MacIntyre Academies

Quest Academy

Long Term Computing Plans 2022 – 2023

KS2						
	Autumn 1	**Autumn 2**	**Spring 1**	**Spring 2**	**Summer 1**	**Summer 2**
	7 weeks	7 weeks	7 weeks	5 weeks	6 weeks	8 weeks
	Overview:	***TOPIC COVERAGE:***			***Objectives:***	

Year 3

Year 3

What is the aim of this Programme of study?
Skills focus: Coding

To gain confidence and move from *encountering* to *mastering* a series of learning aims related to the following topics:

- Algorithms
- Programming
- Data Representation

- ***Engagement factors.***
- ***Enquiry based learning.***
- ***Cross Curricular (particularly with subjects which encounter information technology, computer hardware and processing and digital communication/safety).***
- ***Pupil Led Learning.***
- ***Developing practical skills.***
- ***Developing problem solving and critical thinking skills.***

Autumn

Autumn 1	***Autumn 2***
Introduction to algorithms Learn that programs execute by following clear instructions. Understand that programs respond to inputs to do different things.	**Simple inputs** Learn to combine start and input events to create more advanced apps and programs using precise instructions.

Spring

Spring 1	***Spring 2***
Different sorts of inputs Learn that programs respond to different sorts of inputs, and that the keyboard can be used to control objects on screen, not just by clicking them directly.	**Buttons and Instructions** Learn that one object can be used to control another object, e.g. writing code so clicking a button gives an instruction to make a lorry move.

Summer

Summer 1	***Summer 2***
Sequence and Animation Learn to make things happen in a sequence, creating simple animations and simulations.	**Conditional Events** Learn to code with 'if statements', which select different pieces of code to execute depending on what happens to other objects.

Autumn

Algorithms

- Understands what an algorithm is and is able to express simple linear (non
- branching) algorithms symbolically.
- Understands that computers need precise instructions.
- Demonstrates care and precision to avoid errors.
- Understands what an algorithm is and is able to express simple linear (non-branching) algorithms symbolically.
- Understands that computers need precise instructions.
- Demonstrates care and precision to avoid errors.

Programming and development

- Executes, checks and changes programs.
- Understands that programs execute by following precise instructions.
- Executes, checks and changes programs.
- Understands that programs execute by following precise instructions.

Spring

Algorithms

- Understands what an algorithm is and is able to express simple linear (non-branching) algorithms symbolically.
- Understands that computers need precise instructions.
- Demonstrates care and precision to avoid errors.
- Detects and corrects errors i.e. debugging, in algorithms.

Programming and development

- Executes, checks and changes programs.
- Understands that programs execute by following precise instructions.
- Detects and corrects simple semantic errors i.e. debugging, in programs.

				Summer **Algorithms** • Designs simple algorithms using loops, and selection i.e. if statements. • Uses logical reasoning to predict outcomes. • Detects and corrects errors i.e. debugging, in algorithms. **Programming and development** • Uses logical reasoning to predict the behaviour of programs. • Detects and corrects simple semantic errors i.e. debugging, in programs.

KS2						
	Autumn 1	**Autumn 2**	**Spring 1**	**Spring 2**	**Summer 1**	**Summer 2**
	7 weeks	7 weeks	7 weeks	5 weeks	6 weeks	8 weeks
Year 4 and 5	***Overview:***	***TOPIC COVERAGE:***			***Objectives:***	
	Year 4 and 5 What is the aim of this Programme of study? ***Skills focus: Coding*** To gain confidence and move from *encountering* to *mastering* a series of learning aims related to the following topics: • Algorithms • Programming • Data Representation • ***Engagement factors.*** • ***Enquiry based learning.*** • ***Cross Curricular (particularly with subjects which encounter information technology, computer hardware and***	***Autumn***			***Autumn*** **Algorithms** • Designs solutions (algorithms) that use repetition and two-way selection i.e. if, then and else. • Uses diagrams to express solutions. • Uses logical reasoning to predict outputs, showing an awareness of inputs. **Programming and development** • Creates programs that implement algorithms to achieve given goals. • Declares and assigns variables. ***Spring*** Algorithms • Designs solutions (algorithms) that use repetition and two-way selection i.e. if, then and else. • Uses diagrams to express solutions. • Uses logical reasoning to predict outputs, showing an awareness of inputs. • Designs solutions by decomposing a problem and creates a sub-solution for each of these parts. • Recognises that different solutions exist for the same problem.	
		Autumn 1	***Autumn 2***			
		Introduction to variables Learn how computers use variables to count things and keep track of what is going on, then create simple games which use a score variable.	**Repetition and loops** Learn how computers use repetition and loops to do things again.			
		Spring				
		Spring 1	***Spring 2***			
		Speed, direction and coordinates Learn how computers use numbers to represent things such as how fast things are moving, and where they are.	**Random Numbers and simulations** Learn how computers can generate random numbers and how these can be used in simulations.			
		Summer				
		Summer 1	***Summer 2***			
		More complex variables	**Object properties**			

	processing and digital communication/safety). • ***Pupil Led Learning.*** • ***Developing practical skills.*** • ***Developing problem solving and critical thinking skills.***	Learn to use variables in more complex ways, and to manipulate inputs to create useful outputs.	Learn more about how computers use property values and parameters to store information about objects.	**Programming and development** • Creates programs that implement algorithms to achieve given goals. • Declares and assigns variables. • Designs solutions by decomposing a problem and creates a sub-solution for each of these parts. • Recognises that different solutions exist for the same problem. **Algorithms** • Designs solutions (algorithms) that use repetition and two-way selection i.e. if, then and else. • Uses diagrams to express solutions. • Uses logical reasoning to predict outputs, showing an awareness of inputs. • Designs solutions by decomposing a problem and creates a sub-solution for each of these parts. • Recognises that different solutions exist for the same problem. **Programming and development** • Creates programs that implement algorithms to achieve given goals. • Declares and assigns variables. • Designs solutions by decomposing a problem and creates a sub-solution for each of these parts. • Recognises that different solutions exist for the same problem. • Uses a variable and relational operators within a loop to govern termination. ***Summer*** **Algorithms** • Designs solutions (algorithms) that use repetition and two-way selection i.e. if, then and else. • Uses logical reasoning to predict outputs, showing an awareness of inputs. • Designs solutions by decomposing a problem and creates a sub-solution for each of these parts. • Recognises that different solutions exist for the same problem. • Can identify similarities and differences in situations and can use these to solve problems (pattern recognition).

				Programming and development • Creates programs that implement algorithms to achieve given goals. • Declares and assigns variables. • Designs solutions by decomposing a problem and creates a sub-solution for each of these parts. • Recognises that different solutions exist for the same problem. • Uses a variable and relational operators within a loop to govern termination. • Uses a range of operators and expressions e.g. Boolean, and applies them in the context of program control. • Selects the appropriate data types.

KS2						
	Autumn 1	**Autumn 2**	**Spring 1**	**Spring 2**	**Summer 1**	**Summer 2**
	7 weeks	7 weeks	7 weeks	5 weeks	6 weeks	8 weeks
	Overview:	***TOPIC COVERAGE:***			***Objectives:***	
Year 6 and 7	**Year 6 and 7** What is the aim of this Programme of study? ***Skills focus: Coding*** To gain confidence and move from *encountering* to *mastering* a series of learning aims related to the following topics: • Algorithms • Programming • Data Representation • ***Engagement factors.*** • ***Enquiry based learning.***	***Autumn***			***Autumn***	
		Autumn 1	***Autumn 2***			
		Coding Languages- Introduction to Python Learn to input information in Python and generate an output by running the code. Learn how to use Python to make simple calculations and recognise symbols for multiplication (*) and division (/). Learn how inputs and variables work. Learn how to use selection and inputs. Learn how to use variables to store values.	**Coding Languages- Python Graphics** Use script to draw circles and change their colour. Learn how to use loops to draw shapes in Python. Learn how to program the turtle to create a simple face using coordinates. Develop the use of turtle graphics to affect background colour, line and width. Learn how to use loops to create different images.		***Autumn*** • author a simple program that outputs information • input key information accurately • understand the importance of the order of the code • identify symbols to show multiplication (*) and division (/) • input correct numbers to create a total • understand the effect of using quotes within calculations • run the input command to prompt an answer from the user • understand that they can use input for different questions • recognise a variable within the code • use variables to display the answer from the input command • understand that an 'if' command is a way of coding selection • use an 'if' command to test an input	

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

Cross Curricular (particularly with subjects which encounter information technology, computer hardware and processing and digital communication/safety).
Pupil Led Learning.
Developing practical skills.
Developing problem solving and critical thinking skills.

</td>
<td>Learn how to use the ‘if’ command and the ‘input’ with variables to make a quiz which keeps score.</td>
<td></td>
<td>

recognise that selection is how passwords work
demonstrate and explain the 'if' command
understand why code does not respond when an input is not recognised
understand the effect of changing the value of a variable
use variables to perform calculations
modify the name of a variable and understand its effect
work with multiple variables to perform more complex calculations
create their own quiz, with appropriate scores for each question
use a variable to store and increment a value
demonstrate they understand how the ‘if’ command works understand the code required to display a score and that the score it is a variable
use Python to draw different sized circles
input information to change position and colour
create their own pattern and explain the code they used
understand and explain how the radius of a circle is used in Python
To understand what graphics are and how to code lines and shapes
explain the commands needed to start programming the turtle understand the effects of the commands: forward, right and left know how to effect the direction or length by entering numerical values
understand the importance of sequence in the code
explain how to code a loop and know that a loop is repeated instruction
demonstrate how to use loops and define angles to draw a variety of 2D shapes
understand how to affect the size of shapes using Python
use loops to instruct the turtle to carry out a sequence of repeated commands
show understanding of how coordinates are coded in Python using the 'goto' command
understand how to vary the position, size and colour of circles using Python
use a sequence of code to create the desired outcome
recognise the commands that relate to background colour and line width

</td>
</tr>
<tr>
<td></td>
<td></td>
<td colspan="2">Spring</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Spring 1</td>
<td>Spring 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Coding Languages- Python: Random Numbers and Simulations

Learn how to use the random numbers library in Python

Learn how to combine random numbers and text in Python using different commands.

Learn how arrays can be used to produce phrases within Python.

Use text and arrays in Python to randomise elements of a story.

Learn how to combine the random library and the graphics library to create a variety of effects.

Learn how to combine loops with random numbers and graphics libraries to create 2D shapes.</td>
<td>Coding Languages - Python Functions

Learn how to define and call a function which uses parameters.

Learn how to define and call a function which uses parameters to create turtle graphics.

Learn how to draw a forest scene with the turtle, using random numbers and a loop.

Learn how to use the turtle to create a set of bear faces which are different sizes and in different positions.

Learn how to use the turtle to create a random set of emoticons with different expressions.

Learn how to use the turtle to create a random set of snowflakes in different positions on the screen.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td colspan="2">Summer</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Summer 1</td>
<td>Summer 2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Coding Languages- Intro to HTML

Learn how to get started with HTML by adding paragraphs of text to a page.

Learn how to add images to a web page using HTML.</td>
<td>Coding Languages- HTML Formatting and CSS/HTML Links

Learn how to change the colour of text using the colour property.

Learn how to change the size and font of text using the font-size and font-family properties.</td>
<td></td>
</tr>
</table>

		Understand new vocabulary associated with using HTML, including: images, jpgs, text, headings and paragraphs. Learn how to create a web page using headings, paragraphs and images. Learn how to apply knowledge of HTML to create a web page using headings, paragraphs and images. Learn how to apply knowledge of HTML to create a web page using headings, paragraphs and images. Create a simple web page about food using headings, paragraphs and images. .	Learn how to change the 'background', 'margin' and 'padding' properties of different parts of a web page. Learn how to apply knowledge of HTML to make a web page using text, headings, images and styling. Learn how to apply knowledge of HTML to make a web page using text, headings, images and styling. Learn more about how web developers structure their pages and HTML standards. Learn how to add links to websites and pages. Learn how to make a link using an image. Learn how to make a page with anchor tags and section IDs to navigate within the page. Learn how to use div tags within a web page. Make a web page combining divs, images and anchor links. Create a web page that combines the use of div tags, styles and anchor links.	• experiment with coordinates, pixel width and angles to make their own picture • combine a variety of commands to create the desired outcome • experiment with coding the width of shapes, the background colour and the thickness of lines to create a design • program a loop • recognise how to create an image using loops • use appropriate terminology and commands when explaining their code • demonstrate the effect of modifying the turtle's direction within a loop ***Spring*** • recognise that 'random' is a library of code • demonstrate how to create a variety of outputs using random numbers • select and sequence code correctly to total randomly generated numbers • use randomised values in different contexts, combining numbers and text • demonstrate how to generate a random number from a range • understand how to output a random value from an array • demonstrate how to generate random numbers using a loop • understand how to add an item to an array • create phases of text using the random library with arrays • add adjectives and nouns to an array to affect the random choice • create a four line poem by using a loop • understand how and when to use more than one array • write their own poem by inputting data into arrays and sequencing the code • understand the importance of order when writing code • write, select and sequence code to create their own story • demonstrate the effect of changing an item in an array • understand the effect of using the random library with text • code the turtle to draw randomised circles • use a loop to change both the colour and the direction of turtle

				• understand how to define and set RGB (red, green, blue) values • demonstrate how to affect the colour and position of circles using the random library • demonstrate an understanding of loops, angles and direction to draw a shape • write code to randomly affect the position of the shapes within a loop • use random numbers to create multi-coloured shapes • create a simple program which defines and calls a function • use a loop with a parameter to call the same function more than once • create a program which uses more than one parameter • explain how using functions can makes their code more efficient • create a program which draws 20 trees in random positions • explain how random numbers have been used in their code • write a program which defines and calls a function to create a bear face • use random numbers and a loop to create several bear faces in different positions and of different sizes. • write a program which defines and calls a function to create a set of emoticons with different expressions • use random numbers and a loop to create several emoticons in different positions on screen • write a program which defines and calls a function to draw a snowflake • use random numbers and a loop to create several snowflakes in random positions on screen ***Summer*** • identify opening and closing tags • add paragraph tags and heading tags to create a simple web page understand how to control the size of text using HTML tags • understand the vocabulary associated with HTML, including: angle brackets, tags, paragraphs and headings • select and sequence code, adding images and text to create a simple program in HTML

				• understand new vocabulary associated with this lesson including; images, jpgs, graphics • explain the meaning of tag abbreviation 'img src' and know what 'PNG' is short for • use heading tags, paragraph tags and image tags together to create a web page • correctly select the appropriate tags to format the content • understand associated vocabulary, including: headings, paragraphs, images and source • use heading tags, paragraph tags and image tags together to create a web page • correctly select the appropriate tags to format the content • use heading tags, paragraph tags and image tags together to create a web page • correctly select the appropriate tags to format the content • present their understanding of terms and elements learned in HTML Unit 1 to a partner or class • create a web page about food, sequencing code and repeating a sequence to make their own web page including, headings, paragraphs and images • talk about what they KWL (Know, Want, Learn) from the unit, giving feedback to their peers • use search technologies effectively • understand that styles affect the design of the web page • change the colour of text using words • understand what RGB (red, green, blue) is • use RGB values to change the colour of text within a style attribute • write a style attribute with a colour property • control the size of text in pixels • control the font of text • understand how ‘font-size’ and ‘font-family’ properties are used within a style attribute • understand how to change the ‘background-color’ using text and hex values • change the ‘background-image’ of the web page

				• understand how to use a style section to control the ‘background-color’, ‘margin’ and ‘padding’ of all paragraph tags • use headings, paragraphs and images to build a web page • control the appearance of text using the ‘font-family’, ‘font-size’ and colour properties • control the layout of the web page using the ‘margin’ and ‘padding’ properties • understand how to affect the appearance of all paragraph tags using the style section • understand how to control the appearance of specific elements using inline styles • use headings, paragraphs and images to build a web page • control the appearance of text using the ‘font-family’, ‘font-size’ and colour properties • control the layout of the web page using the ‘margin’ and ‘padding’ properties • understand how to affect the appearance of all paragraph tags using the style section • understand how to control the appearance of specific elements using inline styles • understand why HTML has precise standards and how to implement some of these • use style sections to more efficiently set colour and other values • understand how to use a separate style sheet • understand how to create absolute links to another website • use relative links to make links from one web page to another in the same site • make their own index page using links, and organise it by inserting line breaks • understand how to turn an image into a link • create images with relative and absolute links • use the width attribute of the ‘img’ tag to set the size of the link area • use an anchor tag to jump back to the top of the page they are currently on • make links to different sections of the same page • explain what an anchor tag is, and how to use one together with an ID • understand how a div can be used to separate a page into sections

				• add a style selector to adjust the colour and height of each div tag • use an anchor link to scroll to a div tag within the same page • demonstrate how to select and sequence code to structure a web page • explain how to use style selectors to control the appearance of divs within a web page • understand how anchor links and IDs work together • structure a web page and input their own content • apply styling to sections of the page • demonstrate how to use anchor links to build navigation within a web page

KS3						
	Autumn 1	**Autumn 2**	**Spring 1**	**Spring 2**	**Summer 1**	**Summer 2**
	7 weeks	7 weeks	7 weeks	5 weeks	6 weeks	8 weeks
Year 8	***Overview:***	***TOPIC COVERAGE:***			***Objectives:***	
	Year 8 What is the aim of this Programme of study? *Skills focus:* *Programming techniques and computational thinking* To gain confidence and move from *encountering* to *mastering* a series of learning aims related to the following topics: • Algorithms • Programming	***Autumn***			***Autumn*** • can understand and apply the fundamental principles and concepts of computer science, including abstraction, logic, algorithms and data representation • have repeated practical experience of writing computer programs in order to solve problems • can evaluate and apply information technology • are responsible, competent, confident and creative users of information and communication technology • design, write and debug programs that accomplish specific goals • use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs		
		Autumn 1		***Autumn 2***		
		Sphero- Course 2 (Theme: Empathy) <u>Design & Development:</u> The activities involved in planning, creating and evaluating computing artefacts <u>Programming languages:</u> Sphero Draw/Blocks/Text (based on Java Script)		**Makecode Arcade [Intermediate]: Functions, extensions, animation, difficulty levels, multi-player, tile maps** <u>Algorithms:</u> Being able to comprehend, design, create and evaluate algorithms <u>Programming languages:</u> Creating software to allow computers to solve problems <u>Programming languages:</u> Microsoft MakeCode (Block/Python/Java)		
		Spring				
		Spring 1		***Spring 2***		

	• Data Representation • Hardware and Processing • Information Technology • *Engagement factors* • *Enquiry based learning.* • *Cross Curricular (particularly with subjects which encounter information technology, computer hardware and processing and digital communication/safety).* • *Pupil Led Learning.* • *Developing practical skills.* • *Developing problem solving and critical thinking skills.*	**Makecode Arcade [Intermediate]: Controls, level design, number generation, dialogue scripts, sprite arrays** <u>Algorithms:</u> Being able to comprehend, design, create and evaluate algorithms <u>Programming languages:</u> Creating software to allow computers to solve problems <u>Programming languages:</u> Microsoft MakeCode (Block/Python/Java)	**Makecode Arcade [Intermediate]: Skills development** <u>Algorithms:</u> Being able to comprehend, design, create and evaluate algorithms <u>Programming languages:</u> Creating software to allow computers to solve problems <u>Programming languages:</u> Microsoft MakeCode (Block/Python/Java)	• use sequence, and repetition in programs • can understand and apply the fundamental principles and concepts of computer science, including abstraction, logic, algorithms and data representation • have repeated practical experience of writing computer programs in order to solve problems • can evaluate and apply information technology • are responsible, competent, confident and creative users of information and communication technology • design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; • solve problems by decomposing them into smaller parts • use sequence, selection and repetition in programs; work with ... various forms of input and output • use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs
		Summer		
		Summer 1	***Summer 2***	
		Sphero- Course 2 **(Theme: Storytelling)** <u>Design & Development:</u> The activities involved in planning, creating and evaluating computing artefacts <u>Programming languages:</u> Sphero Draw/Blocks/Text (based on Java Script)	**Sphero- Course 2** **(Theme: Game Design)** <u>Design & Development:</u> The activities involved in planning, creating and evaluating computing artefacts <u>Programming languages:</u> Sphero Draw/Blocks/Text (based on Java Script)	**Spring** • can understand and apply the fundamental principles and concepts of computer science, including abstraction, logic, algorithms and data representation • have repeated practical experience of writing computer programs in order to solve problems • can evaluate and apply information technology • are responsible, competent, confident and creative users of information and communication technology • design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; • solve problems by decomposing them into smaller parts, use sequence, selection, in programs; work with various forms of input and output • use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs • can understand and apply the fundamental principles and concepts of computer science,

				including abstraction, logic, algorithms and data representation ● have repeated practical experience of writing computer programs in order to solve problems ● can evaluate and apply information technology ● are responsible, competent, confident and creative users of information and communication technology ● design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts ● use sequence, selection and repetition in programs; work with variables and various forms of input and output ● use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs ***Summer*** ● can understand and apply the fundamental principles & concepts of computer science. ● practical experience of writing computer programs to solve problems. ● can evaluate and apply information technology, including new or unfamiliar technologies analytically to solve problems ● are responsible, competent, confident and creative users of information and communication technology. ● design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems ● use logical reasoning to compare the utility of alternative algorithms for the same problem ● use two or more programming languages, at least one of which is textual, to solve a variety of computational problems ● understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems ● understand how instructions are stored and executed within a computer system

				● undertake creative projects that involve selecting, using, and combining multiple applications, preferably across a range of devices, to achieve challenging goals, including collecting and analysing data and meeting the needs of known users ● create, re-use, revise and re-purpose digital artefacts for a given audience, with attention to trustworthiness, design and usability ● can understand and apply the fundamental principles & concepts of computer science. ● practical experience of writing computer programs to solve problems. ● can evaluate and apply information technology, including new or unfamiliar technologies analytically to solve problems ● are responsible, competent, confident and creative users of information and communication technology. ● design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems ● use logical reasoning to compare the utility of alternative algorithms for the same problem ● use two 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]; ● understand simple Boolean logic [for example, AND, OR and NOT] and some of its uses in circuits and programming; ● understand a range of ways to use technology safely, respectfully, responsibly and securely, including protecting their online identity and privacy; recognise inappropriate content, contact and conduct and know how to report concerns.

KS3						
	Autumn 1	**Autumn 2**	**Spring 1**	**Spring 2**	**Summer 1**	**Summer 2**
	7 weeks	7 weeks	7 weeks	5 weeks	6 weeks	8 weeks
	Overview:	***TOPIC COVERAGE:***			***Objectives:***	

Year 9

	Year 9 What is the aim of this Programme of study? ***Skills focus: Programming techniques and computational thinking*** To gain confidence and move from *encountering* to *mastering* a series of learning aims related to the following topics: • Algorithms • Programming • Data Representation • Hardware and Processing • Information Technology • *Engagement factors* • *Enquiry based learning.* • *Cross Curricular (particularly with subjects which encounter information technology, computer hardware and processing and digital communication/safety).* • *Pupil Led Learning.* • *Developing practical skills.* • *Developing problem solving and critical thinking skills.*	***Autumn***		***Autumn***
		Autumn 1 **Sphero- Course 3 (Theme: Brain Breakers)** <u>Design & Development:</u> The activities involved in planning, creating and evaluating computing artefacts <u>Programming languages:</u> Sphero Draw/Blocks/Text (based on Java Script)	***Autumn 2*** **Makecode Arcade [Intermediate]: Functions, extensions, animation, difficulty levels, multi-player, tile maps** <u>Algorithms:</u> Being able to comprehend, design, create and evaluate algorithms <u>Programming languages:</u> Creating software to allow computers to solve problems <u>Programming languages:</u> Microsoft MakeCode (Block/Python/Java)	• can understand and apply the fundamental principles & concepts of computer science. • practical experience of writing computer programs to solve problems. • can evaluate and apply information technology, including new or unfamiliar technologies analytically to solve problems • are responsible, competent, confident and creative users of information and communication technology. • design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems • use logical reasoning to compare the utility of alternative algorithms for the same problem • use two or more programming languages, at least one of which is textual, to solve a variety of computational problems • understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems • understand how instructions are stored and executed within a computer system • use two 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]; • understand simple Boolean logic [for example, AND, OR and NOT] and some of its uses in circuits and programming; • understand a range of ways to use technology safely, respectfully, responsibly and securely, including protecting their online identity and
	Spring					
		Spring 1 **Makecode Arcade [Intermediate]: Controls, level design, number generation, dialogue scripts, sprite arrays** <u>Algorithms:</u> Being able to comprehend, design, create and evaluate algorithms <u>Programming languages:</u> Creating software to allow computers to solve problems <u>Programming languages:</u> Microsoft MakeCode (Block/Python/Java).	***Spring 2*** **Makecode Arcade [Intermediate]: Skills development** <u>Algorithms:</u> Being able to comprehend, design, create and evaluate algorithms <u>Programming languages:</u> Creating software to allow computers to solve problems <u>Programming languages:</u> Microsoft MakeCode (Block/Python/Java)	
	Summer					
		Summer 1	***Summer 2***			

		Sphero- Course 3 (Theme: Missions) Design & Development: The activities involved in planning, creating and evaluating computing artefacts Programming languages: Sphero Draw/ Blocks/ Text (based on Java Script)	**Sphero- Course 3 (Theme: Navigation)** Design & Development: The activities involved in planning, creating and evaluating computing artefacts Programming languages: Sphero Draw /Blocks/ Text (based on Java Script)	privacy; recognise inappropriate content, contact and conduct and know how to report concerns. ● undertake creative projects that involve selecting, using, and combining multiple applications, preferably across a range of devices, to achieve challenging goals, including collecting and analysing data and meeting the needs of known users ● create, re-use, revise and re-purpose digital artefacts for a given audience, with attention to trustworthiness, design and usability ***Spring*** ● can understand and apply the fundamental principles and concepts of computer science ● have repeated practical experience of writing computer programs in order to solve problems ● can evaluate and apply information technology ● are responsible, competent, confident and creative users of information and communication technology ● design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems ● use logical reasoning to compare the utility of alternative algorithms for the same problem ● use two or more programming languages, at least one of which is textual, to solve a variety of computational problems ● can understand and apply the fundamental principles and concepts of computer science ● have repeated practical experience of writing computer programs to solve problems ● are responsible, competent, confident and creative users of information and communication technology. ● design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems ● use logical reasoning to compare the utility of alternative algorithms for the same problem ● use two or more programming languages, at least one of which is textual, to solve a variety of computational problems

				• understand a range of ways to use technology safely, respectfully, responsibly and securely, including protecting their online identity and privacy; recognise inappropriate content, contact and conduct and know how to report concerns. ***Summer*** • can understand and apply the fundamental principles and concepts of computer science, including abstraction, logic, algorithms and data representation • can evaluate and apply information technology, including new or unfamiliar technologies, analytically to solve problems • understand simple Boolean logic [for example, AND, OR and NOT] and some of its uses in circuits and programming • understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems • understand how instructions are stored and executed within a computer system • Understand that there are different programming languages, of which Small Basic is one. • Be able to write a basic program by breaking a task down into instructions. --- • Understand what is meant by 'user input' • Know what is meant by 'variable' • Be able to link user input with a variable • Understand how programming languages can use graphics as well as text • Explain how variables can be used • Be able to demonstrate an understanding of computational thinking • Be able to respond effectively to feedback • Be able to use IF and ELSE statements accurately. • Be able to break down a process into instructions which have different outcomes depending on the input. • Understand what ELSEIF is used for. • Understand what is meant by a loop

				• Know why loops are used to make programs more efficient • Be able to change the number of times a loop runs and explain what it will do to a program • Recognise that a while loop can be used as well as a for loop • Understand the difference between a while loop and a for loop • Be able to explain why a while loop could be used efficiently • Recognise that a while loop can be used as well as a for loop • Understand the difference between a while loop and a for loop • Be able to explain why a while loop could be used efficiently

KS3

Year 10

Autumn 1	Autumn 2	Spring 1	Spring 2	Summer 1	Summer 2
7 weeks	7 weeks	7 weeks	5 weeks	6 weeks	8 weeks

Overview:

Year 10

What is the aim of this Programme of study?
Skills focus: Programming techniques, algorithms and problem-solving skills, computational thinking

To become familiar with how computer technology work including:

- Fundamental principles of computer science including problem solving, logic, algorithms and programming.
- Analyse problems in computational terms.

TOPIC COVERAGE:

Autumn

Autumn 1

Programming Part 1- Sequence

Topics:
Lesson 1: Translators
Lesson 2: Sequence
Lesson 3: Variables
Lesson 4: Input
Lesson 5: Flowcharts
Lesson 6: Randomisers

Programming language: Python

Autumn 2

Programming Part 2- Selection

Topics:
Lesson 7: Arithmetic expressions
Lesson 8: Selection
Lesson 9: Selection challenge
Lesson 10: Logical expressions
Lesson 11: Nested selection
Lesson 12: While loops

Programming language: Python

Spring

Spring 1

Programming Part 3- Iteration

Topics:
Lesson 13: Trace tables
Lesson 14: For loops

Spring 2

Programming Part 4- Subroutines

Topics:
Lesson 19: Functions
Lesson 20: Scope

Objectives:

Autumn

- Understand that there are different programming languages, of which python is one.
- Be able to write a basic program by breaking a task down into instructions.
- Understand what is meant by 'user input'
- Know what is meant by 'variable'
- Be able to link user input with a variable
- Understand how programming languages can use graphics as well as text
- Explain how variables can be used
- Be able to demonstrate an understanding of computational thinking
- Be able to respond effectively to feedback
- Be able to use IF and ELSE statements accurately.
- Be able to break down a process into instructions which have different outcomes depending on the input.
- Understand what ELSEIF is used for.
- Understand what is meant by a loop

	• Practical experience of writing computer programs in order to solve problems. • *Engagement factors* • *Enquiry based learning.* • *Cross Curricular (particularly with subjects which encounter information technology, computer hardware and processing and digital communication/safety).* • *Pupil Led Learning.* • *Developing practical skills.* • *Developing problem solving and critical thinking skills.*	Lesson 15: Data validation Lesson 16 and 17: Pseudocode Lesson 18: Subroutines <u>Programming language</u>: Python Learn how to combine the random library and the graphics library to create a variety of effects. Learn how to combine loops with random numbers and graphics libraries to create 2D shapes.	Lesson 21: XOR Lesson 22: Structured programming Lesson 23 and 24: Create a program <u>Programming language</u>: Python	• Know why loops are used to make programs more efficient • Be able to change the number of times a loop runs and explain what it will do to a program • Recognise that a while loop can be used as well as a for loop **Spring** • Understand the difference between a while loop and a for loop • Be able to explain why a while loop could be used efficiently • Recognise that a while loop can be used as well as a for loop • Understand the difference between a while loop and a for loop • Be able to explain why a while loop could be used efficiently • Learners demonstrate knowledge and understanding of ideas related to computational thinking. • Learners demonstrate their ability to recall, select and communicate their knowledge and understanding of concepts, issues and terminology. • Learners demonstrate their ability to analyse problems in computational terms to make reasoned judgement and to design, program, and evaluate solutions. Identify and use variables, operators, inputs, outputs and assignments • Understand and use the three basic programming constructs used to control the flow of a program: • Sequence • Selection • IF Statements • Iteration • Count and condition-controlled loops: WHILE and FOR Understand and use basic string manipulation concatenation only **Summer** • Use different types of data: Integer Boolean Real numbers Text Character and string • Define and use arrays (or equivalent) as appropriate when solving problems
		Summer		
		Summer 1	***Summer 2***	
		Programming Part 5- Strings and Lists <u>Topics:</u> Lesson 25: GUIs Lesson 26: String handling I Lesson 27: String handling II Lesson 28: String handling III Lesson 29: Arrays and lists Lesson 30: List methods <u>Programming language</u>: Python.	**Programming Part 6- Dictionaries and Datafiles** <u>Topics:</u> Lesson 31: Sense HAT I Lesson 32: Sense HAT II Lesson 33: 2D arrays and lists Lesson 34 and 35: 2D lists challenge Lesson 36: Records and dictionaries Lesson 37: Dictionary challenge Lesson 38: Reading text files <u>Programming language</u>: Python Make a web page combining divs, images and anchor links. Create a web page that combines the use of div tags, styles and anchor links.	

				• one dimensional arrays (or similar) • Use the common Arithmetic operators • Use the common Boolean operators

KS3						
	Autumn 1	**Autumn 2**	**Spring 1**	**Spring 2**	**Summer 1**	**Summer 2**
	7 weeks	7 weeks	7 weeks	5 weeks	6 weeks	8 weeks
	Overview:	***TOPIC COVERAGE:***			***Objectives:***	
Year 11	**Year 11** What is the aim of this Programme of study? ***Skills focus: Programming techniques, algorithms and problem-solving skills, computational thinking*** To become familiar with how computer technology work including: • Fundamental principles of computer science • including problem solving, logic, algorithms and programming. • Analyse problems in computational terms. • Practical experience of writing computer programs in order to solve problems. • Evaluate both new and unfamiliar technologies. • Become responsible, confident and creative users of • computer science related technologies. • Understand the components of digital	***Autumn***				**Autumn** • Understand that there are different programming languages, of which Small Basic is one. • Be able to write a basic program by breaking a task down into instructions. • Understand what is meant by 'user input' • Know what is meant by 'variable' • Be able to link user input with a variable • Understand how programming languages can use graphics as well as text • Explain how variables can be used • Be able to demonstrate an understanding of computational thinking • Be able to respond effectively to feedback • Be able to use IF and ELSE statements accurately. • Be able to break down a process into instructions which have different outcomes depending on the input. • Understand what ELSEIF is used for. • Understand what is meant by a loop • Know why loops are used to make programs more efficient • Be able to change the number of times a loop runs and explain what it will do to a program • Recognise that a while loop can be used as well as a for loop • Understand the difference between a while loop and a for loop	
		Autumn 1		***Autumn 2***		
		Sphero- Course 3 (Theme: Brain Breakers) <u>Design & Development:</u> The activities involved in planning, creating and evaluating computing artefacts <u>Programming languages:</u> Sphero Draw/Blocks/Text (based on Java Script)		**Programming project (20%):** **Planning a solution** **Developing a solution** Planning a solution Developing a solution Testing a solution Evaluating the success of the solution Learn key programming ideas and how to use them in block/ text-based code		
		Spring				
		Spring 1		***Spring 2***		
		Programming project (20%): **Testing a solution** **Evaluating the success of the solution** Planning a solution Developing a solution Testing a solution Evaluating the success of the solution		**Sphero- Course 3 (Theme: Missions)** <u>Design & Development:</u> The activities involved in planning, creating and evaluating computing artefacts <u>Programming languages:</u> Sphero Draw/Blocks/Text (based on Java Script)		

<table>
<tr>
<td></td>
<td>systems and how they communicate with one another.
• Understand the impact of digital technology to individuals in wider society.

• Engagement factors
• Enquiry based learning.
• Cross Curricular (particularly with subjects which encounter information technology, computer hardware and processing and digital communication/safety).
• Pupil Led Learning.
• Developing practical skills.
• Developing problem solving and critical thinking skills.</td>
<td>Learn key programming ideas and how to use them in block/text based code</td>
<td></td>
<td>• Be able to explain why a while loop could be used efficiently
• Recognise that a while loop can be used as well as a for loop
• Understand the difference between a while loop and a for loop
• Be able to explain why a while loop could be used efficiently</td>
</tr>
<tr>
<td></td>
<td></td>
<td colspan="2">Summer</td>
<td rowspan="2">Spring
• Learners demonstrate knowledge and understanding of ideas related to computational thinking.
• Learners demonstrate their ability to recall, select and communicate their knowledge and understanding of concepts, issues and terminology.
• Learners demonstrate their ability to analyse problems in computational terms to make reasoned judgement and to design, program, and evaluate solutions. Identify and use variables, operators, inputs, outputs and assignments
• Understand and use the three basic programming constructs used to control the flow of a program:
• Sequence
• Selection
• IF Statements
• Iteration
• Count and condition controlled loops: WHILE and FOR Understand and use basic string manipulation concatenation only
Summer
• Use different types of data:
Integer
Boolean
Real numbers
Text
Character and string
• Define and use arrays (or equivalent) as appropriate when solving problems
• one dimensional arrays (or similar)
• Use the common Arithmetic operators
• Use the common Boolean operators</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Summer 1</td>
<td>Summer 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sphero- Course 3 (Theme: Navigation)

<u>Design & Development:</u>
The activities involved in planning, creating and evaluating computing artefacts

<u>Programming languages:</u> Sphero Draw/Blocks/Text (based on Java Script)</td>
<td>Sphero- Course 3 (Theme: Brain Breakers)

<u>Design & Development:</u>
The activities involved in planning, creating and evaluating computing artefacts

<u>Programming languages:</u> Sphero Draw/Blocks/Text (based on Java Script)</td>
</tr>
</table>