCHMATHS
Mathematics	
Record and compare time in terms of seconds, minutes and hours.	▶ Pupils are required to build timers and stopwatches that can count up or countdown a length of time up to 99:99 (with possible extensions to tenths of seconds and hours).
Read, write and convert time between analogue and digital clocks.	▶ [Extension] Pupils build an analogue representation of their digital timer.
Describe positions on the full coordinate grid.	▶ Pupils are required to use their knowledge of the full coordinate grid to set initial positions of their sprites.
Solve number and practical problems involving place value.	▶ Pupils are required to apply their place value knowledge to the context of time.
Link 360° to calculating angles.	▶ [Extension] Pupils are required to use their knowledge of a 360° full turn to calculate the angle of turn for the second and minute hands.

Build a Stopwatch

Learning Objectives

Explore how to use multiple sprites to count up in seconds and minutes.
Explain why the sprites require different numbers of costumes.

Activity Instructions

Pupils open project **4-Stopwatch**, **Save as a copy** (online) or **Save as** (offline) and rename. *The final version of this project at the end of Activity 4.2.1 will be* **4-Stopwatch FINAL**.

1. Pupils explore the *setup script* and the costumes of the **1 secs** sprite.
2. Pupils duplicate the **1 secs** sprite, rename to **10 secs** and update its initial position in the *setup script* so that it sits atop the correct placeholder. They duplicate two more times, renaming new sprites to **1 mins** and **10 mins**. They modify their *setup scripts*.
3. Pupils start building the behaviour for **1 secs**: as we want it to show current seconds, it will **wait 1 secs** then display the **next costume**… and repeat this **forever**. Pupils add **when this sprite clicked** hat block and run it by clicking the **1 secs**.
4. Similar to Investigation 1 pupils extend the basic script of **1 secs**: if it gets to its 10th costume, it will broadcast *add 10 secs* message. The **10 secs** will react by **next costume**.
5. Pupils modify the script of **10 secs** so that it broadcasts *add 1 min* message when appropriate and the **1 mins** sprite reacts to it.
6. Pupils complete and debug the scripts of all four digits of the stopwatch so that it works correctly.

7. **[Extension]** Pupils experiment with the **wait 1 secs** block in the **forever** script of the **1 secs** sprite:
   - They measure the accuracy of the stopwatch and discuss. They **calibrate it by tweaking** the input of the **wait** block.
   - They double the 'speed of time' or 'slow it down'.
   - They extend their stopwatch so that it shows tenths of seconds as well. *Does it work properly?*

Mathematics Connections

Time is represented using the same digit costumes, however the place values are different. Discuss: *What is the same and what is different between the two representations?*

What happens when 1 secs gets to 0 again? Does it nudge 10 secs?

What happens when the stopwatch gets to 1:59? Does it go to 2:00 as it should? Which digits will the ***10 secs*** *sprite need?*

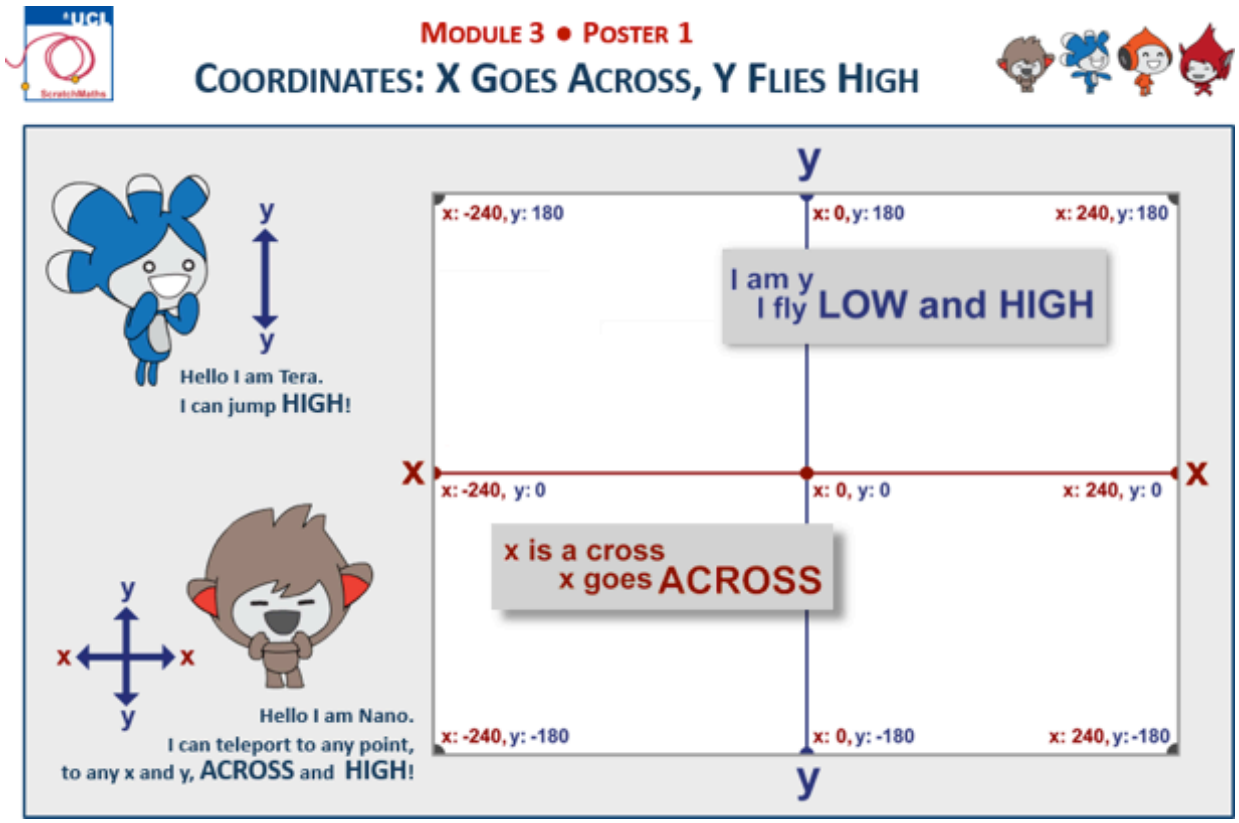
Discuss: *When should the* ***10 secs*** *increase? Which sprite should react?*
Discuss: *What are all possible values of* ***10 secs*** *place holder? Which costume numbers can we remove?*

Discuss: *Explain how the model would work if it was extended to include* ***hours****. How many costumes would each sprite need?*

CONNECTIONS TO Y5 SCRATCHMATHS

*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

2 Each module of Y5 ScratchMaths intervention has various additional and extension materials. Poster 1 of Module 3 illustrates the **co-ordinate system** of Scratch (which can be found on the Module 3 resources page) and could be helpful in Y6 as well.



There are two different strategies how to make sprites move in the stage. One refers to the co-ordinates – **x position** and **y position** of the sprite. We often use blocks like:

We can also set one of the co-ordinates – either **x position** or **y position** – separately:

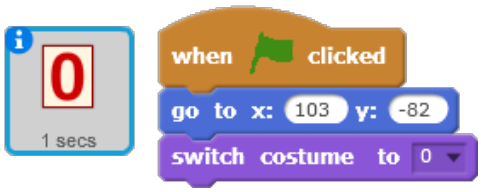
Note the difference between **setting** one of the co-ordinates and **changing** it (increasing or decreasing) by certain number, see below on the left. We can also use the **x position** and **y position** reporter blocks as inputs to other blocks, see examples on the right.

In the ScratchMaths approach we refer to this strategy of moving or placing sprites as **jumping**, as opposed to the "beetle" style of **moving**, which we use for example in Module 2 of Y5 and Module 5 of Y6.

ADDITIONAL SUPPORT

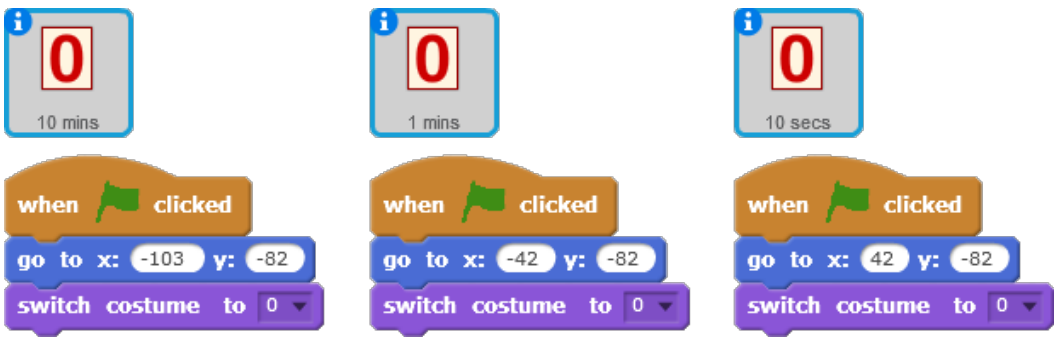
*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

1

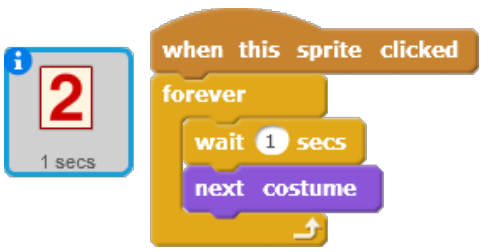


Below are example scripts for each step within 4.2.1.

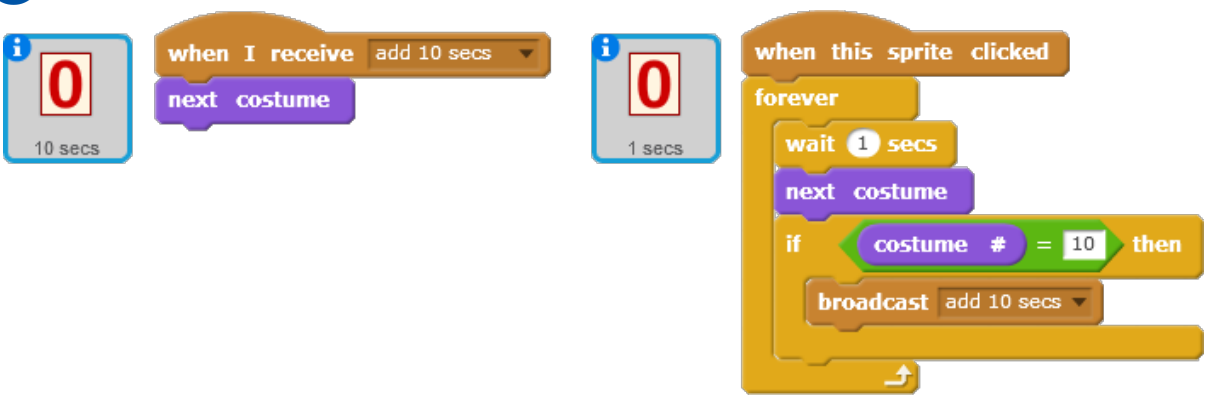
2



3



4



ADDITIONAL SUPPORT CONTINUED

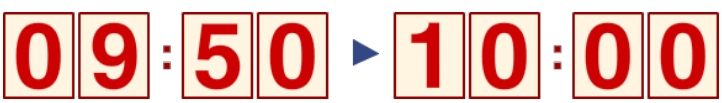
5

The **1 secs** sprite needs no modifications.

The main discovery here is to realise that the **10 secs** sprite will never need its costumes (i.e. digits) **6**, **7**, **8** and **9**. So in alternative **(a)** we deleted them as redundant costumes for that particular sprite. When it gets to its **6th costume** (which is in fact 0 now) it broadcasts its *add 1 min* message.

In alternative solution **(b)** we have not deleted any costumes of **10 secs**. Instead, whenever it gets to its 6th costume (i.e. digit 6 in this case), it immediately switches to 0 and broadcast its message.

6



Encourage pupils to set the initial time (in the *setup scripts*) to e.g. **09:50** and click the **1 secs** sprite (only once as it will run its **forever** process). In 9 seconds it should go to **09:59** and then to **10:00**.

Module 4 • Investigation 2 • Activity 4.2.2

Unplugged: Nudge Nudge Get Get

Learning Objectives

Explore how to "borrow" from tens and hundreds.
bridgE to subtraction and place value as well as time calculations.

Activity Instructions

- Use three of the flip books created for the previous investigation – this activity follows a similar format to activity 4.1.2.
- Choose three pupils to stand at the front.
- Give each flip book to one of the pupils, representing **hundreds**, **tens** and **ones**.

  - All pupils at the front should hold their flip books out in front of them.
  - Set the initial number by choosing a number bigger than 100 (for example 111) and set each of the pupils' flip books to display this number.
  - Tell the two pupils who are representing the **tens** and **hundreds** to position themselves so they cannot see the number on the other pupils' flip books (other pupils in the class can be responsible for making sure there is no cheating!).
  - Have another pupil choose a number (smaller than 10) e.g. by rolling a die and instruct the **ones** pupil to decrease their flip book **by 1** that many times.
  - **Before** the **ones** flips the book decreases by 1, each time they must check if this is possible – **which it is not if the book shows 0**. In that case they nudge the **tens** pupil to decrease their flip book and only then flip their own book.
  - It the **tens** pupil get a nudge, they should decrease their flip book by 1. However, before they do that they have to check if this is possible – **which it is not if the book shows 0**. In that case they nudge the **hundreds** pupil and only them flip the book.
  - Keep repeating this process until you reach a target number e.g. 90.

[Extension]

- Once pupils have got the hang of it try the same activity using time, switch pupils so all can observe the process.

Mathematics Connections

Connect the number card model with place value models seen during KS1 and KS2 mathematics. Ask pupils what the value of each card is to connect with the place value model.

In Activity 4.1.2 the **ones** was **adding 1**, they flipped the book then checked if they should nudge the **tens**.

Now the order of steps is reversed: Before **ones** decreases by 1, they must ensure **if this is possible**. If not (i.e. their book shows 0), they first have to nudge (to "get" 10 ones from **tens**) – to make it possible to flip to 9 again.

Discuss: *What is the smallest (or the largest) number of rolls of the die to get from 111 to 90. How can we calculate this?*
Largest will be the same as the subtraction 111-90

Discuss: *Ask pupils to explain what they have seen.*

MODULE 4 • INVESTIGATION 2 • ACTIVITY 4.2.3

Countdown Conundrum

LEARNING OBJECTIVES

Explore how to use multiple sprites to count down in seconds and minutes.
Explain how to use the concept of "borrowing" in a countdown timer.

ACTIVITY INSTRUCTIONS

MATHEMATICS CONNECTIONS

Pupils open project **4-Timer**, **Save as a copy** (online) or **Save as** (offline) and rename. The final versions of this project at the end of Activity 4.2.3 will be **4-Timer FINAL** and **4-Timer Extensions FINAL**.

1. Pupils explore the project, its sprites **1 secs, 10 secs** and **1 mins**, their costumes and their *setup scripts*.
2. Pupils update the *setup script* of each sprite to set the *initial time* for counting down – by switching to the appropriate costume number (e.g. 3m 25s).

Discuss: *How can we set the initial time e.g. 3m 25s?*

3. For counting up we used the provided **next costume** block. For counting down we will need the opposite command to switch to the *previous costume* but we have to make it ourselves. Pupils use **switch costume to …** to build a short script for the **1 secs** sprite to **switch its costume to the previous one**. They name it **previous costume**.
4. Pupils build *a countdown behaviour* for **1 secs**: when the sprite is clicked, a **forever** process will start which will make the sprite switch to its **previous costume** every second.

Now we have to decide when exactly should the **1 secs** sprite 'nudge' the **10 secs** to get an extra 10 seconds from it.

5. Pupils extend the **1 secs** *count down* script so that it correctly gets an extra 10 seconds from **10 secs**.

Discuss: *What should happen when the **1 secs** sprite is at 0 and wants to decrease by 1?*

6. Pupils modify the **10 secs** sprite so that it:
   - Correctly reacts to the *get 10 secs* message.
   - For that, **10 secs** needs the **previous costume** definition as well.
7. Pupils experiment with their timer making it count down quicker or slower etc., adding the **10 mins** sprite etc.

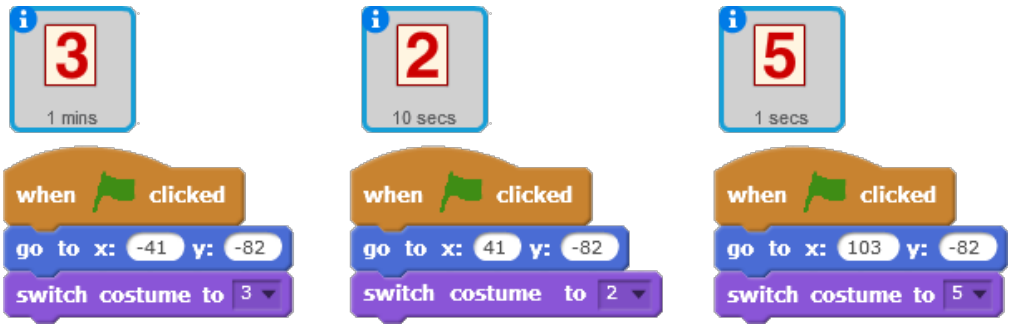
Discuss: *What should happen when the **10 secs** is at 0 and wants to decrease by 1? Where will it get 'more' from? If **10 secs** nudges **1 mins** and that sprite reacts by decreasing by 1, to what value should **10 secs** then 'flip' its costume to?*

8. **[Extension]** What will happen when our timer gets to **0:00**? How could we – instead of having it go in a 'big loop' back to **9:59** – just stop, as we would expect a timer to? In the **Control** group there is a block **stop all** (running scripts). Think about an algorithm and corresponding scripts to use **stop all** to stop the whole timer at **0:00**.

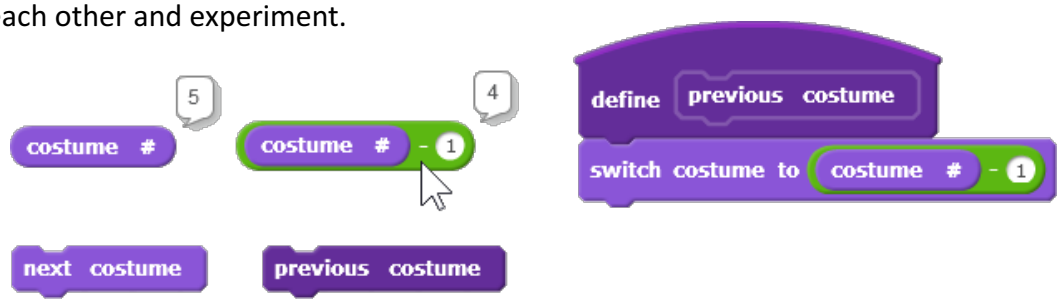
ADDITIONAL SUPPORT

*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

2

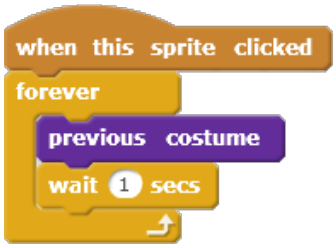


3 First experiment with the **costume #** reporter block in isolation, then build the expression **costume # - 1**. The final definition of **previous costume** can be read: *switch your costume to what it is currently minus 1*. Then put **next costume** and **previous costume** blocks next to each other and experiment.



4

Some pupils will come with the alternative and correct solution, see on the right. Discuss as a class what is the small difference between these two solutions. [in the solution on the right the first second of counting down is 'lost']



5

when counting up

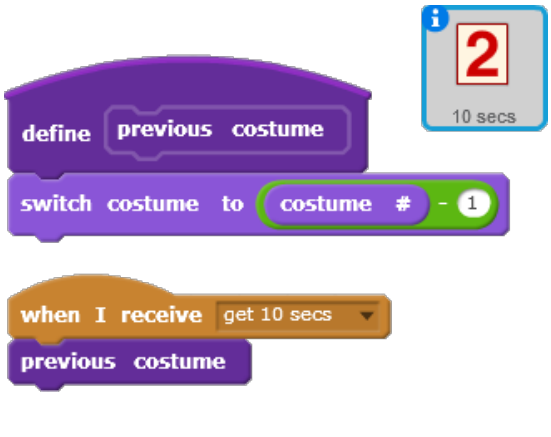
when counting down

Discuss and compare adding 10 when counting up and getting 10 when counting down:

- When counting up, the sprite can safely go to the next costume, then check whether this change requires a 'nudge'.
- When counting down the sprite first has to check whether it has "enough" to decrease from. If not it has to get an extra 10. Only then it is able to decrease.

ADDITIONAL SUPPORT CONTINUED

6



7

First we have to be sure when it might not be possible for the **10 secs** sprite to simply go to its previous costume – this will happen when its value is 0, which here means when its **costume #** is 6. In that case, it must get an extra 10 seconds from **1 mins** first, and only then go to **previous costume**.

Note ensure the **1 mins** sprite has the **previous costume** definition in its scripts area.

EXTENSION SUPPORT

8 This is an advanced extension as pupils will encounter ***real problems of parallel programming***: each sprite operates independently and we need to recognise a moment when **all of them** have got to 0.

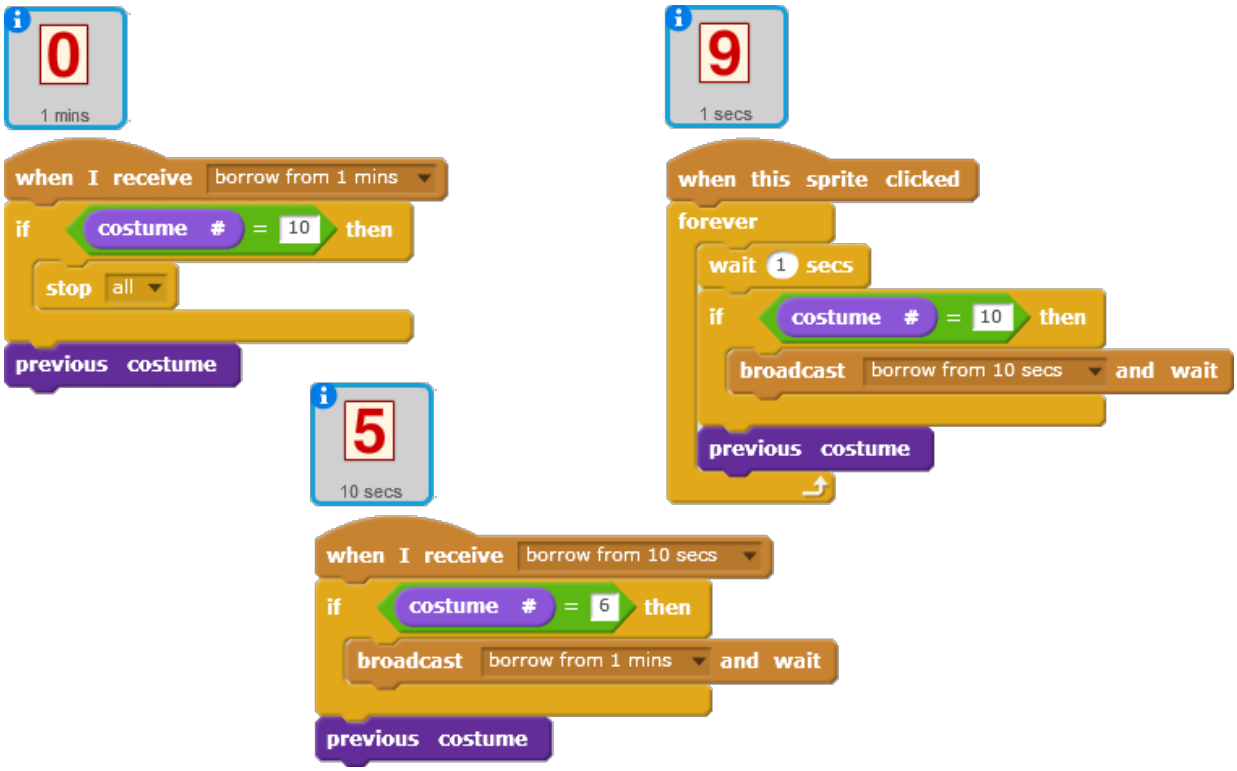
One possible solution might result from the following reflection: If the **1 mins** sprite receives the *get 1 min* message it means the other sprites are currently at 0. Thus, if **1 mins** itself is currently at 0, the timer should stop.

There is one possible danger in this solution: When **1 mins** gets the *get 1 min* message and is checking whether its costume number is 10, the **1 secs** and **10 secs** sprites continue their count down and may get to … **:59** – before the whole project is stopped.

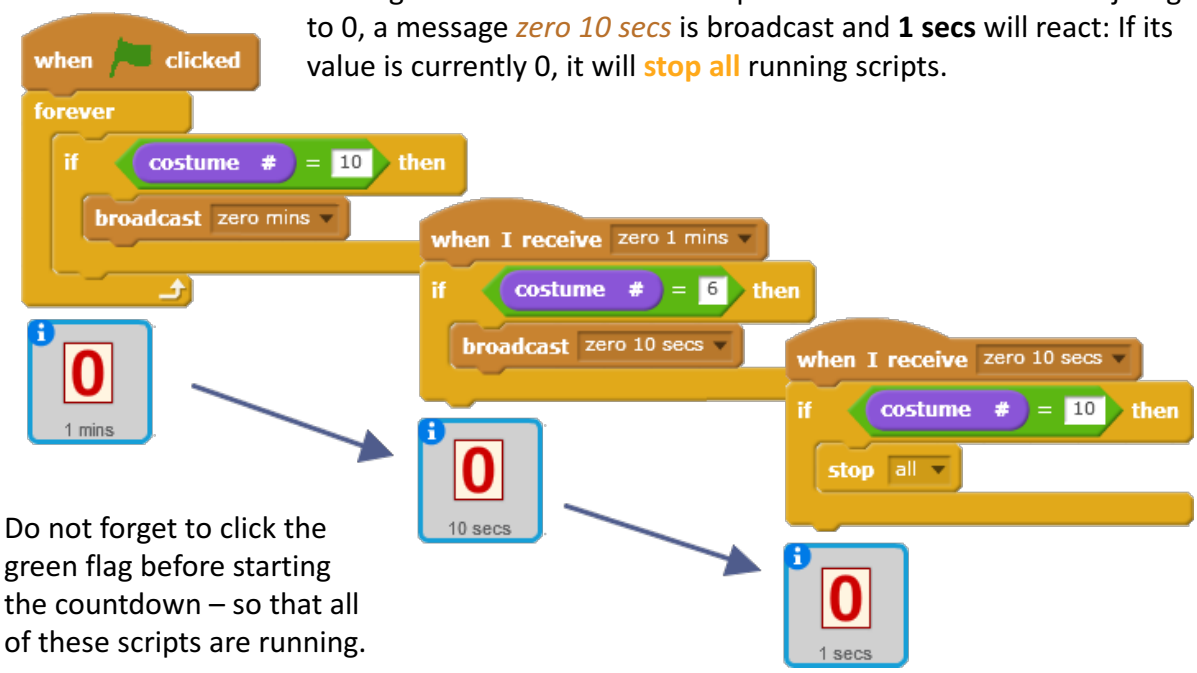
One way around it might be to implement a "safe getting" approach. For instance when **1 secs** realises it needs to get 10 seconds from **10 secs**, it will broadcast its *get 10 secs* **and wait** until the reaction is successful. Only then it decreases its value. The same, however, must be done by **10 secs** – it will continue its count down only if getting an extra minute from **1 mins** was successful. For this purpose Scratch offers a version of broadcasting:

EXTENSION SUPPORT CONTINUED

Solution with the "safe getting" approach: both **1 secs** and **10 secs** will decrease their values only when the reactions to their broadcasting are successfully completed.



Here is another alternative solution which does not call for **broadcast and wait**: In the **1 mins** sprite a monitoring **forever-if** (*whenever*) process is launched. When the sprite gets to 0, it broadcasts a message *zero mins*. The **10 secs** sprite will react: if it itself has just got to 0, a message *zero 10 secs* is broadcast and **1 secs** will react: If its value is currently 0, it will **stop all** running scripts.



Do not forget to click the green flag before starting the countdown – so that all of these scripts are running.

MODULE 4 • INVESTIGATION 2 • ACTIVITY 4.2.4

[Extension] Dizzy Dials

LEARNING OBJECTIVES

Explore how to represent time in both digital and analogue forms.
Explain what angle the second/minute hand should for each unit of time and why.

ACTIVITY INSTRUCTIONS

Pupils open project **4-Dials**, **Save as a copy** (online) or **Save as** (offline) and rename. Final versions of this project at the end of Activity 4.2.4 will be **4-Dials FINAL** and **4-Dials Extension FINAL**.

1. Pupils explore the project, its sprites and their costumes and scripts. They run the project by clicking the green flag then click **1 secs** sprite.
2. Now we want the **seconds hand** to depend on the **1 secs** sprite: whenever **1 secs** "ticks" another second, the **seconds hand** sprite will get a signal (a message) and react by **rotating** (turning it) correspondingly. Pupils add the **broadcast** *tick* block in the **forever** script of **1 secs** and build a corresponding reaction of the **seconds hand**.

We now want to run the **minutes hand** as well.

3. Pupils change the *setup script* of the **minutes hand** so that it is shown (not hidden). Then they discuss and decide which sprite will inform the **minutes hand** to react by turning right. It could be either the **10 secs** sprite when informing the **1 mins** sprite to go to the **next costume**, or the **seconds hand** after finishing the whole turn, i.e. when its **direction** (a reporter block at the bottom of the **Looks** group) gets again to 0. Pupils test out their dial.
4. **[Advanced extension]** Pupils explore the **go to front** block to ensure the **minutes hand** is in front of the **second** hand. Pupils edit their project so that when the **seconds hand** rotates it gradually fills the dial with colour. When it completes the whole turn (i.e. after each minute), the dial is cleared and the filling process starts again.

MATHEMATICS CONNECTIONS

Discuss: *What angle should the second hand sprite turn after one second has passed? Why?*

Discuss: *When should the minute hand sprite turn? How can we tell it to turn at that point?*

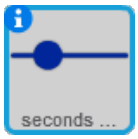
Discuss: *What angle should the minute hand sprite turn after one minute has passed? Why?*

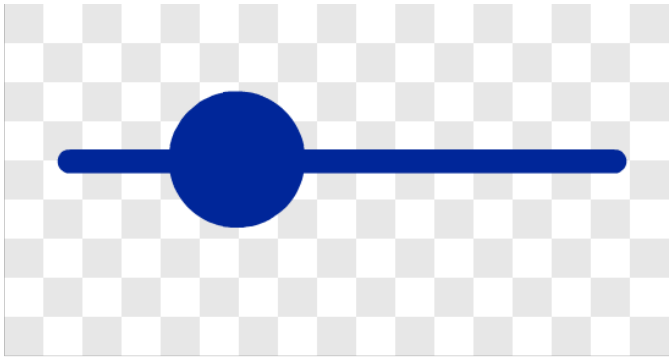
Discuss: *Why is time an example of base 60? Hipparchus and other Greek astronomers used astronomical techniques previously developed by the Babylonians. The Babylonians made astronomical calculations in the sexagesimal (base 60) system inherited from the Sumerians, who developed it around 2000 B.C. Although it is unknown why 60 was chosen, it is convenient for expressing fractions, since 60 is the smallest number divisible by the first six counting numbers as well as by 10, 12, 15, 20 and 30, all factors of 60.*

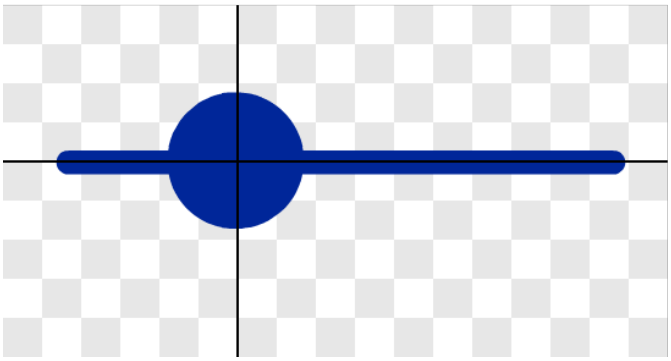
CONNECTIONS TO Y5 SCRATCHMATHS

*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

1







Let us explore the **seconds hand** sprite's costume. To do so, we select it in the sprites area and go to the **Costumes tab**.

A **painting program** opens in which we can modify or create sprites' costume(s). Note two things in the lower right corner of this window:

- We can **zoom in** the costume up to 1600 %. Let us go to 400 %.
- Two buttons there indicate there are two different modes or kinds of costumes in Scratch 2.0 – **bitmap** and **vector**. This costume has been created as a vector shape.

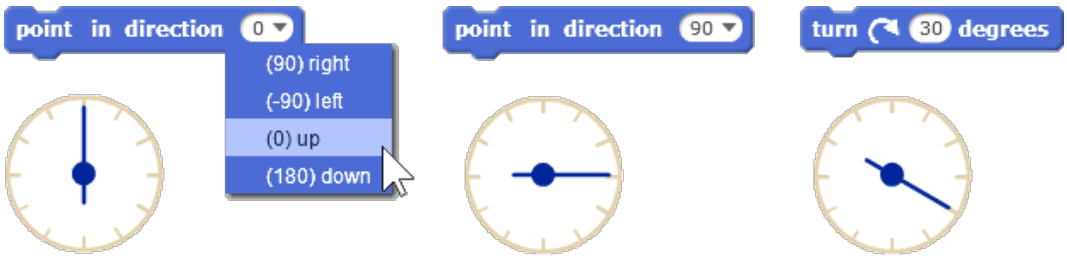
As this is a vector costume it is easy to resize and turn (rotate) the sprite "wearing" it: whichever direction of the sprite we choose, the costume will immediately and automatically be rotated in that direction.

In the painting program the costumes are **always pointing in direction 90**, that is, to the east.

When we turn a sprite, Scratch rotates its costume around the costume **centre**. It can be any position inside (or outside) the costume, which we can inspect or move when we press the **Set costume centre** button in the upper right corner. Black lines appear, see the second picture above. Their crossing point is the costume centre and when the lines are displayed we can drag it elsewhere in the costume.

Note that in the case of the **seconds hand**, the costume centre is in the centre of the big dot. Thus, when we turn the hand by any angle, it will simply rotate like a typical dial hand.

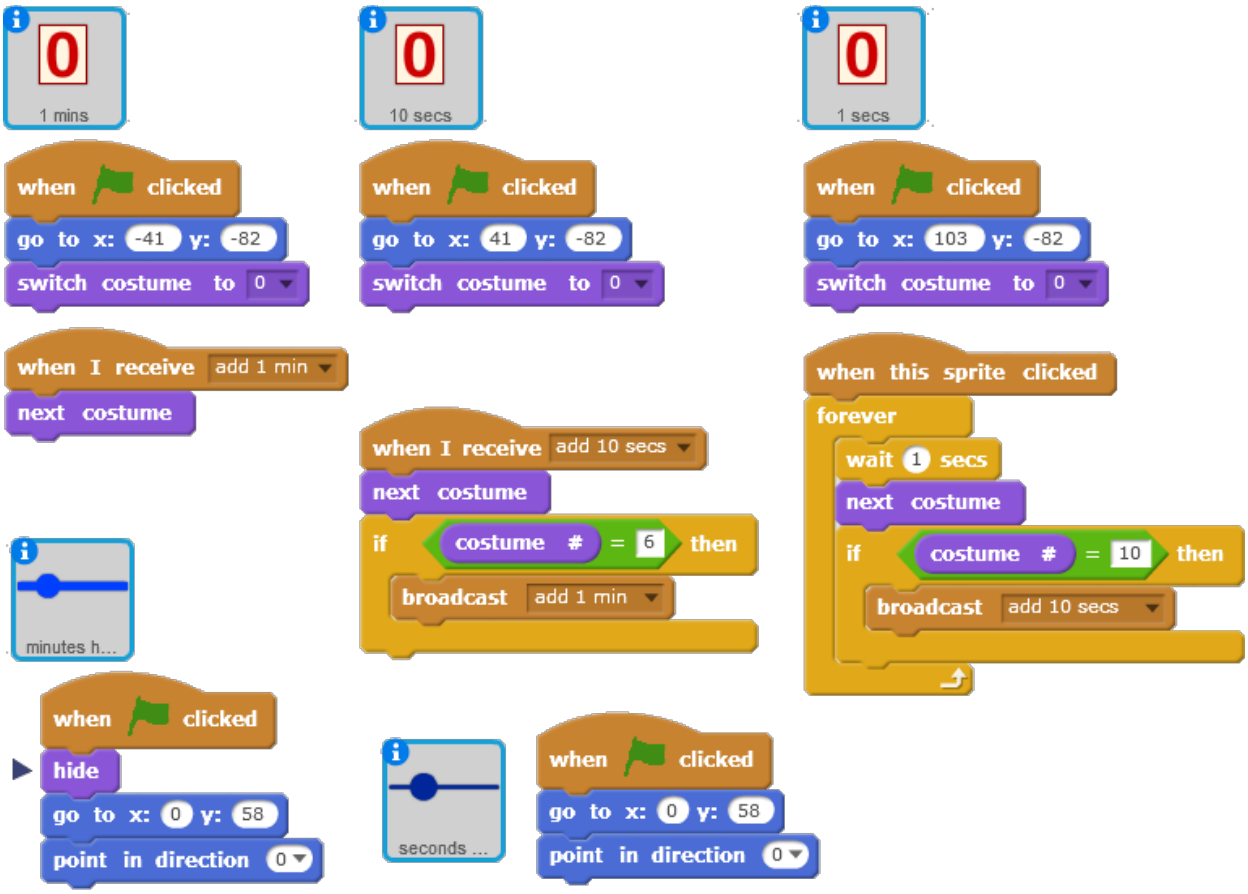
In the *setup script* of the **seconds hand** the initial direction is set to 0, i.e. up or north. Use simple **turn right … degrees** block to explore the **seconds hand**, see pictures bellow.



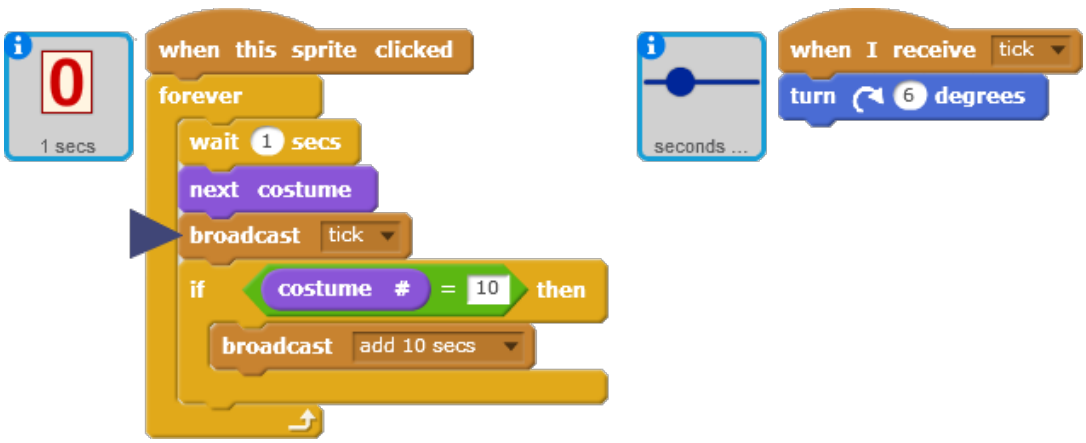
ADDITIONAL SUPPORT

*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

1 Three sprites **1 mins**, **10 secs** and **1 secs** are similar to two previous projects: when the **1 secs** sprite is clicked, all three sprites start measuring time like the stopwatch. Another sprite **minutes hand** is currently hidden and the **seconds hand** does nothing.



2 The **1 secs** sprite will broadcast the *tick* message and the **seconds hand** will react. As it must rotate all around 360 degrees in 60 seconds, each second corresponds to turning right by 6 degrees.

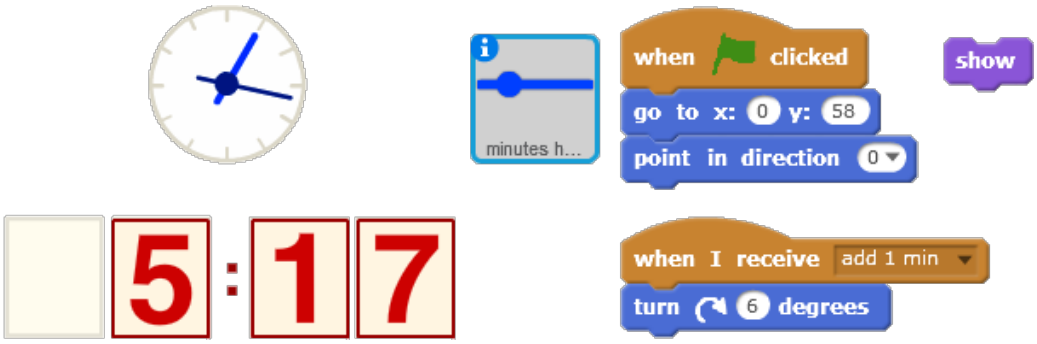


ADDITIONAL SUPPORT CONTINUED

3 It is enough to remove the **hide** block from the **minutes hand** *setup script* and click the **show** block only once – or snap it into the script.

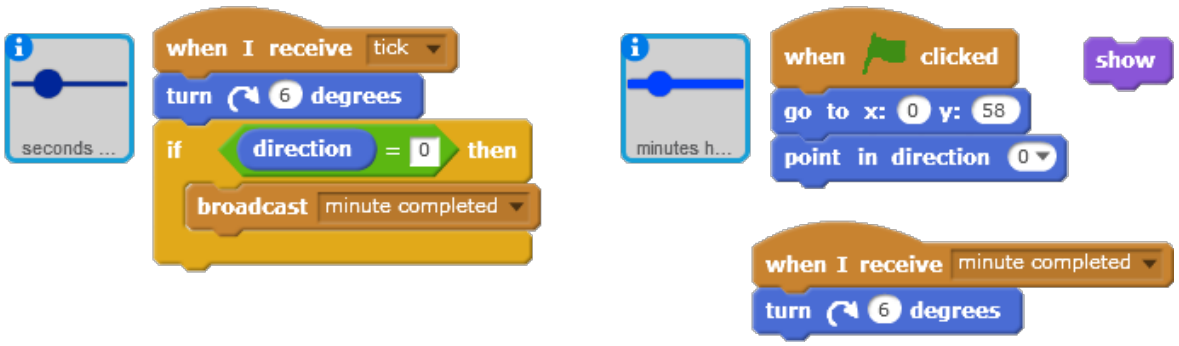
Alternative **(a)**: The main point is to make the sprite react to the correct message: whenever **10 secs** broadcasts its *add 1 min* message, both **1 mins** and **minutes hand** will react.

a



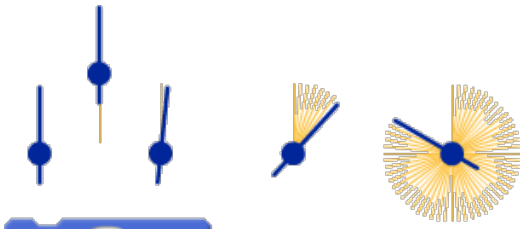
Alternative **(b)**: whenever the **seconds hand** covers the full turning, i.e. whenever its **direction** is 0 again, it will broadcast a message *minute completed*. The **minutes hand** will react.

b



ADVANCED EXTENSION SUPPORT

4 Scratch cannot fill regions with colour, so we must think of another strategy. Fortunately, the **seconds hand** is an ordinary sprite, so it has its pen and can be used to draw lines.



```
move 41 steps
move -41 steps
turn 6 degrees
```

In Module 1 we used the algorithm of moving forward, then moving backwards the same distance and turn… If we repeat it many times with the sprite's **pen down**, it will draw a kind of pompom, see on the left.

We may then experiment with bigger pen sizes, for example 2 or 4 or even more… Or instead of one **turn right 6 degrees** we may draw a line and turn by 3 degrees twice within each *tick* or even 3 times in a *tick*, turning by 2 degrees... see below.

```
set pen size to 2
```



```
set pen size to 4
```

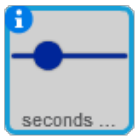
for each tick

```
repeat 2
    move 41 steps
    move -41 steps
    turn 3 degrees
```

for each tick

```
repeat 3
    move 42 steps
    move -42 steps
    turn 2 degrees
```

So here is a possible final solution for the **seconds hand** to fill the dial with colour while turning. When the **seconds hand** points in direction 0 again (after rotating the full turn) it clears the stage (to get rid of the previous yellow minute). As the dial itself is a part of the backdrop it will not be cleared.




```
when clicked
go to x: 0 y: 58
point in direction 0
set pen color to
set pen size to 4
pen down
clear
```

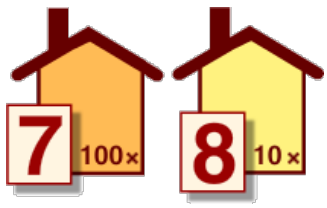
```
when I receive tick
repeat 3
    turn 2 degrees
    move 42 steps
    move -42 steps
if direction = 0 then
    clear
```

MODULE 4: INVESTIGATION 3

The Conversion Game

"The conversion game" (which is built in Investigation 3 and then will be further exploited and modified in Investigation 4) adapts and combines the Scratch building blocks and strategies created in Investigations 1 and 2.

The game is modelled initially as an unplugged activity and is designed to build connections and develop fluency with place value, conversion and number decomposition. Pupils then develop their mathematical reasoning through building, exploring and modifying the game and by setting and exchanging problems with their peers.



- **Activity 4.3.1** – Unplugged: Playing the Conversion Game
- **Ext. Activity 4.3.2** – Building It: The Display
- **Ext. Activity 4.3.3** – Building It: Conversion Houses
- **Ext. Activity 4.3.4** – Building it: Record Keeping

Scratch starter projects

4-Conversion Game
4-Conversion Game INT1
4-Conversion Game INT2
4-Conversion Game FINAL

LINKS TO PRIMARY NATIONAL CURRICULUM

CURRICULUM OBJECTIVES	LINK WITH SCRATCHMATHS
Mathematics	
Perform mental calculations, including with mixed operations.	▶ Pupils are initially required to play the conversion game on paper and perform calculations that involve both multiplication and addition.
Recall and use multiplication to aid fluency.	▶ Pupils are required to perform multiplications involving x1, x10 and x100.
Solve number and practical problems involving place value.	▶ Pupils are required to use their knowledge of place value to build a display that correctly reacts to the addition different multiples of 100s, 10s and 1s.
Use estimation to check accuracy.	▶ Pupils are required to use estimation to check their display is increasing by the correct amount as well as to envisage different possible outcomes whilst playing the conversion game.
(KS3) Work with experiments that involve randomness.	▶ In one variation of the conversion game pupils are required to use knowledge of random numbers to envisage different possible outcomes of the game.
Development of mathematical fluency, reasoning and problem solving.	▶ Pupils are required to apply a range of mathematical knowledge and skills whilst playing (and building) the conversion game providing varied opportunity to develop their fluency, reasoning and problem solving.

Module 4 • Investigation 3 • Activity 4.3.1

Unplugged: Playing the Conversion Game

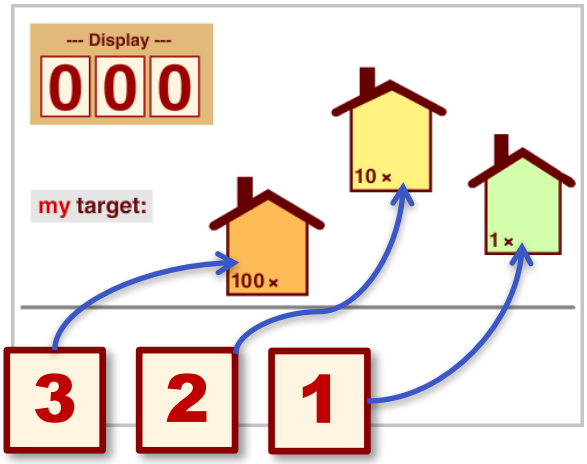
Learning Objectives

Explore how to use place value to create different numbers from the same digits.
Exchange ideas about how to create the biggest or smallest as well as specific numbers.
bridgE to mathematical fluency and reasoning.

Activity Instructions

- Print out a copy of the game screenshot (see next page) for each of your pupils and also give them a small strip of paper (or instead use the small cutout numbers from Activity 4.1.2 of this material).
- Pupils tear their paper strip into three squares and write the numbers 3, 2, 1 on each square (one per square).
- Provide extra paper or individual whiteboards for pupils to write down any calculations.

We now want to play an unplugged version of the Conversion Game. Pupils can build a number by placing the paper digit cards in different combinations of houses e.g. to get the number 321 pupils would place the 3 in the **100 ×** conversion house, the 2 in the **10 ×** conversion house and the 1 in the **1 ×** conversion house.



Multiple digits can be put into the same house and not all digits need to be used. Digits put into the same house are always added together (not subtracted).

- Pupils try to create the biggest number and the smallest number they can by placing their digits in different houses.
- Pupils try to create specific numbers e.g. 420, 15, 321, 240, 50…
- Pupils share with the class what they have discovered.

Mathematics Connections

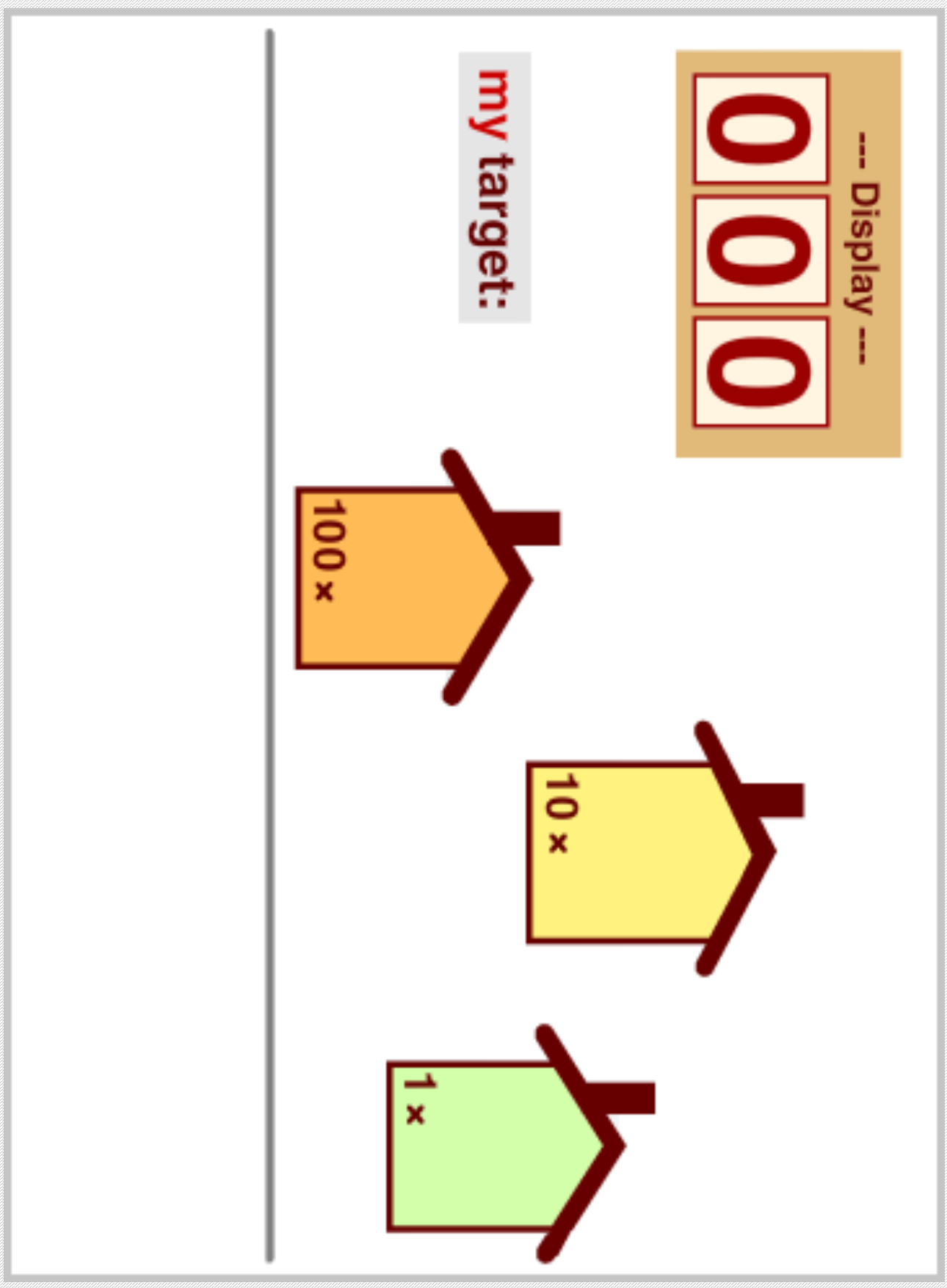
The conversion game explores decomposition of numbers but is restricted to the digit cards, 3, 2, 1 in the first example. Model the calculation on the white board to support instruction of the game.

To provide further challenge for your pupils you can provide all 9 digits. You may also modify the rules of the game or agree the rules as a class to provide variation in the game.

Discuss: *How would you create number 15? Do you now want to change your biggest or smallest number?*

Discuss: *How many different combinations are there using 3,2, 1. How can you be sure you've found all of them?* [Demonstrate an approach to systematically record the combinations.]

Unplugged: Playing the Conversion Game



MODULE 4 • INVESTIGATION 3 • ACTIVITY 4.3.2

[Extension] Building Conversion Game: The Display

LEARNING OBJECTIVES

Explore different ways of creating the same 3 digit number using repeat and broadcast.
bridgE to knowledge of place value (explored during earlier investigations).

ACTIVITY INSTRUCTIONS

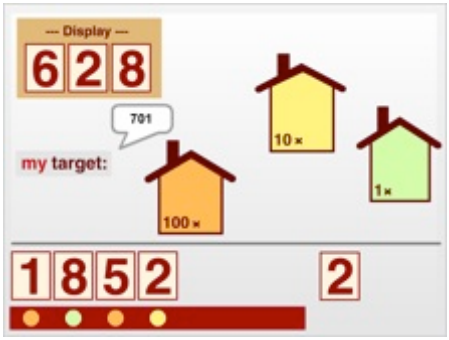
Open the project **4-Conversion Game FINAL** on the IWB to demonstrate how the final game works. Play the game as a class (see additional support for the detailed rules).

Pupils then open project **4-Conversion Game**, **Save as a copy** (online) or **Save as** (offline) and rename. Final version of this project at the end of Activity 4.3.2 will be **4-Conversion Game INT1**.

Our goal is to build a functioning display with three digits that represent a correct place value model (i.e. by broadcasting "nudges" and reacting to them).

1. Pupils explore the starter project with its sprites, costumes and their *setup scripts*.
2. While the **input** sprite increases itself by 1 when clicked, the **ones** sprite increases by 1 only when it receives the *add 1* message. Pupils build a script for the **input** sprite to **broadcast** *add 1* message in **repeat** – as many times as the **costume #** of the **input** is (as in Activity 4.1.3).
3. Pupils duplicate the **ones** sprite twice and rename the new sprites **tens** and **hundreds**. They update their *setup scripts* so they are correctly positioned on the Display panel.
4. While **ones** reacts to **broadcast** *add 1*, **tens** should react to **broadcast** *add 10* and **hundreds** to **broadcast** *add 100*. Pupils use these messages to implement "carrying" from **ones** to **tens** (when **ones** gets to its 10th costume) and "carrying" from **tens** to **hundreds** (when **tens** gets to its 10th costume).
5. In the **input** sprite pupils duplicate the **repeat costume # …** script twice, modify their messages to be *add 100*, *add 10* and *add 1*, set the initial value of **input** to e.g. 7 and experiment with clicking these isolated loops: Does the **Display** panel react properly?
6. For the **target number** sprite pupils build the behaviour: **when this sprite clicked** it will **ask** *What is the target number?* The sprite will then **say** the **answer** (and the speech bubble will keep showing in the stage).

MATHEMATICS CONNECTIONS



Note: Changing the order of the sprites in the Sprites panel to match the order in the stage makes it easier to keep track of which sprite you are working with. (However, reordering the sprites in the Sprites panel is currently possible only in the **online version**.)

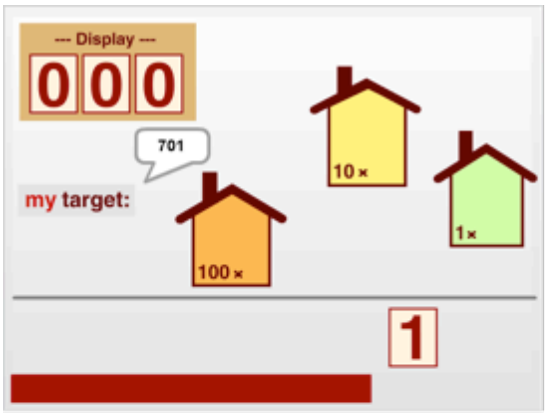
Discuss: *How many ways can you generate the same target number by running different combinations of these* **repeat** *scripts and different* ***input*** *values?*

ADDITIONAL SUPPORT

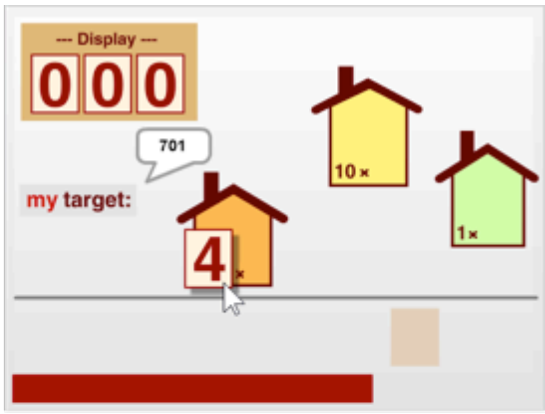
The Conversion Game Rules

- The player will get a **target number** from the teacher or from a peer (it is 701 in the picture below, see 1). The task then is to build that number (or get as close to it as possible) from seven single digits generated randomly by clicking on the **input** sprite.
- If the first input number is e.g. 4, the player will drag it over one of the three conversion houses, chosen by the player.
- If the value is dragged over the **100 ×** (orange) house (see 2 and 3 below), 400 will be added on the Display and the **input** sprite will fly back home. If it is dragged over the **10 ×** (yellow) house, 40 will be added. If it is **1 ×** (green) house, 4 will be added.
- Then the player will click the **input** sprite again, get the second value and repeat the same process another 6 times.
- The record of the player's decisions will be built as a sequence of stamps with the big colour dot above each stamp showing the colour of the house that the player dragged that number on to (see 3 and 4 below). This record can be used to discuss or reconstruct or modify the same process.
- Players will compare the result value on the Display to the target number – the closer it is the better.
- You or your pupils may also like to introduce additional rule(s) for instance: the player will lose the game if they exceed the target number etc.

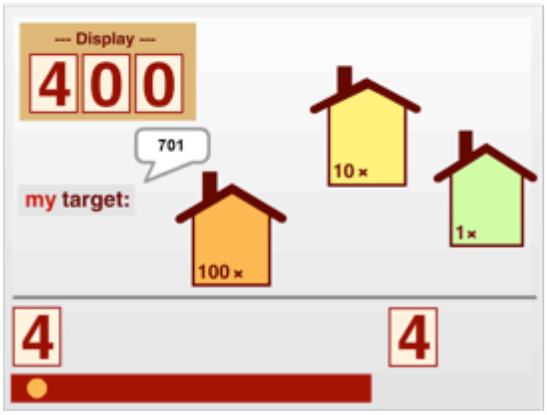
1



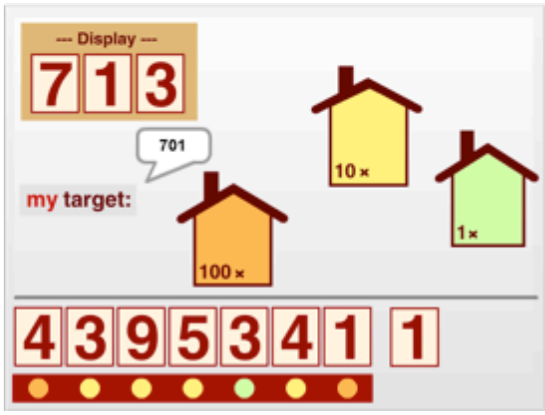
2



3



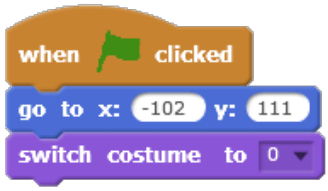
4



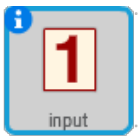
ADDITIONAL SUPPORT

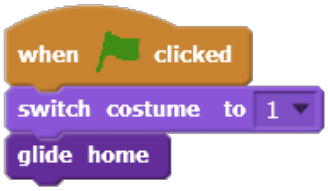
*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

1



This sprite will increase by 1 (i.e. apply the **next costume**) when it receives the *add 1* message.

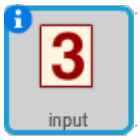




This sprite has only one pre-built (setup) script to place it in an initial position.

This sprite will glide to its initial (home) position **when green flag clicked**. In the Conversion Game it will be dragged around so that is the reason – we will use the **glide home** block to get it back. Also, when clicked it will increase by 1.

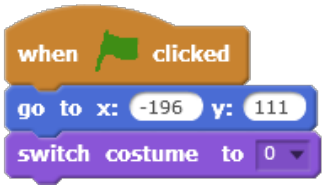
2

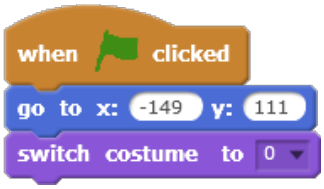


Now we can set the **input** sprite to any digit by clicking it. When we click this new script on the left, the **input**'s value (i.e. its **costume #**) will be added to **ones**.

We will use this 'exploration' script later in other **input**'s scripts.

3





4

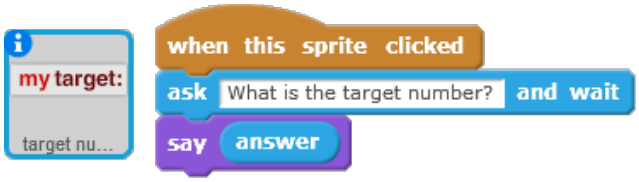
ADDITIONAL SUPPORT CONTINUED

5

The script on the left will add the value of the **input** sprite (i.e. its **costume #**) to the **hundreds**. The script in the middle will add that value to the **tens** – possibly causing an increase (i.e. nudge) to **hundreds**. The script on the right will do the same to **ones**, with possible increase (i.e. nudge) to **tens**.

In the example on the left we initiated the Display to 000 (by clicking the green flag), set **input** to 7, then added it to **ones**, then **tens**, then **ones** again and **tens** again. Discuss and explore: *Does the order of the steps (in this case) matter?*

6



EXTRA CHALLENGE

Using the project as developed to this point can be used for challenging discussions and tasks such as: (note - always start by resetting the Display to 000)

- Using only **input** values 3 and 7 (in any order, once or several times), get to 110 on the Display.
- Using any even **input** value, get to an odd number.
- Using any odd **input** value, get to an even number.
- Using only **input** values smaller than 6, how quickly can we get to 999?
- Using only 5 as an **input** value, how quickly can we get to 100?
- Which **input** value will cause the most 'nudges' on the Display?
- Using only 7 as an input value, is it possible to get to 112? How many steps will be needed?

Module 4 • Investigation 3 • Activity 4.3.3

[Extension] Building the Conversion Game: Coloured Houses

Learning Objectives

Envisage how to update the 1s, 10s and 100s digit sprites based on the value of an input sprite.
Explore how to use the touching colour conditional to update different digit sprites.

Activity Instructions

Pupils continue in their own version of project **4-Conversion Game**, or open the **4-Conversion Game INT1**, **Save as a copy** (online) or **Save as** (offline) and rename. The final version of this project at the end of Activity 4.3.3 will be **4-Conversion Game INT2**.

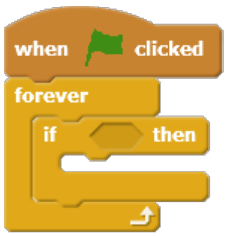
In step 5 of the previous activity we built three similar scripts for the **input** sprite to test the **Display**: they were adding the value of **input** to **hundreds**, to **tens** and to **ones** sprites.

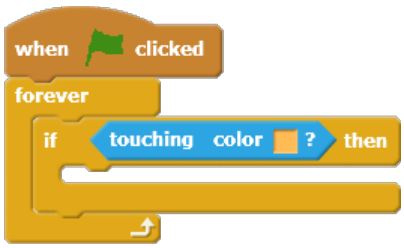
1. Pupils turn three **repeat costume #** … scripts of the **input** sprite into three new blocks named e.g. **in orange**, **in yellow** and **in green** (as they will be run when we put **input** into one of the three differently coloured houses).

In the first extension of Activity 4.1.4 we were adding an **input** value by dragging that number atop **ones**. Now we will build a similar behaviour here: its value will be added to one of the **Display**'s sprites by dragging **input** atop one of the coloured **conversion houses**.

Note that houses are not sprites, they are part of the backdrop. However, as each house has its own distinctive colour, the **input** sprite can react to touching that colour.

2. In the **input** sprite pupils build three *whenever scripts*, to monitor at any time (**forever**) if the sprite is touching the predominant colour of each house. E.g. below is the "empty" script for the orange house (i.e. **100 ×** house):





3. What should happen when the **input** sprite is dragged atop the **100 ×** house? Its value should be added to the **hundreds** sprite, then it should **glide** back home. Pupils complete all three *whenever scripts* so that each house (**100 ×**, **10 ×** and **1 ×**) properly contributes to the **Display**.

4. Finally, pupils modify the **input**'s behaviour: when clicked, it will switch to a random value (costume).

Mathematics Connections

The coloured houses can be thought of as **function machines**, they have an input, an operator (this is like a rule) and calculate an output.

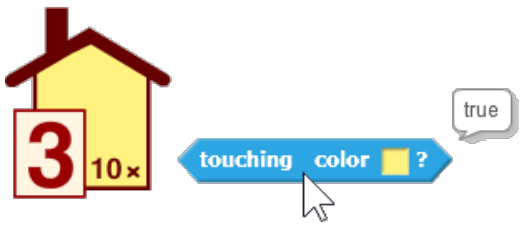
ADDITIONAL SUPPORT

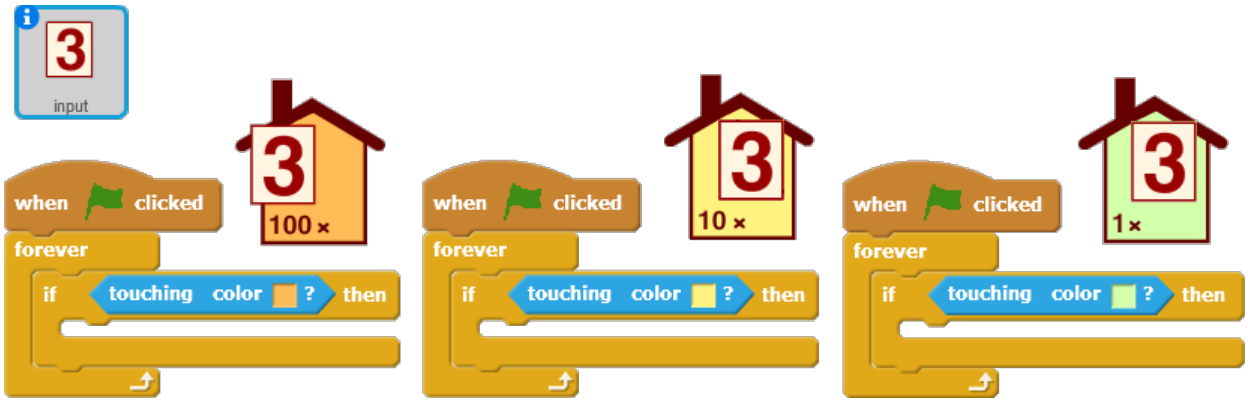
*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

1

2

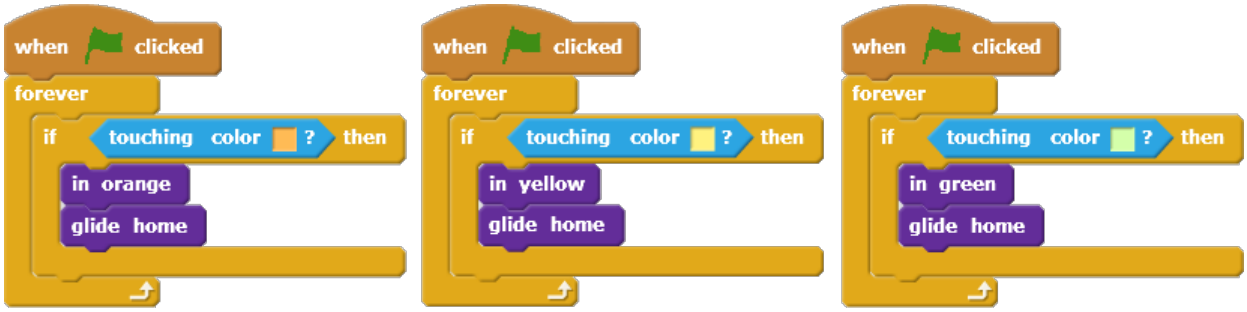
Although each house has two colours – and the **input** sprite is in fact also touching the background colour, the distinctive condition of a house is its predominant and unique colour. Therefore, the **touching color … ?** block can be used by the **input** sprite to know it is over any particular house.





3

These three scripts are the key part of the Conversion Game as they define what will happen when the **input** value is dragged atop one of the 'conversion' houses, i.e. how exactly the Display should react. Tweaking these three scripts will be the main instrument to solve all conversion tasks of Investigation 4.



ADDITIONAL SUPPORT CONTINUED

4 Below there are three different strategies. Pupils may also come with their own.

(a) The sprite switches in one go to a randomly chosen costume.
(b) The sprite goes to the next costume with the random repeat number.
(c) The sprite repeatedly switches to a randomly chosen costume – this approach looks almost like throwing a die.

a

b

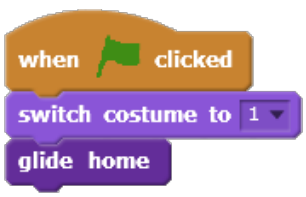
c

All of these solutions, however, will cause a problem when the **input** value is set to 0 – dragging the sprite to any house than causes adding 10 either to **ones**, **tens** or **hundreds**, thus corrupting the Display value. Explore, discuss and explain. The following **[Advanced Extension]** addresses this issue.

ADVANCED EXTENSION AND SUPPORT

You may have noticed that when the **input** value is 0 (i.e. when it displays its 10th costume) and we add it to any house, it ten times increases corresponding digit of the Display – which will end up showing the same costume of that particular sprite, however, one false 'nudge' message will be triggered in between (from **tens** to **hundreds** or from **ones** to **tens**), thus corrupting the final value. How can we solve this problem and debug the scripts?

A simple solution would be not to allow value 0 as a value of **input**. This would require the modification of the **input**'s *setup script* (always start with 1) as well as the **when this sprite clicked** script **(a)** or **(c)** (from the options above) so that it never selects 10th costume – which is 0.



Another approach would be to modify the **add to hundreds**, **add to tens** and **add to ones** scripts so that they do nothing if the **input** value is 0. A further another solution would be to delete 10th costume of **input** completely.

[Extension] Conversion Game: Record Keeping

Learning Objectives

Explore how to keep a record of the input digits and conversion house colour.
Envisage the created number based on the record.

Activity Instructions

Pupils continue in their own version of project **4-Conversion Game**, or open the **4-Conversion Game INT2**, **Save as a copy** (online) or **Save as** (offline) and rename. The final version of this project at the end of Activity 4.3.4 will be **4-Conversion Game FINAL**.

The final phase in the development of the Conversion Game will be building a complete record of the sequence of the input numbers and corresponding colours of selected houses.

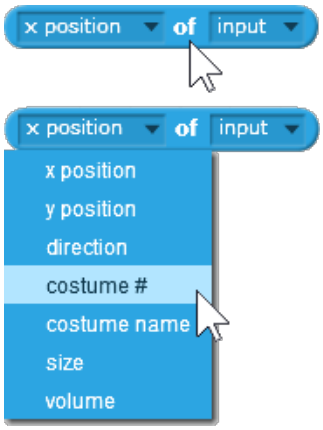
1. Pupils duplicate one of the sprites from the Display, rename it to **record** and delete all its scripts except the *setup script*. They modify the *setup script* so that the sprite's initial position will be in the lower left corner as shown below and it will **forever** show the same costume as the **input** sprite. To do so use the **... of ...** reporter block from the **Sensing** group.

2. The **record** sprite will react to the *record* message (i.e. a signal to record next step). It will **stamp** itself and jump to the right by 46 steps. Pupils build this behaviour.
3. Now, who and when will broadcast that message? It should happen whenever the **input** sprite has been dragged to one of the houses.
   Pupils insert **broadcast** *record* into all three *whenever* scripts of the **input** sprite.

Finally, we want to draw big coloured dots below each stamp of the **record**. This will be done by the **record** sprite itself. To do so, it must know which colour to use in each case.

4. Pupils change the three **broadcast** *record* blocks of the **input** sprite into three different messages:
   broadcast *record orange*, **broadcast** *record yellow* and **broadcast** *record green*.
5. Pupils extend the **record**'s *setup script* so that it sets a big **pen size** and **clears** the stage. Then build three reactions to the broadcasts to draw a dot, stamp itself and jump right.

Mathematics Connections



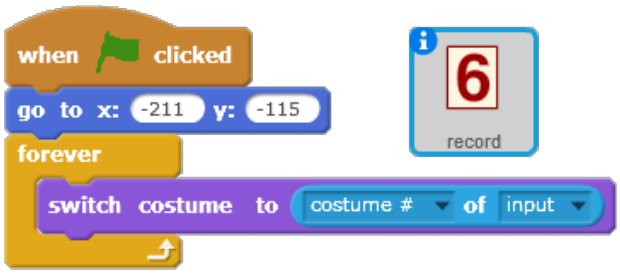
Discuss: *What is the algorithm for the recording sprite?* [switch costume, stamp, move right] *How do we find the position for the next stamp?* [trial and error]

Discuss: *How could we solve the problem of keeping a record of the house colour the digit was dragged to? How does the recording algorithm change?*

CONNECTIONS TO Y5 SCRATCHMATHS

*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

2 In Module 1, Activity 1.1.1 we started using the **stamp** block, which makes a sprite print its costume in the stage.

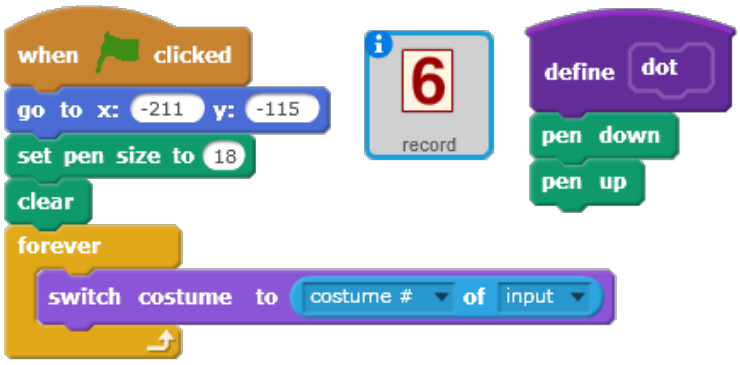


The **forever** block in the *setup script* makes the **record** sprite always display the same number as the **input** sprite. When it receives the *record* message, it will **stamp** itself and jump by increasing its **x position** by 46.

This means that all cards with digits to the left of the **record** sprite are stamps in the stage and will disappear when we use the **clear** block (to clear the stage).

5 In Module 2, Activity 2.1.1 we found out that each sprite had a pen and could draw a line when its pen was on (in Scratch, however, we say **pen down**). When its pen is up, it leaves no line behind when moving. We learned there how to **set pen colour** and how to **set pen size**. We now have to add these blocks to the **record**'s scripts, including the **clear** block to clear the stage from the previous stamps when starting again.

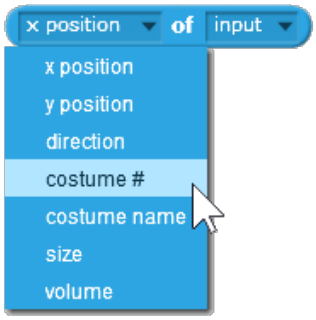
In Module 2, Activity 2.3.1 we discovered how to draw a dot: If a sprite turns its **pen down** and then **pen up**, it will draw a dot. Its colour and size is that of its current pen colour and pen size. Try the following scripts (the one in the bottom left is only for explorations, we will not keep it):

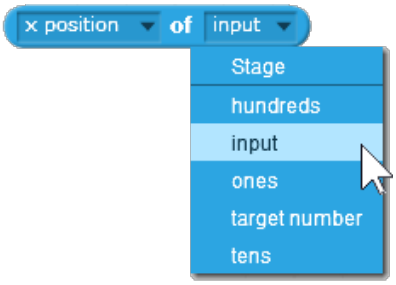


ADDITIONAL SUPPORT

*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

1

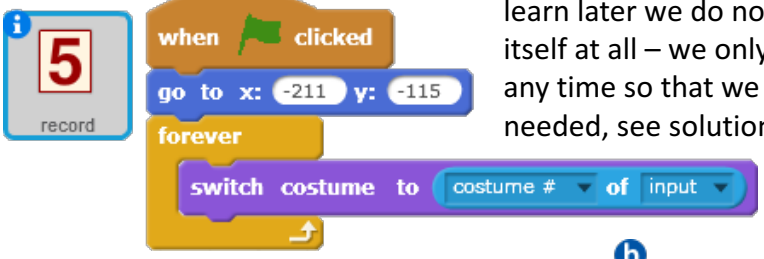




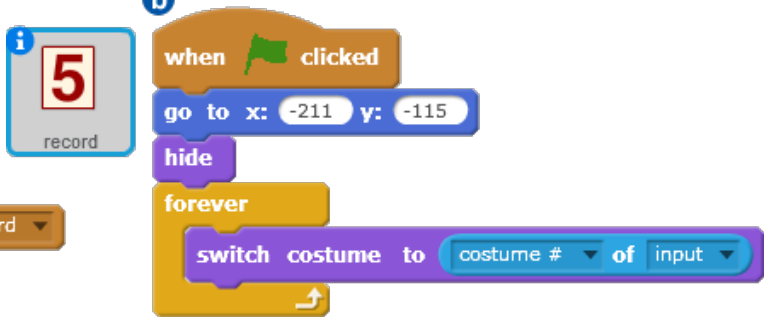
You are going to use here a new powerful reporter block which is useful whenever one sprite needs to learn anything **about another sprite** (e.g. its **x position**, **y position**, **direction**, **costume #**... etc.).

During the whole game the **record** sprite will have exactly the same costume as **input**. However, we will learn later we do not have to see the **record** sprite by itself at all – we only need it to be the same as **input** at any time so that we can **stamp** it as a record whenever needed, see solution (see solution **b** below).

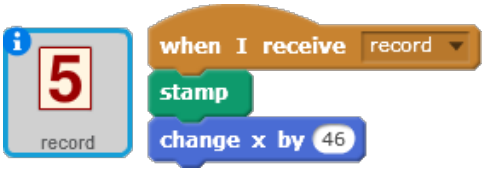
a



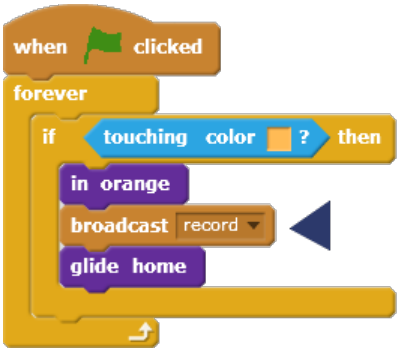
b

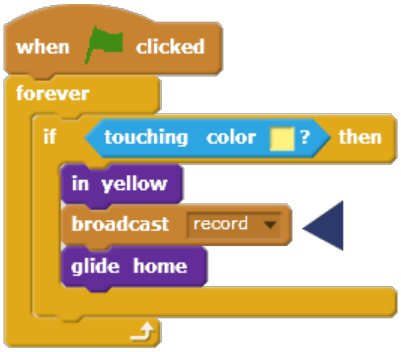


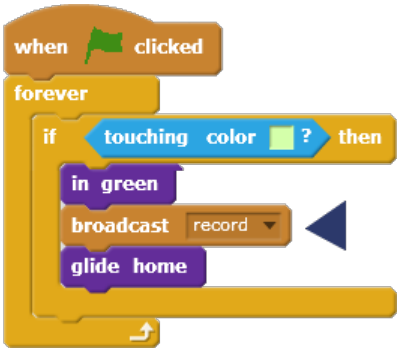
2



3 Whenever the **input** is dragged in one of the houses, the record of that step should be built: the **broadcast** *record* block after **in orange**, **in yellow** and **in green** will do the job (in fact, it can be inserted anywhere inside the *whenever* structure: whenever the **input** sprite touches one of the conversion houses, its value should be added to a corresponding digit of the Display, a record of that step should be taken and then the sprite should glide home).





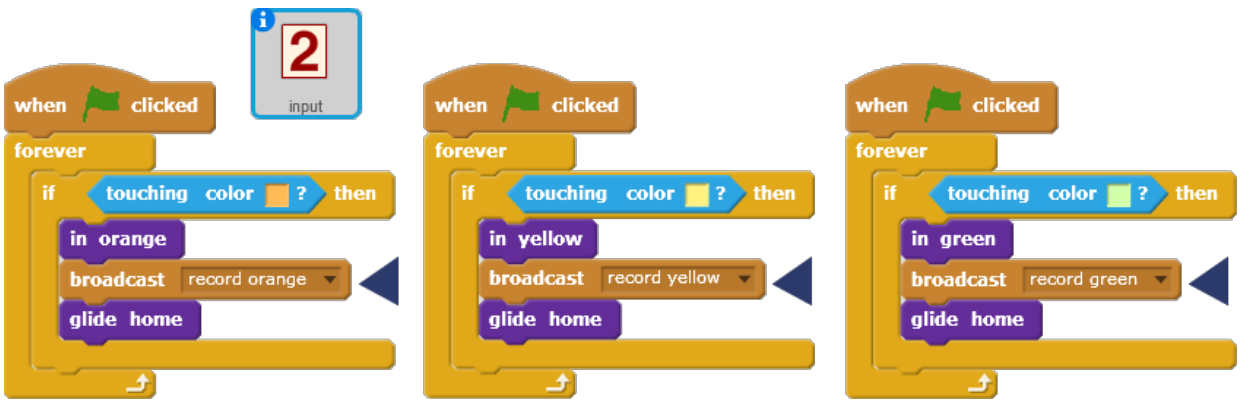


ADDITIONAL SUPPORT CONTINUED

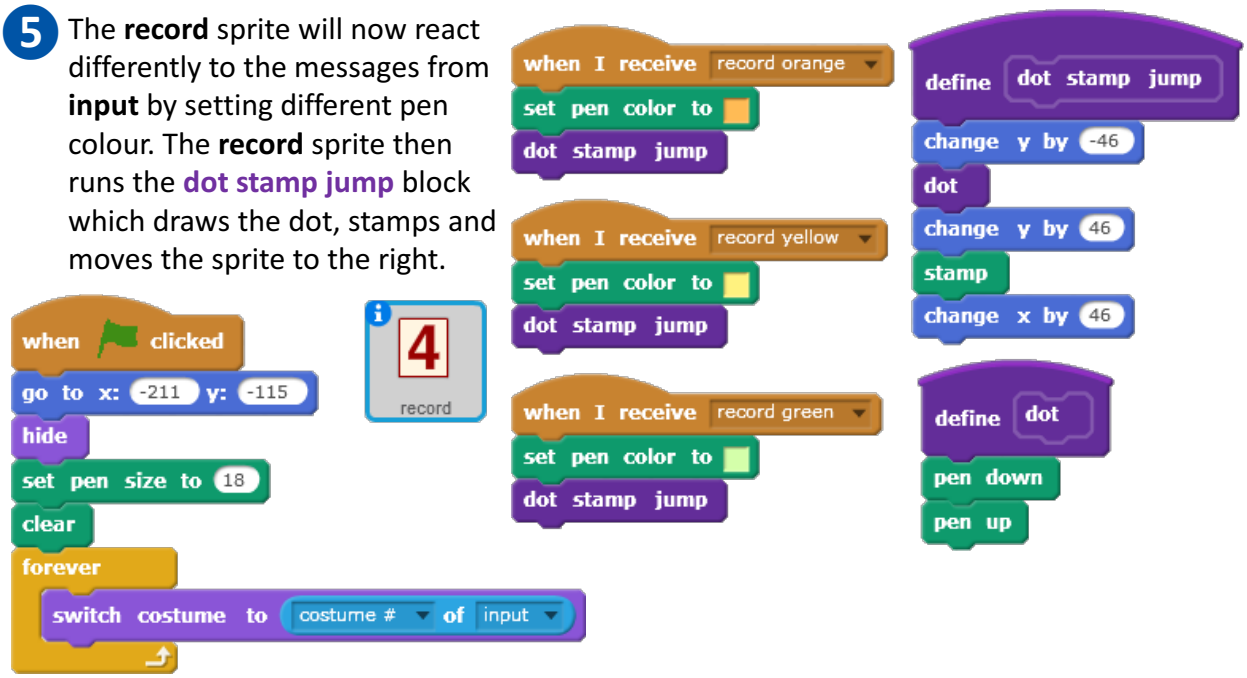
So far we have implemented the first part of keeping the record – we keep the stamps of the input values (see on the left below). However, from this record we would not be able to reconstruct the process because we would not know which value was dragged into which house. Therefore, in the final solution we draw a big dot of the colour of the corresponding house below each recorded value (see on the right below).

To do so, we will use the fact that each sprite – including the **record** sprite – has a pen and can draw (in this case, a big dot).

4 In the **input**'s three *whenever* scripts we replace the **broadcast** *record* by three different messages: **broadcast** *record orange*, **broadcast** *record yellow* and **broadcast** *record green* so that the **record** sprite can react in a slightly different way (setting different pen colours).

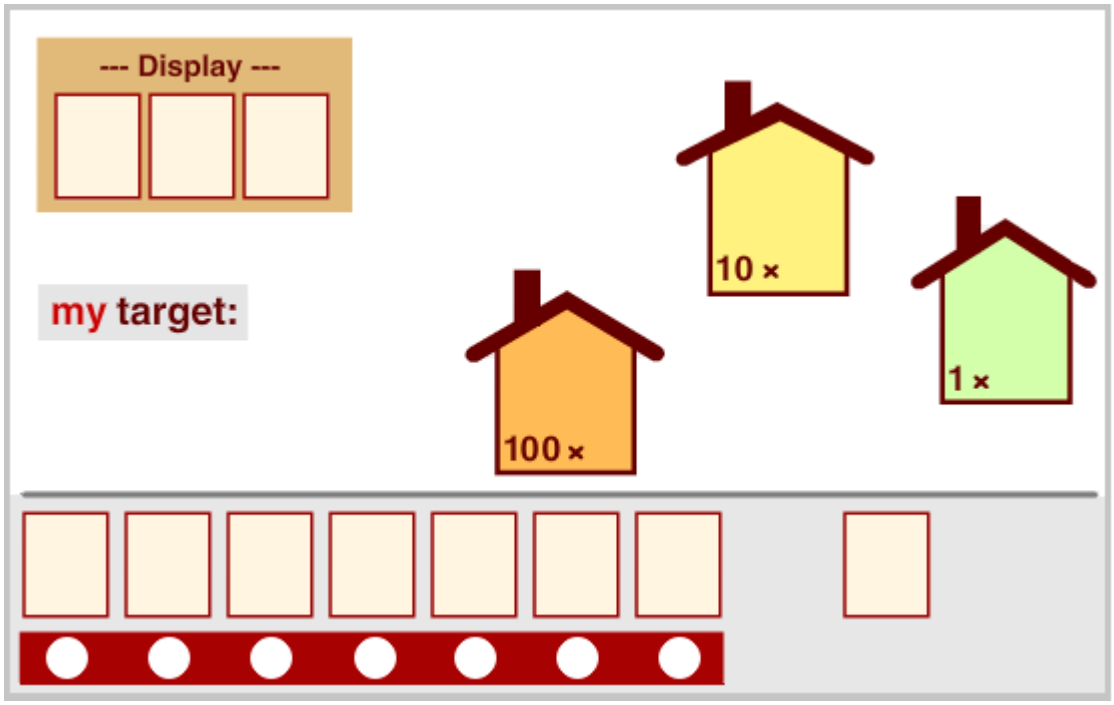


5 The **record** sprite will now react differently to the messages from **input** by setting different pen colour. The **record** sprite then runs the **dot stamp jump** block which draws the dot, stamps and moves the sprite to the right.



ACTIVITY WORKSHEET

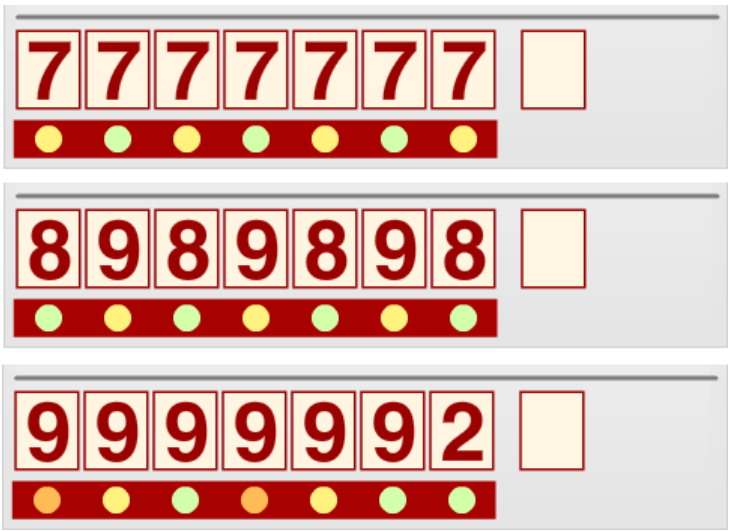
Use the worksheet below to play the entire Conversion Game as an unplugged activity. Each empty line should be used to keep record of one round of the game. Use colour or codes. Label with a key so it's clear and someone can follow your working.



SUGGESTED TASKS

Revisit the unplugged activity that was played at the beginning of this investigation but explore some more challenging questions such as:

- What number would be displayed based on these records?



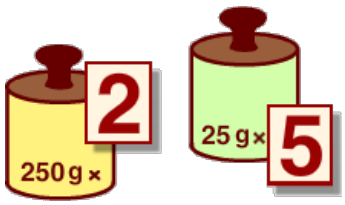
- Select a target number and find three different ways how to build it.
- Is it possible to build 132 using only the input value 6? (use as many times as needed)
- Is it possible to build 201 using only input values 6, 7, 8 and 9? (use as many times as needed)

MODULE 4: INVESTIGATION 4

Exploring Conversions

In the previous investigation pupils developed the Conversion Game project in which they were building a target integer (whole) number (< than 1000) out of randomly generated digits. The player had to decide where to drag an input digit – atop the **100 ×** house, **10 ×** house or **1 ×** house. Now they will explore and play similar games with different measurements: the target number will either be a **length**, e.g. 5.150 km, a **mass**, e.g. 8.300 kg, or a **time**, e.g. 4:29. The Y6 curriculum requires pupils to use, read, write and convert between standard units, converting measurements of length, mass, volume and time from a smaller unit of measure to a larger unit, and vice versa. This investigation provides a context for exploring these conversions, allowing the functionality of the game to be easily adapted to a range of different units and quantities of measurement.

- **Activity 4.4.1** – Converting Length
- **Activity 4.4.2** – Converting Mass
- **Activity 4.4.3** – Converting Time



Scratch starter projects

4-Converting Length
4-Converting Mass
4-Converting Time

LINKS TO PRIMARY NATIONAL CURRICULUM

CURRICULUM OBJECTIVES	LINK WITH SCRATCHMATHS
Mathematics Solve problems involving the calculation and conversion of units of measure, using decimal notation up to three decimal places where appropriate. Use, read, write and convert between standard units, converting measurements of length, mass and time from a smaller unit of measure to a larger unit, and vice versa, using decimal notation to up to three decimal places.	► Pupils are required to envisage conversions between different amounts of metres into kilometres (to 3 decimal places), as well as in extension activities between grams and kilograms and minutes and hours, through playing different versions of the conversion game.
Find fractions of quantities.	► There is opportunity to discuss and explore what fraction of the larger unit (e.g. 1 kilometre) different amounts of the smaller unit are (e.g. 250 m or 100 m).
Solve number and practical problems involving place value. Solve problems involving converting between units of time.	► **[Extension]** Pupils can edit the pre-built conversion games to modify the conversion amounts and are required to understand place value within the context of the different units of measures in order to do this.

Converting Length

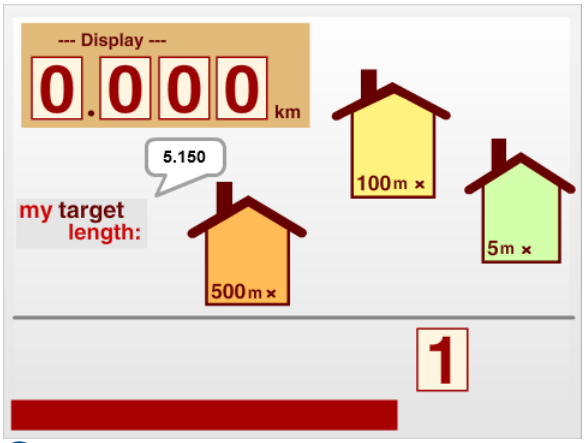
LEARNING OBJECTIVES

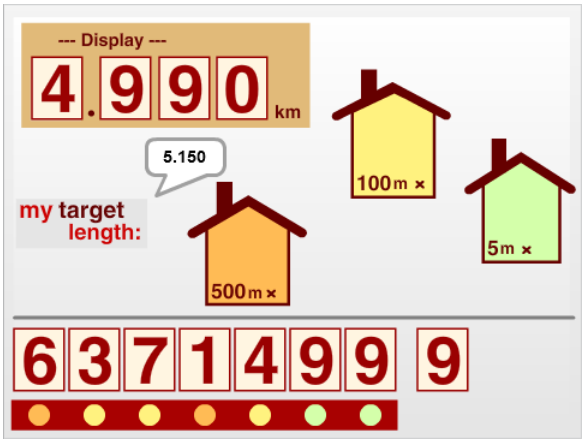
Explore how to convert different amounts of metres into kilometres.
Envisage the increase in the kilometre display when adding different amounts of metres.
bridgE to conversions of units of measure in mathematics.

ACTIVITY INSTRUCTIONS

This is Conversion Game which asks for a target length and provides the player with seven randomly generated one-digit input values. The player has to decide where to drag each input digit – atop the **500m ×** length, **100m ×** length or **5m ×** length. The target number will be a length, e.g. 5.150 km which the player will try to reconstruct – the closer the better. Alternatively you may agree an additional rule: if the player exceeds the target length, they lose the game.

Pupils open project **4-Converting Lengths**, **Save as a copy** (online) or **Save as** (offline) and rename.





1. Pupils explore the project, run it by clicking the green flag and explore how each conversion house contributes to the Display, i.e. how it transforms an input value.
2. Pupils explore the **input**'s scripts in particular so that they understand what happens with the input value when dragged atop the **500 m ×** house, **100 m ×** house or **5 m ×** house.
3. Pupils solve the suggested tasks from the next page or alternatively play the entire Converting Length game as an unplugged activity, using the worksheet provided later in this activity.

MATHEMATICS CONNECTIONS

Discuss how to convert from m to km.

Discuss:

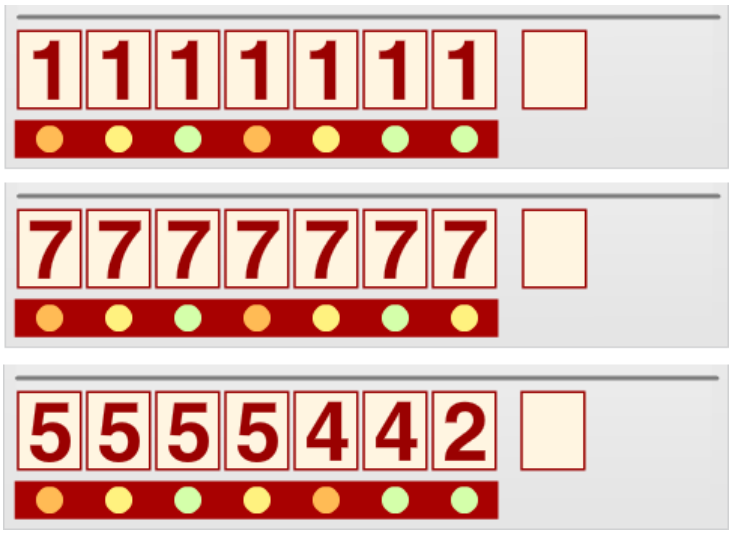
How many metres in 1 km?
How many 500 metres in 1 km?
How many 5 metres in 1 km?

The project is designed to provide opportunities for pupils to explore m to km conversion. They should be encouraged to envisage the change to the display before moving the sprite to a coloured house and to share their thinking aloud; model this approach with the class

SUGGESTED TASKS

Solve with the pupils some of the following or similar tasks:

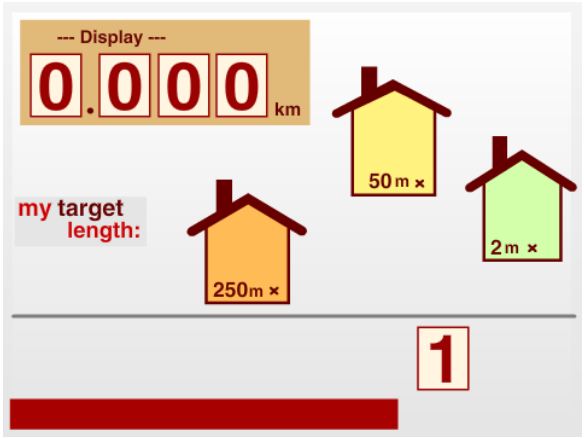
- What length would be displayed based on these input records?



- If instead of 1s in the first input record we use 2s (with the same colour marks), what will the result be? What is the relationship between the outcome lengths displayed when we use all 1s and 2s? What about 1s and 7s?
- Which record of 7 input digits would produce the smallest possible length?
- How could you build the 9.999 km target number?
- What is the biggest length that can be built in 7 steps?

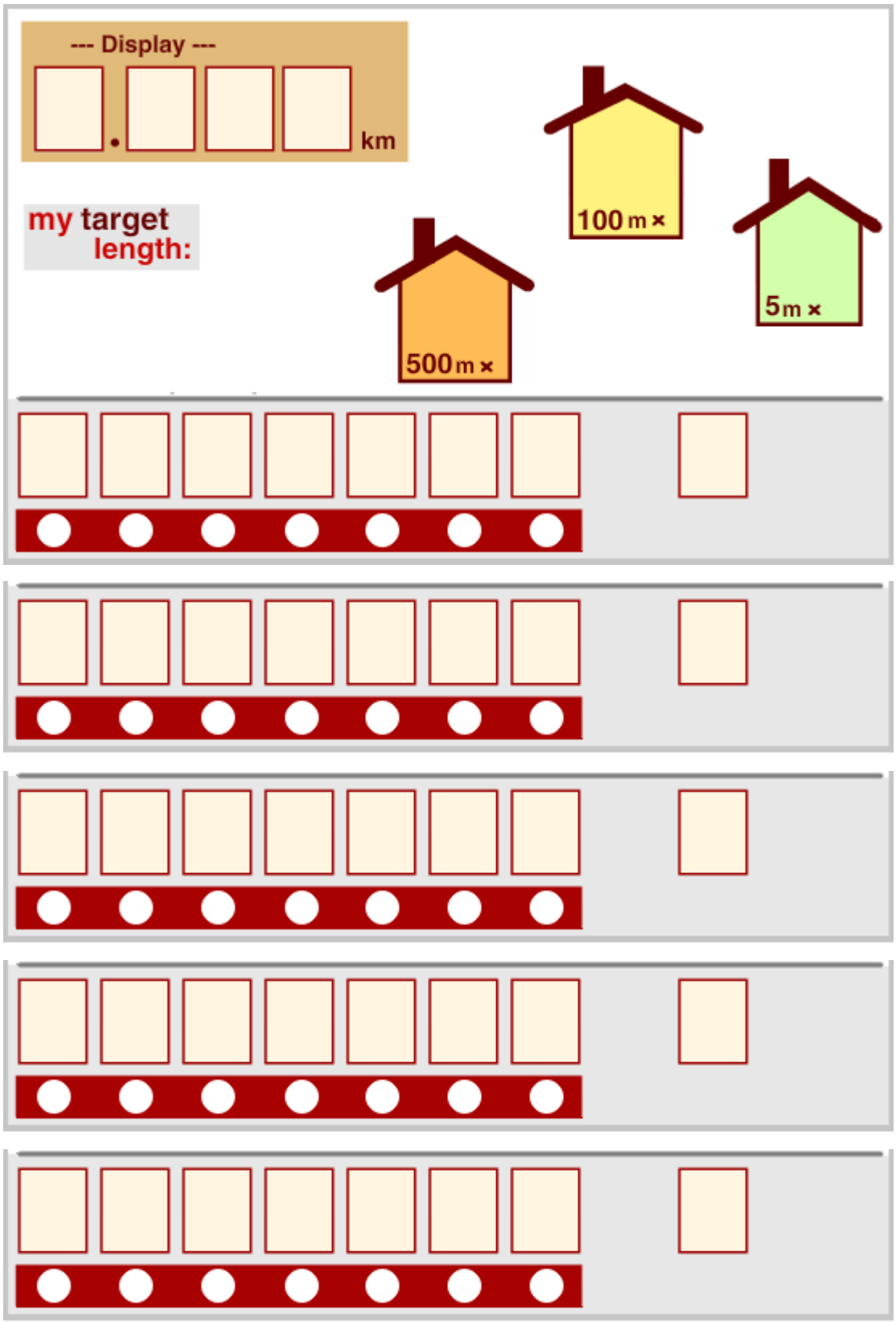
EXTENSION

4 Pupils switch the backdrop to the second one, with slightly different conversion lengths. They modify the **input**'s scripts so that the proper value is always added to the **Display**.



ACTIVITY WORKSHEET

This worksheet can be used to document the task solutions in the Converting Length activity or used unplugged. Think of your own target length and random input values.

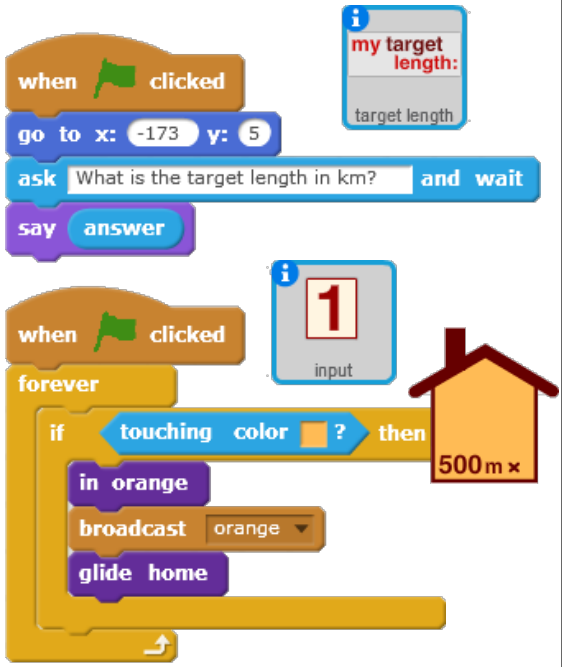


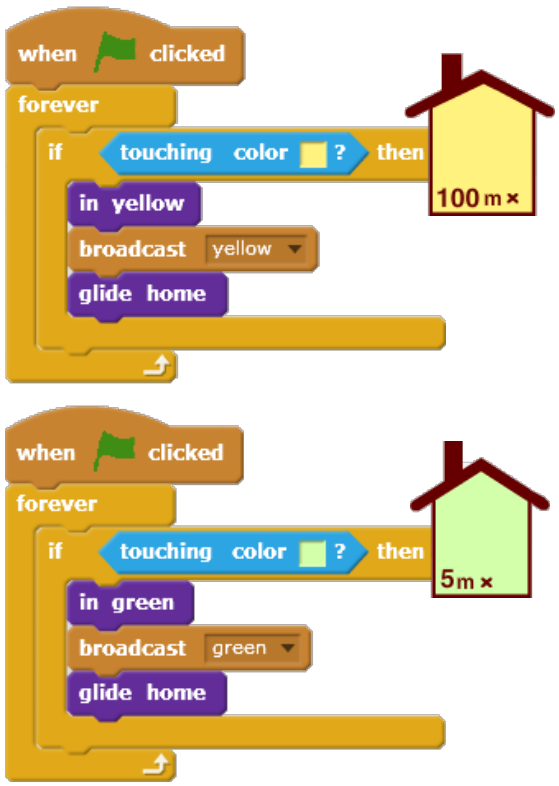
ADDITIONAL SUPPORT

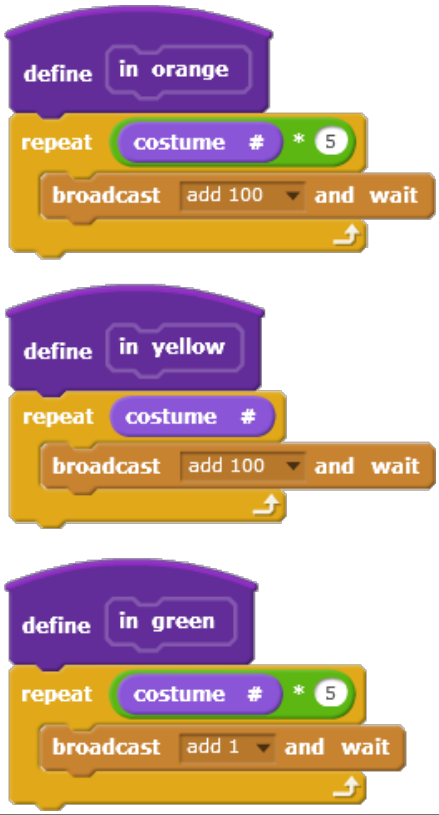
*Please note the **blue** numbers on the left link to the numbered steps in the **activity instructions***

The **4-Converting Length** project is a small modification of what has been developed in Investigation 3. There are three key differences (only the last one is essential):

- The **Display** now consists of four digit sprites – **ones**, **tens**, **hundreds** and **thousands**. They behave in an identical way to before, reacting to the *add 1*, *add 10*, *add 100* and *add 1000* messages being broadcast by the **input** sprite. When appropriate, they "nudge" the digits to the left to implement increase by 1.
- The target number sprite is renamed to **target length** and its two scripts are merged into one, so that it immediately asks for the target length after a player clicks the green flag.
- The most important difference is the way how the **input** sprite reacts to being dragged atop the houses. E.g. if the input value is 3 and we drag it atop the **500 m ×** house, our own **in orange** block is run and increases the **hundreds** sprite not 3 times, but **3 × 5 –** as we are adding 5 hundreds in each go.
  Note also that we removed all **wait** delays to make things happen quicker, but replaced some **broadcast** *message* blocks by **broadcast** *message* **and wait** – this ensures we do not lose any reaction by broadcasting the same message repeatedly at very high speed.







MODULE 4 • INVESTIGATION 4 • ACTIVITY 4.4.2
[Extension] Converting Mass

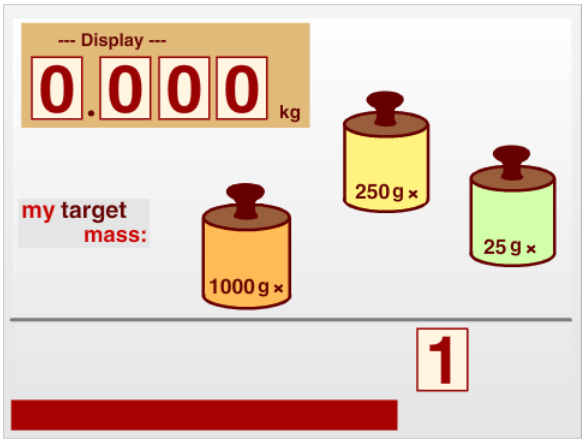
LEARNING OBJECTIVES

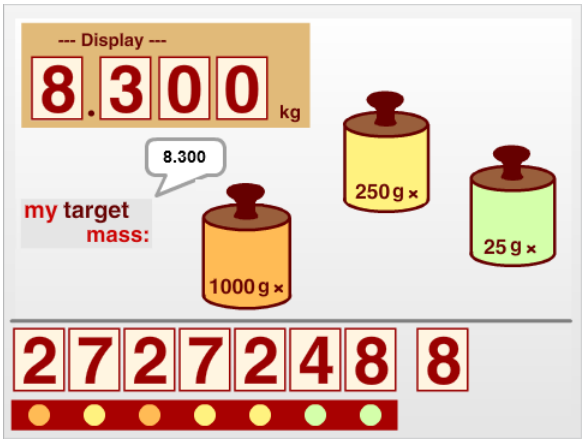
Explore how to convert different amounts of grams into kilograms.
Envisage the increase in the kilogram display when adding different amounts of grams.
bridgE to conversions of units of measure in mathematics.

ACTIVITY INSTRUCTIONS

This is another Conversion Game which asks for a target mass/mass (the terms are used interchangeably in KS2) and provides the player with seven randomly generated one-digit input values. The player has to decide where to drag each input digit – atop the **1000 g ×** mass, **250 g ×** mass or **25 g ×** mass. The target number will be a mass, e.g. 8.300 kg which the player will try to reconstruct – the closer the better. Alternatively you may agree an additional rule: if the player exceeds the target mass, they lose the game.

Pupils open project **4-Converting Mass**, **Save as a copy** (online) or **Save as** (offline) and rename.





1. Pupils explore the project, run it by clicking the green flag and explore how each conversion weight contributes to the Display, i.e. how it transforms an input value.
2. Pupils explore the **input**'s scripts in particular so that they understand what happens with the input value when dragged atop the **1000 g** weight, **250 g** weight or **25 g** weight.
3. Pupils solve the tasks from the next page or alternatively play the entire Converting Mass game as an unplugged activity, using the worksheet provided later in this activity.

MATHEMATICS CONNECTIONS

Discuss how to convert from g to kg.

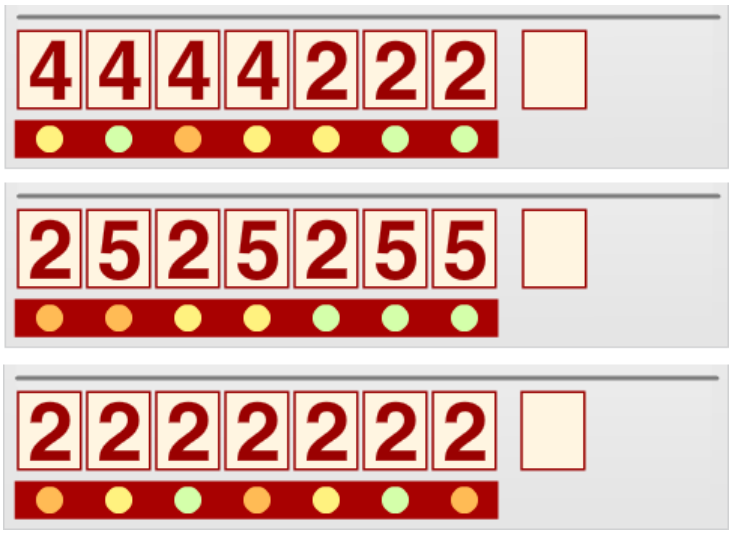
Discuss: *How many grams in 1 kg? How many 250 grams in 1 kg? How many 25 grams 1 kg?*

The questions can be used to make connections with fractions, e.g. 250 grams is ¼ of a kilogram.

SUGGESTED TASKS

Solve with the pupils some of the following or similar tasks:

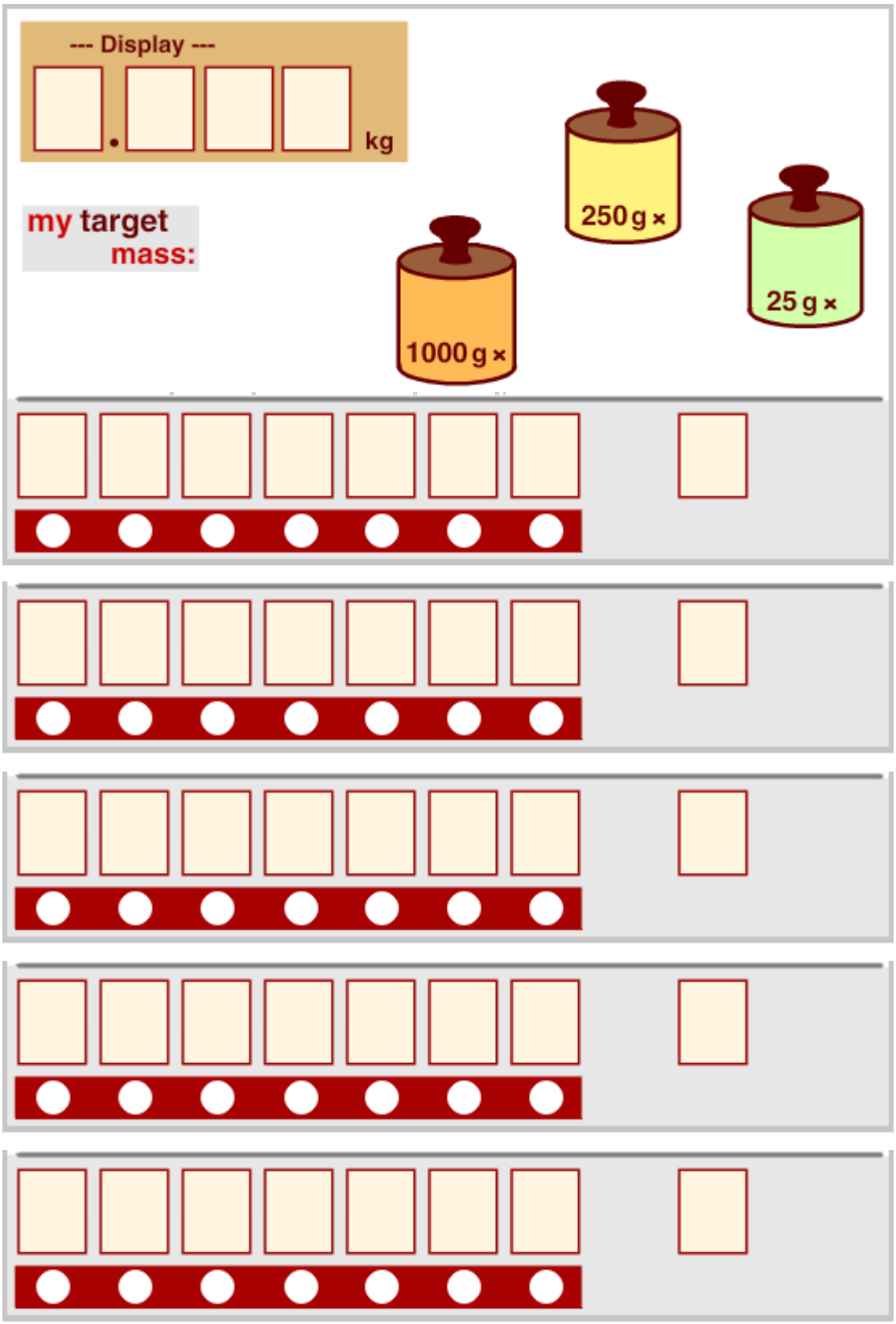
- What mass would be displayed based on these input records?



- Which record of 7 input digits would produce the smallest possible mass?
- How could you build the 9.995 kg outcome value?
- What is the biggest mass that can be built in 7 steps?
- What is the biggest mass you can get using only 9 as the input digit and dragging it always over the **25 g** mass?

ACTIVITY WORKSHEET

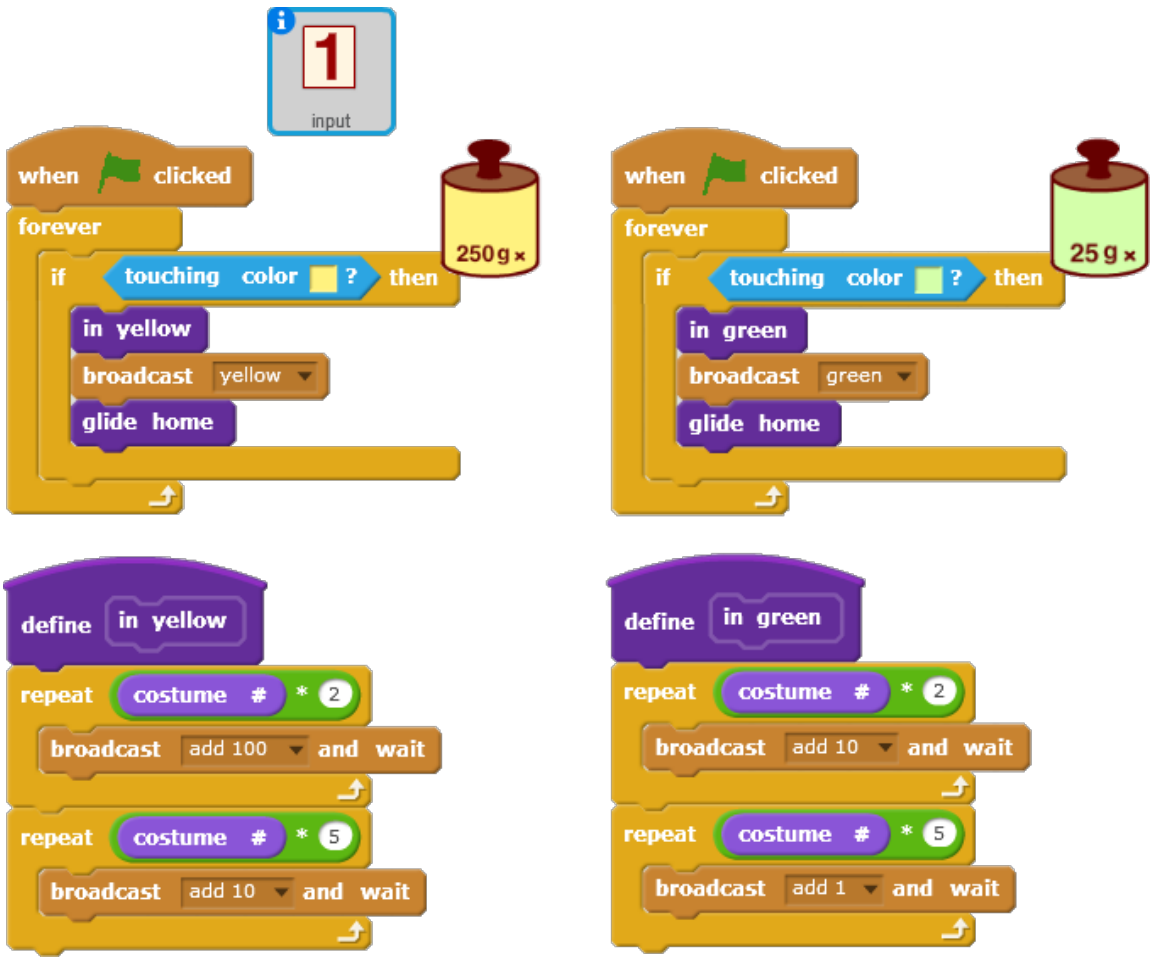
This worksheet can be used to document the task solutions in the Converting Mass activity or used unplugged. Think of your own target mass and random input values.



ADDITIONAL SUPPORT

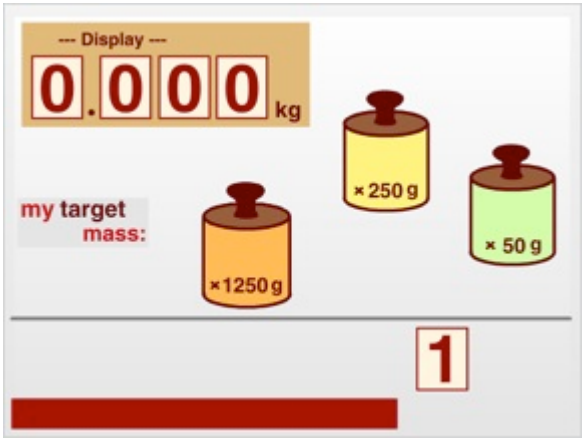
Note that in the case of the **250 g** and **25 g** masses, corresponding reactions **in yellow** and **in green** address not one but two sprites in the Display – increasing e.g. 2 times **hundreds** and 5 times **tens** (the swapped order would make no difference).

An alternative but much slower solution would be to increase 25 times **tens** only.



ADVANCED EXTENSION

4 Pupils switch the backdrop to the second one, with slightly different conversion masses. They modify the **input**'s scripts so that the correct value is always added to the Display.



[Extension] Converting Times

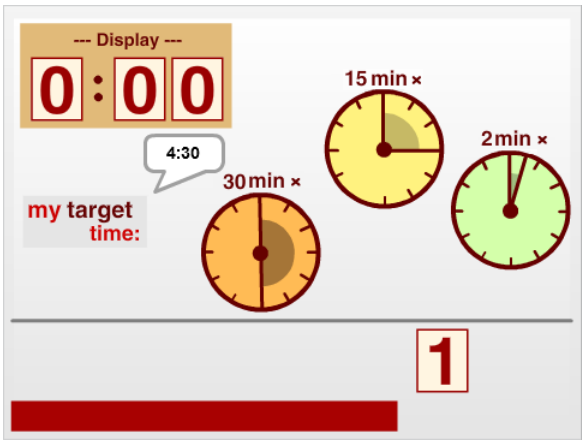
LEARNING OBJECTIVES

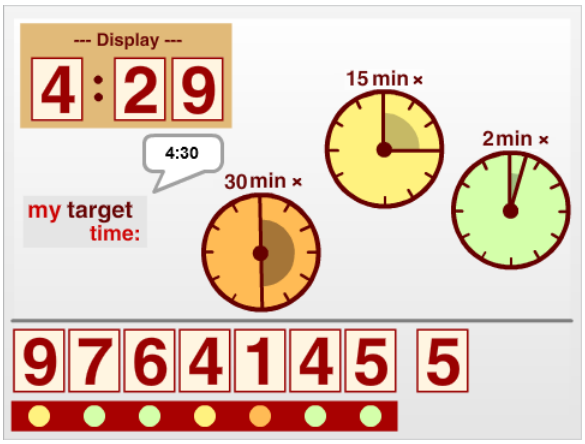
Explore how to convert different numbers of minutes into hours.
Envisage the increase in the hour display when adding different numbers of minutes.
bridgE to conversions of units of measure in mathematics.

ACTIVITY INSTRUCTIONS

This is another Conversion Game which asks for a target time and provides the player with seven randomly generated one-digit input values. The player has to decide where to drag each input digit – atop the **30 min ×** dial, **10 min ×** dial or **2 min ×** dial. The target value will be a time, e.g. 4:29 which the player will try to build – the closer the better. Alternatively you may agree an additional rule: if the player exceeds the target time, they lose the game.

Pupils open project **4-Converting Times**, **Save as a copy** (online) or **Save as** (offline) and rename.





1. Pupils explore the project, run it by clicking the green flag and explore how each conversion house contributes to the Display, i.e. how it transforms an input value.
2. Pupils explore the **input**'s scripts in particular so that they understand what happens with the input value when dragged atop the **30 min ×** dial, **15 min ×** dial or **2 min ×** dial.
3. Pupils solve the tasks from the next page or alternatively play the entire Converting Length game as an unplugged activity, using the worksheet provided later in this activity.

MATHEMATICS CONNECTIONS

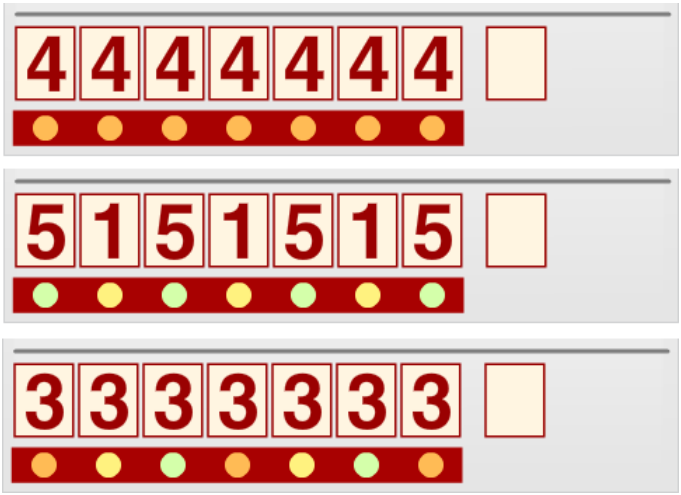
Discuss:

How many minutes in 1hour?
How many 2 minutes in 1 hour?
How many 15 minutes in 1 hour?
How many 30 minutes in 1 hour?

SUGGESTED TASKS

Solve with the pupils some of the following or similar tasks:

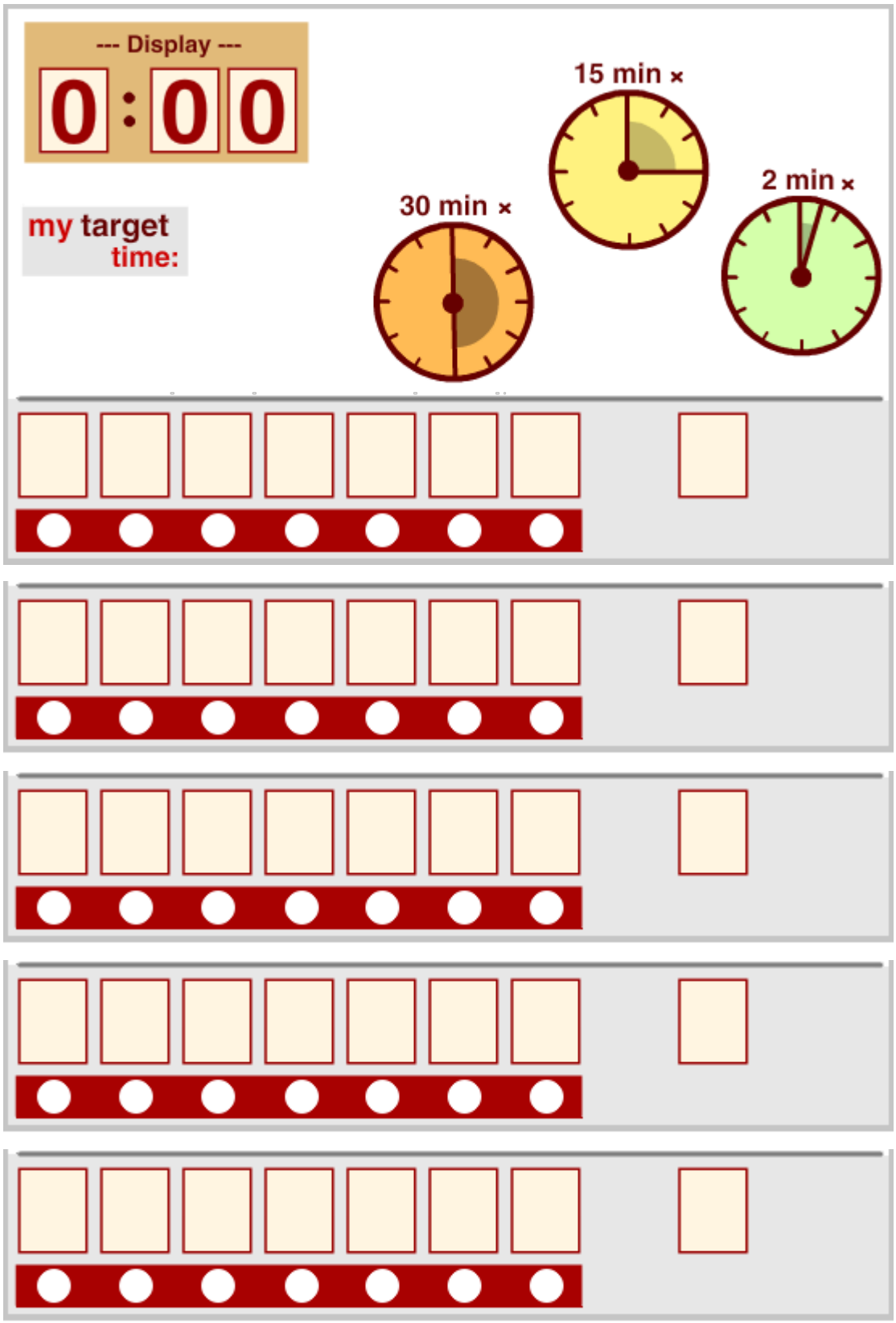
- What time would be displayed based on these input records?



- Which record of 7 input digits would build the greatest possible time?
- Is it possible to build 3:33 in 7 steps?
- Is it possible to build 1:00 in 7 steps? Can you find more than one solution?
- Find at least 3 different times which cannot be built in 7 steps at all.
- What is the greatest time that can be built in 7 steps?

ACTIVITY WORKSHEET

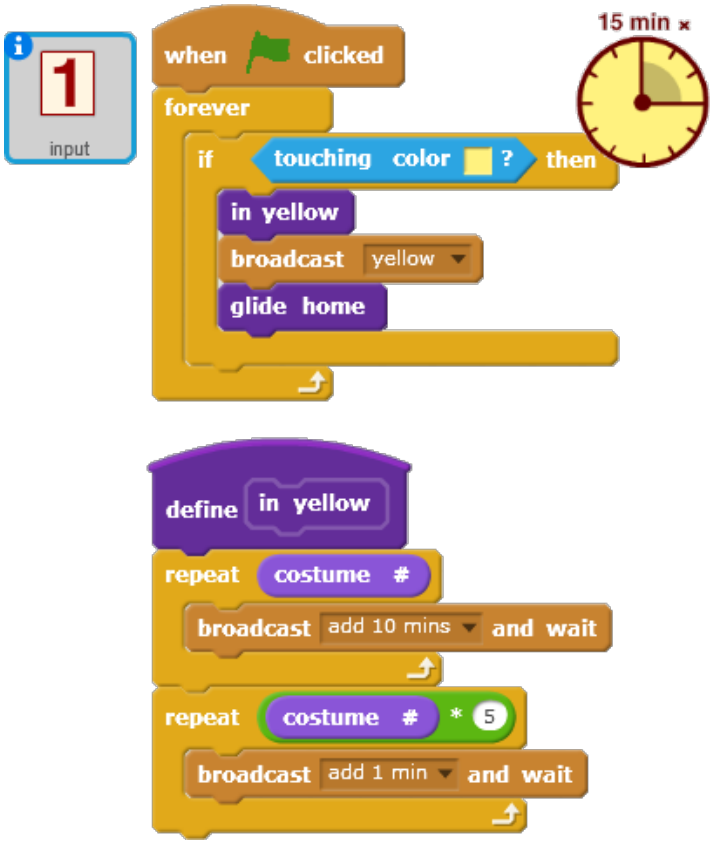
This worksheet can be used to document the task solutions in the Converting Time activity or used unplugged. Think of your own target time and random input values.



ADDITIONAL SUPPORT

Note that in the case of the **15 min ×** dial, the corresponding reaction to **in yellow** addresses not one but two sprites in the Display. If the input value is e.g. 4, it will increase **10 mins** 4 times and **1 mins** 20 times (5 × 4 times).

An alternative but much slower solution would be to increase 60 times **1 mins** only.



ADVANCED EXTENSION

4 Pupils switch the backdrop to the second one, with slightly different conversion times. They modify the **input**'s scripts so that the proper value is always added to the **Display**.