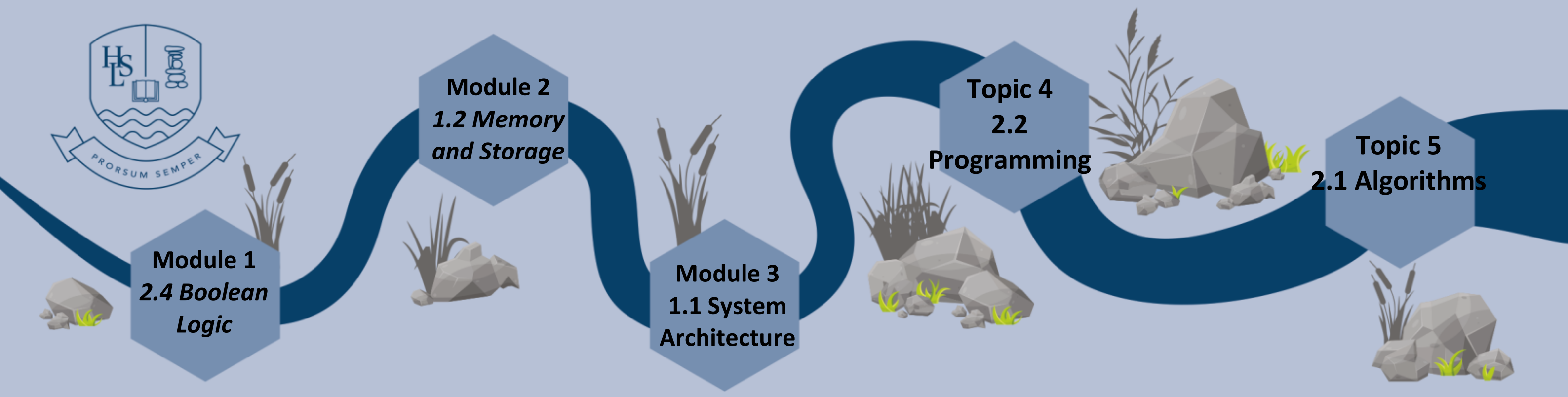
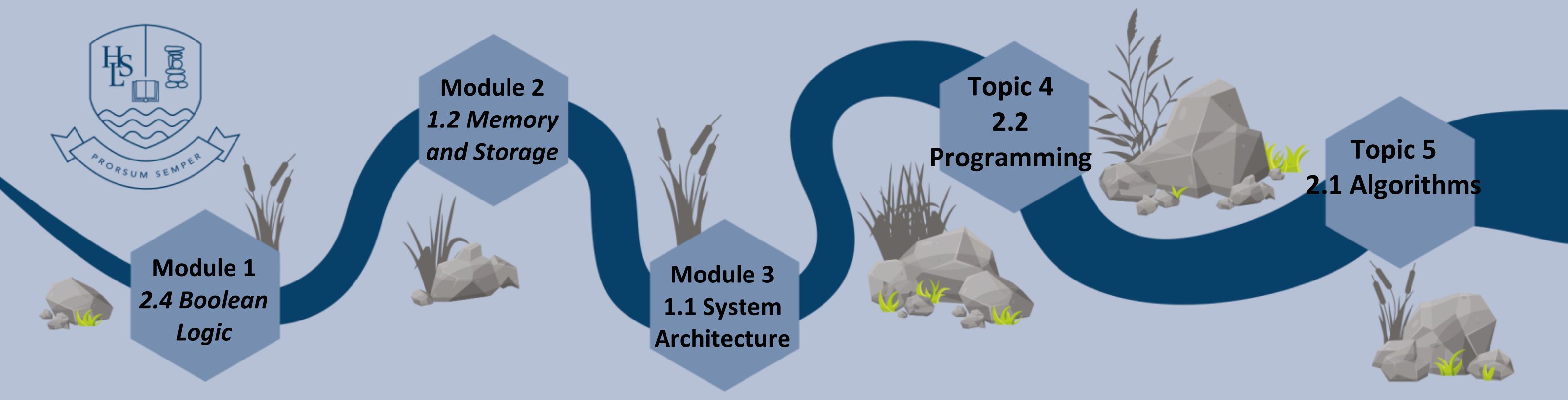


Subject: Computing	**Year Group: 10**	**Term: Autum**
Module/Theme:2.4 Boolean Logic		
Topic Outline & Aims (Intent) Students learn how computers make decisions using Boolean logic. They will construct truth tables, interpret logic gate diagrams, and solve problems using AND, OR, and NOT. This develops computational thinking, preparing them for programming and problem-solving tasks.		
Key Skills and Knowledge taught through this topic: (Intent) Create, complete, and edit truth tables and logic diagrams Recognise AND, OR, NOT gate symbols Apply logic to solve problems Work with multiple logic gates		
Prior Learning: (Context) KS3: Scratch/Python use of IF statements, simple conditions	**Future Learning:** (Context) KS4: Supports programming fundamentals and algorithms	**National Curriculum Links:** Design, use and evaluate computational abstractions Understand Boolean logic and its uses
RRSA Links: Article 28: Right to education (problem-solving and critical thinking) Article 29: Education must develop pupils' abilities fully	**Assessment of Learning:** (Impact) In-class quizzes (truth tables, diagrams) End-of-unit test with exam-style questions Retrieval practice in programming/algorithms	
British Values Links:		
Eco Schools Links:		

Reading / Enrichment:	**Key Vocabulary:** (Literacy)	**Numeracy Opportunities:**	**Career Links:**
Smart Revise – All pupils will be purchased a licence for this. This includes a huge bank of questions and tracks learning to show weak areas PMT Tutor – Great for nots on the course and past papers. **Isaac Computing** – Great for explanations and questions. GCP Textbook – All pupils will be given a copy at the start of year Craig n Dave YouTube videos – This series of videos explains everything in the course. Knowledge Organisers – The will be made available on the pupils Microsoft Teams page	Boolean, Logic gate, AND, OR, NOT, Truth table, Conjunction, Disjunction, Negation	Binary values (0/1) Truth table grids	Electronics engineer, Software developer, AI researcher



Module 1
2.4 Boolean Logic

Module 2
1.2 Memory and Storage

Module 3
1.1 System Architecture

Topic 4
2.2
Programming

Topic 5
2.1 Algorithms

Subject: Computing	**Year Group: 10**	**Term: Autum**

Module/Theme:1.2 Memory and Storage

Topic Outline & Aims (Intent)
Students explore how data is stored in computers using primary memory (RAM, ROM, cache, virtual memory), secondary storage (optical, magnetic, solid state), and units of data (bits, bytes, kilobytes, etc.).

Key Skills and Knowledge taught through this topic: (Intent)
Distinguish between RAM, ROM, cache, and virtual memory
Understand storage media types and pros/cons
Apply knowledge to exam scenarios
Convert between Binary – Denery – Hexadecimal
Use correct units (KB, MB, GB, etc.)

Prior Learning: (Context) KS3: Knowledge of binary data, simple storage concepts	**Future Learning:** (Context) KS4: Builds towards computer systems and networks	**National Curriculum Links:** Data representation, hardware, storage and retrieval

RRSA Links:
Article 17 (access to information in a digital age)

British Values Links:

Eco Schools Links:

Assessment of Learning: (Impact)
In-class quizzes (truth tables, diagrams)

End-of-unit test with exam-style questions

Retrieval practice in programming/algorithms

Reading / Enrichment:

Smart Revise – All pupils will be purchased a licence for this. This includes a huge bank of questions and tracks learning to show weak areas

PMT Tutor – Great for nots on the course and past papers.

Key Vocabulary: (Literacy)

Primary Storage:
- RAM (Random Access Memory)
- ROM (Read Only Memory)
- Cache
- Virtual memory
- Volatile / Non-volatile

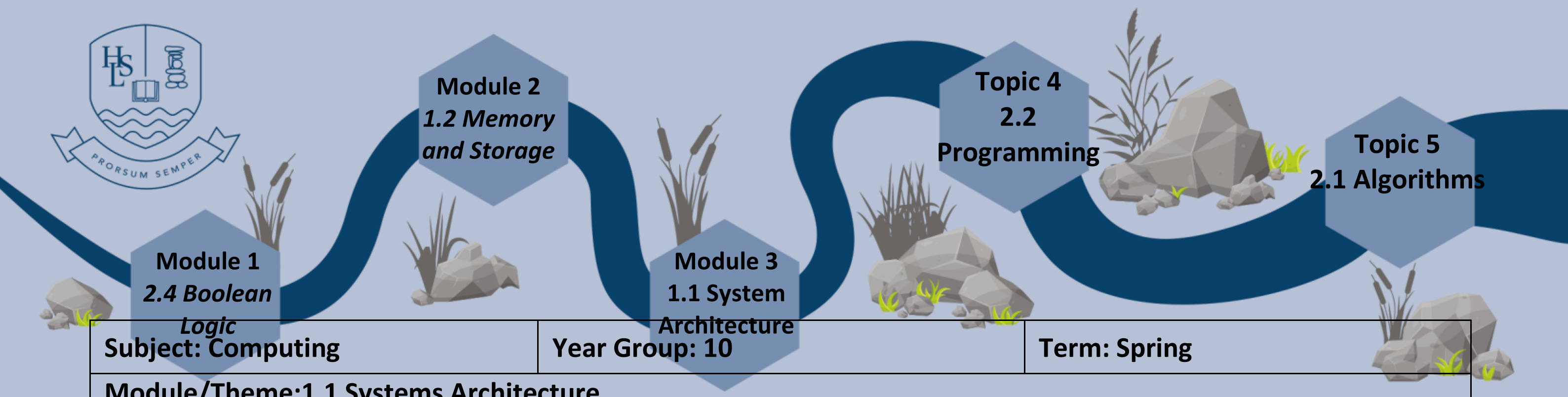
Secondary Storage:

Numeracy Opportunities:

Units of measurement, binary values, capacity comparisons

Career Links:
Cloud computing
Data centre technician
IT support

Isaac Computing – Great for explanations and questions. GCP Textbook – All pupils will be given a copy at the start of year Craig n Dave YouTube videos – This series of videos explains everything in the course. Knowledge Organisers – The will be made available on the pupils Microsoft Teams page	• Magnetic storage • Optical storage • Solid-state storage • Hard Disk Drive (HDD) • Solid State Drive (SSD) • Flash memory • Cloud storage **Units of Data:** • Bit • Nibble • Byte • Kilobyte (KB) • Megabyte (MB) • Gigabyte (GB) • Terabyte (TB) • Petabyte (PB) **Data Representation & Calculations:** • Binary • Denary • Hexadecimal • Binary shift • Overflow error • File size • Sample rate • Bit depth • Resolution • Colour depth • Metadata **Data Types & Files:** • ASCII • Unicode • Character set • Text file • Image file • Sound file **Compression:** • Lossy compression • Lossless compression • File size reduction **Performance & Application:** • Capacity • Speed • Portability • Durability • Reliability • Cost		



Module 1
2.4 Boolean Logic

Module 2
1.2 Memory and Storage

Module 3
1.1 System Architecture

Topic 4
2.2
Programming

Topic 5
2.1 Algorithms

Subject: Computing	**Year Group: 10**	**Term: Spring**

Module/Theme:1.1 Systems Architecture

Topic Outline & Aims (Intent)
Students investigate the CPU's role in computer systems, its components (ALU, CU, registers), the fetch-execute cycle, and CPU performance factors (cores, cache, clock speed). Embedded systems are also covered.

Key Skills and Knowledge taught through this topic: (Intent)
Label and describe CPU components
Explain the fetch-decode-execute cycle
Compare CPU performance factors
Identify and evaluate embedded systems

Prior Learning: (Context) KS3: Basic hardware components	**Future Learning:** (Context) KS4: Links to memory, storage, and software modules	**National Curriculum Links:** Understand how computer hardware and software work togethe

RRSA Links:
Article 28: Right to education (problem-solving and critical thinking)

Article 29: Education must develop pupils' abilities fully

British Values Links:

Eco Schools Links:
Embedded systems in energy-efficient devices

Assessment of Learning: (Impact)
In-class quizzes (truth tables, diagrams)

End-of-unit test with exam-style questions

Retrieval practice in programming/algorithms

Reading / Enrichment:

Smart Revise – All pupils will be purchased a licence for this. This includes a huge bank of questions and tracks learning to show weak areas

PMT Tutor – Great for nots on the course and past papers.

Key Vocabulary: (Literacy)

CPU Components

- CPU (Central Processing Unit)
- ALU (Arithmetic Logic Unit)
- CU (Control Unit)
- Cache
- Register
- MAR (Memory Address Register)
- MDR (Memory Data Register)

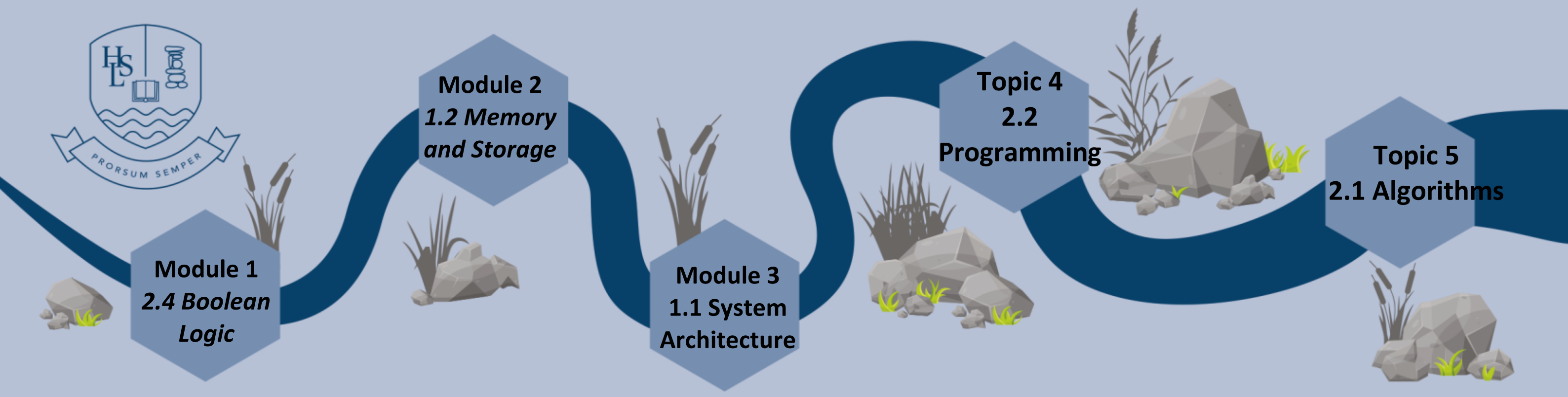
Numeracy Opportunities:

Clock speed (GHz), cache sizes, performance comparisons

Career Links:
Hardware engineer, Embedded systems developer

Isaac Computing – Great for explanations and questions. GCP Textbook – All pupils will be given a copy at the start of year Craig n Dave YouTube videos – This series of videos explains everything in the course. Knowledge Organisers – The will be made available on the pupils Microsoft Teams page	- PC (Program Counter) - Accumulator **Processes** - Fetch-Decode-Execute Cycle (FDE cycle) - Instruction - Operand - Opcode - Clock cycle - Instruction set - Bus (Data bus, Address bus, Control bus) **Performance Factors** - Clock speed (Hz, GHz) - Cache size - Number of cores - Pipelining - Parallel processing - Multitasking - Overclocking - Heat sink / Cooling **Architecture Models** - Von Neumann architecture - Harvard architecture (for contrast/extension) - Stored program concept **Memory & Storage Links** - Main memory - RAM - ROM - Virtual memory - Secondary storage **Embedded Systems** - Embedded system - Microcontroller - Firmware - Dedicated function - Real-time system - IoT (Internet of Things) devices **General Computing Concepts** - Input - Output - Processing - Storage - Hardware - Software		

PRORSUM SEMPER



Module 1
2.4 Boolean Logic

Module 2
1.2 Memory and Storage

Module 3
1.1 System Architecture

Topic 4
2.2
Programming

Topic 5
2.1 Algorithms

Subject: Computing	Year Group: 10	Term: Spring/Summer

Module/Theme:2.2 Programming Fundamentals

Topic Outline & Aims (Intent)
Students develop practical programming skills using a high-level language (e.g. Python). They use variables, constants, operators, input/output, assignment, sequence, selection, iteration, arithmetic and Boolean operators.

Key Skills and Knowledge taught through this topic: (Intent)

Core Constructs & Syntax

- Write programs using **sequence, selection, and iteration** (count-controlled and condition-controlled loops).
- Apply **arithmetic operators** (+, −, ×, ÷, MOD, DIV, ^).
- Apply **comparison operators** (==, !=, <, <=, >, >=).
- Use **Boolean operators** (AND, OR, NOT).
- Correctly implement **assignment statements** with variables and constants.

Inputs, Outputs & Data Types

- Use **input/output statements** to interact with the user.
- Store and manipulate data with appropriate **data types** (integer, real/float, Boolean, string, character).
- Apply **casting** to convert between data types.

Data Structures

- Create and use **1D and 2D arrays** to store and process collections of data.
- Access, update, and iterate through array elements.
- Represent **records** and structured data.

String & File Handling

- Apply **basic string manipulation** (concatenation, slicing, length, case conversion, substring).
- Perform **file handling operations**: open, read, write, close.
- Understand and use **end of file** conditions in loops.

Subprograms & Modular Code

- Define and call **functions and procedures**.
- Use **parameters** (passing by value/reference).
- Understand **local and global variables**.
- Organise code into **modular, reusable blocks** with clear structure, indentation, and naming conventions.

SQL & Data Processing

- Write simple **SQL queries** (SELECT, FROM, WHERE).
- Apply queries to extract meaningful data from sets.

Problem Solving & Program Design

- Analyse problems and break them down into manageable tasks.
- Plan solutions using **flowcharts, pseudocode, or structured English**.
- Translate designs into working code.
- Test programs systematically with normal, boundary, and invalid data.
- Debug errors (syntax and logic) and refine code.

Advanced Concepts (GCSE Extension)

- Apply **random number generation** in programs.
- Recognise and apply **robust design** principles: input validation, authentication, commenting.

- Understand the role of **Integrated Development Environments (IDEs)** in supporting coding.

Prior Learning: (Context)	**Future Learning:** (Context)	**National Curriculum Links:**
KS3: Scratch/Python use of IF statements, simple conditions	KS4: Builds towards robust programming and algorithms	Design, write, and debug programs; apply abstraction and decomposition

RRSA Links:
Article 28: Right to education (problem-solving and critical thinking)

Article 29: Education must develop pupils’ abilities fully

British Values Links:

Eco Schools Links:
Efficient programming reduces energy waste in real-world systems

Assessment of Learning: (Impact)
Mini coding challenges
Extended exam-style questions
Practical programming projects

Reading / Enrichment:

Time2Code – Great website for practicing python code

Smart Revise – All pupils will be purchased a licence for this. This includes a huge bank of questions and tracks learning to show weak areas

PMT Tutor – Great for nots on the course and past papers.

Isaac Computing – Great for explanations and questions.

GCP Textbook – All pupils will be given a copy at the start of year

Craig n Dave YouTube videos – This series of videos explains everything in the course.

Key Vocabulary: (Literacy)

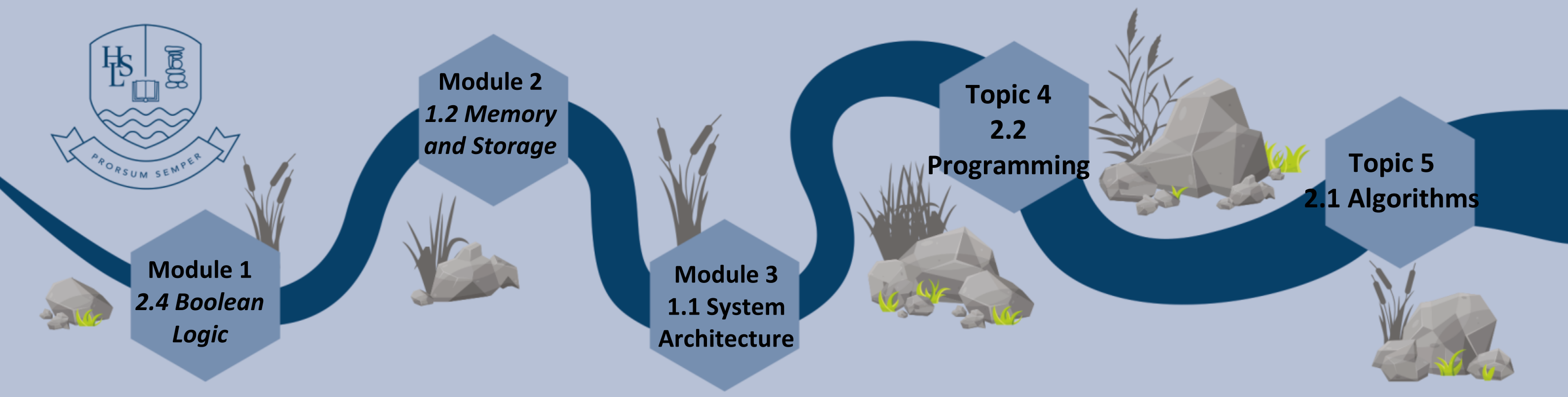
- Algorithm
- Sequence
- Selection
- Iteration
- Variable
- Constant
- Data type
- Casting
- Assignment
- Input/Output
- Arithmetic operator
- Comparison operator
- Boolean operator
- Array
- Index
- Function
- Procedure
- Parameter
- File handling
- Debugging

Numeracy Opportunities:

Operators (MOD, DIV, arithmetic), loops with counters

Career Links:
Software developer, Game designer, Data scientist

Knowledge Organisers – The will be made available on the pupils Microsoft Teams page			



Subject: Computing	**Year Group: 10**	**Term: Summer**
Module/Theme:2.1 Algorithms		
Topic Outline & Aims (Intent) Students learn computational thinking principles (abstraction, decomposition, algorithmic thinking) and how to design, trace, and refine algorithms. They study searching (linear, binary) and sorting algorithms (bubble, merge, insertion).		
Key Skills and Knowledge taught through this topic: (Intent) Apply computational thinking strategies Use pseudocode, flowcharts, trace tables Identify, apply, and evaluate searching/sorting algorithms Refine algorithms for efficiency		
Prior Learning: (Context) KS3:	**Future Learning:** (Context) KS4: Builds upon Boolean logic and programming fundamentals	**National Curriculum Links:** Apply computational thinking, design algorithms, evaluate solutions

RRSA Links: Article 28: Right to education (problem-solving and critical thinking) Article 29: Education must develop pupils’ abilities fully	**Assessment of Learning:** (Impact) Algorithm design tasks Pseudocode challenges Exam-style questions
British Values Links:	
Eco Schools Links: Algorithms in optimising energy/resource use	

Reading / Enrichment: Smart Revise – All pupils will be purchased a licence for this. This includes a huge bank of questions and tracks learning to show weak areas	**Key Vocabulary:** (Literacy) Algorithm Abstraction Decomposition Algorithmic thinking Pseudocode Flowchart	**Numeracy Opportunities:** Algorithm efficiency (time/steps), data set manipulation	**Career Links:** Data analyst, Software engineer, AI/machine learning specialist

PMT Tutor – Great for nots on the course and past papers. **Isaac Computing** – Great for explanations and questions. GCP Textbook – All pupils will be given a copy at the start of year Craig n Dave YouTube videos – This series of videos explains everything in the course. Knowledge Organisers – The will be made available on the pupils Microsoft Teams page	**Keywords** Trace table Dry run Refinement Efficiency Linear search Binary search Bubble sort Merge sort Insertion sort Input Process Output Iteration Selection		