Pioneer Federation
Medium term plan
Cycle 1, Term 5
ICT

Subject: Purple Mash unit 4.1 and 3.1

Key Concept/ Theme:

Prior Learning links: Year ½

Cycle 1

Unit 1.7 Coding

• Following instructions • Creating simple programs • Computer simulation of real life events

Unit 1.9 Technology Outside School

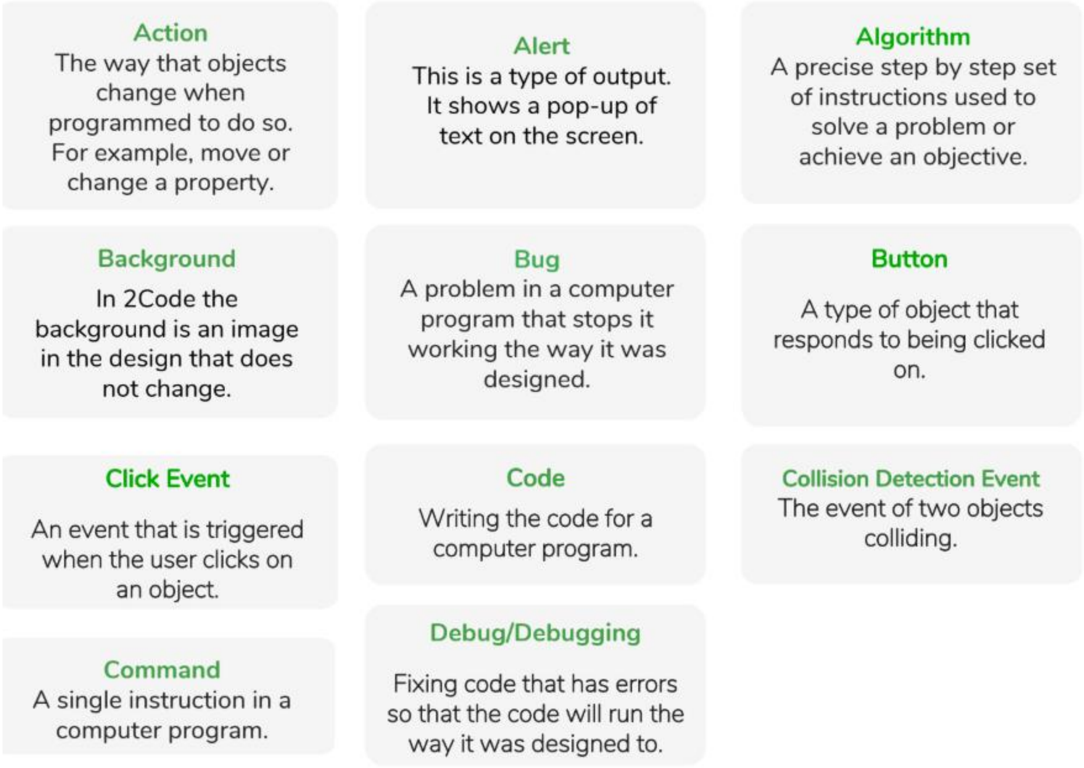
• Understanding the term ‘technology’ • Recognising the use of technology around them

Unit 2.1 Coding

• Algorithms • Collision detection - simulating air traffic control • Object types • Debugging

Vocabulary:

Pioneer Federation
Medium term plan
Cycle 1, Term 5
ICT



Action
The way that objects change when programmed to do so. For example, move or change a property.

Alert
This is a type of output. It shows a pop-up of text on the screen.

Algorithm
A precise step by step set of instructions used to solve a problem or achieve an objective.

Background
In 2Code the background is an image in the design that does not change.

Bug
A problem in a computer program that stops it working the way it was designed.

Button
A type of object that responds to being clicked on.

Click Event
An event that is triggered when the user clicks on an object.

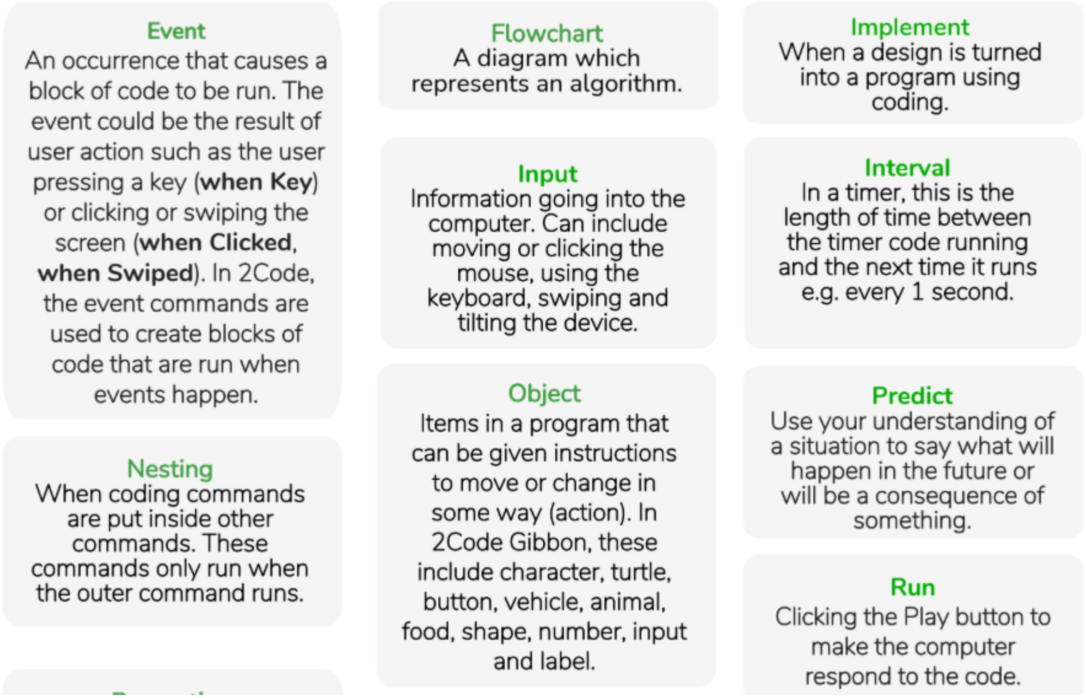
Code
Writing the code for a computer program.

Collision Detection Event
The event of two objects colliding.

Command
A single instruction in a computer program.

Debug/Debugging
Fixing code that has errors so that the code will run the way it was designed to.

Pioneer Federation
Medium term plan
Cycle 1, Term 5
ICT



Event
An occurrence that causes a block of code to be run. The event could be the result of user action such as the user pressing a key (**when Key**) or clicking or swiping the screen (**when Clicked**, **when Swiped**). In 2Code, the event commands are used to create blocks of code that are run when events happen.

Nesting
When coding commands are put inside other commands. These commands only run when the outer command runs.

Flowchart
A diagram which represents an algorithm.

Input
Information going into the computer. Can include moving or clicking the mouse, using the keyboard, swiping and tilting the device.

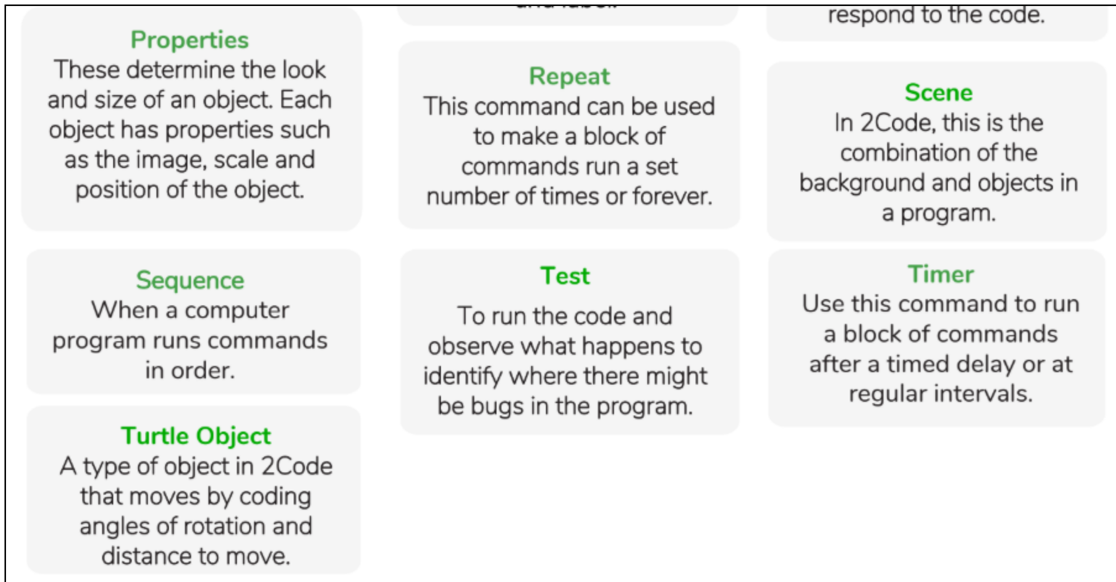
Object
Items in a program that can be given instructions to move or change in some way (action). In 2Code Gibbon, these include character, turtle, button, vehicle, animal, food, shape, number, input and label.

Implement
When a design is turned into a program using coding.

Interval
In a timer, this is the length of time between the timer code running and the next time it runs e.g. every 1 second.

Predict
Use your understanding of a situation to say what will happen in the future or will be a consequence of something.

Run
Clicking the Play button to make the computer respond to the code.



Properties
These determine the look and size of an object. Each object has properties such as the image, scale and position of the object.

Sequence
When a computer program runs commands in order.

Turtle Object
A type of object in 2Code that moves by coding angles of rotation and distance to move.

and label.

Repeat
This command can be used to make a block of commands run a set number of times or forever.

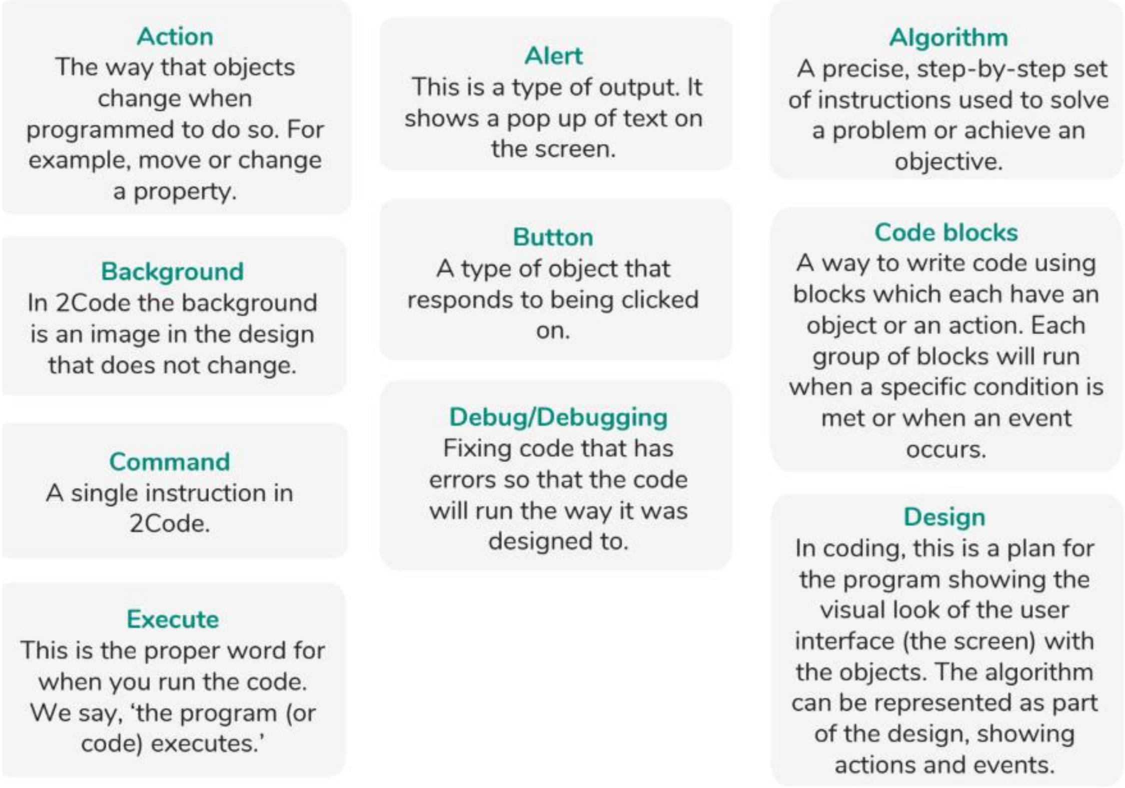
Test
To run the code and observe what happens to identify where there might be bugs in the program.

respond to the code.

Scene
In 2Code, this is the combination of the background and objects in a program.

Timer
Use this command to run a block of commands after a timed delay or at regular intervals.

Pioneer Federation
Medium term plan
Cycle 1, Term 5
ICT



Action
The way that objects change when programmed to do so. For example, move or change a property.

Background
In 2Code the background is an image in the design that does not change.

Command
A single instruction in 2Code.

Execute
This is the proper word for when you run the code. We say, 'the program (or code) executes.'

Alert
This is a type of output. It shows a pop up of text on the screen.

Button
A type of object that responds to being clicked on.

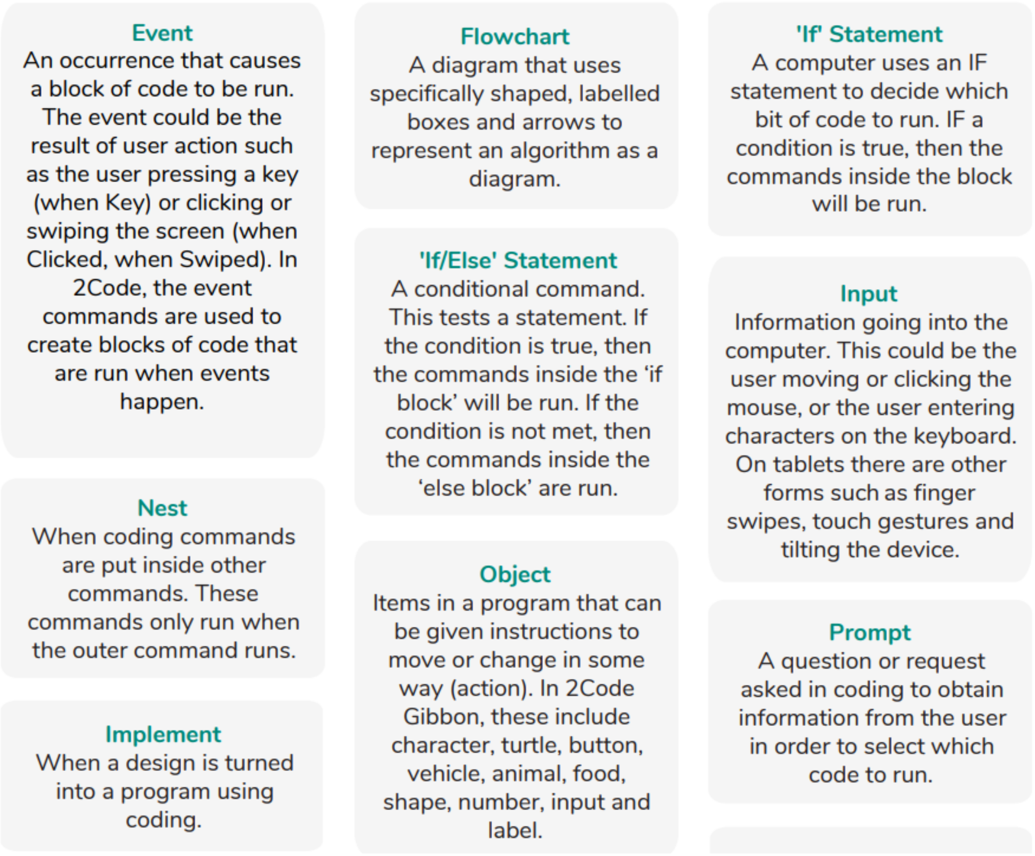
Debug/Debugging
Fixing code that has errors so that the code will run the way it was designed to.

Algorithm
A precise, step-by-step set of instructions used to solve a problem or achieve an objective.

Code blocks
A way to write code using blocks which each have an object or an action. Each group of blocks will run when a specific condition is met or when an event occurs.

Design
In coding, this is a plan for the program showing the visual look of the user interface (the screen) with the objects. The algorithm can be represented as part of the design, showing actions and events.

Pioneer Federation
Medium term plan
Cycle 1, Term 5
ICT



Event
An occurrence that causes a block of code to be run. The event could be the result of user action such as the user pressing a key (when Key) or clicking or swiping the screen (when Clicked, when Swiped). In 2Code, the event commands are used to create blocks of code that are run when events happen.

Nest
When coding commands are put inside other commands. These commands only run when the outer command runs.

Implement
When a design is turned into a program using coding.

Flowchart
A diagram that uses specifically shaped, labelled boxes and arrows to represent an algorithm as a diagram.

'If/Else' Statement
A conditional command. This tests a statement. If the condition is true, then the commands inside the 'if block' will be run. If the condition is not met, then the commands inside the 'else block' are run.

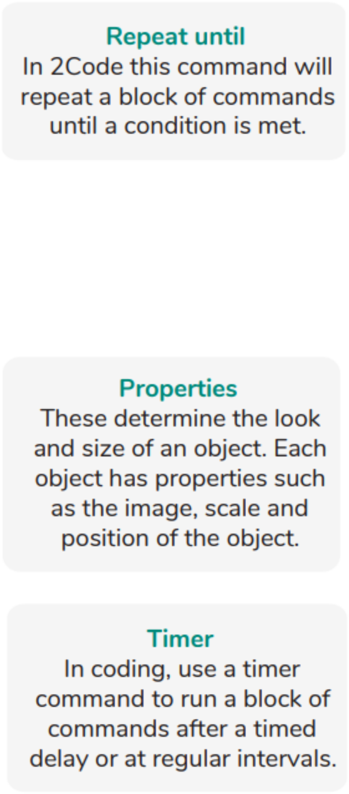
Object
Items in a program that can be given instructions to move or change in some way (action). In 2Code Gibbon, these include character, turtle, button, vehicle, animal, food, shape, number, input and label.

'If' Statement
A computer uses an IF statement to decide which bit of code to run. IF a condition is true, then the commands inside the block will be run.

Input
Information going into the computer. This could be the user moving or clicking the mouse, or the user entering characters on the keyboard. On tablets there are other forms such as finger swipes, touch gestures and tilting the device.

Prompt
A question or request asked in coding to obtain information from the user in order to select which code to run.

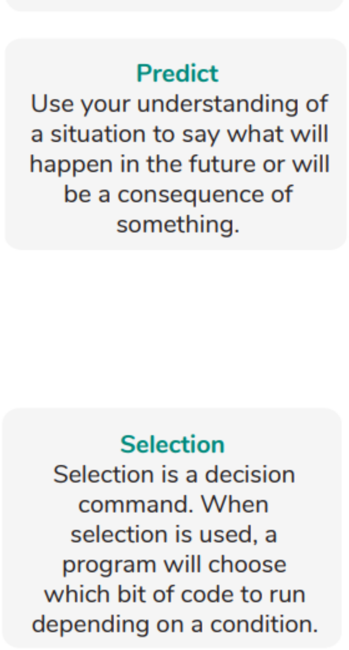
Pioneer Federation
Medium term plan
Cycle 1, Term 5
ICT



Repeat until
In 2Code this command will repeat a block of commands until a condition is met.

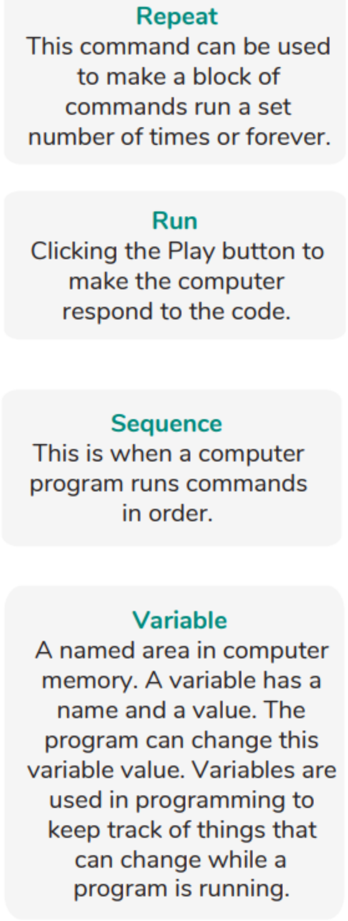
Properties
These determine the look and size of an object. Each object has properties such as the image, scale and position of the object.

Timer
In coding, use a timer command to run a block of commands after a timed delay or at regular intervals.



Predict
Use your understanding of a situation to say what will happen in the future or will be a consequence of something.

Selection
Selection is a decision command. When selection is used, a program will choose which bit of code to run depending on a condition.



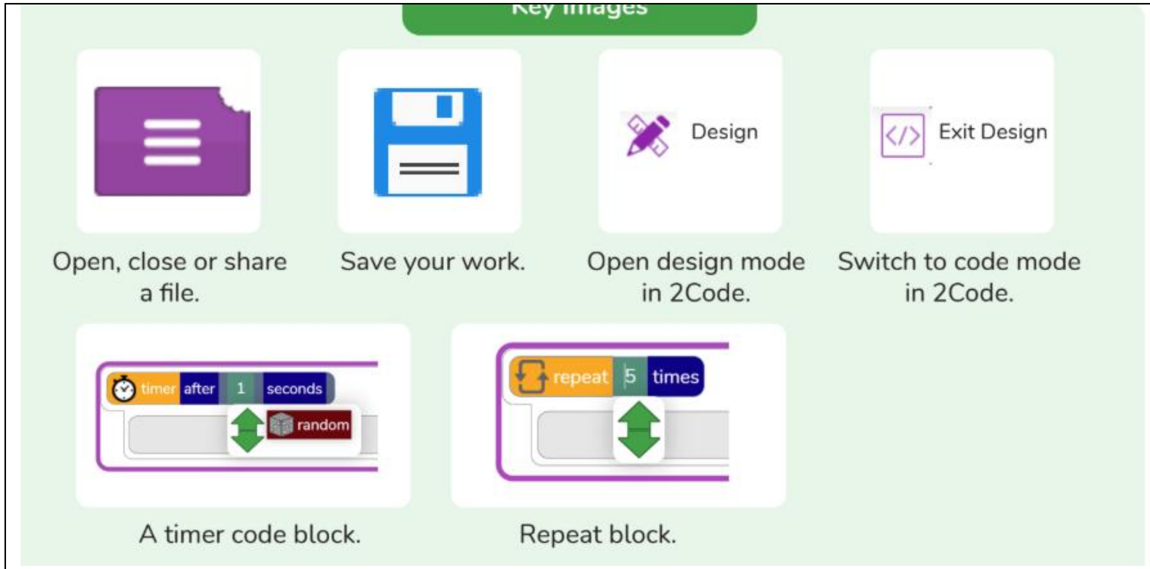
Repeat
This command can be used to make a block of commands run a set number of times or forever.

Run
Clicking the Play button to make the computer respond to the code.

Sequence
This is when a computer program runs commands in order.

Variable
A named area in computer memory. A variable has a name and a value. The program can change this variable value. Variables are used in programming to keep track of things that can change while a program is running.

Key images



Key images

Open, close or share a file.

Save your work.

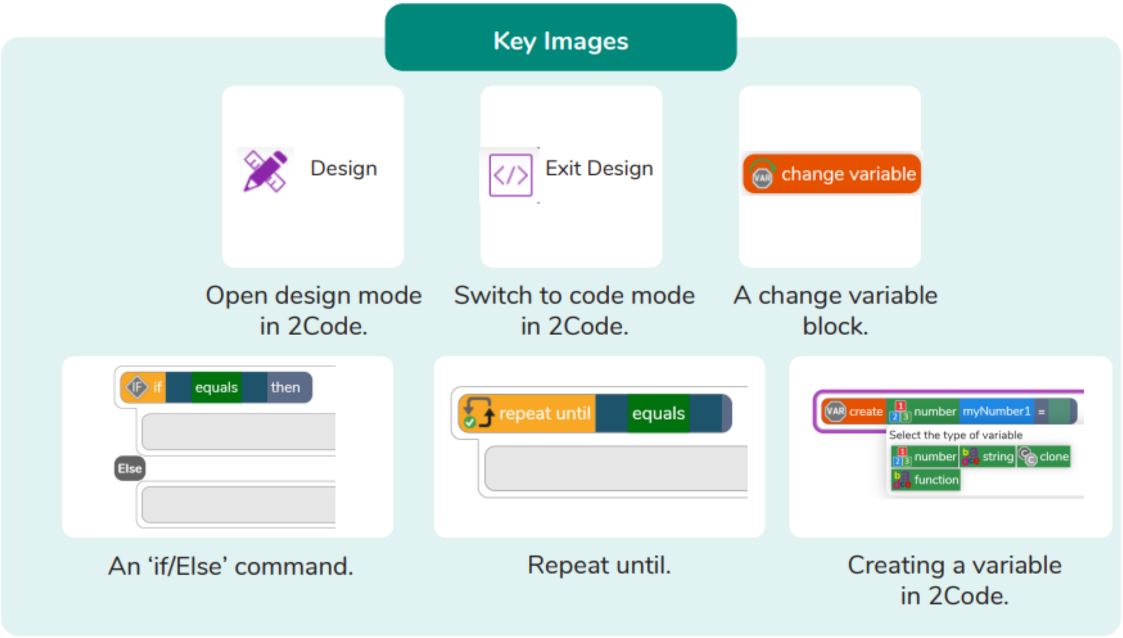
Open design mode in 2Code.

Switch to code mode in 2Code.

A timer code block.

Repeat block.

Pioneer Federation
Medium term plan
Cycle 1, Term 5
ICT



Key Images

Open design mode in 2Code.

Switch to code mode in 2Code.

A change variable block.

An 'if/Else' command.

Repeat until.

Creating a variable in 2Code.

Resources needed for each lesson – 2dos to set.

Lesson 1:

Unless otherwise stated, all resources can be found on the main unit 3.1 page. From here, click on the icon to set a resource as a 2Do for your class. Use the links below to preview the resources; right-click on the link and 'open in new tab' so you do not lose this page. • Vocabulary Quiz Year 3. • Animal Character Flowchart • Free Code Chimp (this is found on the main 2Code page). • (Optional) Vocabulary flash cards. The teacher flash cards have been created so you can print them on A4 paper, cut them to size, fold them in half and glue them together. You can display and use these throughout coding lessons to support use of vocabulary. Preparation • Set Free Code Chimp as a 2Do, call it 'Flowchart Program'. You can select the following objectives when setting the 2Dos to make future assessment easier: • You could print and copy Animal Character Flowchart for children to refer to while coding.

Lesson 2

Unless otherwise stated, all resources can be found on the main unit 3.1 page. From here, click on the icon to set a resource as a 2do for your class. Use the links below to preview the resources; right-click on the link and 'open in new tab' so you do not lose this page. • Magician. • Night and Day. • Tick Tock Challenge. • When Lightning Strikes Worksheet. Preparation • Set Night and Day as a 2Do • Set Tick Tock Challenge as a 2Do. You can select the following objectives when setting the 2Dos to make future assessment easier: • Print out copies of When Lightning Strikes Worksheet

Lesson 3

Unless otherwise stated, all resources can be found on the main unit 4.1 page. From here, click on the icon to set a resource as a 2Do for your class. Use the links below to preview the resources; right-click on the link and 'open in new tab' so you do not lose this page. • Coding Vocabulary Quiz Y4 • Free Code Gibbon (this is found on the main 2Code page). • Storyboard Planner OR individual whiteboards. • (Optional) Vocabulary flash cards. The Teacher flash cards have been created so you can print them on A4 paper, cut them to size, fold them in half and glue them together. You can display and use these throughout coding lessons to support use of vocabulary. Preparation • Set Free Code Gibbon as a 2Do. You can select the following objectives when setting the 2Dos to make future assessment easier: • Print copies of the Storyboard Planner for children to use if you are using it (see main plan).

Lesson 4

Unless otherwise stated, all resources can be found on the main unit 4.1 page. From here, click on the icon to set a resource as a 2Do for your class. Use the links below to preview the resources; right-click on the link and 'open in new tab' so you do not lose this page. • Coding Vocabulary Quiz Y4 • Free Code Gibbon (this is found on the main 2Code page). • Storyboard Planner OR individual whiteboards. • (Optional) Vocabulary flash cards. The Teacher flash cards have been created so you can print them on A4 paper, cut them to size, fold them in half and glue them together. You can display and use these throughout coding lessons to support use of vocabulary. Preparation • Set Free Code Gibbon as a 2Do. You can select the following objectives when setting the 2Dos to make future assessment easier: • Print copies of the Storyboard Planner for children to use if you are using it (see main plan)

Lesson 5

Unless otherwise stated, all resources can be found on the main unit 4.1 page. From here, click on the icon to set a resource as a 2Do for your class. Use the links below to preview the resources; right-click on the link and 'open in new tab' so you do not lose this page. • Knights Castle flowchart. • Guard the Castle (Gibbon). • Have printed storyboard templates available for program design. • Football Goal 2Code activity. Preparation • Set Guard the Castle (Gibbon) as a 2Do. You can select the following objectives when setting the 2Dos to make future assessment easier: • Set Football Goal as a 2Do (if planning to include extension)

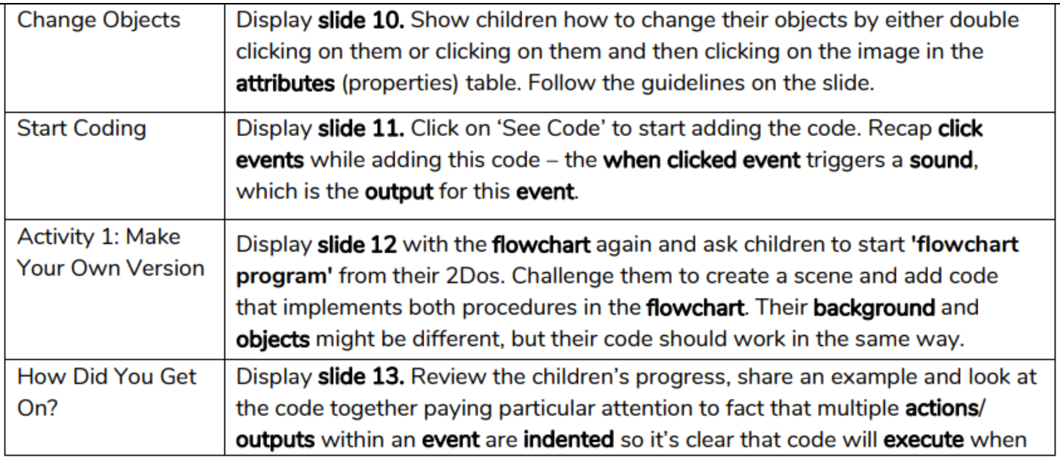
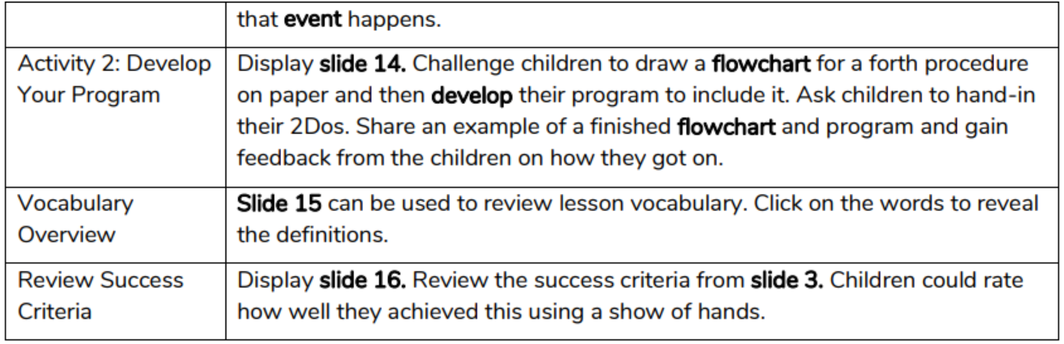
Lesson 6

Unless otherwise stated, all resources can be found on the main unit 4.1 page. From here, click on the icon to set a resource as a 2Do for your class. Use the links below to preview the resources; right-click on the link and 'open in new tab' so you do not lose this page. • Is It Raining 2Code Example. • Is It Raining IF Flowchart. • Is it Raining IF ELSE Flowchart. • Reginald Rocket 2Code Example. • Reginald Rocket IF ELSE Flowchart. • Free Code Gibbon (this is found on the main 2Code page). • Storyboard Planner. Preparation • Set Free Code Gibbon as a 2Do. • Set Reginald Rocket 2Code Example as a 2Do. You can select the following objectives when setting the 2Dos to make future assessment easier: • Print copies of the Storyboard Planner for children to use if you are using it (see step 7

Pioneer Federation
Medium term plan
Cycle 1, Term 5
ICT

1. Unit Lesson 1	**Deeper learning questions: When would a simulation be a good thing? Can you think of any problems with simulations?** **Reconnection:** prior online learning. – remind children of the idea of online safety and why it is important to be safe online. introduce new vocabulary. The vocabulary is repeated at the end of the lesson where it can be used to review lesson vocabulary. **LO:** • To review previous coding knowledge. • To understand what a flowchart is and how flowcharts are used in computer programming. **Activity:**

Introduction	Display **slide 2** and outline the lesson aims. Display **slide 3** and outline the success criteria.
Vocabulary	Display **slide 4,** Use the Y3 Coding Vocabulary Quiz as a class to help refresh coding knowledge. It is set up so that you attempt all questions and then click the hand in button to check the answers. Click 'OK' to see which are correct and incorrect. Run through the answers to the questions together. You could use the vocabulary cards to find the answers and display in the classroom or use slide 5 which has definitions.
	Slide 5 can be used to review previous vocabulary. The use of this vocabulary is recapped during the lesson. The vocabulary is repeated at the end of the lesson where it can be used to review new vocabulary.
Flowcharts	Display **slide 6.** Display the Animal and Character Flowchart. Tell children that this is a **flowchart** for a computer program. Give them some time to discuss with their talking partner what this program might do. Follow the activities on the slide.
Procedures	Display **slide 7.** Tell children that there are three parts to this flowchart and that each part is called a **procedure**. Ask children to help you create this program in 2Code following the next slides.
Add a Background	Display **slide 8.** Open Free Code Chimp and click on 'Design'. Ask children to remind you how to add the **background.** Choose a **background** with some land that the animal and character **objects** might move on.
Add Objects	Display **slide 9.** Add an **animal object** and a **character object**. (the **objects** they identified in the **flowchart**).

Change Objects	Display **slide 10.** Show children how to change their objects by either double clicking on them or clicking on them and then clicking on the image in the **attributes** (properties) table. Follow the guidelines on the slide.
Start Coding	Display **slide 11.** Click on 'See Code' to start adding the code. Recap **click events** while adding this code – the **when clicked event** triggers a **sound**, which is the **output** for this **event**.
Activity 1: Make Your Own Version	Display **slide 12** with the **flowchart** again and ask children to start **'flowchart program'** from their 2Dos. Challenge them to create a scene and add code that implements both procedures in the **flowchart**. Their **background** and **objects** might be different, but their code should work in the same way.
How Did You Get On?	Display **slide 13.** Review the children's progress, share an example and look at the code together paying particular attention to fact that multiple **actions/ outputs** within an **event** are **indented** so it's clear that code will **execute** when that **event** happens.
Activity 2: Develop Your Program	Display **slide 14.** Challenge children to draw a **flowchart** for a forth procedure on paper and then **develop** their program to include it. Ask children to hand-in their 2Dos. Share an example of a finished **flowchart** and program and gain feedback from the children on how they got on.
Vocabulary Overview	**Slide 15** can be used to review lesson vocabulary. Click on the words to reveal the definitions.
Review Success Criteria	Display **slide 16.** Review the success criteria from **slide 3.** Children could rate how well they achieved this using a show of hands.

Extension:

2.

Unit

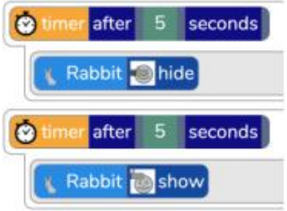
Lesson 2

LO:

To understand that there are different types of timers. • To be able to select the right type of timer for a purpose.

Activity:

	Introduction	Display **slide 2** and outline the lesson aims. Display **slide 3** and outline the success criteria.	
	Vocabulary	**Slide 4** can be used to review previous vocabulary. The use of this vocabulary is recapped during the lesson. The vocabulary is repeated at the end of the lesson where it can be used to review new vocabulary.	
		Display **slide 5.** Display the key vocabulary **timer** on the board and recap knowledge from Year 2.	
		Display **slide 6.** Display the key vocabulary **sequence** on the board and recap knowledge from Year 2.	
		Display **slide 7.** Display the key vocabulary **timer** and **sequence** on the board and recap knowledge from Year 2.	
	Instructions With Delays	Display **slide 8.** Look at the flowchart together and then ask the children to follow the instructions it gives. Ask children to draw their own version on an individual whiteboard or piece of paper. Ask children to swap their flowchart with a partner and have a go at following each other's instructions.	
	Magician	Work through **slide 9** and watch the video for stage 1 of the Magician guided lesson.	
	Magician – Stage 1	Display **slide 10.** Use the slide to talk through stage 1. This is a bit like using an **event** code block – it sets a **timer** and after the specified time the **object** (rabbit) will hide. Open Magician and work through the first stage as a whole class. Watch the videos and remind them that they can unlock a hint if they get stuck.	

		Make mistakes as you add the code and get the children to help you **debug** and fix the problems.	
	Magician - Stage 2	Display **slide 11. Stage 2:** Add this incorrect code, **test** it and ask the children to help you **debug:**  timer after 5 seconds Rabbit hide timer after 5 seconds Rabbit show This code doesn't work because when you run the program both timers start at the same time (if you click stop and run again you could notice they both highlight orange at the same time) and the code to 'hide' and 'show'	

	the rabbit executes at the same time – after 5 seconds. Point out that the timer for the rabbit to 'show' needs to start AFTER the rabbit has hidden. You need to add the second timer inside the first timer OR work out that the rabbit 'shows' 10 seconds after the start (5 + 5) and alter the second timer to reflect that, so either of the solutions shown on the slide would work.
Activity 1: Night and Day	Display **slide 12.** Ask children to start the Night and Day activity from their 2Dos and see if they can complete it. This works in a very similar way to Magician.
Activity 2: Tick Tock Challenge	Display **slide 13.** Once they have completed Night and Day and they have recapped using **timer-after**, tell children that there is another kind of **timer**, and they are going to learn about it by working through the Tick Tock Challenge. Set them to start and complete this challenge form their 2Dos. Review how they have got on – what have they learnt? Ask children: What is the difference between **timer-after** and a **timer-every?**
Activity 3: Extension	Display **slide 14**. Ask children to look at the scene and read the code, then predict what would happen when the code is run. Discuss with children how they could use a timer-every command to develop this program. You could set this activity as an extension 2Do for children to develop during or after the lesson.
Vocabulary Overview	**Slide 15** can be used to review lesson vocabulary. Click on the words to reveal the definitions.
Review Success Criteria	Display **slide 16.** Review the success criteria from **slide 3.** Children could rate how well they achieved this using a show of hands.

Extension:

3.

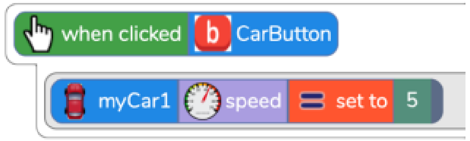
Unit
3.7

LO:
To review coding vocabulary and knowledge. • To create a simple computer program

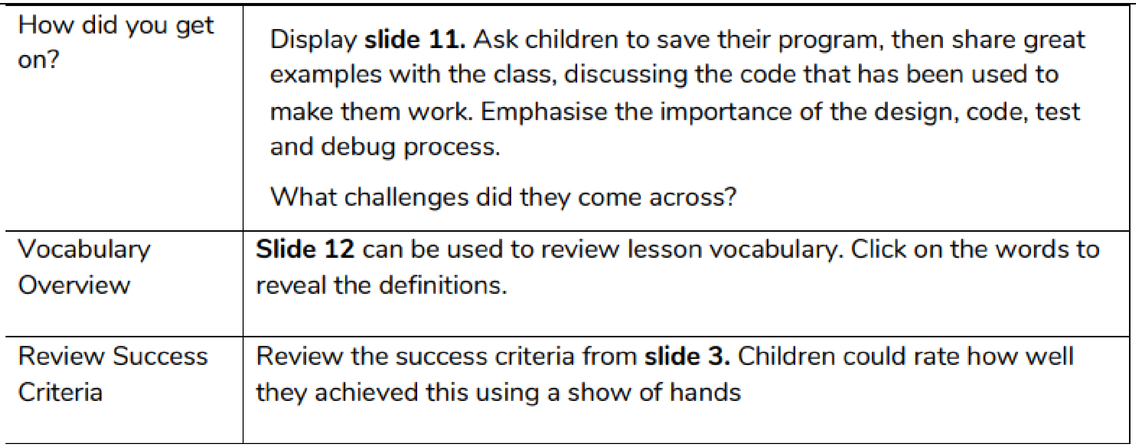
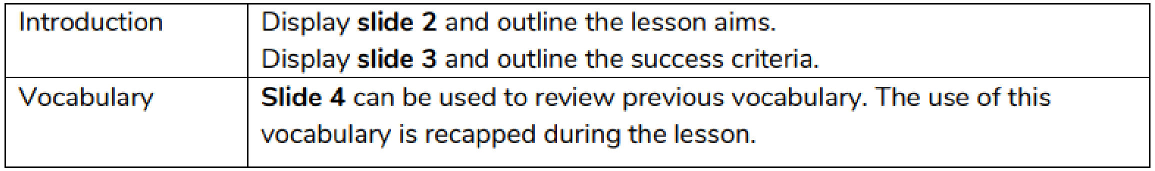
Activity:
Go over vocab for lesson.

Pioneer Federation
Medium term plan
Cycle 1, Term 5
ICT

Lesson 3		Introduction	Display **slide 2** and outline the lesson aims. Display **slide 3** and outline the success criteria.
		Vocabulary	Display **slide 4.** Use the Coding Vocabulary Quiz Y4 as a class. It is set up so that you attempt all questions and then click the Hand in button to check the answers. Click 'OK' to see which are correct and incorrect: Run through the answers to the questions together. You could use the vocabulary cards to find the answers and display in the classroom or use slide 5 which has definitions.
			Slide 5 can be used to review previous vocabulary. The use of this vocabulary is recapped during the lesson. The vocabulary is repeated at the end of the lesson where it can be used to review new vocabulary.
		Free Code Gibbon	Display **slide 6.** Put Free Code Gibbon on the board. Review how to add objects in 2Code by going into Design View. Talk about creating a scene using a background and some **objects**. Then run through the design steps shown on the slide ending with 'Running' the program and testing the code.
			Display **slide 7**. Stop the program, click on 'Design' and look in more detail at the **object** attributes. Ask children to **predict** what would happen if you edited the animal **object attributes** to change the speed or allow off screen. Look at the **attributes** for the **button** (click on it to display them). Ask the children to help you to make the car move when you click on the **button**:

		Change the text on the **button** to e.g., ‘Car Go’ and the name of it to e.g., ‘CarButton’ (it is one **object** so the name can only be one word – no spaces). Add a click **event** that makes the car move at a set speed when CarButton is clicked on  when clicked b CarButton myCar1 speed set to 5 (a speed between 3 and 6ish is sensible, try the children’s suggestions and correct if the speed is too fast).	

		Display **slide 8**. Notice the car goes up – you didn't have up, down, left or right **action** options like you did with the animal **object**. Ask children to suggest how you could make the car go along to the right? Go back to design view and look through the **attributes** of a vehicle – notice and adjust the angle to 90 degrees. Run the program and test the code. Stop the program and return to Design View, discuss how you've seen that different **object** types have different **attributes** and different **actions** available when you use their **code blocks**. Show children how to delete an **object** (click on it, then click on the bin).	
		Display **slide 9**. Exit Design View using the 'See Code' button and look at the different **code blocks** available – inputs, outputs, **timers** etc. Ask the children which ones they recognise and to explain what they might do – it doesn't matter if they don't know them all yet, they'll be learning more in this unit!	
	Activity: Create a Computer Program	Use **slide 10** to Explain to children that today they are going to explore Design view in Free Code Gibbon and make their own designs by adding background and **objects**. Ask them to log into Purple Mash and open Free Code Gibbon from their 2Dos, then work through the following: Set a background. - Experiment with adding different **object** types and exploring their **attributes** and **actions**. - Use a whiteboard or Storyboard Planner to plan what will happen in their program. - Use code to implement their plan - running, testing and **debugging** as they go.	

<table>
<tr><td></td><td>

<table>
<tr><td>How did you get on?</td><td>Display <b>slide 11.</b> Ask children to save their program, then share great examples with the class, discussing the code that has been used to make them work. Emphasise the importance of the design, code, test and debug process.
What challenges did they come across?</td></tr>
<tr><td>Vocabulary Overview</td><td><b>Slide 12</b> can be used to review lesson vocabulary. Click on the words to reveal the definitions.</td></tr>
<tr><td>Review Success Criteria</td><td>Review the success criteria from <b>slide 3.</b> Children could rate how well they achieved this using a show of hands</td></tr>
</table>
</td></tr>
<tr><td>Lesson 4</td><td>
<b>LO:</b>
• To begin to understand selection in computer programming. • To understand how an IF statement works.
<b>Activity:</b>
Go over vocab for lesson.

<table>
<tr><td>Introduction</td><td>Display <b>slide 2</b> and outline the lesson aims.
Display <b>slide 3</b> and outline the success criteria.</td></tr>
<tr><td>Vocabulary</td><td><b>Slide 4</b> can be used to review previous vocabulary. The use of this vocabulary is recapped during the lesson.</td></tr>
</table>
</td></tr>
</table>

			The vocabulary is repeated at the end of the lesson where it can be used to review new vocabulary.	
			Use **slide 5** to introduce the term – 'Selection' in relation to computer programming. Reveal the slide.	
		IF Statement	Display **slide 6**. Say to the children '**IF** my class is quiet for 30 seconds, then I will [insert **action**/ activity here!!] Start a **timer** and then check **IF statement** is true. If it is, carry out stated **action**/ activity. In pairs, get children to write an **IF statement** on their boards, then check if it's true and run the **action** if it is, or not if it isn't. Discuss as a class: When tested, were any not true? Explain that in code we can use **IF statements** to help our programs work – for example, **IF** the countdown has reached 0 the game is over, or **IF** the score equals 10 the 'amazing' sound plays.	
		Selection Video	Display **slide 7**. Play Selection video to children (Video should play from slide).	
		Is It Raining?	Display **slide 8**. Display Is It Raining 2Code activity – show how the chart in the video looks in a program – look at the design together; two people under some rain clouds and a hidden umbrella (you can hide **objects** at the start using the **attributes** table). Talk through the code – it starts with a prompt for input. If the user notices the rain clouds and puts 'yes' into the input, the **IF statement** runs and shows the umbrella.	
		Lost	Display **slide 9**. Open Lost from your 2Dos by clicking on 'Preview'. Look at the design together and discover that there is a background and 2 objects:	

	Click on 'See Code' and see if children can 'read' the code and predict what will happen when the program is run. Run the program twice, putting in different inputs to see what happens. You could ask children to help you draw a flowchart for this program. Delete the code and see if the children can help you put it back in again – you may need to emphasise the difference between alert and prompt for input. Ask children – what could happen after they get to the sea? Click on 'Design' and remind children how to change the backgrounds and objects – remind them to change the name in the attributes table if they change the object so it matches what it is.
Activity: Lost	Display **slide 10**. Explain to the children that they are going to create their own 'Lost' program which should include a **timer** and an **IF statement**. They will start with a background and two **objects** but they can add more if they wish – but be careful not to get distracted by adding too many. Give children the Storyboard Planner– tell them to sketch inside each box and make notes including timings. You could challenge them to draw the flowchart for the **IF statement** (or one **IF statement** if they have more than one) on the back of their plan. Once children have finished their designs, they have a go at making them by going to their 2Dos and starting 'Lost'.
How did you get on?	Review children's work together against the lesson aims – this could be done by sharing some good examples from the 2Dos folder.
Vocabulary Overview	**Slide 12** can be used to review lesson vocabulary. Click on the words to reveal the definitions.
Review Success Criteria	Review the success criteria from **slide 3.** Children could rate how well they achieved this using a show of hands.

Pioneer Federation

Medium term plan

Cycle 1, Term 5

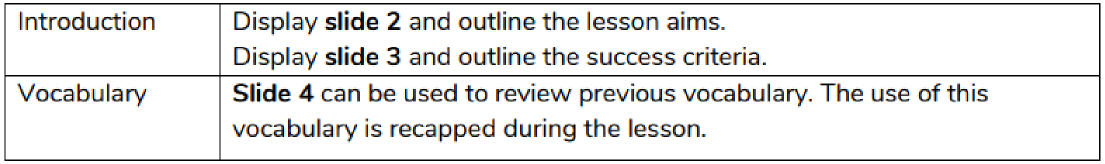
ICT

<table>
<tr>
<td>Lesson 5</td>
<td>

L.O

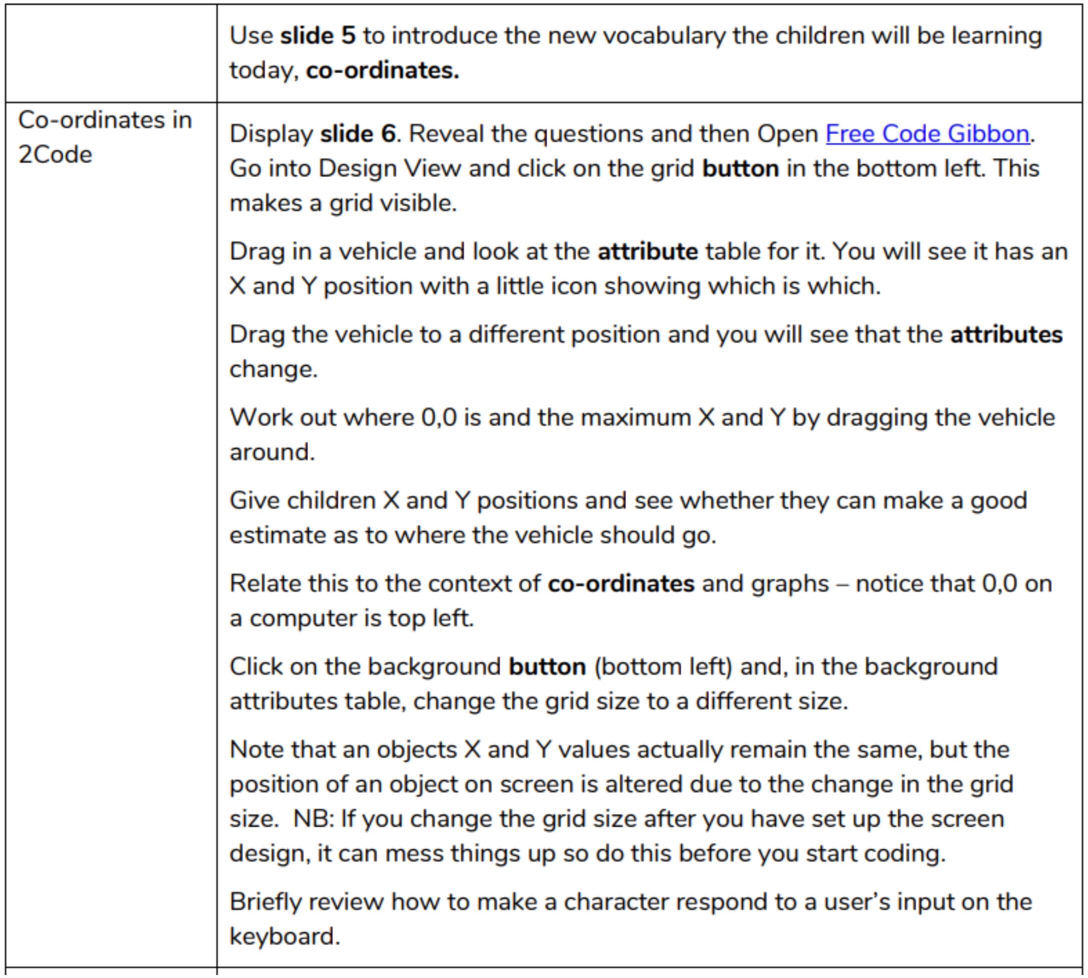
To understand how to use co-ordinates in computer programming. • To understand how an IF statement works

Activities

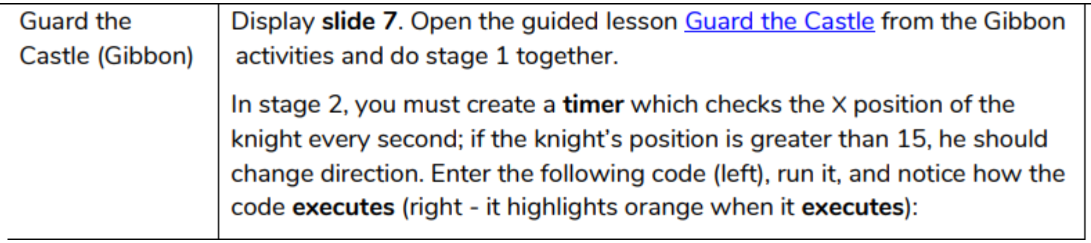


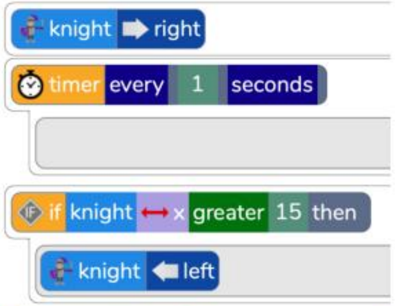
<table>
<tr><td>Introduction</td><td>Display <b>slide 2</b> and outline the lesson aims.
Display <b>slide 3</b> and outline the success criteria.</td></tr>
<tr><td>Vocabulary</td><td><b>Slide 4</b> can be used to review previous vocabulary. The use of this vocabulary is recapped during the lesson.</td></tr>
</table>

</td>
</tr>
</table>

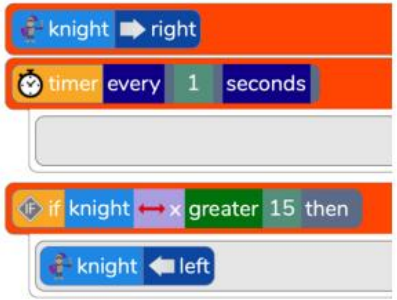


	Use **slide 5** to introduce the new vocabulary the children will be learning today, **co-ordinates.**
Co-ordinates in 2Code	Display **slide 6**. Reveal the questions and then Open [Free Code Gibbon](). Go into Design View and click on the grid **button** in the bottom left. This makes a grid visible. Drag in a vehicle and look at the **attribute** table for it. You will see it has an X and Y position with a little icon showing which is which. Drag the vehicle to a different position and you will see that the **attributes** change. Work out where 0,0 is and the maximum X and Y by dragging the vehicle around. Give children X and Y positions and see whether they can make a good estimate as to where the vehicle should go. Relate this to the context of **co-ordinates** and graphs – notice that 0,0 on a computer is top left. Click on the background **button** (bottom left) and, in the background attributes table, change the grid size to a different size. Note that an objects X and Y values actually remain the same, but the position of an object on screen is altered due to the change in the grid size. NB: If you change the grid size after you have set up the screen design, it can mess things up so do this before you start coding. Briefly review how to make a character respond to a user's input on the keyboard.

	 Guard the Castle (Gibbon) — Display **slide 7**. Open the guided lesson Guard the Castle from the Gibbon activities and do stage 1 together. In stage 2, you must create a **timer** which checks the X position of the knight every second; if the knight's position is greater than 15, he should change direction. Enter the following code (left), run it, and notice how the code **executes** (right - it highlights orange when it **executes**):

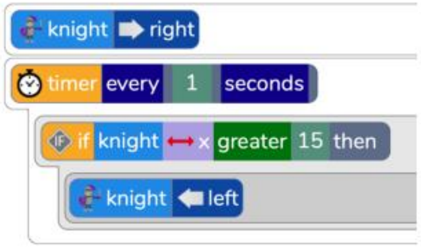


```
knight ➡ right
timer every 1 seconds

if knight ↔ x greater 15 then
    knight ⬅ left
```



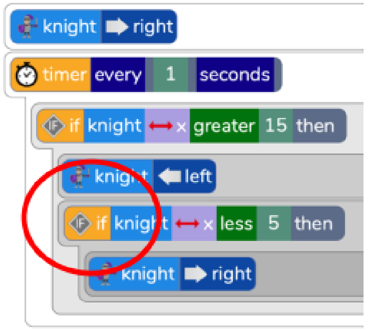
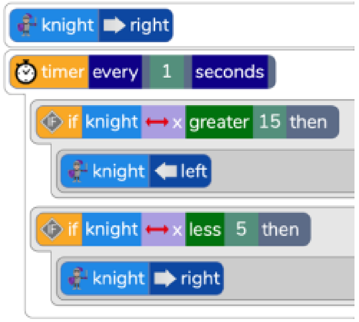
```
knight ➡ right
timer every 1 seconds

if knight ↔ x greater 15 then
    knight ⬅ left
```

Can children spot a problem? – The **IF statement executes** and the computer checks the knight's X position when the program is run, not every second – the **IF statement** needs to **nest** within the **timer** to trigger the computer to check the knight's X position every second:



```
knight ➡ right
timer every 1 seconds
    if knight ↔ x greater 15 then
        knight ⬅ left
```

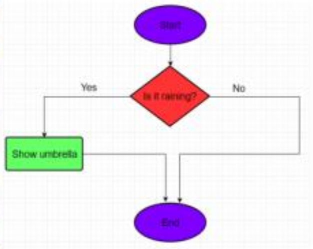
		Run the program with the corrected code and notice with children when the code **executes** and what effect it has.	
	Activity: Guard the Castle	Display **slide 8**. Ask children to log into Purple Mash and open Guard the Castle from their 2Dos area, then work through stages 1-4. At stage 3, children commonly add the second **IF statement** inside the first **IF statement** so it looks like the following:	

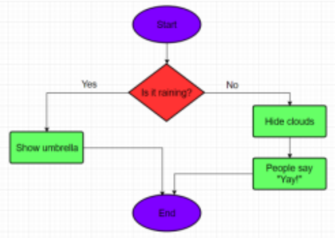
	
	knight ➡ right
	timer every 1 seconds
	  if knight ↔ x greater 15 then
	    knight ⬅ left
	    if knight ↔ x less 5 then
	      knight ➡ right
	If this happens, **run** their code with them and notice when the statements **execute**. Fix the problem together, the code should look like this (the second **IF statement nested** within the **timer**, not within the first **IF statement** – so both **IF statement**s are triggered by the **timer** – every second):
	
	knight ➡ right
	timer every 1 seconds
	  if knight ↔ x greater 15 then
	    knight ⬅ left
	  if knight ↔ x less 5 then
	    knight ➡ right

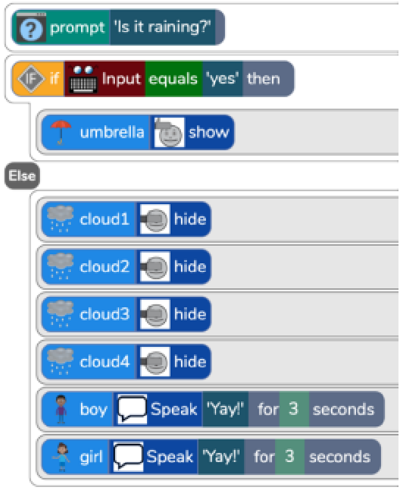
Flowchart: Guard the Castle	Display **slide 9**. Open the Knights Castle Flowchart for this activity and read it together, relating it to what they have done in the lesson. They could then either do the challenge activity (last stage) or the extension or both depending on what time you have.

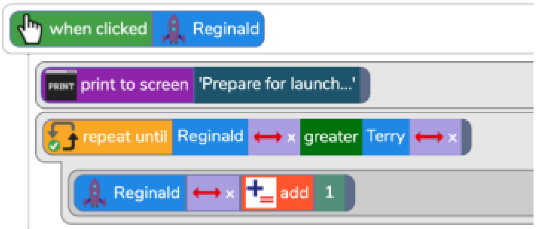
Extension: Football Goal	Display **slide 10**. Football Goal Challenge. Can children program the goalie to defend the goal using an if statement and co-ordinates? Can they add code for the football so a player can shoot at the goal and if the football collides with the goalie it is saved? They could try to draw a flow chart on a piece of paper that they could use to explain their code.
Vocabulary Overview	**Slide 11** can be used to review lesson vocabulary. Click on the words to reveal the definitions.
Review Success Criteria	Review the success criteria from **slide 3.** Children could rate how well they achieved this using a show of hands.

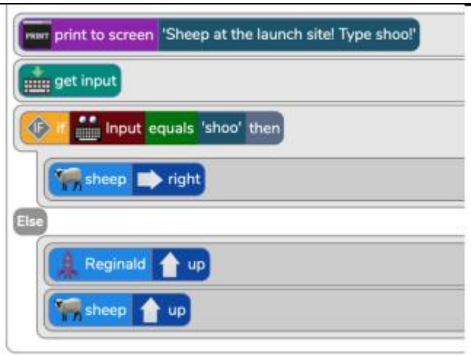
Lesson 6	**L.O** **To understand the repeat until command. • To begin to understand selection in computer programming. • To understand how an IF/ ELSE statement works** **Activities**

		Introduction	Display **slide 2** and outline the lesson aims. Display **slide 3** and outline the success criteria.	
		Vocabulary	**Slide 4** can be used to review previous vocabulary. The use of this vocabulary is recapped during the lesson.	
		IF	Start this lesson by returning to the Is It Raining 2Code example from lesson 2. Display the design view in one tab, and the Is It Raining IF Flowchart in another (or **display slide 5**).  Start Yes Is it raining? No Show umbrella End Ask the children to remind you what **selection** is and explain how an **IF statement** works. Explain that today we are going to start by looking at how to program something to happen if the condition is not met e.g. program something to happen in our scene if it is not raining.	
		IF/ ELSE	Display **slide 6.** Say to the children 'IF my class is quiet for 30 seconds, then I will [insert action/ activity here!!] Start a timer and then check IF statement is true. If it is, carry out stated action/ activity. In pairs, ask one child to write an IF statement on their small whiteboard, then the other to check if it's true and run the action if it is, or not if it isn't. Discuss as a class: When tested, were any not true?	

			Explain that in code we can use IF statements to help our programs work – for example, IF the countdown has reached 0 the game is over, or IF the score equals 10 the 'amazing' sound plays. Show children the Is It Raining IF ELSE **flowchart**:  Start Is it raining? Yes No Show umbrella Hide clouds People say "Yay!" End

		Talk it through with the children and then go back to the Is It Raining 2Code activity and ask them to help you add code to program it (ask them to look at the **code blocks** and see if they can pick out **IF/ ELSE** as a sensible one to use, if not direct them to it). The code should look like this:  prompt 'Is it raining?' if Input equals 'yes' then umbrella show Else cloud1 hide cloud2 hide cloud3 hide cloud4 hide boy Speak 'Yay!' for 3 seconds girl Speak 'Yay!' for 3 seconds Run the program a couple of times, once typing 'yes' in the prompt for input box, once typing 'no'. Notice how the code **executes** and what happens each time.
	Reginald Rocket	Display **slide 7**. Load Reginald Rocket 2Code example. Look at the design first, and then the code. Look at the first part of the code and pick out the repeat until **command**.

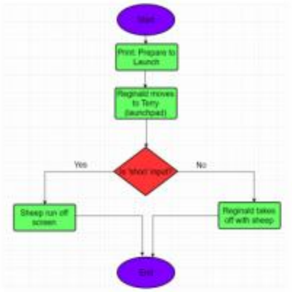
		 when clicked Reginald print to screen 'Prepare for launch...' repeat until Reginald ⟷ x greater Terry ⟷ x Reginald ⟷ x add 1 Can children 'read' the code to see what this **command** is doing? When the user clicks on Reginald (the rocket), a message is printed to the screen – Prepare for Launch – then Reginald will move right (adding 1 to the X **attribute**) this repeats until the X is greater than the X position of Terry. Ask children what/who is Terry? - Terry is the launch pad (you can work this out by looking for the **object** called Terry in Design View). And what is the purpose of this first section of code? - This first section of code moves the rocket onto the launch pad. Now look at the next part of the code:	



```
print to screen 'Sheep at the launch site! Type shoo!'
get input
if input equals 'shoo' then
    sheep right
Else
    Reginald up
    sheep up
```

What do children think will happen when this program is run?
When the children click on Reginald, he will move along to the launchpad. If the input is 'shoo' the sheep will run out of the way, if not (else) Reginald will take off with the sheep!

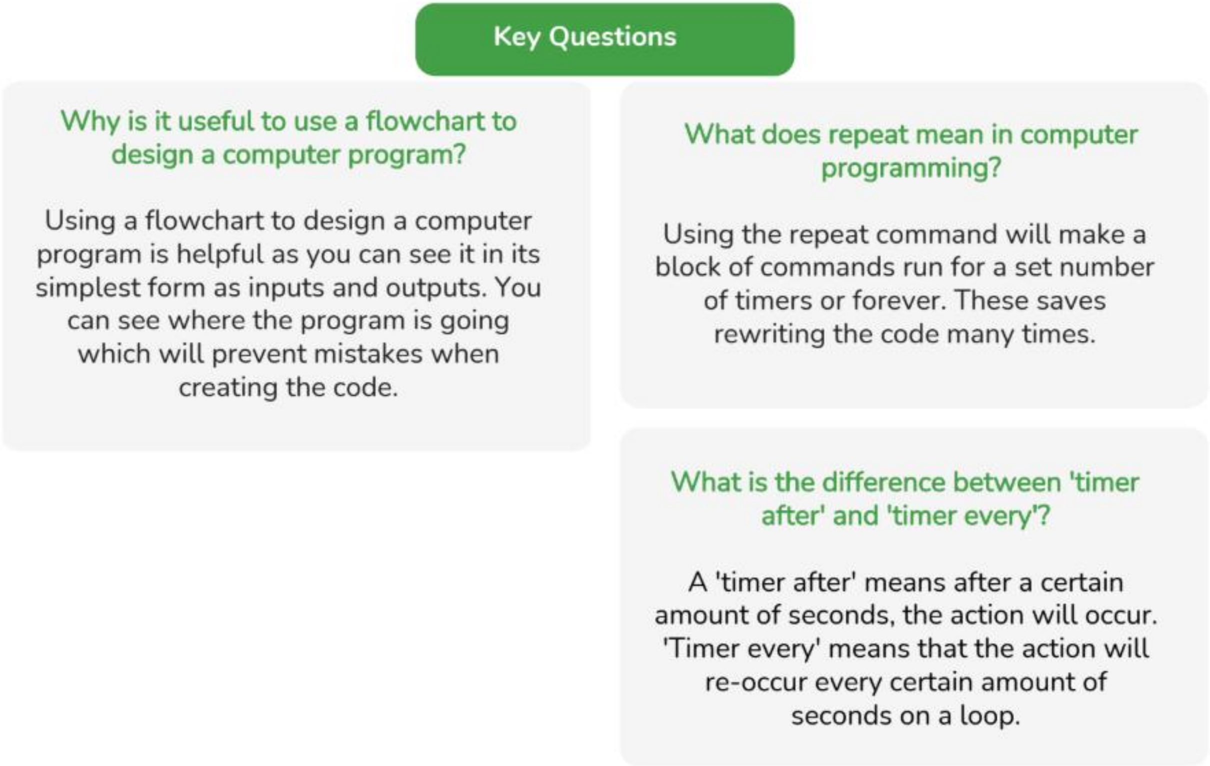
Display **slide 8**. Show children the Reginald If Else Flowchart – if they look on the 'statement is true' side Reginald doesn't take off. What code would they need to add for Reginald to take off 3 seconds after the sheep was shooed away? Can we add a blast off sound?



Activity: Create a Program	Display **slide 9**. Ask children to make a written plan with the following task specification: **Task:** Create a short program that uses **Repeat Until** and **IF/ ELSE** commands. Ask children to use the Storyboard Planner to plan their program. Challenge them to plan how their code will work using a **flowchart** on the back of their storyboard. Remind children not to be too ambitious, and to think about the knowledge they have when making their plans, so they know they will be able to create them in 2Code. Children could have Free Code Gibbon in front of them as they plan so they can look at available **backgrounds** and **objects**.
	Children open the Free Code Gibbon task from their 2Dos area and start to make their plan into a working computer program.
How did you get on?	Display **slide 10.** Review children's work together against the lesson aims – this could be done by sharing some good examples from the 2Dos folder.
Vocabulary Overview	**Slide 11** can be used to review lesson vocabulary. Click on the words to reveal the definitions.
Review Success Criteria	Review the success criteria from **slide 3.** Children could rate how well they achieved this using a show of hands.

End Points:

Pioneer Federation
Medium term plan
Cycle 1, Term 5
ICT



Key Questions

Why is it useful to use a flowchart to design a computer program?

Using a flowchart to design a computer program is helpful as you can see it in its simplest form as inputs and outputs. You can see where the program is going which will prevent mistakes when creating the code.

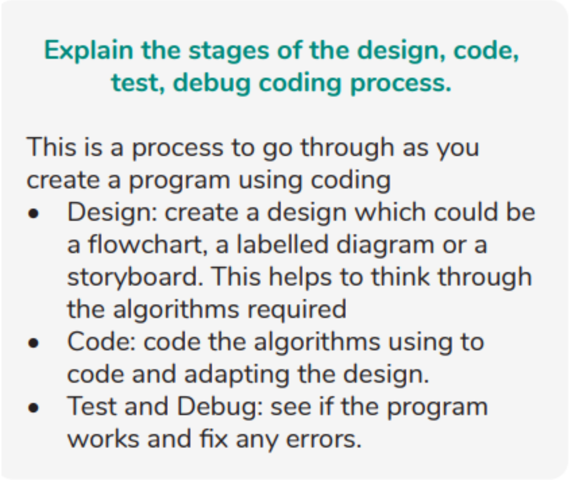
What does repeat mean in computer programming?

Using the repeat command will make a block of commands run for a set number of timers or forever. These saves rewriting the code many times.

What is the difference between 'timer after' and 'timer every'?

A 'timer after' means after a certain amount of seconds, the action will occur. 'Timer every' means that the action will re-occur every certain amount of seconds on a loop.

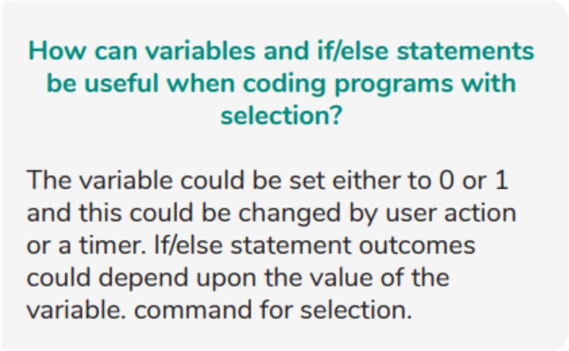
Pioneer Federation
Medium term plan
Cycle 1, Term 5
ICT



Explain the stages of the design, code, test, debug coding process.

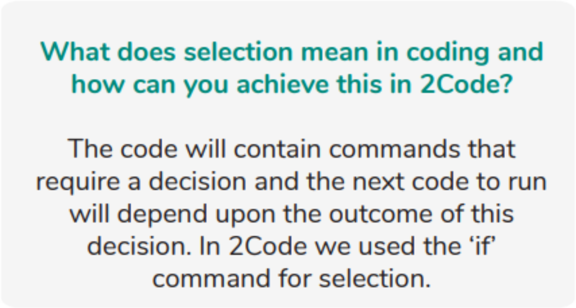
This is a process to go through as you create a program using coding

- Design: create a design which could be a flowchart, a labelled diagram or a storyboard. This helps to think through the algorithms required
- Code: code the algorithms using to code and adapting the design.
- Test and Debug: see if the program works and fix any errors.



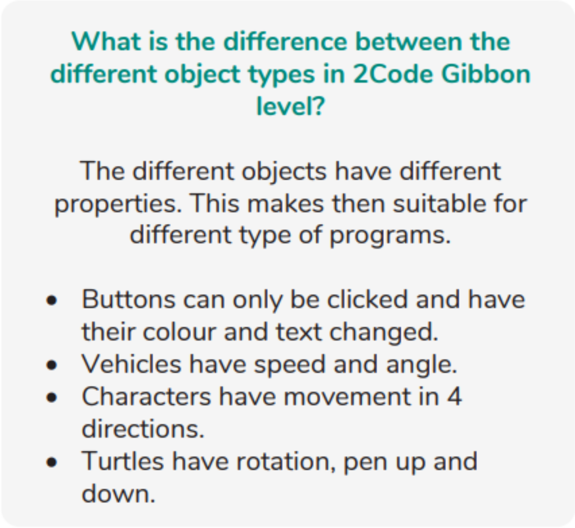
How can variables and if/else statements be useful when coding programs with selection?

The variable could be set either to 0 or 1 and this could be changed by user action or a timer. If/else statement outcomes could depend upon the value of the variable. command for selection.



What does selection mean in coding and how can you achieve this in 2Code?

The code will contain commands that require a decision and the next code to run will depend upon the outcome of this decision. In 2Code we used the 'if' command for selection.



What is the difference between the different object types in 2Code Gibbon level?

The different objects have different properties. This makes then suitable for different type of programs.

- Buttons can only be clicked and have their colour and text changed.
- Vehicles have speed and angle.
- Characters have movement in 4 directions.
- Turtles have rotation, pen up and down.

Evaluation: What have the end of unit quizzes, pupil self-reflections and termly work told you about what the children can remember and recall? What are the gaps? Ensure that the areas that need further reinforcement are documented in the next subject unit MTP. **Plan in time to revisit gaps within units, determined by the quizzes.**

Pioneer Federation

Medium term plan

Cycle 1, Term 5

ICT