Computer Science

Curriculum Overview

At Dixons Kings we develop students to lead successful and happy lives and make a positive contribution to their community. Our curriculum in each year is designed to provide experiences, opportunities, knowledge and skills that enrich and challenge our students. We understand that the curriculum is key to determining the life chances and choices for our students and therefore we will not compromise on providing the very best. We achieve this in Computer Science through the below:

Knowledge, skills and understanding to be gained at each stage:

		Cycle 1	Cycle 2	Cycle 3
Year 9	**Knowledge Introduced**	Computer architecture, components of a CPU, CPU performance factors, memory types, fetch-execute cycle. Python basic syntax, expressions and operators, sequence, selection, iteration, data types.	Hexadecimal number systems. Image, text and sound representation in binary (including Unicode). Python lists and Python function definitions.	Database structure and SQL syntax. Multi-table relationships and keys. SQL data types.
	Knowledge Revisited	Computer system components, input and output devices, storage devices. Simple Python syntax, input and output functions.	Binary number systems. Text representation in ASCII. B&W image representation.	Simple data tables. Data types. Ascending and descending sorts.
	Skills Introduced	Comparison of CPU features for different contexts. Programming using different types of selection and iteration.	Conversion from bin2hex, hex2bin, dec2hex, hex2dec. Decoding and encoding colour images. Programming using lists. Programming using subroutines.	Designing table schema. Writing and interpreting SQL select statements. Writing and interpreting SQL create / insert / update statements.
	Skills Revisited	Comparison of storage features for different contexts. Programming using simple selection and count-controlled iteration.	Conversion from bin2dec, dec2bin. Decoding and encoding B&W images. Programming using different types of selection and iteration.	Interpreting simple table queries. Sorting and filtering.

		Cycle 1	Cycle 2	Cycle 3
Year 10	**Knowledge Introduced**	Linear and binary search. Bubble sort and merge sort. Comparisons of different algorithms. Network topologies, network stack layers. Packets and packet switching. Network security.	Structured and secure programming. Types of test data. Types of validation test. Subroutine interfaces. Cyber security threats and mitigation, including encryption, access levels, password security, MFA. Penetration testing. Social engineering.	Logic gates, symbolic logic and truth tables. Types of software and language translators.
	Knowledge Revisited	Purpose of algorithms. Selection and iteration (count and condition controlled). Variables and constants. Network hardware. LANs and WANs.	Purpose of testing. Python function definitions. Types of malware. Network security.	Boolean logic and basic Boolean operators. Machine code. Application and utility software.
	Skills Introduced	Dry-running algorithms using trace tables. Performing linear and binary search, bubble sort and merge sort.	Choosing test data and designing test plans. Choosing and writing validation routines.	Producing and interpreting truth tables. Producing and interpreting logic gate circuit diagrams.
	Skills Revisited	Tracing through code. Finding and fixing errors in code.	Boolean logic, especially used in range checks.	Interpreting CPU instructions (machine code). Operation of the fetch-execute cycle.

		Cycle 1	Cycle 2	Cycle 3
Year 11	**Knowledge Introduced**			
	Knowledge Revisited	Python syntax, expressions and operators, sequence, selection, iteration, data types. Computer architecture, components of a CPU, CPU performance factors, memory types, fetch-execute cycle. Network topologies, network stack layers. Packets and packet switching. Network security.	Binary number systems. Text representation in ASCII. B&W image representation. Cyber security threats and mitigation, including encryption, access levels, password security, MFA. Penetration testing. Social engineering.	Database structure and SQL syntax. Multi-table relationships and keys. SQL data types. Logic gates, symbolic logic and truth tables. Types of software and language translators.
	Skills Introduced			
	Skills Revisited	Dry-running algorithms using trace tables. Advanced Python programming. File IO and 1 and 2D array operations.	Advanced Python programming. Project-based problem solving, design, implementation and testing.	Dry-running algorithms using trace tables. Performing linear and binary search, bubble sort and merge sort.

A powerful, knowledge-rich curriculum teaches both declarative knowledge (facts; knowing that something is the case; what we think about) and non-declarative or procedural knowledge (skills and processes; knowing how to do something; what we think with). There are no skills without bodies of knowledge to underpin them. In some subjects, a further distinction can be made between substantive knowledge (the domain specific knowledge accrued e.g. knowledge of the past) and disciplinary knowledge (how the knowledge is accrued e.g. historical reasoning). Please refer to the DAT Curriculum Principles, published on our website, for further information about how we have designed our curriculum.