SAIL Academy Curriculum, Skills and Intent SECONDARY Computing

Year Group 7	Term One	Term Two	Term Three	Term Four	Term Five	Term Six
Intent of Study	In Year 7 at SAIL, computing supports students as they transition from primary school to secondary education by building on their existing knowledge and introducing new concepts in a clear, accessible way. The curriculum recognises the skills many students bring from primary computing lessons, such as basic use of digital devices, simple programming, and understanding of online safety and aims to deepen their confidence and independence. Students begin by exploring 'Clear messaging in digital media', where they learn how to communicate effectively using images, text, and sound; building on their early experiences with multimedia tools at primary school. This prepares them for more complex media projects in later years. Next, they investigate 'Networks from semaphores to the internet', which introduces the idea that computers and people can share information in many ways. This broadens their understanding of communication beyond everyday devices, linking early concepts of connectivity from primary school to the digital world. In 'Programming essentials in Scratch', students develop their coding skills in a supportive environment. Scratch often forms a part of primary education, so Year 7 focuses on consolidating this knowledge and introducing logical thinking and problem-solving strategies that will underpin future programming challenges. Through 'Modelling data using spreadsheets', students learn how to organise and analyse information. This builds on simple data handling they may have done before and prepares them for more advanced data science and database topics in later years. Finally, in 'Using media – gaining support for a cause', students combine their computing and communication skills to create persuasive digital content. This encourages creativity, social awareness, and real-world application of their learning. Overall, Year 7 computing at SAIL provides a strong foundation by connecting prior learning from primary school to new skills and knowledge. It establishes a stepping stone for increasingly complex computing topics throughout Years 8 to 11, supporting students' personal growth, digital confidence, and preparation for future study and life in a digital world.					
Year 7	Clear messaging in digital media	Networks from semaphores to the internet	Programming essentials in Scratch – part I	Modelling data using spreadsheets	Programming essentials in Scratch – part II	Using media – Gaining support for a cause
	• Choose search terms relating to a particular issue – online safety • Use tools to copy an image into another application • Identify key features of a good poster focusing on online safety • Plan a poster to clearly convey a message • Choose and download a suitable image following the theme • Create a poster using a desktop publishing application • Modify a logo using a graphic editing program • Choose how to combine text and graphics in a slide • Use digital tools to provide feedback on design choices	• Define what a computer network is and explain how data is transmitted between computers across networks • Define 'protocol' and provide examples of non-networking protocols • List examples of the hardware necessary for connecting devices to networks • Compare wired to wireless connections and list examples of specific technologies currently used to implement such connections • Define 'bandwidth', using the appropriate units for measuring the rate at which data is transmitted, and discuss familiar examples where bandwidth is important • Define what the internet is • Explain how data travels between computers across the internet	• Compare how humans and computers understand instructions (understand and carry out) • Define a sequence as instructions performed in order, with each executed in turn • Predict the outcome of a simple sequence • Modify a sequence • Define a variable as a name that refers to data being stored by the computer • Recognise that computers follow the control flow of input/process/output • Predict the outcome of a simple sequence that includes variables • Trace the values of variables within a sequence • Make a sequence that includes a variable • Define a condition as an expression that will be evaluated as either true or false • Identify that selection uses conditions to	• Identify columns, rows, cells, and cell references in spreadsheet software • Use formatting techniques in a spreadsheet • Use basic formulas with cell references to perform calculations in a spreadsheet (+, -, *, /) • Use the autofill tool to replicate cell data • Explain the difference between data and information • Explain the difference between primary and secondary sources of data • Collect data • Analyse data • Create appropriate charts in a spreadsheet • Use the functions SUM, COUNTA, MAX, and MIN in a spreadsheet • Analyse data • Use a spreadsheet to sort and filter data • Use the functions AVERAGE, COUNTIF, and IF in a spreadsheet	• Define a subroutine as a group of instructions that will run when called by the main program or other subroutines • Define decomposition as breaking a problem down into smaller, more manageable subproblems • Identify how subroutines can be used for decomposition • Identify where condition-controlled iteration can be used in a program • Implement condition-controlled iteration in a program • Evaluate which type of iteration is required in a program • Define a list as a collection of related elements that are referred to by a single name • Describe the need for lists • Identify when lists can be used in a program • Use a list	• Select the most appropriate software to use to complete a task • Identify the key features of a word processor • Apply the key features of a word processor to format a document • Evaluate formatting techniques to understand why we format documents • Select appropriate images for a given context • Apply appropriate formatting techniques • Demonstrate an understanding of licensing issues involving online content by applying appropriate Creative Commons licences • Demonstrate the ability to credit the original source of an image • Critique digital content for credibility • Apply techniques to identify whether or not a source is credible

	• Plan a consistent layout for a set of slides • Modify a logo so that it fits in with the planned slide styles • Create a styled set of slides based on a plan • Search for suitable text for slides • Search for and add a suitable image • Evaluate content against a rubric • Plan how to deliver a presentation • Explain your work to others through a presentation • Evaluate your work against a rubric	• Describe key words such as 'protocols', 'packets', and 'addressing' • Explain the difference between the internet, its services, and the World Wide Web • Describe how services are provided over the internet • List some of these services and the context in which they are used • Explain the term 'connectivity' as the capacity for connected devices ('Internet of Things') to collect and share information about me with or without my knowledge (including microphones, cameras, and geolocation) • Describe how internet-connected devices can affect me • Describe components (servers, browsers, pages, HTTP and HTTPS protocols, etc.) and how they work together	control the flow of a sequence • Identify where selection statements can be used in a program • Modify a program to include selection • Create conditions that use comparison operators (>,<,=) • Create conditions that use logic operators (and/or/not) • Identify where selection statements can be used in a program that include comparison and logical operators • Define iteration as a group of instructions that are repeatedly executed • Describe the need for iteration • Identify where count-controlled iteration can be used in a program • Implement count-controlled iteration in a program • Detect and correct errors in a program (debugging) • Independently design and apply programming constructs to solve a problem (subroutine, selection, count-controlled iteration, operators, and variables)	• Use conditional formatting in a spreadsheet • Apply all of the spreadsheet skills covered in this unit	• Decompose a larger problem into smaller subproblems • Apply appropriate constructs to solve a problem	• Apply referencing techniques and recognise the concept of plagiarism • Evaluate online sources for use in own work • Construct a blog using appropriate software • Create content for a blog based on credible sources • Apply referencing techniques that credit authors appropriately • Design the layout of the content to make it suitable for the audience • Construct a blog using appropriate software • Create content for a blog based on credible sources • Apply referencing techniques that credit authors appropriately • Design the layout of the content to make it suitable for the audience
Year Group 8	Term One	Term Two	Term Three	Term Four	Term Five	Term Six
Intent of Study	In Year 8 at SAIL, computing continues to develop students' digital understanding by deepening their technical knowledge and creative skills. Building on the foundations from Year 7, the curriculum encourages more independence, structured thinking and exploration of how computer systems work and are used in everyday life. Students begin with 'Media – Vector graphics'; where they learn to create and edit images using shapes, lines and layers, strengthening their visual design and attention to detail while linking back to their Year 7 work on digital messaging. This is followed by Layers of computing systems, introducing students to the different components that make up computing devices from hardware to software helping them understand the machines they use every day. In 'Developing for the Web', students build on their digital communication skills by learning how websites are made and how to structure content for a real audience. They also explore how data and ideas are stored and shared through 'Representations from clay to silicon', connecting physical and digital worlds by understanding how text, images and numbers are converted into binary. As the year progresses, students explore mobile app development, using familiar ideas from Scratch and digital media to design simple, functional applications, encouraging creative problem-solving. Finally, students are introduced to Python programming, where they begin to develop text-based coding skills in a structured, accessible way, building on the logic and sequencing they learned in Scratch.					

	Through real-world examples, supported practice, and purposeful projects, students gain confidence in using technology to express themselves, solve problems and understand the digital systems around them. This prepares them for more complex thinking in Year 9 and supports their independence as digital citizens.					
Year 8	Media - Vector graphics	Layers of computing systems	Developing for the Web	Representations – from clay to silicon	Mobile app development	Introduction to Python programming
	• Use tools to draw and modify shapes • Change the position and rotation shapes • Explain how z-order determines what is visible • Use tools to align and distribute objects to create uniformity • Explain how grouping can be used to work with several objects at once • Combine two shapes using union, intersection, and difference • Explain that vector graphics are made up of paths • Create and modify straight and curved paths • Change shapes to paths and edit them • Choose a project and plan a design • Combine tools and techniques to create a vector image • Evaluate the project against its given purpose • Explain how markup defines what a vector graphic looks like • Change an object by modifying its markup • Plan improvements and implement them to develop a project • Explain key differences between vector and bitmap images • Outline which image type best suits which uses • Evaluate their image against a rubric	• 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 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	• Describe what HTML is • Use HTML to structure static web pages • Modify HTML tags using inline styling to improve the appearance of web pages • Display images within a web page • Apply HTML tags to construct a web page structure from a provided design • Describe what CSS is • Use CSS to style static web pages • Assess the benefits of using CSS to style pages instead of in-line formatting • Describe what a search engine is • Explain how search engines 'crawl' through the World Wide Web and how they select and rank results • Analyse how search engines select and rank results when searches are made • Use search technologies effectively • Discuss the impact of search technologies and the issues that arise by the way they function and the way they are used • Create hyperlinks to allow users to navigate between multiple web pages • Implement navigation to complete a functioning website • Complete summative assessment	• List examples of representations • Recall that representations are used to store, communicate, and process information • Provide examples of how different representations are 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 the different ways that binary digits are physically represented in digital devices • Apply all of the skills covered in this unit	• Understand the objectives and requirements of the Health and Fitness Tracker app. • Recognise the importance of user-centred design principles. • Be able to brainstorm and design your own version of the app, considering key features and user interface design. • Understand key online safety concepts and their importance in app development. • Identify specific online safety measures applicable to mobile apps. • Identify online safety best practices and potential pitfalls. • Understand the hardware components of mobile phones and their functions in app development. • Recognize how hardware capabilities influence app functionality and user experience. • Consider the implications of hardware limitations on the design and performance of mobile apps. • Learn strategies for securely handling user data in app development. • Understand the importance of user privacy and consent in app design and development. • Explore the ethical considerations related to online safety in app development and identify ways to address them. • Understand the app development process and tools used in mobile app development.	• 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 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

		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			● Be able to start building the core features of your Health and Fitness Tracker app. ● Collaborate with peers to share ideas, solve problems, and provide feedback on app development progress. ● Continue developing your Health and Fitness Tracker apps. ● Collaborate with peers to troubleshoot problems, test app functionality, and provide feedback on each other's app development progress. ● Reflect on the app development experience and identify areas for improvement in future projects.	● Use Boolean variables as flags
Year Group 9	Term One	Term Two	Term Three	Term Four	Term Five	Term Six
Intent of Study	In Year 9 at SAIL, computing aims to consolidate and extend students' knowledge, giving them more opportunities to apply what they've learned in creative and meaningful ways. Building on the communication, problem-solving and technical skills developed in Years 7 and 8, the Year 9 curriculum encourages independence, deeper thinking and real-world connections. Students begin with Python programming with sequences of data, extending their skills from Year 8's introduction to Python by working with variables, lists and loops. This helps them develop structured thinking and a deeper understanding of how to use code to solve problems. They then explore 'Media – Animations', using design and storytelling to bring digital content to life. This builds on previous work with vector graphics and digital messaging, giving students further experience in planning, creating and reviewing multimedia work. In 'Data science', students are introduced to how data is collected, analysed and used to make decisions, drawing on their earlier skills in spreadsheets and web content. They learn to question data, spot patterns, and consider how information can be used to inform choices in the real world. 'Representations – going audiovisual' deepens their understanding of binary and data encoding by looking at how sound and video are stored and processed. This links to previous representation units and helps students understand how the media they use every day is powered by code and data. Next, students learn about cybersecurity, where they explore how to protect data and systems, understand online threats, and develop safe behaviours. This unit strengthens their awareness of digital responsibility, following on from previous learning in online safety and digital systems. Finally, students apply their coding knowledge in a hands-on project with physical computing, programming devices to interact with the real world. This practical experience builds confidence and shows students how digital skills can be used creatively and purposefully beyond the classroom. Throughout the year, students continue to develop their independence, resilience and technical confidence, preparing them for the next stage of their learning journey and for life in a digital world.					
Year 9	Python programming with sequences of data	Media – Animations	Data science	Representations – going audiovisual	Introduction of cybersecurity	Applying programming skills with physical computing

	• Write programs that display messages, receive keyboard input, and use simple arithmetic expressions in assignment statements • Use selection (if-elif-else statements) to control the flow of program execution • Locate and correct common syntax errors • Create lists and access individual list items • Perform common operations on lists or individual items • Use iteration (while statements) to control the flow of program execution • Perform common operations on lists or individual items • Perform common operations on strings or individual characters • Use iteration (for statements) to iterate over list items • Perform common operations on lists or strings • Use iteration (for loops) to iterate over lists and strings • Use variables to keep track of counts and sums • Combine key programming language features to develop solutions to meaningful problems	• Add, delete, and move objects • Scale and rotate objects • Use a material to add colour to objects • Add, move, and delete keyframes to make basic animations • Play, pause, and move through the animation using the timeline • Create useful names for objects • Join multiple objects together using parenting • Use edit mode and extrude • Use loop cut and face editing • Apply different colours to different parts of the same model • Use proportional editing • Use the knife tool • Use subdivision • Add and edit set lighting • Set up the camera • Compare different render modes • Create a 3–10 second animation • Render out the animation	• 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 • 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 steps of the investigative cycle • 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	• Describe how digital images are composed of individual elements • Recall that the colour of each picture element is represented using a sequence of binary digits • Define key terms such as 'pixels', 'resolution', and 'colour depth' • Describe how an image can be represented as a sequence of bits • Describe how colour can be represented as a mixture of red, green, and blue, with a sequence of bits representing each colour's intensity • Compute the representation size of a digital image, by multiplying resolution (number of pixels) with colour depth (number of bits used to represent the colour of individual pixels) • Describe the trade-off between representation size and perceived quality for digital images • Perform basic image editing tasks using appropriate software and combine them in order to solve more complex problems requiring image manipulation • Explain how the manipulation of digital images amounts to arithmetic operations on their digital representation • Describe and assess the creative benefits and ethical drawbacks of digital manipulation (Education for a Connected World) • Recall that sound is a wave • Explain the function of microphones and speakers as components that capture and generate sound	• Explain the difference between data and information • Critique online services in relation to data privacy • Identify what happens to data entered online • Explain the need for the Data Protection Act • Recognise how human errors pose security risks to data • Implement strategies to minimise the risk of data being compromised through human error • Define hacking in the context of cyber security • Explain how a DDoS attack can impact users of online services • Identify strategies to reduce the chance of a brute force attack being successful • Explain the need for the Computer Misuse Act • List the common malware threats • Examine how different types of malware causes problems for computer systems • Question how malicious bots can have an impact on societal issues • Compare security threats against probability and the potential impact to organisations • Explain how networks can be protected from common security threats • Identify the most effective methods to prevent cyberattacks	• Describe what the micro:bit is • List the micro:bit's input and output devices • Use a development environment to write, execute, and debug a Python program for the micro:bit • Write programs that use the micro:bit's built-in input and output devices • Write programs that use GPIO pins to generate output and receive input • Write programs that communicate with other devices by sending and receiving messages wirelessly • Design a physical computing artifact purposefully, keeping in mind the problem at hand, the needs of the audience involved, and the available resources • Decompose the functionality of a physical computing system into simpler features • Implement a physical computing project, while following, revising, and refining the project plan • Implement a physical computing project, while following, revising, and refining the project plan

				● Define key terms such as 'sample', 'sampling frequency/rate', 'sample size' ● Describe how sounds are represented as sequences of bits ● Calculate representation size for a given digital sound, given its attributes ● Explain how attributes such as sampling frequency and sample size affect characteristics such as representation size and perceived quality, and the trade-offs involved ● Perform basic sound editing tasks using appropriate software and combine them in order to solve more complex problems requiring sound manipulation ● Recall that bitmap images and pulse code sound are not the only binary representations of images and sound available ● Define 'compression', and describe why it is necessary		
Year Group 10	Term One	Term Two	Term Three	Term Four	Term Five	Term Six
Intent of Study	In Year 10 at SAIL, computing continues to strengthen students' digital skills and deepen their understanding of how computers work, how we interact with them, and how they affect the world around us. Building on the creative, technical, and logical thinking developed from Years 7 to 9, this year supports students in becoming more independent, reflective, and capable users of technology. Students begin by developing their programming skills, building on previous experiences in Scratch (Year 7), Python (Years 8 and 9), and physical computing. They revisit sequencing, then progress into selection and iteration, allowing them to write more efficient, structured code. Problem-solving, debugging, and logical thinking are key skills developed throughout this unit. In 'Computer systems', students explore how hardware and software work together to carry out tasks. This draws on their earlier learning from Year 8 about layers of computing systems and gives them a more complete picture of how digital devices function from the inside out. Next, students develop their understanding of algorithms, learning how to plan, represent and evaluate step-by-step solutions to problems. This builds on the logical structuring of code introduced in earlier years and sets the groundwork for more advanced algorithm work in Year 11. In 'Data representations', students revisit how text, numbers, images and sound are stored in binary. Building on their Year 8 and 9 knowledge of representation, this unit helps them appreciate how computers handle data at the most fundamental level. The impacts of technology unit encourages students to think critically about how digital tools and systems affect individuals, society and the environment. Drawing from earlier online safety, media, and cybersecurity topics, this encourages thoughtful discussion and responsible digital citizenship. Finally, in 'Computer networks', students explore how devices connect and share data. They build on their Year 7 introduction to networks and Year 8's deeper look at how systems communicate, now learning how the internet, protocols and security play a role in everyday digital communication.					

	By the end of Year 10, students are more confident in applying their computing knowledge to both practical tasks and broader thinking. They are well-prepared to handle more complex coding, technical understanding, and ethical discussions in Year 11.					
Year 10	Programming	Computer systems	Algorithms	Data representations	Impacts of technology	Computer networks
	• Compare how humans and computers interpret instructions • Explain the differences between high- and low-level programming languages • Describe why translators are necessary • List the differences, benefits, and drawbacks of using a compiler, or an interpreter • Describe the tools an IDE provides (editors, error diagnostics, runtime environment, translators) • Use subroutines in programs • Define a sequence as instructions performed in order, with each executed in turn • Predict the outcome of a sequence and modify it • Interpret error messages and define error types and identify them in programs (logic, syntax) • Use meaningful identifiers • Determine the need for variables • Distinguish between declaration, initialisation, and assignment of variables • Demonstrate appropriate use of naming conventions • Output data (e.g. print (my_var)) • Obtain input from the keyboard in a program • Differentiate between the data types: integer, real, Boolean, character, string • Cast variables by calling a function that will return a new value of the desired data type	• Compare embedded and general purpose computer systems • Describe the role of system software as part of a computer system • Explore the role of the operating system and utility software • Describe the basic components of the CPU • Describe the roles and purpose of each component of the CPU in computation • Explain how the fetch-decode-execute cycle works by describing what happens at each stage • Describe the role of each part of the CPU as part of the fetch-decode-execute cycle • Describe the characteristics of RAM and ROM • Explain the role of main memory as part of a computer system • Define cache memory • Describe the role of cache in a computer system • Explain why a computer system needs secondary storage • State the different types of secondary storage and describe their functional characteristics • State how solid-state memory works and describe its characteristics • Explain how optical and magnetic memory stores data in the form of binary • Describe how data is read from and written to	• Define the terms decomposition, abstraction, and algorithmic thinking • Recognise scenarios where each of these computational thinking techniques is applied • Apply decomposition, abstraction, and algorithmic thinking to help solve a problem • Describe the difference between algorithms and computer programs • Identify algorithms that are defined as written descriptions, flow charts, and code • Analyse and create flow charts using the flow chart symbols • Use a trace table to walk through code that contains a while loop, a for loop, and a list of items • Use a trace table to detect and correct errors in a program • Identify why computers often need to search data • Describe how linear search is used for finding the position of an item in a list of items • Perform a linear search to find the position of an item in a list • Describe how binary search is used for finding the position of an item in a list of items • Perform a binary search to find the position of an item in a list • Identify scenarios when a binary search can and cannot be carried out • Compare the features of linear and binary search and decide which is	• Give examples of the use of representation • Explain that computers use binary to represent all data and instructions • Explain how binary relates to two-state electrical signals • Explain the difference between base-2 and base-10 numbers • Convert between binary and decimal numbers • Count in binary • Perform addition in binary on two binary numbers • Perform addition in binary on three binary numbers • Perform subtraction in binary • Perform binary shifts • Describe situations where binary shifts can be used • Explain how overflow errors can occur • Explain how underflow occurs • Compare signed and unsigned integers • Use sign and magnitude to represent positive and negative integers • Use two's complement to represent positive and negative integers • Explain why and where hexadecimal notation is used • Explain how numbers are represented using hexadecimal • Convert decimal numbers to and from hexadecimal numbers	• Apply the terms 'privacy', 'legal', 'ethical', 'environmental', and 'cultural' • Explain data legislation, including an organisation's obligation to protect and supply data • Explain the term 'stakeholder' • Explain the right to be forgotten • Distinguish the differences between legitimate creative uses and clear infringement of material subject to copyright • Explain the Freedom of Information Act • Define 'computer misuse' and the associated offences • Identify situations that would be classified as an offence under the Act • Define 'downtime' and explain the associated impact on an organisation • Explain what is meant by the 'digital divide' and measures to mitigate its effect • Identify positive and negative aspects of the use of mobile technology • Identify the implications of having personal data online • Explain the social and environmental impacts of social media • Explain the positive and negative effects of online content • Explain the environmental effects of the use of technology • Explain the ethical issues surrounding the use of AI in society	• Define a computer network • Discuss the advantages and disadvantages of computer networks • Describe the role of a computer in a peer-to-peer network • Describe the role of a computer in a client–server network • Describe the purpose of a PAN, LAN, and a WAN • Describe the tasks performed by the network hardware: wireless access point, router, switch, hub, NIC, and bridge • Define a MAC address • Draw and describe a star, bus, mesh, and ring topology • Describe the advantages and disadvantages of the star, bus, mesh, and ring topologies • Select an appropriate topology for a given scenario • Define a wired and a wireless network • Define transmission media • Describe the attributes of fibre optic and copper cables used in wired networks • Describe Bluetooth as a mode of connection • Discuss the advantages and disadvantages of wireless networks compared to wired networks • Describe the factors that affect network performance (bandwidth,

	• Define runtime errors in programs • Define validation checks • Identify flowchart symbols and describe how to use them (start, end, input, output, subroutine) • Translate a flowchart into a program sequence • Design a flowchart for a program • Be able to locate information using the language documentation • Import modules into your code • Demonstrate how to generate random numbers • Evaluate arithmetic expressions using rules of operator precedence (BIDMAS) • Write and use expressions that use arithmetic operators (add, subtract, multiply, real division, integer division, MOD, to the power) • Assign expressions to variables • Define a condition as an expression that can be evaluated to either True or False • Identify flowchart symbols and describe how to use them (decision) • Identify that selection uses conditions to control the flow of execution • Walkthrough code that includes selection (if, elif, else) • Use selection statements in a program • Identify when selection statements should be used in programs • Write and use expressions that use comparison operators (equal to, not equal to, less than, greater than, less than	optical and magnetic memory • Apply knowledge of storage devices to compare the three mediums of storage • Apply the knowledge of storage devices to recommend an appropriate device • Describe the limitations of secondary storage • Explain the definition of 'cloud storage' and describe the characteristics of cloud storage • Explore the factors that impact a CPU's performance • Select components to create a computer system • Evaluate a computer's suitability for a given task • Revise computer systems content covered so far • Design and implement a software project • Discover the logic gates AND, NOT, and OR, including their symbols and truth tables • Learn how logic gates are used in carrying out computation • Design a logical circuit, combining logic gates to solve a problem • Construct truth tables for a three-input logic circuit • Write a Boolean expression to describe a logical circuit • Describe how combinations of logic gates can perform mathematical operations • Determine that assembly language has a 1:1 relationship with machine code	most suitable in a given context • Interpret the code for linear search and binary search • Trace code for both searching algorithms with input data • Identify why computers often need to sort data • Traverse a list of items, swapping the items that are out of order • Perform a bubble sort to order a list containing sample data • Insert an item into an ordered list of items • Describe how insertion sort is used for ordering a list of items • Perform an insertion sort to order a list containing sample data • Interpret the code for bubble sort and insertion sort • Trace code for both sorting algorithms with input data • Identify factors that could influence the efficiency of a bubble sort implementation • Merge two ordered lists of items into a new ordered list • Describe how merge sort is used for ordering a list of items • Perform a merge sort to order a list containing sample data • Interpret algorithms and suggest improvements • Analyse and fix errors in a flow chart • Perform searching and sorting algorithms on samples of data • Develop a linear search function in Python	• Determine the maximum number of states that can be represented by a binary pattern of a given length • Explain how ASCII is used to represent characters, and its limitations • Explain what a character set is • Describe how character codes are commonly grouped and run in sequence within encoding tables • Explain the need for Unicode • State that Unicode uses the same codes as ASCII up to 127 • Calculate the number of bytes needed to store a piece of text • Describe what a pixel is and how pixels relate to bitmap images • Describe colour depth and resolution • Define 'metadata' • Give examples of metadata applied to a bitmap image • Calculate the file size of bitmaps • Describe how the number of pixels and colour depth can affect the file size of a bitmap image, using examples • Explain why analogue sound data needs to be converted into binary digits • Describe the concepts of sampling, sample rate, and sample resolution • Calculate file size requirements of sound files • Describe the effect of sample rate, duration, and sample resolution on the playback quality and the size of a sound file	• Explain the ethical impact of using algorithms to make decisions • Demonstrate knowledge of the five impacts of technology	range, latency, number of devices) • Determine how network speeds are measured and construct expressions involving file size, transmission rate, and time • Determine methods of routing traffic on a network and calculation of routing costs • Describe the internet as a network of computer networks • Describe the function of an IP address • Describe a DNS and its role in the conversion of a URL to an IP address • Describe the role and function of a web browser • Describe how servers are used for hosting services across the internet • Describe the role of web servers and clients • Describe how the cloud provides services for software and storage • List the advantages and disadvantages of the cloud • Determine the need for standards in network communications • Define the term network protocol • Define the purpose and common use of the network protocols: Ethernet, WiFi, HTTP, HTTPS. FTP, POP, SMTP, and IMAP • Describe the four layers of the TCP/IP model (Link, Internet, Transport, Application) • Define the purpose and common use of the network protocols: TCP, IP, UDP • Describe that the HTTP, HTTPS, SMTP, IMAP, and FTP protocols operate at the application layer

	or equal to, greater than or equal to) ● Describe how Boolean/logical operators can be used in expressions ● Walk through code that use conditions with Boolean/logical operators (AND, OR) ● Write and use expressions that use Boolean/logical operators (AND, OR) ● Define nested selection ● Walk through code that uses nested selection ● Modify a program that uses nested selection ● Define iteration as a group of instructions that are repeatedly executed ● Modify a program to incorporate a while loop ● Use a trace table to walkthrough code that uses a while loop ● Use a trace table to detect and correct errors in programs ● Define a for loop ● Walk through code that uses a for loop ● Modify a program that uses a for loop ● Compare a while loop and a for loop ● Determine the need for validation checks ● Use iteration to perform validation checks ● Describe the purpose of pseudocode ● Translate pseudocode into a program ● Design and build a program using pseudocode ● Describe a subroutine ● Describe the purpose of parameters in subroutines ● Use procedures that accept arguments through parameters	● Explain the basic commands in the LMC's assembly code: INP, OUT, STA, LDA, ADD, SUB, and BRP ● Design and write your own program in assembly language		● Give examples of metadata applied to sound files ● Define the terms 'bit', 'nibble', 'kilobyte', 'megabyte', 'gigabyte', 'terabyte', and 'petabyte' ● Compare 'kibibyte', 'mebibyte', 'gibibyte', and 'tebibyte' to 'megabyte', 'gigabyte', and 'terabyte'' ● Convert between units of measurement ● Explain what data compression is ● Explain why data may be compressed, and that there are different ways to compress data ● Define 'lossy compression' and 'lossless compression' ● Explain how data can be compressed using run length encoding (RLE) ● Represent data in RLE frequency/data pairs ● Calculate compression ratios ● Explain how data can be compressed using Huffman coding ● Interpret a Huffman tree ● Calculate the number of bits required to store a piece of data compressed using Huffman coding ● Summarise learning through a final assessment		● Describe that the TCP and UDP protocols operate at the transport layer ● Describe that the IP protocol operates at the internet layer ● Describe the typical contents of a TCP/IP packet and packet switching ● Describe the purpose of each layer in the seven-layer Open Systems Interconnection model (OSI model) ● Describe the use of contemporary networking protocols in the seven-layer OSI model ● Determine the need for and importance of network security ● Identify different forms of attacks on networks (social engineering, malicious software) ● Explain network security methods ● Recall knowledge of networks through a final, summative assessment

	● Describe how subroutines are used for decomposition ● List the advantages of subroutines ● Explain the difference between a function and a procedure ● Use trace tables to investigate functions ● Use functions to return values in programs ● Describe scope of variables ● Describe how parameters can reduce the need for global variables ● Identify when to use global variables ● Describe a constant ● Use a truth table ● Describe the function of an XOR operator ● Design and create a function for an XOR operator ● Describe the structured approach to programming ● Explain the advantages of the structured approach ● Use the structured approach in programming ● Describe iterative testing ● Describe the types of testing (erroneous, boundary, normal) ● Design and create a program ● Define the term GUI ● Import third-party libraries ● Use guizero to create an event-driven program that uses a GUI ● Describe the function of string operators ● Use string-handling techniques ● Use for loops with string operations ● Use a substring in a program					

	● Use the in operator to check for a substring ● Use chr() and ord() to perform ASCII conversions ● Create a program that uses string-handling techniques ● Define a data structure ● Define a list and an array ● Describe the differences between lists and arrays ● Use a list in a program ● Append to a list ● Traverse a list of elements ● Use list methods ● Create a function that returns a list ● Import custom-built functions ● Use lists to display output on a physical computing device ● Use randomisation to append items to a list ● Define a 2D array and a list ● Use a 2D list in a program ● Use a 2D list as part of a programming challenge ● Describe the record data structure ● Use a dictionary to represent a record in a program ● Use a dictionary with a list to represent records in a database ● Describe the dictionary data structure ● Use a dictionary to produce key-value pairs ● Determine the purpose of external data files ● Read data from an external text file ● Write to text files ● Append to text files					

	• Describe a CSV file • Read from a CSV file • Use the split() method • Select data from a collection of values • Write data from a 1D list to a CSV file • Write data from a 2D list to a CSV file • Determine the good habits of a programmer • Explore alternative approaches to programming solutions • Append to a CSV file • Write success criteria for a challenging project • Design the program for a challenging project using flowchart or pseudocode • Create the solution for the battle boats program • Perform final testing of the solution to a challenging problem • Evaluate a challenging program					
Year Group 11	Term One	Term Two	Term Three	Term Four	Term Five	Term Six
Intent of Study	In Year 11 at SAIL, computing brings together all the key skills and knowledge developed from Years 7 to 10 and introduces more advanced concepts that prepare students for future study, employment, or confident participation in a digital world. The curriculum focuses on applying practical computing skills, deepening technical understanding, and developing secure, responsible digital behaviours. The year begins with a focus on cyber security, where students explore how data and systems can be protected from threats. Building on their knowledge of computer systems, networks, and the impacts of technology covered in Year 10, students examine real-world risks such as hacking, phishing, and malware, and learn how individuals and organisations can stay safe. Next, students learn to manage and organise information using databases and SQL. This builds on their previous experience with spreadsheets (Year 7) and data science (Year 9), now introducing them to structured data storage, queries, and commands. They gain confidence in handling real-world data and understanding how information systems underpin everyday technology. In HTML and web development, students return to web skills introduced in Year 8 ("Developing for the web"), applying them in a more structured and purposeful way. They learn how to build and style simple websites, giving them an opportunity to express ideas, structure content, and consider user needs. Later in the year, students are introduced to object-oriented programming (OOP) — a new way of thinking about code. Building on their procedural programming work in Python from Years 9 and 10, students learn how to create and use classes and objects. This supports the development of organised, reusable code and introduces concepts used in professional software development. In the summer term, students take part in two practical, reflective projects to consolidate their skills and celebrate their progress: Designing a digital portfolio gives students the chance to showcase their work and learning from previous years. Using HTML and digital media skills, they create a simple website that presents their achievements, project work, and reflections in an accessible and personal way.					

	Creative computing project allows students to work independently or in small groups to design and create a digital solution to a problem or theme they care about. They might build a quiz in Python, develop a small database, create a campaign website, or combine multimedia tools to raise awareness for a cause. This project encourages creativity, confidence, and independent thinking, drawing on everything they've learned across KS3 and KS4. Throughout Year 11, students apply problem-solving, logic and creativity while strengthening their independence and digital responsibility. By the end of the year, they have developed a well-rounded and practical understanding of computing. They are equipped with transferable digital skills, critical awareness of technology's role in society, and the confidence to take their learning further, whether in academic, vocational or everyday settings.					
Year 11	Cyber security	Databases and SQL	HTML	Object-oriented programming	Designing a digital portfolio	Creative computing project
	• Describe the impact of cybercrime on businesses and individuals • Analyse an attack on a company and identify what motivated the hackers • Define the terms cybersecurity and network security, explain their importance, and distinguish between the two concepts • Describe the features of a network that make it vulnerable to attack • Identify and describe non-automated forms of cyberattacks, and learn how humans can be the weak links in an organisation • Demonstrate knowledge of social engineering through role playing activities and case studies • Describe automated forms of cyberattacks • Analyse a real cyberattack and identify the network or software weaknesses that enabled it to happen • Identify how software can be used to protect from cyberattacks • Describe how organisations use software to protect against cyberattacks • Explain the need for, and the importance of, network security • Explain a number of methods of achieving network security • Describe different methods of identifying	• Describe a database • Define database key terms (table, record, field, primary key, foreign key) • Describe a flat file database • Describe a relational database • Describe the function of SQL • Use SQL to retrieve data from a table in a relational database • Use SQL to retrieve data from more than one table in a relational database • Describe the function of different data types • Use SQL to insert, update, and delete data in a relational database • Interrogate and update an existing database • Interrogate and update an existing database	• Describe the purpose of HTML and tags when designing a website • Create a simple webpage using basic tags • Describe what is meant by the term 'accessibility' • Extend an HTML page to include: ○ Images <img> ○ Hyperlinks <a href> • Identify the common features of existing websites and the basics of what makes good web design • Design and create pages for a mini website • Create hyperlinks between pages stored locally within a folder • Insert images stored locally within a folder • Describe the purpose of CSS and why it is needed in addition to HTML • Experiment with CSS by changing the style of the tags learnt so far in this unit • Describe the purpose of DIV tags • Apply knowledge of CSS to DIVs within webpages using classes • Explain how to plan a website by developing a house style and sketched wireframe • Describe the box model in CSS • Apply skills to position items within a page • Construct a three-page website to showcase the skills learnt throughout this unit of study	• Describe the role of conventions in programming • Recall that there are different paradigms for programming • Define object oriented programming • Identify a class and object as a part of a program • Describe the relationship between a class and an object • Model a real world problem using object oriented programming conventions. • Define attributes and methods as a part of a class • Use a constructor to create objects • Use a method and access an attribute on an object • Create a class • Define the use of a self parameter in object-oriented Python • Create a method on a class • Access and modify attributes using getters and setters • Define the principle of inheritance • Define the terms superclass and subclass • Select appropriate uses of inheritance • Create a subclass in a program • Explore a program written using OOP • Explain the key concepts of OOP	• Identify key components of an effective portfolio (e.g. projects, skills, about me) • Review previous computing work (e.g. Python, media, HTML, databases) to select what to include • Organise past work into categories (e.g. programming, media, data handling) • Reflect on what you've enjoyed or learned in computing across KS3 and KS4 • Recap basic HTML structure: <!DOCTYPE html>, <html>, <head>, <body> • Create a simple homepage using headings (<h1> to <h3>), paragraphs, and lists • Use <img> tags to add images with appropriate file paths and alt text • Insert links using <a href> to navigate between pages or link to work • Add simple styling using inline or internal CSS (colours, fonts, spacing) • Understand folder structure for a website (e.g. index.html, images, styles folder)	• Understand what a creative computing project is and explore examples of digital solutions • Identify a topic, theme or problem that matters to you (e.g. wellbeing, hobbies, community) • Choose an appropriate digital format for your project (e.g. Python app, website, database, animation, quiz) • Plan your project using a visual aid (e.g. mind map, storyboard, flowchart or checklist) • List the digital skills you will use and revisit past projects for inspiration • Break the project into manageable steps with support (e.g. design, create, test, present) • Create or collect resources needed for your project (e.g. images, code, sound, text) • Begin designing or coding your project using your chosen platform or tool • Apply HTML, Python, SQL, media, or other digital skills learned in previous years • Test your work regularly and fix any

	cybersecurity vulnerabilities, such as: ○ Penetration testing ○ Ethical hacking ○ Network forensics ○ Commercial analysis tools ○ Review of network and user policies ● Evaluate the potential for cybersecurity careers ● Apply your knowledge of cybersecurity to GCSE-style questions		● Self/peer evaluate the produced webpage using a rubric ● Extend/finish the assessed website ● Showcase the assessed website ● Demonstrate how much has been learnt by taking an end of unit test		• Write an "About Me" section with support for sentence starters and scaffolds • Describe a favourite project using accessible sentence frames • Insert screenshots of work and label them clearly • Add embedded media (e.g. animation video, Scratch link, Python code snippet) • Create a "Skills" section highlighting what you can now do in computing • Use bullet points to list areas of improvement and learning goals • Plan the layout of your portfolio using a visual storyboard or template • Choose appropriate colours and fonts for readability • Ensure text is easy to read and images are appropriately sized • Use headings to structure content clearly • Add alt text for accessibility and explain why this matters • Create a navigation bar or menu with clear links to each section • Use consistent page layouts for ease of use • Test all links to make sure they work correctly • Ensure filenames and folders are clearly labelled and well-organised	errors or bugs with support • Use simple evaluation tools to check that your project is clear, working, and purposeful • Write simple explanations of what your project does and why it is useful or meaningful • Share your project with a peer or adult and collect supportive feedback • Make improvements to your project based on feedback and your own reflections • Keep track of your progress using a log, checklist or journal • Add simple finishing touches (e.g. design improvements, extra features, help instructions) • Prepare to present your project clearly, using sentence frames or cue cards if needed • Present your final product to a small group or staff member, explaining how it was made • Reflect on what you learned during the project process • Celebrate your progress in computing and identify your next steps or future interests

					• Create a site map (diagram or list) showing the structure of your portfolio • View a classmate's portfolio and give simple structured feedback (e.g. "I liked...", "Next time you could...") • Use a checklist to self-assess: Have I included all required sections? Do all links work? • Update and improve your work based on feedback • Reflect on what you are most proud of in your portfolio • Write a short evaluation of your final product: What went well? What would you do differently? • Prepare to present your portfolio to a small group or trusted adult • Practise explaining the contents of your website using key vocabulary • Export, upload, or save the finished website for presentation	