Bishop Stopford School
faith | justice | responsibility | truth | compassion

Computing Department

Curriculum – Key Stage 5

Key Stage 5 Curriculum Overview 2025-26

		AUTUMN 1	AUTUMN 2	SPRING 1	SPRING 2	SUMMER 1	SUMMER 2
11	Unit description (teacher 1)	Reading and writing algorithms and programs (1.2.1, 2.2.1, 2.3.1) Students recap core Python skills and extend their knowledge with advanced techniques such as recursion and passing parameters by value or reference. They learn to write, trace, and evaluate algorithms, using Big O notation to assess efficiency.	Algorithms and data structures (1.3.1 & 1.3.2) Students learn how to implement and manipulate key data structures such as stacks, queues, linked lists, trees, and hash tables using code. They apply this knowledge to explore and program pathfinding algorithms, including Dijkstra's and A*, developing an understanding of their use, efficiency, and real-world applications.	Object oriented programming (2.2.1) Students build on prior programming experience by learning the principles of object-oriented programming (OOP) in Python. They develop classes, create objects, and apply concepts such as inheritance, encapsulation, and polymorphism to structure programs more efficiently and maintainable.	Web development (1.4.2 & 2.2.1) Students develop dynamic websites using HTML, CSS, and JavaScript, applying programming concepts to enhance interactivity and design. They learn how websites are structured, styled, and scripted, and explore key web technologies such as the client-server model and the use of APIs.	Data Types and Databases (1.3.1 & 1.3.3) Students explore fundamental data types and how data is structured and stored efficiently. They then apply this understanding to relational databases, learning to create, query, and manipulate data using SQL. Topics include normalisation, primary and foreign keys, and the role of databases in modern information systems.	**Compression, Encryption and Hashing (1.3.1)** Students explore how data is secured, reduced, and verified through various techniques. They learn the principles and purposes of lossy and lossless compression, symmetric and asymmetric encryption methods, and hashing algorithms, with a focus on their use in secure communications and data integrity.
	Unit description (teacher 2)	**Computational thinking (1.21-1.2.5)** Students explore core principles of computational thinking, including abstraction, decomposition, and procedural design. They learn to plan structured solutions, and identify inputs, outputs, and preconditions. While concurrency introduces how tasks can run in parallel to improve system performance. Student will learn different sorting and searching algorithms.	2.2.2 – Computational Methods Students explore techniques for approaching and solving complex problems using computational methods. This includes problem recognition, performance modelling, and simulation. They learn how to apply algorithms to real-world scenarios, evaluate the effectiveness of solutions, and understand when approximation or heuristic methods are appropriate.	2.1.2 – Software Development (with NEA-style Project) Students explore software development methodologies including agile, waterfall, and spiral models, understanding how they guide real-world projects. Alongside this, they complete a group project that mimics the NEA, practising each stage of the development lifecycle—analysis, design, implementation, testing, and evaluation—to build a functioning program collaboratively.		NEA – Analysis Section Students begin their NEA by identifying a real-world problem and conducting in-depth analysis. They gather and interpret user requirements, define key objectives, and explore existing solutions. This section emphasises understanding the problem domain and setting a clear foundation for the design and development stages.	NEA – Planning Section Students translate their analysis into a clear development plan, outlining system requirements, data structures, algorithms, and interface designs. They produce detailed designs using tools such as flowcharts, pseudocode, and wireframes, setting out a logical, structured approach to guide the implementation of their solution.
	Assessment	Baseline programming assessment End of topic test (writing algorithms / Big O notation) End of topic test (Computational thinking)	End of topic test with a programming element (data structures) End of topic test (Computational methods)	Programming assessment (OOP) End of topic test (Software development)	Web development programming challenge End of unit test (web development) Hand in and marking of group project	End of unit test (databases and datatypes) Hand in and marking of NEA analysis section	End of topic test (compression, encryption and hashing) Y12 mocks (hybrid paper featuring paper 1 / 2 content) Hand in and marking of NEA planning section.

Key Stage 5 Curriculum Overview 2025-26

		AUTUMN 1	AUTUMN 2	SPRING 1	SPRING 2	SUMMER 1
11	Unit description (teacher 1)	<u>Databases (1.3.1 & 1.3.3)</u> Students explore relational databases, learning to create, query, and manipulate data using SQL. Topics include normalisation, primary and foreign keys, and the role of databases in modern information systems.	<u>Data Types and Boolean Logic (1.4.1 & 1.4.3)</u> Students explore a variety of data types, including integer, real, character, string, and Boolean, along with how data is represented and manipulated in memory. They also study Boolean algebra, using logic gates and truth tables to simplify logical expressions and understand how decisions are made within computer systems.	<u>Paper 1 revision</u> • Computer systems and architecture (including the CPU, memory, and storage) • System software, networking, cyber security, and ethical/legal issues • Databases, web technologies, and emerging technologies Revision for spring 2 and summer 1 will be decided based on the data from the February mocks.		
	Unit description (teacher 2)	<u>Algorithms (2.3.1 Algorithms)</u> Students study a range of standard algorithms, including searching and sorting methods such as binary search, merge sort, and quick sort. They analyse algorithm efficiency using Big O notation and learn to compare and evaluate algorithmic approaches to problem-solving. The unit also reinforces tracing, pseudocode, and structured design skills.	<u>NEA – Implementation Section</u> Students implement their solution by writing well-structured, efficient code based on their designs. They apply appropriate programming techniques and maintain good documentation throughout development. This section emphasizes iterative testing and refinement to ensure the program meets the user requirements effectively.	<u>NEA – Testing and evaluation Section</u> Students systematically test their program to identify and fix errors, using a variety of testing methods to ensure functionality and reliability. They evaluate the final solution against the original requirements, reflecting on its strengths, weaknesses, and potential improvements for future development. Students will make final changes to their coursework before final hand in.	<u>Paper 2 revision</u> • Fundamentals of programming, data structures, algorithms, and computational thinking • Programming techniques such as data types, structures, searching, sorting, and algorithm efficiency • Theory on Boolean logic, system architecture, and software development principles • Focus on practical coding skills, algorithm design, and problem-solving in Python Revision for spring 2 and summer 1 will be decided based on the data from the February mocks.	
	Assessment	End of topic test (databases) End of topic test (Algorithms)	End of topic test (Datatypes and Boolean logic) Hand in and marking of NEA implementation section	Y13 mocks – paper 1 / paper 2 Hand in and marking of NEA	exam style questions for topics covered in the revision. Past papers	

>Something More?

Curriculums at BSS are designed to nurture not only intellectual and physical development but also the spiritual growth of students. This will be through:

Encouraging students to reflect on their experiences, beliefs and purpose and to contemplate the big Questions of Who am I? Why am I here? What is my purpose?

Highlighting extraordinary people, events, and discoveries that inspire awe or investigates how a sense of awe has led to breakthroughs and creativity.

Using art, music, literature, and nature to inspire awe, wonder, and spiritual insight.

Encouraging creative expression to connect with the inner self and the transcendent.

Fostering a sense of belonging and interconnectedness with others, nature, and the universe.

Encouraging self-awareness, emotional intelligence, and moral reasoning.

Promoting open-ended investigations rather than just seeking right answers.

Using hands-on activities, field trips and experiments to immerse students in learning and evoke wonder.

How does our curriculum do >Something More?

Encouraging students to reflect on their experiences, beliefs and purpose and to contemplate the big Questions of Who am I? Why am I here? What is my purpose?

- ✓ **Discussions on AI, social media, or data ethics can prompt reflection on personal values and the role of humans in a digital world.**

Encouraging creative expression / Promoting open-ended investigation

- ✓ **Once students have mastered the fundamentals of Python programming, they can begin to explore a wide range of exciting pathways, including game design, app development, computer networking, and even music through code. They are actively encouraged to pursue their own interests and experiment with different branches of computing beyond the limits of the A-level specification.**
- ✓ **Student will complete an NEA project. Student may choose any program to make and are encouraged to pick something that supports their studies, links to a hobby or that brings them joy.**

Fostering a sense of belonging and interconnectedness

- ✓ **Students are introduced to a range of different technologies to encourage communication and collaborative work through office 365 and paired programming though the sense hats.**

Encouraging self-awareness, emotional intelligence, and moral reasoning

- ✓ **Topics like AI bias, data privacy, or cybersecurity ethics can prompt students to think deeply about fairness, justice, and personal responsibility**.