Computing Year 3 2015—2016

Unit 3.2 Step 1:

Karin Clark

Learning Expectations

This unit will enable the children to:

- develop a number of strategies for finding errors in programs
- build up resilience and strategies for problem solving
- increase their knowledge and understanding of Scratch
- recognise a number of common types of bug in software..

Main Session

Share the *Learning expectations* and explain the success criteria. Revise the term ‘algorithm’: an unambiguous procedure or precise step-by-step guide to solve a problem or achieve a particular objective.

Ask the pupils to recite their three-times table. What did they say? Could they phrase the instruction more clearly? Ask them to write an algorithm for reciting the three-times table, e.g. ‘Starting from 1 × 3, multiply each counting number by three, up to and including 12 × 3.’

Show the children the Scratch multiplication script on the CD-ROM. It is meant to read the twelve-times table up to 12 × 3. What’s wrong with it? (It stops at 11 × 3, because it repeats *until* 12 × 3, but not *including* 12 × 3.) Explain that ‘off-by-one’ errors like this are common mistakes in programming.

Ask the pupils to compare the Scratch script to their algorithm. Can they find the *script* block that needs changing so that the cat says the three-times table all the way through? Ask the pupils to fix the program by editing the script and testing the changes they make.

If time allows, you could ask the pupils to record a screencast explaining how they debugged the program.

Resources and e-safety

Pupils don’t need accounts to download Scratch 1.4 or to use Scratch 2.0 or Snap! online.
Children register with a class account
Once registered, pupils can share their corrected and refined programs with the global Scratch community in a safe online space. Alternatively, pupils can upload their completed projects to the school’s learning platform or blog.
If pupils upload screencasts of their solutions, make sure you take the usual precautions to protect their identity.
If pupils use the web for research (see *Extensions*), ensure all usual internet safety protocols are in place Scratch file—example animation

Use Scratch multiplication script

Use Scratch multiplication script: http://scratch.mit.edu/projects/11932181

Mrs Clark - 3 times table

https://scratch.mit.edu/projects/83161952/

Curriculum Links for Unit 3.1—steps 1-6

Literacy	✓✓✓
Numeracy	✓✓✓
Science	✓✓✓
D&T	

Computing Year 3 2015—2016

Unit 3.2 Step 2:

Karin Clark

Learning Expectations

This unit will enable the children to:

- develop a number of strategies for finding errors in programs
- build up resilience and strategies for problem solving
- increase their knowledge and understanding of Scratch
- recognise a number of common types of bug in software..

Main Session

Ask the pupils to draw a circle on a piece of paper. Discuss the differences between how different children drew these. Did some draw much more slowly than others? Did any use compasses or objects to draw round? Were the faster pupils less accurate than the slower ones?

Remind the pupils what an algorithm is (unambiguous, step by-step instructions). Invite them to write an algorithm for drawing a circle, in terms of the steps or rules they followed. Compare the algorithms that they have written.

Head out into the playground or gym, if you can, and practise the following possible approach to drawing a circle.

Repeat until you get back to the start:

- Walk forward one step.
- Turn right a bit.

Ask the pupils to look at the Scratch project at http://scratch.mit.edu/projects/11932410 and try to explain how it works. Run the script. What do the pupils think? The program is meant to draw a circle, or at least a close approximation to one. Did it work? Did it work well? Try to elicit a response of: 'The program takes too long to draw the circle.'

Ask the pupils to look for ways to improve the performance of the program while keeping the size of the 'circle' the same. Ask the pupils to experiment with changing the numbers. (It is not necessary to explain what the angle values mean as pupils can discover for themselves how changing this affects the shape drawn.) At what point does it become obvious that their shape is no longer a circle?

The pupils could record a screencast or audio file explaining how the program works and what they did to improve its performance.

Mrs Clark—Draw a circle

https://scratch.mit.edu/projects/83160628/

Resources and e-safety

Pupils don't need accounts to download Scratch 1.4 or to use Scratch 2.0 or Snap! online.

Children register with a class account

Once registered, pupils can share their corrected and refined programs with the global Scratch community in a safe online space. Alternatively, pupils can upload their completed projects to the school's learning platform or blog.

If pupils upload screencasts of their solutions, make sure you take the usual precautions to protect their identity.

If pupils use the web for research (see *Extensions*), ensure all usual internet safety protocols are in place Scratch file—example animation

Use Scratch circle script

Use Scratch circle script: http://scratch.mit.edu/projects/11932410

Mrs Clark—Draw a circle

https://scratch.mit.edu/projects/83160628/

Curriculum Links for Unit 3.1—steps 1-6

Literacy	✓✓✓
Numeracy	✓✓✓
Science	✓✓✓
D&T	

Computing Year 3 2015—2016

Unit 3.2 Step 3:

Karin Clark

Learning Expectations

This unit will enable the children to:

- develop a number of strategies for finding errors in programs
- build up resilience and strategies for problem solving
- increase their knowledge and understanding of Scratch
- recognise a number of common types of bug in software..

.

Main Session

Ask the pupils to think about occasions when everybody in the class is doing something at the same time, such as collecting things from their trays or tidying up. What sort of problems do they encounter? How can these be avoided? Could the pupils write an algorithm for tidying up the classroom at the end of the day? Would this algorithm work if all pupils followed it at the same time?

Explain that in modern computers, several things can be happening at once, and that sometimes this can cause difficulties – when one process races ahead without another catching up, or when several processes are all waiting for the same shared resource.

Show the pupils the Scratch project listed in *Resources*, which is meant to be two penguins telling a joke. Ask the pupils to identify the problem with this project and try to fix it. It may be helpful to introduce pupils to (or remind them of) the *say ( ) for ( ) secs* and *wait ( ) secs* blocks, and/or the *broadcast ( )* and *when I receive ( )* blocks. (See Step 5 of Unit 3.1 – We are programmers.)

The pupils could record a screencast explaining how the program works and what they did to fix it.
The pupils could adapt their corrected script for other jokes or characters

Mrs Clark Knock, knock

https://scratch.mit.edu/projects/83164482/

Resources and e-safety

Pupils don't need accounts to download Scratch 1.4 or to use Scratch 2.0 or Snap! online.
Children register with a class account
Once registered, pupils can share their corrected and refined programs with the global Scratch community in a safe online space. Alternatively, pupils can upload their completed projects to the school's learning platform or blog.
If pupils upload screencasts of their solutions, make sure you take the usual precautions to protect their identity.
If pupils use the web for research (see *Extensions*), ensure all usual internet safety protocols are in place Scratch file—example animation

Use Scratch penguin script

UseScratch penguin script: http://scratch.mit.edu/projects/11932160

Mrs Clark Knock, knock

https://scratch.mit.edu/projects/83164482/

Curriculum Links for Unit 3.1—steps 1-6

Literacy	✓✓✓
Numeracy	✓✓✓
Science	✓✓✓
D&T	

Computing Year 3 2015—2016

Unit 3.2 Step 4:

Karin Clark

Learning Expectations

This unit will enable the children to:

- ♦ develop a number of strategies for finding errors in programs
- ♦ build up resilience and strategies for problem solving
- ♦ increase their knowledge and understanding of Scratch
- ♦ recognise a number of common types of bug in software..

Main Session

Ask the pupils to describe what happens if they roll a ball towards a wall, either straight on or at an angle. Can they draw a picture to show how the ball would bounce? Do the experiment, changing the angle of impact. Are the pupils surprised? Can they come up with a rule to describe what happens?

Explain that often, programs have bugs because the programmer hasn't fully understood the idea of what's supposed to happen in the program – the bug lies in the concept for the program rather than the code. These sorts of logic bugs can be tricky to find and fix.

Let the pupils play the simple 'pong'-style game listed in *Resources*. Do the pupils notice anything odd about the game? They should spot that the ball doesn't bounce back correctly when it hits the bat (it always bounces off in the direction it came).

Ask the pupils to study the script to work out how the game works, and then identify which block has the bug (if necessary, remind them they need to have the ball sprite selected to see the appropriate script). Ask them to correct the bug, drawing on their knowledge of how a ball bounces from discussions at the start of the session. The correct solution is to use *point in direction* (*180 – direction*). Use your judgement about how much scaffolding to provide.

You might want pupils to record a screencast explaining how the program works and what they did to fix it.

You could ask the pupils to make further changes to the game, perhaps allowing a number of lives, and keeping score.

Mrs Clark—Pong

https://scratch.mit.edu/projects/83166924/

Resources and e-safety

Pupils don't need accounts to download Scratch 1.4 or to use Scratch 2.0 or Snap! online.

Children register with a class account

Once registered, pupils can share their corrected and refined programs with the global Scratch community in a safe online space. Alternatively, pupils can upload their completed projects to the school's learning platform or blog.

If pupils upload screencasts of their solutions, make sure you take the usual precautions to protect their identity.

If pupils use the web for research (see *Extensions*), ensure all usual internet safety protocols are in place Scratch file—example animation

Use Scratch 'Pong' script

Use Scratch 'Pong' script: http://scratch.mit.edu/projects/11932263

Mrs Clark—Pong

https://scratch.mit.edu/projects/83166924/

Curriculum Links for Unit 3.1—steps 1-6

Literacy	✓✓✓
Numeracy	✓✓✓
Science	✓✓✓
D&T	

Computing Year 3 2015—2016

Unit 3.2 Step 5:

Karin Clark

Learning Expectations

This unit will enable the children to:

- develop a number of strategies for finding errors in programs
- build up resilience and strategies for problem solving
- increase their knowledge and understanding of Scratch
- recognise a number of common types of bug in software..

Main Session

Ask the pupils to practise their division in pairs. One pupil could give a number, the other the number to divide by, and both could work out the answer. Ask the pupils to consider how they worked out the answers to the division questions. Can they write their method as an algorithm? For example, 'Start with the number to be divided (the dividend). Count how many times you can subtract the divisor until you get to zero.' Another example might be 'Look up the answer on the times table and the number you multiply by tells you the quotient.' Are there any divisions their algorithm won't work for? A test plan (a list of divisions to try) would be useful to see if there were any special cases when their algorithm didn't work. Check if the pupils' algorithms work for dividing where there is a remainder, e.g. 7 ÷ 3. What about when the children try to divide by zero?

Show the pupils the project listed in *Resources* and ask them to explain to each other how this program works. Ask them to test the program. Does Scratch always give the right answer when the divisor 'goes in' exactly? What about when it doesn't?

Ask the pupils to edit this project so that Scratch will work out remainders.

Ask the pupils to edit the script to use Scratch's built-in *division* block, and to test the program again. Were they surprised by what happened?

The pupils could record a screencast explaining what the script does and how they have used logical reasoning to improve it.

Mrs Clark—Division

https://scratch.mit.edu/projects/83183382/

Resources and e-safety

Pupils don't need accounts to download Scratch 1.4 or to use Scratch 2.0 or Snap! online.

Children register with a class account

Once registered, pupils can share their corrected and refined programs with the global Scratch community in a safe online space. Alternatively, pupils can upload their completed projects to the school's learning platform or blog.

If pupils upload screencasts of their solutions, make sure you take the usual precautions to protect their identity.

If pupils use the web for research (see *Extensions*), ensure all usual internet safety protocols are in place Scratch file—example animation

Use Scratch division script

Use Scratch division script: http://scratch.mit.edu/projects/13550313

Mrs Clark—Division

https://scratch.mit.edu/projects/83183382/

Curriculum Links for Unit 3.1—steps 1-6

Literacy	✓✓✓
Numeracy	✓✓✓
Science	✓✓✓
D&T	

Computing Year 3 2015—2016

Unit 3.2 Step 6:

Karin Clark

Learning Expectations

This unit will enable the children to:

- develop a number of strategies for finding errors in programs
- build up resilience and strategies for problem solving
- increase their knowledge and understanding of Scratch
- recognise a number of common types of bug in software..

Main Session

Show the pupils the video clip about creating a driving simulator. The clip ends with the pupils discovering what appears to be a bug in their scripts. The pupils might also enjoy watching the second clip, about simulating Formula 1. Show the pupils the Scratch project referenced in *Resources*, which uses the same code as the scripts in the video.
Ask the pupils to study the script and explain how it works. What happens when the speed is increased? At what speed does the car go off the track? Does it matter where the car starts or which direction it's facing? Does the track shape make a difference?
Can the pupils explain why this script seems to work for low speeds but breaks for high speeds? Can they think of a way to fix this? What could they change? Be aware that this is quite hard, owing to limitations in Scratch, but the pupils will still benefit from exploring this, and thinking of ways to work around these limitations.
The pupils could record a screencast explaining how the program works and what they've done to improve it.
Show the pupils the clip about Google's self-driving cars. The pupils should discuss what an algorithm for driving would be like. This is an example of where it's better to think of algorithms as sets of rules, e.g. if the road ahead is clear, then accelerate up to a safe speed; if the speed is greater than the speed limit, then stop accelerating or brake until the speed is below the speed limit, etc. It might be easier to do this in terms of the rules that drivers should follow. What events do they encounter? How should they react to them? The pupils should think about how important it would be to debug the program driving the car, and what the consequences of bugs in this software might be.
Finally, pupils should evaluate the success of their work

Resources and e-safety

Pupils don't need accounts to download Scratch 1.4 or to use Scratch 2.0 or Snap! online.
Children register with a class account
Once registered, pupils can share their corrected and refined programs with the global Scratch community in a safe online space. Alternatively, pupils can upload their completed projects to the school's learning platform or blog.
If pupils upload screencasts of their solutions, make sure you take the usual precautions to protect their identity.
If pupils use the web for research (see *Extensions*), ensure all usual internet safety protocols are in place Scratch file—example animation

Scratch racing car script
Pupil self-assessment information
Scratch racing car script: http://scratch.mit.edu/projects/11932304
Mrs Clark—Race Track

https://scratch.mit.edu/projects/85016430/

Curriculum Links for Unit 3.1—steps 1-6

Literacy	✓✓✓
Numeracy	✓✓✓
Science	✓✓✓
D&T	