Subject Curriculum Plan – KS5

Subject	**Computer Science (AQA A-Level 7517)**
Year Group	**12**
Subject Intent	The Key Stage 5 A-Level Computer Science (AQA 7517) Year 12 curriculum introduces advanced computational theory, object-oriented software engineering, mathematical data representations, and systematic problem-solving. Students develop rigorous mastery over abstract data structures (queues, stacks, lists, hash tables, graphs, trees, vectors), algorithmic efficiency (Big-O analysis), relational database design (normalisation to 3NF and SQL), and modern paradigms (OOP and Functional Programming). Crucially, 20% of lesson time throughout Year 12 is structured for students to work on their AQA Non-Exam Assessment (NEA) Programming Project. By the end of Year 12, students apply formal software life cycle methodologies to complete and submit a comprehensive first draft of their project report containing the full Analysis and Design sections.

Subject Curriculum Plan – KS5

HASMONEAN
אל תקרי בניך אלא בוניך

TERM	TOPIC & CORE LEARNING	KEY CONCEPTS
Autumn (Term 1)	Unit 7: Data Structures & Unit 8: Algorithms • Unit 7 Data Structures: Abstract Data Types (ADTs). Implementation, operations, and applications of Linear, Circular, and Priority Queues (add, remove, test empty/full). Stacks (push, pop, peek, empty/full, subroutine call stack frames). Lists (static vs dynamic data structures). Hash Tables (hashing functions, collision handling via rehashing, dictionaries). Graphs (directed/undirected, weighted, adjacency matrices vs adjacency lists). Trees (binary trees, rooted tree traversals). Vectors (list/dictionary representation, vector addition, scalar multiplication, dot product, angle between vectors). • Unit 8 Algorithms: Recursive algorithms (tracing, call stacks, tree traversals: pre-order, in-order, post-order). Big-O notation (constant, linear, polynomial, exponential, logarithmic time complexity). Searching and Sorting algorithms (Linear, Binary, Binary Tree search, Bubble sort, Merge sort analysis and tracing). Graph traversal (Depth-First Search using stacks, Breadth-First Search using queues). Optimisation algorithms (Dijkstra's shortest path algorithm). Limits of Computation (tractable vs intractable problems, Halting problem, uncomputable problems). • NEA Project Allocation (20% Lesson Time): Project initiation: identifying a complex problem, client/stakeholder analysis, and preliminary requirement gathering.	Abstract Data Types, Queues, Stacks, Call Stack Frames, Hash Tables, Collisions, Graphs, Adjacency Matrix/List, Binary Trees, Vectors, Dot Product, Recursion, Big-O Notation, Time Complexity, Depth-First Search, Breadth-First Search, Dijkstra's Algorithm, Tractability, Halting Problem, NEA Scope & Requirements.
SEQUENCING	Front-loads core algorithmic thinking and dynamic abstract data structures early in Year 12. Establishing ADTs, recursion, and search/sort mechanics early enables students to architect sophisticated data models and algorithms required both for AQA Paper 1 theoretical computation and for designing non-trivial data structures in their 20% NEA project.	

Subject Curriculum Plan – KS5

HASMONEAN
אל תקרי בניך אלא בוניך

TERM	TOPIC & CORE LEARNING	KEY CONCEPTS
Winter (Term 2)	Unit 1: Fundamentals of Programming & Unit 11: Relational Databases and Software Development • Unit 1 Fundamentals of Programming: Programming basics, primitive data types, arithmetic/string operations, variables vs constants. Control structures: Selection (relational/Boolean operators, nested IF statements) and Iteration (WHILE, REPEAT, FOR loops). Data structures: 1D and 2D arrays. Subroutines (functions vs procedures, parameters, arguments, return values, local vs global scope). File handling (reading/writing text and binary files) and robust Exception Handling (try-catch/try-except blocks). • Unit 11 Relational Databases & Software Development: Entity-Relationship (ER) modeling (entities, attributes, relationships, ER diagrams). Relational database concepts (relations, primary keys, composite keys, foreign keys). Normalisation process (1NF, 2NF, 3NF, transitive dependency elimination). Structured Query Language (SQL) for data querying (SELECT, WHERE, ORDER BY, JOIN) and data definition/manipulation (CREATE TABLE, INSERT, UPDATE, DELETE). Concurrent database access control. Systematic software development life cycle (waterfall vs agile/prototyping, data modeling, evaluation criteria). • NEA Project Allocation (20% Lesson Time): Translating user requirements into database schemas (3NF) and structural system designs for the NEA project. Drafting initial entity models, pseudocode subroutines, and UI mockups.	Control Structures, Arrays (1D/2D), Subroutines, Scope, File I/O, Exception Handling, Entity-Relationship Diagrams, Normalisation (1NF, 2NF, 3NF), Primary/Foreign Keys, SQL (DDL/DML), Multi-table JOINs, Concurrency Control, Software Life Cycle, Agile/Prototyping.
SEQUENCING	Consolidates core procedural programming syntax with database engineering and system design methodology. Equips students with persistent data storage skills (SQL/3NF) and structured coding habits essential for building robust back-end systems in their NEA project and scoring highly on Paper 1 & Paper 2 database modules.	

Subject Curriculum Plan – KS5

HASMONEAN
אל תקרי בניך אלא בוניך

TERM	TOPIC & CORE LEARNING	KEY CONCEPTS
Spring (Term 3)	Unit 12: OOP and Functional Programming & Unit 3: Data Representation • Unit 12 OOP and Functional Programming: Object-Oriented Programming (OOP) fundamentals: Classes, Objects, Instantiation, Encapsulation. Advanced OOP design principles: Inheritance, Polymorphism, Method Overriding, Aggregation vs Composition vs Association. OOP design guidelines (encapsulate what varies, favor composition, program to interfaces). Drawing and interpreting UML Class Diagrams. Functional Programming paradigm: Pure functions, domain & co-domain, first-class objects, function composition, partial function application. Higher-order functions (Map, Filter, Fold/Reduce). Functional list processing (head, tail, prepend/append). Big Data characteristics (Volume, Velocity, Variety), fact-based data modeling, and graph schema. • Unit 3 Data Representation: Number systems (Natural, Integer, Rational, Irrational, Real, Ordinal numbers; base conversions between Decimal, Binary, Hexadecimal). Binary prefix units (Ki, Mi, Gi vs K, M, G). Character sets (ASCII, Unicode). Error-checking mechanisms (parity bits, checksums, Hamming code). Binary arithmetic: unsigned addition/multiplication, Two's Complement signed representation, subtraction, Fixed-Point real numbers. Image representation: Bitmaps (resolution, color depth, file size formula), Vector graphics (lists of objects, comparative analysis). Sound representation: sampling rate, bit depth, sound file size calculations. Data compression (Lossy vs Lossless, RLE, Huffman Coding) and Encryption (Caesar and Vernam ciphers). • NEA Project Allocation (20% Lesson Time): Applying OOP architecture (defining classes, encapsulation, inheritance hierarchies) or functional data transformations to the NEA software design. Documenting system architecture and module interfaces.	OOP, Encapsulation, Inheritance, Polymorphism, UML Class Diagrams, Functional Programming, First-Class Objects, Higher-Order Functions (Map/Filter/Fold), Big Data, Two's Complement, Fixed-Point Binary, Vector Graphics, Huffman Coding, Vernam Cipher, System Architecture.
SEQUENCING	Introduces advanced programming paradigms (OOP and Functional) alongside low-level mathematical data representation. Applying OOP design principles directly supports modular architecture in the NEA project, while data representation secures critical mathematical and binary calculation skills tested on AQA Paper 2.	

Subject Curriculum Plan – KS5

HASMONEAN
אל תקרי בניך אלא בוניך

TERM	TOPIC & CORE LEARNING	KEY CONCEPTS
Spring (Term 4)	Unit 2: Problem Solving and Theory of Computation & Unit 9: Regular Languages • Unit 2 Problem Solving & Theory of Computation: Computational thinking techniques (Abstraction, Decomposition, Algorithmic Thinking). Structured programming approach (hierarchy charts, modularity benefits). Algorithm development and interpretation in formal pseudocode. Systematic Testing strategies: designing test plans, test data selection (normal, boundary, invalid/erroneous), hand-tracing algorithms using trace tables. Abstraction types (procedural, functional, data, abstraction by generalisation). • Unit 9 Regular Languages: Finite State Machines (FSMs) with and without output (Mealy machines, state transition diagrams, state transition tables). Set Theory: set notation, set comprehension, finite/infinite/countably infinite sets, cardinality, Cartesian product, set operations (union, intersection, difference, membership, subsets). Regular Expressions (Regex) syntax, matching, and equivalence to FSMs. Turing Machines: tape, transition functions, state diagrams, Universal Turing Machine concept and significance. Backus-Naur Form (BNF) production rules and Syntax Diagrams. Reverse Polish Notation (RPN) conversion to/from infix and evaluation using stacks. • NEA Project Allocation (20% Lesson Time): Formulating comprehensive test plans (normal, boundary, invalid test cases), designing FSM state logic, and finalizing pseudocode algorithms for the NEA project.	Computational Thinking, Hierarchy Charts, Algorithm Hand-Tracing, Abstraction, Finite State Machines, Mealy Machines, Set Theory, Regular Expressions, Universal Turing Machine, Backus-Naur Form (BNF), Reverse Polish Notation (RPN), Test Plan Design.
SEQUENCING	Establishes formal theoretical computer science foundations and rigorous testing protocols. Deepens understanding of language syntax, automata theory, and notation (BNF/RPN) required for Paper 1 theory, while embedding robust test planning into the student's NEA documentation.	

Subject Curriculum Plan – KS5

HASMONEAN
אל תקרי בניך אלא בוניך

TERM	TOPIC & CORE LEARNING	KEY CONCEPTS
Summer (Term 5)	Unit 4: Hardware and Software & Unit 10: The Internet • Unit 4 Hardware and Software: Relationship between Hardware and Software. Classification of software: Systems software (Operating Systems, Utilities, Translators, Libraries) vs Application software. Role and functions of an Operating System: resource management, hiding operational complexity, processor scheduling algorithms, virtual memory management. Programming language levels: High-level (imperative) vs Low-level (Machine Code, Assembly). Translators: Compilers, Interpreters, Assemblers; bytecode and intermediate code execution. Boolean Logic: truth tables for logic gates, logic circuit diagrams, Boolean algebra identities, De Morgan's Laws, and algebraic expression simplification. • Unit 10 The Internet: Structure and topology of the Internet. Uniform Resource Locators (URLs), Domain Name System (DNS) hierarchy, Domain Name Servers, IP addresses, Internet Registries. Packet Switching mechanics, packet structure, role of routers and gateways. Internet security: Firewalls, Symmetric vs Asymmetric encryption, Digital Signatures, Digital Certificates, Malware (Viruses, Worms, Trojans) and mitigation strategies. TCP/IP Protocol Stack (4 layers), sockets, port numbers, MAC addressing, FTP (anonymous/authenticated), SSH, email servers (SMTP/IMAP/POP3), web servers and browsers. Network addressing: IPv4 vs IPv6, subnet masks, network/host ID split, routable vs non-routable IPs, DHCP, NAT, port forwarding. Web paradigms: Client-Server model, WebSockets, RESTful web services (CRUD), JSON vs XML, Thin-client vs Thick-client architectures. • NEA Project Submission Milestone (20% Lesson Time): Finalizing and consolidating all Year 12 project documentation. END OF YEAR 12 DELIVERABLE: Formal submission of the complete FIRST DRAFT of the NEA Project Report containing full Analysis and Design sections.	Operating System, Processor Scheduling, Compilers vs Interpreters, Bytecode, Logic Circuits, De Morgan's Laws, DNS, Packet Switching, TCP/IP Stack, Subnetting, Public/Private Keys, REST, WebSockets, JSON, NEA Analysis & Design Draft.
SEQUENCING	Completes hardware, software systems, and internet infrastructure modules. Provides a holistic view of modern networked computing environments for Paper 2, while concluding Year 12 with the critical submission of the full Analysis and Design sections for the NEA Programming Project.	

Subject Curriculum Plan – KS5

Subject	**Computer Science (AQA A-Level 7517)**
Year Group	**13**
Subject Intent	The Key Stage 5 A-Level Computer Science (AQA 7517) Year 13 curriculum consolidates advanced theoretical systems, completes the Non-Exam Assessment (NEA) Programming Project lifecycle, and drives students toward examination excellence in Paper 1 and Paper 2. Students finalize their practical software technical implementation, execute rigorous systematic testing, and produce a critical evaluation, submitting all sections as a full draft by Term 2 and the final polished version in Term 3. Concurrently, students conduct in-depth analysis and modification of the pre-released Paper 1 Skeleton Program, master low-level computer architecture (F-D-E cycle, assembly), and examine communication technologies alongside ethical/legal computing frameworks. Comprehensive past paper practice using Exampro question banks, mark scheme analysis, and command word drills ensure top performance in final May/June examinations.

Subject Curriculum Plan – KS5

HASMONEAN
אל תקרי בניך אלא בוניך

TERM	TOPIC & CORE LEARNING	KEY CONCEPTS
Autumn (Term 1)	Finalising Programming Project (NEA Implementation) & Unit 5: Computer Organisation and Architecture & Paper 1 Skeleton Program • Finalising Programming Project (NEA): Intensive practical software development phase. Code implementation in chosen high-level language (e.g., Python/C#/Java), implementing database connectivity, modular subroutine structure, exception handling, and custom class hierarchies. • Unit 5 Computer Organisation & Architecture: Internal computer components: Processor, Main Memory, I/O controllers, System Buses (Control, Address, Data Bus). Addressable memory and stored program concept (Von Neumann & Harvard architectures). Processor architecture: Internal registers (MAR, MDR, PC, CIR, Accumulator), stages of the Fetch-Decode-Execute (F-D-E) cycle. Factors affecting processor performance (clock speed, cache size/type, core count, bus width). Interrupts and Interrupt Service Routines (ISRs). Instruction Sets: format (opcode & operand), Direct vs Immediate addressing modes. Assembly Language programming: mnemonic machine code operations, conditional/unconditional branching. Input/Output devices: Barcode readers, Digital cameras, Laser printers, RFID. Secondary storage technologies: Optical, Magnetic (HDD), Solid-State (SSD) principles of operation and comparative evaluation. • Skeleton Programming Project (Paper 1): Introduction to the AQA pre-released Skeleton Program code. Analyzing software architecture, identifying key data structures, tracing code flow, and understanding object-oriented class interactions in the pre-released material.	NEA Technical Implementation, System Buses, Fetch-Decode-Execute Cycle, Processor Registers, Interrupts, Instruction Sets, Addressing Modes, Assembly Language, HDD vs SSD, Skeleton Code Architecture, Object Interaction Tracing.
SEQUENCING	Pairs low-level machine architecture and assembly programming with hands-on code investigation of the AQA Paper 1 Skeleton Program. Simultaneously accelerates the technical coding phase of the NEA, moving from design blueprints into actual functional software execution.	

Subject Curriculum Plan – KS5

HASMONEAN
אל תקרי בניך אלא בוניך

TERM	TOPIC & CORE LEARNING	KEY CONCEPTS
Winter (Term 2)	Unit 6: Communication Technologies and Consequences & Skeleton Program & NEA Full Draft Report Submission • Unit 6 Communication Technologies & Consequences: Data communication methods: Serial vs Parallel transmission, Synchronous vs Asynchronous transmission. Network performance metrics: Baud rate, Bit rate, Bandwidth, Latency, Protocol definitions; relationship between bit rate and bandwidth. Network topologies: Star vs Bus topologies, Physical vs Logical network layout comparison. Client-Server vs Peer-to-Peer networks; Cloud computing advantages, security risks, and operational trade-offs. Wireless networking: Wi-Fi setup, Wireless Access Points (WAPs), CSMA/CA protocol with RTS/CTS, SSID broadcasting and WPA/WPA2/WPA3 security encryption. Moral, Social, Legal, and Cultural Issues: legal frameworks (Data Protection Act 2018 / GDPR, Computer Misuse Act 1990, Copyright Designs and Patents Act 1988), ethical implications of AI and automated decision-making, digital divide, tech environmental impacts, software engineer power and responsibility. • Skeleton Programming Project: Advanced manipulation and expansion of the Skeleton Code. Practicing simulated exam tasks, modifying methods, adding new features, and optimizing algorithms within the pre-released codebase. • NEA Project Submission Milestone: Finalizing the complete draft of the NEA Project Report. END OF TERM 2 DELIVERABLE: Mandatory submission of ALL SECTIONS OF THE REPORT COMPLETED AS A FIRST DRAFT (Analysis, Design, Technical Solution, Testing, and Evaluation).	Serial vs Parallel, Baud/Bit Rate, CSMA/CA, RTS/CTS, Client-Server vs P2P, Tech Legislation (GDPR/CMA/CDPA), Ethics in Computing, Skeleton Code Modifications, Complete First Draft NEA Submission.
SEQUENCING	Completes the final theoretical unit of the AQA specification (Unit 6), connecting networking hardware to societal, legal, and ethical frameworks assessed in Paper 2 essay questions. Achieves a major NEA milestone with the submission of the complete 5-section draft report.	

Subject Curriculum Plan – KS5

HASMONEAN
אל תקרי בניך אלא בוניך

TERM	TOPIC & CORE LEARNING	KEY CONCEPTS
Winter (Term 3)	Paper 1 Exam Techniques & Revision & NEA Final Submission & Skeleton Programming Mastery • Paper 1 Exam Techniques & Synoptic Revision: Intensive Paper 1 revision using Exampro question banks covering Units 1 (Programming Fundamentals), 2 (Problem Solving & Computation), 7 (Data Structures), 8 (Algorithms), 9 (Regular Languages), and 12 (OOP & Functional Programming). Systematic practice on high-tariff pseudocode writing, trace tables, recursive algorithm execution, Big-O proof, and theoretical computation. • Skeleton Programming Project Mastery: Final targeted preparation on the pre-released Skeleton Program. Practicing timed section C and D exam tasks, writing custom subroutines, implementing new algorithms into the skeleton framework, and anticipating likely exam scenarios. • NEA Final Submission Deliverable: Incorporating educator and self-review feedback from the Term 2 draft submission. Refining testing evidence (screen captures, trace logs, client feedback) and evaluation against original system requirements. END OF TERM 3 DELIVERABLE: Formal submission of the FINAL VERSION of the NEA Programming Project for internal moderation and AQA submission.	Paper 1 Exampro Questions, Pseudocode Mastery, Algorithm Tracing, Skeleton Code Exam Tasks, Mark Scheme Analysis, Final NEA Project Submission.
SEQUENCING	Focuses heavily on Paper 1 preparedness alongside the final submission of the NEA project. Resolving all NEA moderation requirements by the end of Term 3 frees students to focus 100% of their remaining time on exam revision.	

Subject Curriculum Plan – KS5

HASMONEAN
אל תקרי בניך אלא בוניך

TERM	TOPIC & CORE LEARNING	KEY CONCEPTS
Spring (Term 4)	Paper 2 Exam Techniques & Synoptic Revision (Exampro) & Final Exam Preparation • Paper 2 Exam Techniques & Synoptic Revision: Intensive Paper 2 revision using Exampro question banks covering Units 3 (Data Representation), 4 (Hardware & Software), 5 (Computer Architecture), 6 (Communication & Consequences), 10 (The Internet), and 11 (Databases & SQL). • Targeted Exam Mastery: Practicing binary arithmetic, floating-point conversions, Boolean algebra simplification using De Morgan's laws, multi-table SQL query construction, network subnetting calculations, and structuring 12-mark extended prose essay responses using AO1 (Knowledge), AO2 (Application), and AO3 (Evaluation) criteria. Full timed mock exams, command word analysis (Explain, Discuss, Calculate, Compare), and eliminating common student misconceptions prior to final May/June examinations.	Paper 2 Exampro Questions, Floating-Point Calculations, Boolean Simplification, SQL Query Mastery, Subnetting Drills, 12-Mark Essay Structure, Synoptic Command Words, Exam Timing.
SEQUENCING	Completes full synoptic exam preparation for Paper 2 immediately prior to final examinations, cementing low-level architectural concepts, network protocols, mathematical data representations, and high-tariff evaluation essay techniques.	

TERM	TOPIC & CORE LEARNING	KEY CONCEPTS
Spring (Term 5)	AQA A-Level Computer Science 7517 Final Examinations & Synoptic Drop-in Clinics • Final Revision Drop-in Support: Targeted formula/concept refresher sessions, command word review, and final exam warm-ups. • Sitting Public Examinations: Sitting AQA A-Level Computer Science 7517 Paper 1 (On-Screen Skeleton Code & Computational Theory - 2hrs 30mins) and Paper 2 (Written Computing Concepts - 2hrs 30mins).	AQA A-Level Computer Science 7517 Examinations (Paper 1 & Paper 2), Synoptic Recall, Final Assessment.
SEQUENCING	Final assessment phase where students demonstrate mastery across all 12 specification units and practical programming skills built over the two-year course.	