COMPUTING Year 9 Curriculum End Points and Key Vocabulary

	Autumn Term 1	Autumn Term 2	Spring Term 1	Spring Term 2	Summer Term 1	Summer Term 2
Ethos Links	**STEM** – problem solving through creating programs that use sequence, variables, selection, and iteration, and through analysing and interpreting large data sets to identify patterns, trends, correlations, and outliers. Students apply knowledge of correct data types, comparison and logical operators, and understand when to use For and While loops effectively. They also follow the investigative cycle — defining a question, collecting, cleansing, analysing, and reporting data — using visualisation tools such as charts and graphs to communicate insights. Debugging skills are developed by identifying and resolving syntax errors, interpreting error messages, and using comments to make code more efficient and maintainable. This builds computational thinking, the ability to translate algorithms into functional Python programs, and the analytical skills needed for evidence-based reasoning and data-driven decision-making **Character** – resilience in tackling coding challenges and in cleaning, analysing, and interpreting complex data sets, critical thinking in designing algorithms, applying control structures, and drawing justified conclusions from evidence, responsibility in writing clear, efficient, and ethical code while handling	**STEM** – problem solving through applying knowledge of digital graphics and database design, understanding how images and data are stored, represented, and manipulated using appropriate tools and structures. Students explore how file type, resolution, and colour depth affect image quality and file size, and how data types, validation, and queries ensure the accuracy and efficiency of databases. They develop practical skills in using layers, transformations, and editing tools such as Magic Wand and Magnetic Lasso to isolate and modify elements, and in designing and constructing databases with tables, fields, and relationships. Students analyse how design and editing choices affect suitability for purpose, and how calculated fields, forms, and reports support clear presentation and meaningful insight, developing computational thinking, problem-solving, and digital creativity across both contexts **Character** – creativity in producing purposeful digital content and user-focused data solutions, resilience in mastering complex tools, software, and design principles, responsibility in handling data ethically and using images lawfully while respecting copyright, critical thinking in selecting effective methods to ensure technical accuracy and	**STEM** – problem solving through applying knowledge of data representation and database design, converting between binary, denary, and hexadecimal systems, and understanding how data is stored, processed, and managed within digital systems. Students explore units of digital storage (bit, nibble, byte, kilobyte, megabyte, gigabyte, terabyte, petabyte) and learn how binary and hexadecimal are used in computing for efficiency and readability. They also design and query relational databases using SQL, applying commands such as SELECT, FROM, and WHERE to retrieve and filter data, using comparison and logical operators (AND, OR, NOT) to refine results, and implementing aggregate functions such as SUM() and COUNT(). Students develop understanding of primary and foreign keys, relational structures, and data manipulation statements (INSERT, UPDATE, DELETE), while recognising vulnerabilities such as SQL injection and the importance of secure coding. Together, these skills build deep computational understanding of how data is represented, processed, and protected in digital systems **Character** – resilience in mastering both abstract and technical concepts such as binary arithmetic, character	**STEM** – problem solving through applying logical and creative thinking across computational and design contexts. Students use Boolean logic and logic gates to analyse and represent conditions in algorithms, interpret logic using written expressions and Venn diagrams, and apply decomposition and abstraction to simplify complex problems. They develop efficiency through the use of loops and nested loops and explore network (graph) theory concepts such as nodes and edges to understand connectivity and relationships within systems. Alongside this, students apply animation techniques including frame-by-frame, shape tweening, and motion tweening, using frame rate and keyframes to control speed and smoothness, and employ symbols, motion paths, and scripting to add interactivity. They also examine data compression, understanding lossy and lossless methods and their importance in efficient digital communication. Through these activities, students combine computational logic with digital creativity to design, optimise, and communicate effective technical and visual solutions **Character** – creativity in designing and producing purposeful digital and algorithmic solutions, resilience in mastering abstract logical concepts and refining complex	**STEM** – problem solving through analysing different types of cybersecurity threats, including malware, brute force, and DDoS attacks, applying knowledge of hacking and preventative methods to reduce risk, and understanding how human error can compromise data security. Students evaluate the impact of various threats in terms of likelihood and severity, explore the use of technical and organisational protection methods such as firewalls, antivirus software, and strong authentication, and develop an understanding of how online services collect, store, and process user data. They also investigate how malicious bots and cyberattacks can affect not only individuals but also wider social systems through misinformation and service disruption **Character** – responsibility in handling personal and digital data securely, resilience in understanding and addressing complex cybersecurity threats, critical thinking in evaluating the impact and effectiveness of protection strategies, and professionalism in applying legislation, secure practices, and ethical behaviour to protect individuals, organisations, and society from harm	**STEM** – problem solving through planning, designing, and developing interactive digital products such as games and mobile applications. Students apply event-driven programming principles to control gameplay and app behaviour, using variables, conditions, and logic to create engaging and functional experiences. They design structured objectives, multiple levels, and user interactions, integrating features such as particle effects, navigation systems, icons, galleries, and maps to enhance usability and visual appeal. Concept art, themes, and wireframes are used to plan user interfaces and environments effectively, while testing and debugging processes are applied to refine functionality and performance. Through the design–test–evaluate cycle, students strengthen their computational thinking, logical reasoning, and understanding of user-centred design, developing the technical and creative skills used in the digital and games industries **Character** – creativity in producing original, purposeful, and user-focused digital solutions, resilience in testing, debugging, and refining gameplay mechanics and app functionality, responsibility in ensuring that content is inclusive, ethical, and age-appropriate, critical thinking in

	data accurately and securely, and professionalism in testing, refining, and presenting digital solutions that are purposeful, reliable, and effectively communicated	meaningful output, and professionalism in producing high-quality, user-friendly, and ethically sound digital work that meets defined objectives	encoding, and SQL syntax, critical thinking in designing efficient solutions and applying logical reasoning to data problems, responsibility in handling and modifying data accurately, securely, and ethically, and professionalism in writing clear, well-structured code and presenting precise, logical explanations of complex processes	animations or programs, critical thinking in evaluating efficiency, accuracy, and quality in both technical and creative outputs, responsibility in applying logical reasoning, ethical decision-making, and suitability for audience and purpose, and professionalism in testing, refining, and presenting polished work that demonstrates precision, structure, and innovation		evaluating usability, engagement, and design effectiveness, and professionalism in presenting polished, high-quality products that demonstrate both technical competence and thoughtful design
Learning End Points	**Python Programming** By the end of this unit students will know and understand: • How to use a text-based programming language to create programs using sequence, variables, selection and iteration • The rules for creating variables • What a syntax error is and how to interpret an error message within the small basic environment • The use and value of using comments • The importance of using correct data types • How to use different comparison operators • How to use different logical operators • When to use a For Loop • When to use a While Loop **Data Science** By the end of this unit students will know and understand: • The definition of data science and its purpose in analysing information. • How visualising data (e.g. charts, graphs) can help identify patterns, trends, and insights. • Examples of where large data sets are used in daily life (e.g. social media, shopping, transport, healthcare).	**Graphics** By the end of this unit students will know and understand: • The characteristics of vector graphics, how they are stored, and their typical uses. • The components of vector graphics, including shapes, fills, outlines (strokes), merging, and grouping. • That bitmap graphics are made from individual pixels and can represent black and white, grayscale, or colour images. • How colour maps are used to create colour images. • That the number of bits per pixel determines the number of available colours in an image. • How the choice between vector and bitmap graphics affects file quality and file size. • The impact of altering image resolution on quality, file size, and suitability for purpose. • How layers work and their role in creating composite images. • That resizing, transforming, and rotating images can change appearance while preserving or altering quality. • How image editing tools (e.g. Magic Wand with tolerance, Magnetic Lasso) are used to	**Data Representation** By the end of this unit students will know and understand: • Why computer systems use binary • How to convert numbers to and from binary • Define the terms bit, nibble, byte, kilobyte, megabyte, gigabyte, terabyte, petabyte • Understand that data needs to be converted into a binary format to be processed by a computer • The rules when adding binary numbers together • What a binary overflow error is • How to convert denary numbers into Hexadecimal and vice versa • Why Hexadecimal is used by programmers over binary • The term 'character set' • The relationship between the number of bits per character in a character set, and the number of characters that can be represented using: ○ ASCII ○ Extended ASCII ○ Unicode **Practical Skills in SQL** By the end of this unit students will know and understand: • That databases consist of one or more tables, and that tables store data in rows and columns.	**Computational Thinking & Logic** By the end of this unit students will know and understand: • The common Boolean operators: ○ AND ○ OR ○ NOT • Different logic gates including: ○ AND gates ○ OR gates ○ NOT gates • What an algorithm is • How Boolean operators can be represented in written expressions and • Venn diagrams • How logic is used in different situations • How loops can be used to reduce the amount of code required for a solution • The difference between lossy and lossless compression • Why compression is needed for video transmission and photo storage • How abstractions are used in everyday life • How networks are used to make an abstraction of a maze • How decomposition can be used to break down problems into more manageable components	**Introduction to Cybersecurity** By the end of this unit students will know and understand: • The difference between data and information. • That data entered online is collected, stored, and processed by online services, often affecting data privacy. • The purpose of the Data Protection Act and how it safeguards personal data. • That human error can create security risks for data and why it is important to minimise these risks. • The definition of hacking in the context of cybersecurity. • What a DDoS attack is and how it can affect users of online services. • That a brute force attack is an attempt to break security by trying many combinations and that strategies exist to reduce its effectiveness. • The purpose of the Computer Misuse Act and the offences it addresses. • Common types of malware (e.g. viruses, worms, trojans, ransomware, spyware). • The problems that different types of malware can cause for computer systems and data. • That malicious bots can influence wider societal issues (e.g. spreading	**GDevelop** By the end of this unit students will know and understand: • That a new game project can be created in GDevelop and customised (e.g. background colour, objects). • The purpose of a particle emitter object and how changing its properties affects gameplay visuals. • That objects can have multiple instances and orientations within a game. • That events are used to control game behaviour, such as creating new object instances. • The key components of a platform game, including: ○ Game objectives ○ Characters and sprites ○ Levels and interactions • That themes and concept art inform the design of characters, sprites, and environments. • How collision detection works and why collision masks are used to control interactions between objects. • That camera controls (e.g. smooth camera scrolling) can extend the play area beyond the screen. • That platform games include typical mechanics such as: ○ Falling off the screen and respawning at the start

	• The meaning of correlation and outliers in relation to data trends. • The steps of the investigative cycle (e.g. define a question, collect data, clean data, analyse, conclude, report). • That different criteria can be applied to a data set to make predictions. • That findings can be used to support or challenge a prediction or argument. • The importance of data cleansing to remove errors, inconsistencies, or missing values. • That data capture forms are used to collect relevant and reliable data. • That visualisations can be analysed to identify patterns, trends, correlations, and anomalies. • That conclusions drawn from data should be communicated and reported clearly.	isolate images from backgrounds. • How text properties (e.g. size, fill colour, leading/line height) can be altered to enhance image design. • That layer properties (e.g. drop shadow, stroke) can be adjusted to enhance visual presentation. **Databases** By the end of this unit students will know and understand: • What a database is and common scenarios where databases are used. • The terms table, record, and field, and how they are used in databases. • That database tables must be designed with suitable data types for fields. • That validation rules and validation text can be applied to fields to reduce data entry errors. • That queries are used to interrogate data within a database. • That queries can use logical and comparison operators (BETWEEN, >, >=, <, <=, =, <>, AND, OR, NOT). • That queries can include criteria, Boolean operators, comparison operators, and parameters to refine results. • That calculated fields can be used to generate new values within a query. • That data entry forms allow users to enter, edit, and delete records more easily, and can include user-friendly features (e.g. text prompts, combo boxes). • That websites and applications often use similar form features for data entry. • The purpose of reports in presenting database	• Basic SQL query structure and purpose: SELECT <columns> FROM <table> WHERE <condition>. • How wildcards are used in SQL (e.g. * to select all columns). • How WHERE conditions filter records and the common comparison operators (>, >=, <, <=, =, <>). • Logical operators used in conditions: AND, OR, NOT. • How to order query results with ORDER BY and sorting directions ASC (ascending) and DESC (descending). • How to limit the number of returned records using LIMIT. • More advanced condition keywords such as LIKE, IS NULL, and IS NOT NULL and their purposes. • Use and purpose of simple aggregate functions such as SUM() and COUNT(). • How relational databases can be built from two or more tables to represent related data. • The purpose of primary keys (unique record identifiers) and foreign keys (links between tables). • How to retrieve related data from two tables using appropriate SELECT and WHERE clauses (e.g. joining logic conceptually). • The SQL statements for changing data at a basic level: ○ Insert records: INSERT INTO ... VALUES (...) ○ Update records: UPDATE ... SET ... WHERE ... ○ Delete records: DELETE FROM ... WHERE ... • What SQL injection is	• How nested loops can be used to improve solutions further • Network (graph) theory terms including: ○ Nodes ○ Edges **Animation** By the end of this unit students will know and understand: • That drawing tools can be used to create and edit shapes (add, subtract, distort). • The difference between frame-by-frame animation, shape tweening, and motion tweening. • The role of frame rate and keyframes in determining the speed and smoothness of an animation. • That graphics can be imported into a library for use in animations. • The purpose of symbols and instances in animation. • That motion tweens can animate a symbol, including movement along a motion path. • That animations can be created within symbols (nested animation). • That text can also be animated using the same principles as objects. • That interactive buttons allow users to control elements of an animation. • That ActionScript (or similar scripting) can add interactivity to animations. • The techniques commonly used in animations (e.g. tweens, frame-by-frame, interactivity). • That a storyboard is used to plan an animation's purpose, sequence, techniques, and timings.	misinformation, disrupting services). • That different security threats can be evaluated in terms of their likelihood and impact on organisations. • That networks can be protected from threats using technical and organisational methods (e.g. firewalls, antivirus, strong authentication). • That there are a range of preventative methods to reduce the risk of cyberattacks.	○ Reaching the end of a level/game • The purpose of variables in storing and updating values during gameplay (e.g. score). • That hazard objects can be used to trigger events (e.g. returning a character to the start). • That testing and refining are essential to improve playability and user experience. • That self-evaluation helps identify areas of strength and gaps in understanding. **AppShed** By the end of this unit students will know and understand: • The different types of mobile apps and their features. • The features and capabilities of the AppShed platform. • That apps should be planned before development, including purpose, audience, and layout. • That app design uses Graphical User Interfaces (GUIs) and can be represented with wireframes using standard symbols. • That a typical home screen includes navigation features, icons, and links to other screens. • The purpose of a navigation bar with tabs to organise and access app sections. • That screens can contain icons, links, galleries, maps, and interactive elements. • That icons can link to specific content (e.g. information on chemical elements). • That external links (e.g. to websites or video platforms) can be embedded within apps. • The purpose of photo galleries in presenting collections of images.

		information in a structured way. • That reports can be customised, edited, and formatted to provide a consistent design.		• That sound can be added to animations to enhance impact. • That animations must be suitable for their purpose and target audience. • That animations must be exported in the correct format for sharing or publishing. • That evaluation of finished work involves reviewing strengths, weaknesses, and suitability for purpose.		• That map screens can display and edit map points for location-based content. • That the Blockly visual programming editor can add interactivity to apps, including quizzes. • The role of variables and if statements in quiz logic and app programming. • That apps must be tested and debugged to ensure links, media, and screens function correctly. • That a refined app should be suitable for its purpose and user-friendly for the intended audience.
Key Vocabulary	• Python Programming Keywords • Data Science Keywords	• Graphics Keywords • Databases Keywords	• Data Representation Keywords • Practical Skills in SQL Keywords	• Computational Thinking & Logic Keywords • Animations Keywords	• Introduction to Cybersecurity Keywords	• GDevelop Keywords • AppShed Keywords