Ashwell Primary School
Computing Curriculum
Programming Progression - Knowledge Organiser

Reception (EYFS) – Programming

Core Knowledge / skills to be acquired:

All about instructions
- To know that being able to follow and give simple instructions is important in computing.
- To understand that it is important for instructions to be in the right order.
- To understand why a set of instructions may have gone wrong.

Programming Bee-Bots
- To know that you can program a Bee-Bot with some simple commands.
- To understand that debugging means how to fix some simple programming errors.
- To understand that an algorithm is a set of clear and precise instructions.

Key Vocabulary:

All about instructions
adjective, algorithm, bend down, blindfold, debug, describe, duck, first, follow, give, hop, instructions, last, left, next, order, predict, prediction, right, run, second, sequence, shuffle, skip, stand still, step over, stop, straight on, third, tiptoe, timer, turn, two-part instructions, under, walk around.

Programming Bee-Bots
algorithm, arrow, back, backwards, Bee-Bot, circle, debug, direction, directions, forward, instructions, left, program, right, route, sequence, straight on, turn

Curriculum Enrichment / Cultural Capital Opportunities / key questions
- Exploration of a variety of types of technology in everyday life and the world around us.
- What is technology? What are instructions?

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

Communication and Language
- Understand how to listen carefully and why listening is important.
- Describe events in some detail.
- Use talk to help work our problems and organise thinking and activities, and to explain how things work and why they might happen.

Personal, Social and Emotional Development
- ELG: Self-Regulation> Give focused attention to what the teacher says, responding appropriately even when engaged in activity, and show an ability to follow instructions involving several ideas or actions.
- ELG: Managing Self> Be confident to try new activities and show independence, resilience, and perseverance in the face of challenge.
- ELG: Building Relationships> Work and play cooperatively and take turns with others.

Physical Development
- Know and talk about the different factors that support their overall health and wellbeing.
- Further develop the skills they need to manage the school day successfully.

Mathematics
- Count objects, actions and sounds.
- Link the number symbol (numeral) with its cardinal number value.
- Count beyond 10.

What comes next:

Algorithms unplugged (Year 1)
- To understand that an algorithm is when instructions are put in an exact order.
- To know that input devices get information into a computer and that output devices get information out of a computer.
- To understand that decomposition means breaking a problem into manageable chunks and that it is important in computing.
- To know that we call errors in an algorithm 'bugs' and fixing these 'debugging'.

Programming Bee-Bots (Year 1)
- To understand the basic functions of a Bee-Bot.
- To know that you can use a camera/tablet to make simple videos.
- To know that algorithms move a bee-bot accurately to a chosen destination.

	▪ To know humans need to give robots instructions to follow and that they will carry out these instructions exactly, even if they are wrong. ▪ To know humans need to give instructions in the correct language for the robot to understand. ▪ To know an algorithm is a set of instructions used to carry out a task. ▪ To know algorithms must give every step of a task. ▪ To know algorithms must give clear, sequenced instructions. ▪ To know there may be an error if a set of instructions (an algorithm) does not give the expected result. ▪ To know errors could result from sequencing issues, unclear instructions or missing steps.

Year 1 – Programming

Core Knowledge / skills to be acquired:

Algorithms unplugged (Year 1)
- To understand that an algorithm is when instructions are put in an exact order.
- To know that input devices get information into a computer and that output devices get information out of a computer.
- To understand that decomposition means breaking a problem into manageable chunks and that it is important in computing.
- To know that we call errors in an algorithm 'bugs' and fixing these 'debugging'.

Programming Bee-Bots (Year 1)
- To understand the basic functions of a Bee-Bot.
- To know that you can use a camera/tablet to make simple videos.
- To know that algorithms move a bee-bot accurately to a chosen destination.
- To know humans need to give robots instructions to follow and that they will carry out these instructions exactly, even if they are wrong.
- To know humans need to give instructions in the correct language for the robot to understand.
- To know an algorithm is a set of instructions used to carry out a task.
- To know algorithms must give every step of a task.
- To know algorithms must give clear, sequenced instructions.
- To know there may be an error if a set of instructions (an algorithm) does not give the expected result.
- To know errors could result from sequencing issues, unclear instructions or missing steps.

Key Vocabulary:

Algorithms unplugged
algorithm, artificial intelligence, bug, **chunks, code, computer,** debug, **decompose,** device, directions, **input**, instructions, **manageable, order, organise, output,** program, problem, **solution, specific, tasks, virtual assistant.**

Programming Bee-Bots
algorithm, **Bee-Bot,** code, debug, **demonstration,** explain, **explore, filming, inputting,** instructions, **pause**, precise, **predict, program, review, test, tinker, video**

Curriculum Enrichment / Cultural Capital Opportunities / key questions

- Why do we need to debug code?

Prior knowledge / skills this builds on:	**What comes next:**

All about instructions (Reception) ▪ To know that being able to follow and give simple instructions is important in computing. ▪ To understand that it is important for instructions to be in the right order. ▪ To understand why a set of instructions may have gone wrong. **Programming Bee-Bots (Reception)** ▪ To know that you can program a Bee-Bot with some simple commands. ▪ To understand that debugging means how to fix some simple programming errors. ▪ To understand that an algorithm is a set of clear and precise instructions.	**Algorithms and debugging (Year 2)** ▪ To understand what machine learning is and how that enables computers to make predictions. ▪ To know that loops in programming are where you set a certain instruction (or instructions) to be repeated multiple times. ▪ To know that abstraction is the removing of unnecessary detail to help solve a problem. **Introduction to block coding – ScrathJr (Year 2)** ▪ To know that coding is writing in a special language so that the computer understands what to do. ▪ To understand that the character in ScratchJr is controlled by the programming blocks. ▪ To know that you can write a program to create a musical instrument or tell a joke. ▪ That programming a computer or device involves giving it instructions to perform specific tasks. ▪ That video games, phones, websites and apps are all created using programming. ▪ That different devices and programs use different programming languages or 'codes'. ▪ That an algorithm becomes a program when it is coded. ▪ That programs execute the exact instructions they are given, even if they are incorrect. ▪ That a program is a series of instructions (algorithms) that are written for a computer to follow. ▪ That an error in a computer program is known as a 'bug' and fixing errors is known as 'debugging'.

Year 2 – Programming

Core Knowledge / skills to be acquired:	**Key Vocabulary:**
Algorithms and debugging (Year 2) ▪ To understand what machine learning is and how that enables computers to make predictions. ▪ To know that loops in programming are where you set a certain instruction (or instructions) to be repeated multiple times. ▪ To know that abstraction is the removing of unnecessary detail to help solve a problem. **Introduction to block coding – ScrathJr (Year 2)** ▪ To know that coding is writing in a special language so that the computer understands what to do. ▪ To understand that the character in ScratchJr is controlled by the programming blocks. ▪ To know that you can write a program to create a musical instrument or tell a joke. ▪ That programming a computer or device involves giving it instructions to perform specific tasks. ▪ That video games, phones, websites and apps are all created using programming. ▪ That different devices and programs use different programming languages or 'codes'. ▪ That an algorithm becomes a program when it is coded.	**Algorithms and debugging** **abstraction**, algorithm, **artificial intelligence**, bug, clear, correct, data, debug, decompose, error, **key features, loop,** predict, **unnecessary.** **Introduction to block coding – ScrathJr** algorithm, bug, debug, programming, sequence, as above, plus: animation, **blocks**, button, **CGI, computer code, fluid, icon, imitate,** instructions, loop, '**on tap**', repeat, **Scratch Jr** , sound recording.

▪ That programs execute the exact instructions they are given, even if they are incorrect. ▪ That a program is a series of instructions (algorithms) that are written for a computer to follow. ▪ That an error in a computer program is known as a 'bug' and fixing errors is known as 'debugging'.	
Curriculum Enrichment / Cultural Capital Opportunities / key questions ▪ Why do we need to debug code? ▪ 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: **Algorithms unplugged (Year 1)** ▪ To understand that an algorithm is when instructions are put in an exact order. ▪ To know that input devices get information into a computer and that output devices get information out of a computer. ▪ To understand that decomposition means breaking a problem into manageable chunks and that it is important in computing. ▪ To know that we call errors in an algorithm 'bugs' and fixing these 'debugging'. **Programming Bee-Bots (Year 1)** ▪ To understand the basic functions of a Bee-Bot. ▪ To know that you can use a camera/tablet to make simple videos. ▪ To know that algorithms move a bee-bot accurately to a chosen destination. ▪ To know humans need to give robots instructions to follow and that they will carry out these instructions exactly, even if they are wrong. ▪ To know humans need to give instructions in the correct language for the robot to understand. ▪ To know an algorithm is a set of instructions used to carry out a task. ▪ To know algorithms must give every step of a task. ▪ To know algorithms must give clear, sequenced instructions. ▪ To know there may be an error if a set of instructions (an algorithm) does not give the expected result. ▪ To know errors could result from sequencing issues, unclear instructions or missing steps.	**What comes next:** **Scratch (Year 3)** ▪ To know that Scratch is a programming language and some of its basic functions. ▪ To understand how to use loops to improve programming. ▪ To understand how decomposition is used in programming. ▪ To understand that you can remix and adapt existing code. ▪ To know decomposition is the process of breaking down a task or problem into smaller parts. ▪ To know breaking down a problem into smaller parts makes it easier to solve.

Year 3 – Programming

Core Knowledge / skills to be acquired: **Scratch (Year 3)** ▪ To know that Scratch is a programming language and some of its basic functions. ▪ To understand how to use loops to improve programming.	**Key Vocabulary:** **Scratch** algorithm, animation, **application**, code, code block, debug, decompose, game, interface, loop, predict, program, remixing code, repetition code,

▪ To understand how decomposition is used in programming. ▪ To understand that you can remix and adapt existing code. ▪ To know decomposition is the process of breaking down a task or problem into smaller parts. ▪ To know breaking down a problem into smaller parts makes it easier to solve.	review, **Scratch,** sprite, tinker.
Curriculum Enrichment / Cultural Capital Opportunities / key questions ▪ Why do we need to debug code? ▪ What is an algorithm? Why is it useful in coding? ▪ If you are good at coding, you don't need to debug. Is this true?	
Prior knowledge / skills this builds on: **Algorithms and debugging (Year 2)** ▪ To understand what machine learning is and how that enables computers to make predictions. ▪ To know that loops in programming are where you set a certain instruction (or instructions) to be repeated multiple times. ▪ To know that abstraction is the removing of unnecessary detail to help solve a problem. **Introduction to block coding – ScrathJr (Year 2)** ▪ To know that coding is writing in a special language so that the computer understands what to do. ▪ To understand that the character in ScratchJr is controlled by the programming blocks. ▪ To know that you can write a program to create a musical instrument or tell a joke. ▪ That programming a computer or device involves giving it instructions to perform specific tasks. ▪ That video games, phones, websites and apps are all created using programming. ▪ That different devices and programs use different programming languages or 'codes'. ▪ That an algorithm becomes a program when it is coded. ▪ That programs execute the exact instructions they are given, even if they are incorrect. ▪ That a program is a series of instructions (algorithms) that are written for a computer to follow. ▪ That an error in a computer program is known as a 'bug' and fixing errors is known as 'debugging'.	**What comes next:** **Further coding with Scratch (Year 4)** ▪ To understand that a variable is a value that can change (depending on conditions) and know that you can create them in Scratch. ▪ To know what a conditional statement is in programming. ▪ To understand that variables can help you to create a quiz on Scratch. ▪ To know breaking down a problem into smaller parts makes it easier to solve the problem. ▪ To know loops are used to save time when writing code by reducing repetition. **Computational Thinking (Year 4)** ▪ To know that combining computational thinking skills (sequence, abstraction, decomposition etc) can help you to solve a problem. ▪ To understand that pattern recognition means identifying patterns to help them work out how the code works. ▪ To understand that algorithms can be used for a number of purposes e.g. animation, games design etc. **Skills showcase – HTML (Year 4)** ▪ To understand and identify examples of HTML tags. ▪ To understand what changing the HTML and CSS does to alter the appearance of an object on the web**.**

Year 4 – Programming

Core Knowledge / skills to be acquired: **Further coding with Scratch (Year 4)**	**Key Vocabulary:** **Further coding with Scratch**

▪ To understand that a variable is a value that can change (depending on conditions) and know that you can create them in Scratch. ▪ To know what a conditional statement is in programming. ▪ To understand that variables can help you to create a quiz on Scratch. ▪ To know breaking down a problem into smaller parts makes it easier to solve the problem. ▪ To know loops are used to save time when writing code by reducing repetition. **Computational Thinking (Year 4)** ▪ To know that combining computational thinking skills (sequence, abstraction, decomposition etc) can help you to solve a problem. ▪ To understand that pattern recognition means identifying patterns to help them work out how the code works. ▪ To understand that algorithms can be used for a number of purposes e.g. animation, games design etc. **Skills showcase – HTML (Year 4)** ▪ To understand and identify examples of HTML tags. ▪ To understand what changing the HTML and CSS does to alter the appearance of an object on the web**.**	code block, **conditional statement, coordinates,** decompose, **feature,** information, **negative number, orientation, position,** program, project, **script,** sprite, **stage,** tinker, **variable** **Computational Thinking** abstraction, algorithm, code, **computational thinking,** decomposition, input, **logical reasoning,** output, **pattern recognition,** script, sequence, variable **Skills showcase – HTML** code, content, copyright, **CSS, end tag,** fake news, **hacker, heading, HTML, HTML tags, internet browser, paragraph, remixing, start tag,** text, unplugged, **URL, web page, web page elements.**
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? ▪ 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?	
Prior knowledge / skills this builds on: **Scratch (Year 3)** ▪ To know that Scratch is a programming language and some of its basic functions. ▪ To understand how to use loops to improve programming. ▪ To understand how decomposition is used in programming. ▪ To understand that you can remix and adapt existing code. ▪ To know decomposition is the process of breaking down a task or problem into smaller parts. ▪ To know breaking down a problem into smaller parts makes it easier to solve.	**What comes next:** **Music (Year 5)** ▪ To know that a soundtrack is music for a film/video and that one way of composing these is on programming software. ▪ To understand that using loops can make the process of writing music simpler and more effective. ▪ To know how to adapt their code while performing their music. ▪ To know that programmers often save time when creating code by taking code from one program and turning it into another. ▪ To know that nested loops are loops within loops. **Micro:bit (Year 5)** ▪ To know that a Micro:bit is a programmable device. ▪ To know that Micro:bit uses a block coding language similar to Scratch. ▪ To understand and recognise coding structures including variables. ▪ To know what techniques to use to create a program for a specific purpose (including decomposition).

Year 5 – Programming

Core Knowledge / skills to be acquired:

Music (Year 5)

- To know that a soundtrack is music for a film/video and that one way of composing these is on programming software.
- To understand that using loops can make the process of writing music simpler and more effective.
- To know how to adapt their code while performing their music.
- To know that programmers often save time when creating code by taking code from one program and turning it into another.
- To know that nested loops are loops within loops.

Micro:bit (Year 5)

- To know that a Micro:bit is a programmable device.
- To know that Micro:bit uses a block coding language similar to Scratch.
- To understand and recognise coding structures including variables.
- To know what techniques to use to create a program for a specific purpose (including decomposition).

Key Vocabulary:

Music

beat, bugs, coding, command, debug, decompose, error, instructions, loop, **melody, mind map,** music, output, **performance, pitch,** plan, **play,** predict, programming, repeat, **rhythm,** scratch, **soundtrack, spacing**, **tempo, timbre,** tinker, **tutorials,** typing.

Micro:bit

algorithm, animation, **app,** blocks, **Bluetooth,** code block, connection, create, debug, decompose, designing, desktop, device, download, images, input, instructions, laptop, **load,** loop, **Micro:bit, outputs, pairing, pedometer, polling,** predict, program, repetition, **reset, sabotage, scoreboard,** screen, **systematic,** tablet, **tinkering, USB,** variables, Wi-Fi, wireless, wires.

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

Prior knowledge / skills this builds on:

Further coding with Scratch (Year 4)

- To understand that a variable is a value that can change (depending on conditions) and know that you can create them in Scratch.
- To know what a conditional statement is in programming.
- To understand that variables can help you to create a quiz on Scratch.
- To know breaking down a problem into smaller parts makes it easier to solve the problem.
- To know loops are used to save time when writing code by reducing repetition.

Computational Thinking (Year 4)

- To know that combining computational thinking skills (sequence, abstraction, decomposition etc) can help you to solve a problem.
- To understand that pattern recognition means identifying patterns to help them work out how the code works.

What comes next:

Programming into Python (Year 6)

- To know that there are text-based programming languages such as Logo and Python.
- To know that nested loops are loops inside of loops.
- To understand the use of random numbers and remix Python code.

Skills showcase – Inventing a product (Year 6)

- To know which programming software/ language is best to achieve a purpose.
- To know the building blocks of computational thinking e.g. sequence, selection, repetition, variables and inputs and outputs.

▪ To understand that algorithms can be used for a number of purposes e.g. animation, games design etc. **Skills showcase – HTML (Year 4)** ▪ To understand and identify examples of HTML tags. ▪ To understand what changing the HTML and CSS does to alter the appearance of an object on the web**.**	

Year 6 – Programming

Core Knowledge / skills to be acquired: **Programming into Python (Year 6)** ▪ To know that there are text-based programming languages such as Logo and Python. ▪ To know that nested loops are loops inside of loops. ▪ To understand the use of random numbers and remix Python code. **Skills showcase – Inventing a product (Year 6)** ▪ To know which programming software/ language is best to achieve a purpose. ▪ To know the building blocks of computational thinking e.g. sequence, selection, repetition, variables and inputs and outputs.	**Key Vocabulary:** **Programming into Python** algorithm, code, command, design, import, **indentation,** input, instructions, loop, output, patterns**, random**, **remix,** repeat, **shape** **Skills showcase – Inventing a product** **abstraction,** adapt, **advert,** algorithm, bug, code, coding, debug, design, edit, **electronic,** evaluate, **image rights,** images, information, **input,** loop, output, photos, **product,** program, repetition, **screenshot, selection,** sequence, software, **structure,** variable, video, website
Curriculum Enrichment / Cultural Capital Opportunities / key questions ▪ What is the difference between a text-based language like Python and a block-based language like Scratch? ▪ Can you explain what a loop is and why we use it in programming? Can you think of an example where we might need to use nested loops, like drawing shapes or creating patterns? ▪ What do you think "remixing" means when it comes to writing or changing code?	
Prior knowledge / skills this builds on: **Music (Year 5)** ▪ To know that a soundtrack is music for a film/video and that one way of composing these is on programming software. ▪ To understand that using loops can make the process of writing music simpler and more effective. ▪ To know how to adapt their code while performing their music. ▪ To know that programmers often save time when creating code by taking code from one program and turning it into another. ▪ To know that nested loops are loops within loops. **Micro:bit (Year 5)** ▪ To know that a Micro:bit is a programmable device. ▪ To know that Micro:bit uses a block coding language similar to Scratch. ▪ To understand and recognise coding structures including variables. ▪ To know what techniques to use to create a program for a specific purpose (including decomposition).	**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.