Ashwell Primary School
Computing Curriculum
Coding Progression - Knowledge Organiser

Year 1 – Coding

Core Knowledge / skills to be acquired: (Unit 1.4 – Lego Builders)

- To compare the effects of adhering strictly to instructions to completing tasks without complete instructions.
- To follow and create simple instructions on the computer.
- To consider how the order of instructions affects the result.

Key Vocabulary:

Instruction - Information about how something should be done.
Algorithm - A precise, step-by-step set of instructions used to solve a problem or achieve an objective.
Computer - An electronic device for storing and processing data.
Program - To provide (a computer or other machine) with coded instructions.
Debug - To find and remove errors from computer hardware or software.

Curriculum Enrichment / Cultural Capital Opportunities / key questions

- Why do we need to debug code?

Prior knowledge / skills this builds on: (EYFS Framework)

-

What comes next: (Unit 1.5 – Maze Explorers)

- To understand the functionality of the direction keys.
- To understand how to create and debug a set of instructions (algorithm).
- To use the additional direction keys as part of an algorithm.
- To understand how to change and extend the algorithm list.
- To create a longer algorithm for an activity.
- To set challenges for peers.
- To access peer challenges set by the teacher as 2Dos.

Year 1 – Coding (continued)

Core Knowledge / skills to be acquired: (Unit 1.5 – Maze Explorers)

- To understand the functionality of the direction keys.
- To understand how to create and debug a set of instructions (algorithm).
- To use the additional direction keys as part of an algorithm.
- To understand how to change and extend the algorithm list.
- To create a longer algorithm for an activity.
- To set challenges for peers.
- To access peer challenges set by the teacher as 2Dos.

Key Vocabulary:

Direction - A course along which someone or something moves.
Challenge - A task or situation that tests someone's abilities.
Arrow - A mark or sign resembling an arrow, used to show direction or position.
Undo - Cancel or reverse the instruction.
Rewind - Move back several steps or to the start.
Forward - To move in the direction that one is facing or travelling.
Backwards - To move in the opposite direction to which one is facing.
Right turn - To move the object in a clockwise direction.
Left turn - To move the object in an anti-clockwise direction.
Debug - To find and remove errors from computer hardware or software.
Instruction - Information about how something should be done.
Algorithm - A precise, step-by-step set of instructions used to solve a problem or achieve an objective.

Curriculum Enrichment / Cultural Capital Opportunities / Key questions

- What is 2Go?
- How do I undo a mistake on 2Go?

Prior knowledge / skills this builds on: (Unit 1.4 – Lego Builders)

- To compare the effects of adhering strictly to instructions to completing tasks without complete instructions.
- To follow and create simple instructions on the computer.
- To consider how the order of instructions affects the result.

What comes next: (Unit 1.7 - Coding)

- To understand what instructions are and predict what might happen when they are followed.
- To use code to make a computer program.
- To understand what object and actions are.
- To understand what an event is.
- To use an event to control an object.
- To begin to understand how code executes when a program is run.
- To understand what backgrounds and objects are.
- To plan and make a computer program.

Year 1 – Coding (continued)

Core Knowledge / skills to be acquired: (Unit 1.7 – Coding)

- To understand what instructions are and predict what might happen when they are followed.
- To use code to make a computer program.
- To understand what object and actions are.
- To understand what an event is.
- To use an event to control an object.
- To begin to understand how code executes when a program is run.
- To understand what backgrounds and objects are.
- To plan and make a computer program.

Key Vocabulary:

Action - Types of commands which are run on an object. They could be used to move an object or change a property.
Code - Instructions written using symbols and words that can be interpreted by a computer.
Event - Something that causes a block of code to be run.
Algorithm - A precise step by step set of instructions used to solve a problem or achieve an objective.
Command - A single instruction in a computer program.
Execute - To run a computer program.
Input Information going into the computer. Can include moving or clicking the mouse, using the keyboard, swiping and tilting the device.
Debug/Debugging - Finding a problem in the code and fixing it.
Background - The part of the program design that shows behind everything else. It sets the scene for the story or game.
Properties - All objects have properties that can be changed in design or by writing code e.g. image, colour and scale properties.
Scene - The background and objects together create a scene.
Object - An element in a computer program that can be changed using actions or properties.

Curriculum Enrichment / Cultural Capital Opportunities / Key questions

- What is 2Go?
- How do I undo a mistake on 2Go?

Prior knowledge / skills this builds on: (Unit 1.5 – Maze Explorers)

- To understand the functionality of the direction keys.
- To understand how to create and debug a set of instructions (algorithm).
- To use the additional direction keys as part of an algorithm.
- To understand how to change and extend the algorithm list.
- To create a longer algorithm for an activity.
- To set challenges for peers.
- To access peer challenges set by the teacher as 2Dos.

What comes next: (Unit 2.1 - Coding)

- To understand what an algorithm is.
- To create a computer program using an algorithm.
- To create a program using a given design.
- To understand the collision detection event.
- To understand that algorithms follow a sequence.
- To design an algorithm that follows a timed sequence.
- To understand that different objects have different properties.
- To understand what different events do in code.
- To understand the function of buttons in a program.
- To understand and debug simple programs.

Year 2 – Coding

Core Knowledge / skills to be acquired: (Unit 2.1 – Coding)

- To understand what an algorithm is.
- To create a computer program using an algorithm.
- To create a program using a given design.
- To understand the collision detection event.
- To understand that algorithms follow a sequence.
- To design an algorithm that follows a timed sequence.
- To understand that different objects have different properties.
- To understand what different events do in code.
- To understand the function of buttons in a program.
- To understand and debug simple programs.

Key Vocabulary:

Action - Types of commands which are run on an object. They could be used to move an object or change a property.
Event - Something that causes a block of code to be run.
Algorithm - A precise step by step set of instructions used to solve a problem or achieve an objective.
Debug/Debugging - Looking for any problems in the code, fixing and testing them.
Background - The part of the program design that shows behind everything else. It sets the scene for the story or game.
Properties - All objects have properties that can be changed in design or by writing code e.g. image, colour and scale properties.
Scene - The background and objects together create a scene.
Object - An element in a computer program that can be changed using actions or properties.
Test - When code is run to check that it works correctly.
Sequence - When a computer program runs commands in order.
Collision Detection - Detecting when two characters on the screen touch each other.
Timer - Use this command to run a block of commands after a timed delay or at regular intervals.
When clicked/swiped - An event command. It makes code run when you click or swipe on something (or press/swipe your finger on a touchscreen).
Nesting - When you write a command inside something else e.g. a block of commands could be nested inside a timer.

Curriculum Enrichment / Cultural Capital Opportunities / Key questions

- What is an algorithm? Why is it useful in coding?
- Why is it important to know there are different object types?
- If you are good at coding, you don't need to debug. Is this true?

Prior knowledge / skills this builds on: (Unit 1.7 – Coding)

- To understand what instructions are and predict what might happen when they are followed.
- To use code to make a computer program.
- To understand what object and actions are.
- To understand what an event is.
- To use an event to control an object.
- To begin to understand how code executes when a program is run.
- To understand what backgrounds and objects are.
- To plan and make a computer program.

What comes next: (Unit 3.1 - Coding)

- To understand what a flowchart is and how flowcharts are used in computer programming.
- To understand that there are different types of timers and select the right type for purpose.
- To understand how to use the repeat command.
- To understand the importance of nesting.
- To design and create an interactive scene.

Year 3 – Coding

Core Knowledge / skills to be acquired: (Unit 3.1 – Coding)

- To understand what a flowchart is and how flowcharts are used in computer programming.
- To understand that there are different types of timers and select the right type for purpose.
- To understand how to use the repeat command.
- To understand the importance of nesting.
- To design and create an interactive scene.

Key Vocabulary:

Action - Types of commands which are run on an object. They could be used to move an object or change a property.
Alert - This is a type of output. It shows a pop-up of text on the screen.
Event - Something that causes a block of code to be run.
Algorithm - A precise step by step set of instructions used to solve a problem or achieve an objective.
Debug/Debugging - Looking for any problems in the code, fixing and testing them.
Background - The part of the program design that shows behind everything else. It sets the scene for the story or game.
Blocks of Command - A series of a few code instructions.
Properties - All objects have properties that can be changed in design or by writing code e.g. image, colour and scale properties.
Scene - A visual aspect of a program.
Object - An element in a computer program that can be changed using actions or properties.
Test - When code is run to check that it works correctly.
Sequence - When a computer program runs commands in order.
Collision Detection - Detecting when two characters on the screen touch each other.
Command - A single instruction in a computer program.
Execute - To run a computer program.
Flowchart - A diagram which represents an algorithm.
Timer - Use this command to run a block of commands after a timed delay or at regular intervals.
Nesting - When you write a command inside something else e.g. a block of commands could be nested inside a timer.

Curriculum Enrichment / Cultural Capital Opportunities / Key questions

- Why is it useful to use a flowchart to design a computer program?
- What does repeat mean in computer programming?
- What is the difference between 'timer after' and 'timer every'?

Prior knowledge / skills this builds on: (Unit 2.1 – Coding)

- To understand what an algorithm is.
- To create a computer program using an algorithm.
- To create a program using a given design.
- To understand the collision detection event.
- To understand that algorithms follow a sequence.
- To design an algorithm that follows a timed sequence.
- To understand that different objects have different properties.
- To understand what different events do in code.
- To understand the function of buttons in a program.
- To understand and debug simple programs.

What comes next: (Unit 4.1 - Coding)

- To begin to understand selection in computer programming.
- To understand how an IF statement works.
- To understand how to use co-ordinates in computer programming.
- To understand the 'repeat until' command.
- To understand how an IF/ELSE statement works.
- To understand what a variable is in programming.
- To use a number variable.
- To create a playable game.

Year 4 – Coding

Core Knowledge / skills to be acquired: (Unit 4.1 – Coding)

- To begin to understand selection in computer programming.
- To understand how an IF statement works.
- To understand how to use co-ordinates in computer programming.
- To understand the 'repeat until' command.
- To understand how an IF/ELSE statement works.
- To understand what a variable is in programming.
- To use a number variable.
- To create a playable game.

Key Vocabulary:

Action - Types of commands which are run on an object. They could be used to move an object or change a property.
Alert This is a type of output. It shows a pop-up of text on the screen.
Background - The part of the program design that shows behind everything else. It sets the scene for the story or game.
Code Block An individual code command represented visually by a block on the screen.
Command A single instruction in a computer program.
Co-ordinates Numbers which determine the position of a point, shape or object in a particular space.
Debug/Debugging - Looking for any problems in the code, fixing and testing them.
Execute - To run a computer program.
Flowchart - A diagram which represents an algorithm.
If - A conditional command. This tests a statement. If the condition is true, then the commands inside the block will be run.
If/Else - 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.
Nesting - When you write a command inside something else e.g. a block of commands could be nested inside a timer.
Blocks of Command - A series of a few code instructions.
Properties - All objects have properties that can be changed in design or by writing code e.g. image, colour and scale properties.
Repeat - This command can be used to make a block of commands run a set number of times or forever.
Repeat Until - This command can be used to make a block of commands run until something certain happens.
Timer - Use this command to run a block of commands after a timed delay or at regular intervals.
Variable - A named area in computer memory. A variable has a name and a value. The program can change this variable value.

Curriculum Enrichment / Cultural Capital Opportunities / Key questions

- Explain the stages of the design, code, test, debug coding process.
- How can variables and if/else statements be useful when coding programs with selection?
- What is the difference between the different object types in 2Code Gibbon level?

Prior knowledge / skills this builds on: (Unit 3.1 – Coding)

- To understand what a flowchart is and how flowcharts are used in computer programming.
- To understand that there are different types of timers and select the right type for purpose.
- To understand how to use the repeat command.
- To understand the importance of nesting.
- To design and create an interactive scene.

What comes next: (Unit 5.1 - Coding)

- To begin to simplify code.
- To create a playable game.
- To understand what a simulation is.
- To program a simulation using 2Code.
- To know what decomposition and abstraction are in computer science.
- To a take a real-life situation, decompose it and think about the level of abstraction.
- To understand how to use friction in code
- To begin to understand what a function is and how functions work in code.
- To understand what the different variables types are and how they are used differently.
- To understand how to create a string.
- To understand what concatenation is and how it works.

Year 5 – Coding

Core Knowledge / skills to be acquired: (Unit 5.1 – Coding)

- To begin to simplify code.
- To create a playable game.
- To understand what a simulation is.
- To program a simulation using 2Code.
- To know what decomposition and abstraction are in computer science.
- To a take a real-life situation, decompose it and think about the level of abstraction.
- To understand how to use friction in code
- To begin to understand what a function is and how functions work in code.
- To understand what the different variables types are and how they are used differently.
- To understand how to create a string.
- To understand what concatenation is and how it works.

Key Vocabulary:

Action - Types of commands which are run on an object. They could be used to move an object or change a property.
Abstraction - A way of de-cluttering and removing unnecessary details to get a program functioning.
Algorithm - A precise step by step set of instructions used to solve a problem or achieve an objective.
Called - A line of code that triggers a function to be executed.
Co-ordinates - Numbers which determine the position of a point, shape or object in a particular space.
Decomposition - A method of breaking down a task into manageable components. This makes coding easier as the components can then be coded separately and then brought back together in the program.
Event - Something that causes a block of code to be run.
Function - A block or sequence of code that you can access when you need it, so you don't have to rewrite the code repeatedly Instead, you simply 'call' the function each time you want it.
If - A conditional command. This tests a statement. If the condition is true, then the commands inside the block will be run.
Nesting - When you write a command inside something else e.g. a block of commands could be nested inside a timer.
Background - The part of the program design that shows behind everything else. It sets the scene for the story or game.
Object - An element in a computer program that can be changed using actions or properties.
Physical System - A system or process which happen in the real world using robotics, sensors or motors e.g. traffic lights.
Properties - All objects have properties that can be changed in design or by writing code e.g. image, colour and scale properties.
Simulation - A model that represents a real or imaginary situation.
Timer - Use this command to run a block of commands after a timed delay or at regular intervals.
Variable - A named area in computer memory. A variable has a name and a value. The program can change this variable value.

Curriculum Enrichment / Cultural Capital Opportunities / Key questions

- What does simulating a physical system mean?
- Describe how you would use variables to make a timer countdown and a scorepad for a game.
- What do the terms decomposition and abstraction mean? Use examples to explain them.

Prior knowledge / skills this builds on: (Unit 4.1 – Coding)

- To begin to understand selection in computer programming.
- To understand how an IF statement works.
- To understand how to use co-ordinates in computer programming.
- To understand the 'repeat until' command.
- To understand how an IF/ELSE statement works.
- To understand what a variable is in programming.
- To use a number variable.
- To create a playable game.

What comes next: (Unit 6.1 - Coding)

- To design a playable game with a timer and a score.
- To plan and use selection and variables.
- To understand how the launch command works.
- To use functions and understand why they are useful.
- To understand how functions are created and called.
- To use flowcharts to create and debug code.
- To create a simulation of a room in which devices can be controlled.
- To understand how user input can be used in a program.
- To understand how 2Code can be used to make a text-adventure game.

Year 6 – Coding

Core Knowledge / skills to be acquired: (Unit 6.1 – Coding)

- To design a playable game with a timer and a score.
- To plan and use selection and variables.
- To understand how the launch command works.
- To use functions and understand why they are useful.
- To understand how functions are created and called.
- To use flowcharts to create and debug code.
- To create a simulation of a room in which devices can be controlled.
- To understand how user input can be used in a program.
- To understand how 2Code can be used to make a text-adventure game.

Key Vocabulary:

Action - Types of commands which are run on an object. They could be used to move an object 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 - The part of the program design that shows behind everything else. It sets the scene for the story or game.
Called - A line of code that triggers a function to be executed.
Co-ordinates - Numbers which determine the position of a point, shape or object in a particular space.
Decomposition - A method of breaking down a task into manageable components. This makes coding easier as the components can then be coded separately and then brought back together in the program.
Developer - A person who writes, debugs and executes code to create a program.
Event - Something that causes a block of code to be run.
Function A block or sequence of code that you can access when you need it, so you don't have to rewrite the code repeatedly Instead, you simply 'call' the function each time you want it.
Get Input - This puts the text that a user types into the computer's temporary memory to be used to control the program flow.
Launch Command - A command that launches another program within the existing program.
If/Else - 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.
Nesting - When you write a command inside something else e.g. a block of commands could be nested inside a timer.
Procedure - A set of coded instructions that perform a certain task.
Properties - All objects have properties that can be changed in design or by writing code e.g. image, colour and scale properties.
Prompt - A question or request asked in coding to obtain information from the user in order to select which code to run.
Simulation - A model that represents a real or imaginary situation.
Selection - This is a conditional/decision command. When selection is used, a program will choose a different outcome depending on a condition.
Timer - Use this command to run a block of commands after a timed delay or at regular intervals.
Variable - A named area in computer memory. A variable has a name and a value. The program can change this variable value.

Curriculum Enrichment / Cultural Capital Opportunities / Key questions

- How can you use Tabs in 2Code Gorilla?
- What is a function in coding? Give an example that you have used in 2Code Gorilla.
- In 2Code Gorilla, how can a program receive user input?

Prior knowledge / skills this builds on: (Unit 5.1 – Coding)

- To begin to simplify code.
- To create a playable game.
- To understand what a simulation is.
- To program a simulation using 2Code.
- To know what decomposition and abstraction are in computer science.
- To a take a real-life situation, decompose it and think about the level of abstraction.
- To understand how to use friction in code
- To begin to understand what a function is and how functions work in code.
- To understand what the different variables types are and how they are used differently.
- To understand how to create a string.
- To understand what concatenation is and how it works.

What comes next: (Key Stage 3 - Coding)

- To design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems.
- To understand several key algorithms that reflect computational thinking [for example, ones for sorting and searching]; use logical reasoning to compare the utility of alternative algorithms for the same problem.
- To use 2 or more programming languages, at least one of which is textual, to solve a variety of computational problems; make appropriate use of data structures [for example, lists, tables or arrays]; design and develop modular programs that use procedures or functions.
- To understand simple Boolean logic [for example, AND, OR and NOT] and some of its uses in circuits and programming; understand how numbers can be represented in binary, and be able to carry out simple operations on binary numbers [for example, binary addition, and conversion between binary and decimal]