EMMANUEL COLLEGE

CORE KNOWLEDGE

What I will know and understand by the end of Year 10.

	This year in Computer Science – Computer Systems we will be learning:	This links to:	Key Vocabulary:	
1	**Memory and Storage** • The units of data storage • How numbers are representing in a computer system and in Computer Science • Converting between number systems and binary addition and shifts • How characters, images, sound and instructions are represented in computer systems • How to estimate the size of character, raster image and sound files • Compression • Check digits • Primary Storage (Memory): the need for primary storage, the difference between RAM and ROM, the purpose of ROM in a computer system, the purpose of RAM in a computer system, virtual memory • Secondar Storage: the need for secondary storage, common types of storage: optical, magnetic, solid state, suitable storage devices and storage media for a given application, the advantages and disadvantages of different storage devices and storage media relating to capacity, speed, portability, durability, reliability and cost	You will build and develop your understanding of how numbers, characters, images and sound are represented and transferred. You will also extend your understanding of different types of memory and storage hardware, covered in the Computer Systems unit in Year 9. All data is represented and processed as electrical pulses hence this unit links to both the previous Boolean Logic unit and the systems architecture unit where we also consider memories role in the fetch decode execute cycle.	• Binary • Bit • Byte • Nibble • Hexadecimal	• Memory • Primary Storage • Storage • Volatile • Persistent • Capacity • Speed • Portability • Durability • Reliability

Target Grade:		AP1:		AP2:		AP3:	

EMMANUEL COLLEGE

CORE KNOWLEDGE

What I will know and understand by the end of Year 10.

	This year in Computer Science – Computational Thinking we will be learning:	This links to:	Key Vocabulary:	
1	**Boolean Logic** • Simple logic diagrams using the operators AND, OR and NOT • Truth tables • Combining Boolean operators using AND, OR and NOT • Applying logical operators in truth tables to solve problems	The work you covered in Year 9 on the CPU, logic gates and circuits are used to carry out the operations carried out by the CPU in the fetch execute cycle.. Logic gates and circuits are what perform the operations within all instructions executed by the CPU, therefore the information in this unit is foundation to looking at the components and operation of the CPU: the systems architecture unit.	• Boolean • Logic Gate • Logic Circuit • Truth Table	• Input • Output • Combined
2	**Programming Fundamentals** • The use of variables, constants, operators, inputs, outputs and assignments • The use of the three basic programming constructs used to control the flow of a program: sequence, selection, iteration (count- and condition-controlled loops) • The common arithmetic operators • The common Boolean operators AND, OR and NOT • The use of data types: integer, real, Boolean, character and string, casting • The use of basic string manipulation • The use of basic file handling operations: open, read, write, close • The use of records to store data • The use of SQL to search for data • The use of arrays (or equivalent) when solving problems, including both one-dimensional (1D) and two-dimensional arrays (2D) • How to use sub programs (functions and procedures) to produce • structured code • Random number generation	You will build on your programming skills, developing during the Year 8 and 9 programming units, advancing your knowledge of programming concepts and applying them to produce more complex Python programs. This unit gives practical experience of applying computational thinking algorithm design skills and hence links closely with the Algorithms unit.	• Variable • Constanta • Assign • Identifier • Integer • Real • Boolean • Character • String • Sequence • Condition • Outcome	• Selection • Iteration • Loop • Subprogram • Function • Procedure • Definition • Call • Record • Table • Field
3	**Algorithms: Computational Thinking & Designing, Creating and Refining Algorithms** • Principles of computational thinking: abstraction, decomposition, algorithmic thinking • Identify the inputs, processes, and outputs for a problem • Structure diagrams • Create, interpret, correct, complete, and refine algorithms using: pseudocode, flowcharts , reference language/high-level programming language • Identify common errors • Trace tables	You will draw on skills from Year 7 , 8 and 9 to understand how programmers analyse complex problems and design and test algorithms before starting to code them. You will meet several tools used to design and test such algorithms. This unit links closely to the programming fundamentals unit and gives the tools to develop the skills learnt in this unit in order to produce programs for more complex real-world scenarios,.	• Computational thinking • Decomposition • Abstraction • Algorithmic thinking	• Algorithm • Instruction • Trace • Dry run
4	**Programming languages and Integrated Development Environments** • Characteristics and purpose of different levels of programming language: high-level languages, low-level languages • The purpose of translators • The characteristics of a compiler and an interpreter • Common tools and facilities available in an Integrated • Development Environment (IDE): editors, error diagnostics, run-time environment, translators	You will build on knowledge acquired in the data representation sub unit and programming unit to consider how what programmers write is converted to a form that the computer can store and execute.	• Translate • Execute • Source code • Executable code • Machine code	• Assembler • Interpreter • Compiler
5	**Producing Robust Programs; Testing** • The purpose of testing • Types of testing: iterative, final/terminal • Identify syntax and logic errors • Selecting and using suitable test data: normal, boundary, invalid/erroneous • Refining algorithms	You will build on your algorithm design and programming skills to ensure appropriate devices are included to withstand inappropriate use and to ensure that programs are tested to identify and remove errors. . This unit links closely to the programming fundamentals unit and gives the tools to develop the skills learnt in this unit in order to produce robust error free programs for more complex real-world scenarios,.	• Iterative • Terminal • Syntax error • Logic error • Test data	• Normal data • Boundary data • Invalid data • Erroneous data

Target Grade:		AP1:		AP2:		AP3:	