ICKNIELD HIGH SCHOOL **KEY STAGE 3 CURRICULUM MAP**

'Raising the Achievement of All'

WELL-SEQUENCED **CONTEXUALISED/INCLUSIVE** **AMBITIOUS**

SUBJECT: Computing **Year: 8**

	Autumn 1 **Representations: from clay to silicon** Representing numbers and text using binary digits.	**Autumn 2** **Computing systems** Exploring the fundamental elements that make up a computer system.	**Spring 1** **Developing for the web** Using HTML and CSS to create webpages.	**Spring 2** **Data science** Using data to investigate problems and make real-world changes.	**Summer 1** **Media: vector graphics** Creating vector graphics through objects, layering, and path manipulation.	**Summer 2** **Introduction to Python programming** Applying the programming constructs of sequence, selection, and iteration in Python.
Key Questions	-What is binary? -How does it work in circuitry? -Why do computers use binary? -How do I Convert between binary and decimal (vice versa) - What are the different ways binary digits are Physically represented in digital devices?	- How instructions are stored and executed within a computer system? - How is binary used to store various data types? - How can computers collect data from various input devices, including sensors and application software. - What is the difference between hardware and application software, and their roles within a computer system.	- What are the benefits to websites in terms of communication in using 1 universal scripting language? - What are the benefits of using CSS to a website? - How can I develop online? based platforms for a specific purpose?	-What is Data Science? Why do we need Data Science to analyse data sets? What is Big Data? -Why are Data Scientists key in our society? How can we use that data analysis for effectiveness? How can it be used for different technologies? - What are the latest technological advances that will affect society? -What are the benefits and pitfalls of specific current and future technology initiatives? How do data scientists play a key role?	- How can I create and reuse digital artefacts and multiple applications across a range of devices to present information suitable for my audience? -How do I use media effectively so that I can get the message I need across? -How do I ensure my digital artefact is fit for purpose?	- How can I use computational thinking to solve problems? - Can I apply block-Based programming concepts to high level programming? - Can I create programs that implement algorithms to achieve given goals? - Can I use loops and a sequence of selection statements in programs, including an IF, THEN and ELSE statement? - Can I use sequence, selection to develop a program to solve a problem?

ICKNIELD HIGH SCHOOL **KEY STAGE 3 CURRICULUM MAP**

Content: 'Know what'	-Understand what a computer is, and how its constituent parts function together as a whole. -Understand how data is stored, organised, and used to represent real-world artefacts and scenarios.	-What is a computer, how do it's constituent parts function together as a whole. -How is data stored, organized and used to represent real world artefacts and scenarios. -How individuals, systems and society as a whole interact with computer systems. -Creating software to allow computers to solve problems	-Being able to comprehend, design, create and evaluate algorithms. -Select and create a range of media including text, images, sounds and video. -The activities involved in planning, creating and evaluating computing artefacts. -Use software tools to support computing work. -Understanding risks when using technology and how to protect individuals and systems.	-Select and create a range of media including text, images, sounds and video. -How is data stored, organized and used to represent real world artefacts and scenarios. -Use software tools to support computing work. -How individuals, systems and society as a whole interact with computer systems.	-Select and create a range of media including text, images, sounds and video. -The activities involved in planning, creating and evaluating computing artefacts. -How is data stored, organized and used to represent real world artefacts and scenarios. - Use software tools to support computing work.	-Being able to comprehend, design, create and evaluate algorithms. -What is a computer, how do it's constituent parts function together as a whole.
Expertise: 'Know how'	-Understand how instructions are stored and executed within a computer system; understand how data of various types (including text, sounds and pictures) can be represented and manipulated digitally, in the form of binary digits. - List examples of representations. - Recall that Representations are used to store, communicate, and process information. - Provide examples of how different representations are	- 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]. - 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	- 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, reuse, revise and repurpose digital artefacts for a given audience, with attention to trustworthiness, design and usability. - Draw basic shapes (rectangle, ellipse, polygon, star) with	- 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. - Define data science. - Explain how visualising data can help identify patterns and trends in order to help us gain insights. - Use an appropriate software tool to visualise data sets and look for patterns or trends.	- 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, reuse, revise and repurpose digital artefacts for a given audience, with attention to trustworthiness, design and usability. - Draw basic shapes (rectangle, ellipse, polygon, star) with	- Design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems. - 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. - Use 2 or more programming languages, at least one of which is textual, to solve a variety of computational

	appropriate for different tasks. - Recall that characters can be represented as sequences of symbols and list examples of character coding schemes. - Measure the length of a representation as the number of symbols that it contains. - Provide examples of how symbols are carried on physical media. - Explain what binary digits (bits) are, in terms of familiar symbols such as digits or letters. - Measure the size or length of a sequence of bits as the number of binary digits that it contains. - Describe how natural numbers are represented as sequences of binary digits. - Convert a decimal number to binary and vice versa. - Convert between different units and multiples of representation size. - Provide examples of	executed within a computer system; understand how data of various types (including text, sounds and pictures) can be represented and manipulated digitally, in the form of binary digits. - Recall that a general-purpose computing system is a device for executing programs. - Recall that a program is a sequence of instructions that specify operations that are to be performed on data. - Explain the difference between a general-purpose computing system and a purpose-built device. - Describe the function of the hardware components used in computing systems. - Describe how the hardware components used in computing systems work together in order to execute programs. - Recall that all computing systems, regardless of form, have a similar structure ('architecture'). - Analyse how the hardware components used in computing systems work together in order to execute programs. - Define what an operating system is, and recall its role in	different properties (fill and stroke, shape-specific attributes). - Manipulate individual objects (select, move, resize, rotate, duplicate, flip, z-order). - Recall that a general-purpose computing system is a device for executing programs. - Recall that a program is a sequence of instructions that. specify operations that are to be performed on data. - Explain the difference between a general-purpose computing system and a purpose-built device. - Describe what HTML is. - Use HTML to structure static web pages. - Modify HTML tags using inline styling to improve the appearance of web pages. - List examples of representations. - Recall that representations are used to store, communicate, and process information.	- Recognise examples of where large data sets are used in daily life. - Select criteria and use data set to investigate predictions. - Evaluate findings to support arguments for or against a prediction. - Define the terms 'correlation' and 'outliers' in relation to data trends. - Identify the steps of the investigative cycle. - Solve a problem by implementing steps of the investigative cycle on a data set. - Use findings to support a recommendation. - Identify the data needed to answer a question defined by the learner. - Create a data capture form. - Describe the need for data cleansing. - Apply data cleansing techniques to a data set. - Visualise a data set. - Analyse visualisations to identify patterns, trends, and outliers. - Draw conclusions and report findings	different properties (fill and stroke, shape-specific attributes). - Manipulate individual objects (select, move, resize, rotate, duplicate, flip, z-order). - Manipulate groups of objects (select, group/ungroup, align, distribute). - Combine paths by applying operations (union, difference, intersection). - Convert objects to paths. - Draw paths. - Edit path nodes. Combine multiple tools and techniques to create a vector graphic design. - Explain what vector graphics are. - Provide examples where using vector graphics would be appropriate. - Peer assess another pair's project work. - Improve your own project work based on feedback. - Complete a summative assessment.	problems; make appropriate use of data structures [for example, lists, tables or arrays]; design and develop modular programs that use procedures or functions. Understand how instructions are stored and executed within a computer system; understand how data of various types (including text, sounds and pictures) can be represented and manipulated digitally, in the form of binary digits. - Describe what algorithms and programs are and how they differ. - Recall that a program written in a programming language needs to be translated in order to be executed by a machine. - Write simple Python programs that display messages, assign values to variables, and receive keyboard input. - Locate and correct common syntax errors. - Describe the semantics of assignment statements. - Use simple arithmetic

	the different ways that binary digits are physically represented in digital devices. - Apply all of the skills covered in this unit.	controlling program execution. - Describe the NOT, AND, and OR logical operators, and how they are used to form logical expressions. - Use logic gates to construct logic circuits, and associate these with logical operators and expressions. - Describe how hardware is built out of increasingly complex logic circuits. Recall that, since hardware is built out of logic circuits, data and instructions alike need to be represented using binary digits. - Provide broad definitions of 'artificial intelligence' and 'machine learning'. - Identify examples of artificial intelligence and machine learning in the real world. - Describe the steps involved in training machines to perform tasks (gathering data, training, testing). - Describe how machine learning differs from traditional programming. - Associate the use of artificial intelligence with moral dilemmas. - Explain the implications of sharing program code.	- Provide examples of how different representations are appropriate for different tasks. - Identify when a problem needs to be broken down. - Implement and customise GUI elements to meet the needs of the user. - Describe what algorithms and programs are and how they differ. - Recall that a program written in a programming language needs to be translated in order to be executed by a machine. - Write simple Python programs that display messages, assign values to variables, and receive keyboard input. - Locate and correct common syntax errors.			expressions in assignment statements to calculate values. Receive input from the keyboard and convert it to a numerical value. - Use relational operators to form logical expressions - Use binary selection (if, else statements) to control the flow of program execution. - Generate and use random integers. - Use multi-branch selection (if, elif, else statements) to control the flow of program execution. - Describe how iteration (while statements) controls the flow of program execution. - Use iteration (while loops) to control the flow of program execution. - Use variables as counters in iterative programs. - Combine iteration and selection to control the flow of program execution. - Use Boolean variables as flags
Literacy/ Numeracy/ SMSC/	- The impact of (social) communication today on society (positive and			- Collecting and analysing real world data.	**Literacy** - Glossary tier 3 terminology.	**Literacy** - Glossary tier 3 terminology.

	negative).			- Ability to follow instructions.	**Numeracy** - Drawing geometrical shapes and manipulating these using software. -Combing paths between shapes by applying operation (union, difference, intersection). -Problem solving. **SMSC** - Ability to follow series of instructions to complete tasks. -Understanding the uses of vector graphics. -Group work. -Creativity. -Initiative.	**Numeracy** -Students make use of arithmetic operations to construct programmes. - Use of iteration **SMSC** - Peerassessments. - Group work. - Constructing algorithms based on real work scenarios. Ability to follow set of instructions.
Careers	https://icould.com/stories/nicky-s/ https://icould.com/stories/tariq-s/	https://icould.com/stories/tariq-s/ https://icould.com/stories/tariq-s/	https://www.youtube.com/watch?v=fU47qvOCmlQ https://www.youtube.com/watch?v=fU47qvOCmlQ	https://www.youtube.com/watch?v=fU47qvOCmlQ https://www.youtube.com/watch?v=fU47qvOCmlQ	https://www.youtube.com/watch?v=fU47qvOCmlQ https://www.youtube.com/watch?v=WPC2kBzpq_k https://www.youtube.com/watch?v=g_cAsmciL7g	https://www.youtube.com/watch?v=g_cAsmciL7g https://icould.com/stories/malcolm-b/
Disciplinary Literacy	• Prioritise Tier 2 and 3 vocabulary. • Provide explicit vocabulary instruction to help students access and use academic language. • Reading and Analysing Technical Texts: Encourage students to read and critically analyse technical texts related to computing. This could include textbooks, articles, or online resources that explain various concepts or technologies. Students can then discuss and reflect on the ideas presented in these texts. Reading strategies, such as activating prior knowledge, prediction and questioning can improve students' comprehension. • Writing for Different Purposes: Assign tasks that require students to write for different purposes within the context of computing. For example, they can write instructions for a coding project, create technical documentation for a program, or write a report analysing the ethical implications of a technology. Targeted support should be provided to students who struggle to write fluently, as this may affect writing quality. • Understanding Domain-Specific Vocabulary: Teach students the specific vocabulary and terminology used in computing. Make sure they understand the meanings of words such as algorithm, variable, loop, binary, cybersecurity, etc. Encourage them to use this vocabulary correctly and effectively in their written and verbal communication. Teachers can also use high quality talk to introduce new vocabulary and encourage students to incorporate these terms into their discussions. • Collaborative Discussions: High quality talk is typically well-structured and guided by teachers. High quality talks promotes collaborative discussions and group work in which students can engage					

	in discourse about computing-related topics. Encourage them to share their ideas, ask questions, and challenge each other's viewpoints. This will help develop their ability to communicate effectively and engage in disciplinary conversations. Modelling high quality talk, for example key vocabulary and metacognitive reflection. By engaging in discussions, students can deepen their understanding of key terms and concepts. • Presentations and Debates: Assign presentations or debates on computing-related topics to allow students to present their ideas, arguments, and evidence to an audience. This will provide them with an opportunity to develop their oral communication skills, research abilities, and critical thinking. • Combining reading activities and writing instruction to improve students' skills in both, particularly with coding units. • Teachers can use questioning techniques to prompt students to reflect on their thought processes, identify gaps in their understanding, and develop strategies to improve their learning.