Computer Science Rationale

We aim to equip students to use computational thinking and creativity to understand and change the world. The core of computing is computer science, in which pupils are taught the principles of information and computation, how digital systems work, and how to put this knowledge to use through programming. Armed with this knowledge and understanding, students are equipped to use information technology to create programs, systems and a range of content. Computing also ensures that students become digitally literate: able to use and express themselves and develop their ideas through information and communication technology at a level suitable for the future workplace and as active participants in a digital world.

Computer Science GCSE Aims

OCR GCSE Computer Science aims to:

- Offer pupils a qualification that builds on the knowledge, understanding and skills established through the Computer Science elements of the Key Stage 3 programme of study
- enable learners to develop computational thinking skills built on a sound base of conceptual learning and understanding
- Enthuse and engage learners through the practical application of computational theory

OCR's GCSE (9–1) in Computer Science will encourage learners to:

• understand and apply the fundamental principles and concepts of Computer Science, including abstraction, decomposition, logic, algorithms, and data representation
• analyse problems in computational terms through practical experience of solving such problems, including designing, writing and debugging programs
• think creatively, innovatively, analytically, logically and critically
• understand the components that make up digital systems, and how they communicate with one another and with other systems
• understand the impacts of digital technology to the individual and to wider society
• apply mathematical skills relevant to Computer Science.

YEAR 10	Content	Skills	Rationale
Autumn Half Term 1	**Systems architecture, memory and storage** **At the end of this Unit all students should be able to:** · Understand the purpose of the CPU · Explain the role and operation of the following CPU registers used in Von Neumann architecture · Explain the function of the CPU as fetch and execute instructions stored in memory · Describe how common characteristics of CPUs affect their performance: clock speed, cache size, number of cores · Explain the purpose and give examples of embedded systems · Describe the difference between RAM and ROM	Gain experience in exam technique for answering difficult questions relating to describing common CPU components and their function: ALU (Arithmetic Logic Unit), CU (Control Unit), Cache Using key terminology accurately and with understanding: o MAR (Memory Address Register), o MDR (Memory Data Register), o Program Counter, o Accumulator	Component 1.1, 1.2, 1.3 **1.1 Systems architecture** **1.2 memory and** **1.3 storage** No prior knowledge is essential with this unit. However students will benefit from a basic understanding of Computer Hardware

	· Describe the purpose RAM and ROM in a computer system · Explain the need for virtual memory · Describe flash memory · Discuss the need for secondary storage including optical, magnetic and solid state storage · Discuss data capacity of storage devices · Calculate data capacity requirements · Evaluate suitable storage devices and storage media for a given application using the following characteristics: capacity, speed, portability, durability, reliability, cost		delivered as part of the Key Stage 3 National Curriculum. It is a good place to start year 10 as the challenging concepts will emphasise to the students the "step-up" required for GCSE study.
Autumn Half Term 2 Spring Half Term1	**Programming techniques OCR board set programming project FOR 2020** **At the end of this Unit all students should be able to:** · identify and use variable types integer, real, Boolean, character and string · identify variables and constants in a program · use meaningful identifier names and know why it is important to use them · use arithmetic operations including mod and div · use Boolean operators in pseudocode solutions · show the results of basic string manipulation functions · use random number generation · follow through pseudocode solutions to simple problems involving sequence, selection and iteration · explain why functions and procedures are used in creating solutions to problems · use simple functions and procedures that return values to the calling program **Most students will be able to:** · write pseudocode solutions to simple problems involving sequence, selection and iteration · use nested selection and iteration statements · use Boolean operations NOT, AND and OR within conditions for iterative and selection structures · use basic string manipulation functions in pseudocode solutions · give examples of data structures:	**To cover the following 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: 1. sequence 2. selection 3. iteration (count and condition controlled loops) Also: the use of basic string manipulation the use of records to store data the use of arrays including both one and two dimensional arrays Importantly, students will be introduced to sub programs (functions and procedures) to produce structured code.	Component 2.2 and the practical programming project (NEA) are to run concurrently throughout Autumn 2 and Spring 1 It works well to teach the programming theory distinctly in some lessons, alongside students working independently in other lessons. For example if they all come to a problem that requires more knowledge about 2-D lists, then more teaching time can be devoted to that, before they return to their NEA.

	arrays and records · use one-dimensional arrays in the design of solutions to simple problems · write simple functions and procedures using parameters · read from and write to a text file **Some students will be able to:** · explain what is meant by a data structure and why these are used · use two-dimensional arrays in the design of solutions to simple problems · explain why it is good practice to use local variables		
Spring Half Term 2	**Wired and wireless networks** **Network topologies, protocols and layers** **At the end of this Unit all students should be able to:** • explain the advantages of networking stand-alone computers into a local area network • explain the difference between a client-server and a peer-to-peer network • describe, using diagrams or otherwise, the star and mesh network topologies and the differences between a local area network and a wide area network • describe the nature of the Internet as a worldwide collection of computer networks • state the advantages of different transmission media • explain the terms IP addressing, MAC addressing, packet and protocols • describe network policies such as acceptable use, disaster recovery, backup and archiving • describe the hardware needed to connect to the Internet including routers and switches • explain the need for IP addressing of resources on the Internet and how this can be facilitated by the role of DNS servers • explain the concept of encryption, giving examples • describe the concept of hosting	Gain experience in exam technique for answering difficult questions relating to wired and wireless networks Using key terminology accurately and with understanding: LAN, WAN, topology, star, mesh, peer-to-peer, client-server, hub, switch, router, wireless access point, NIC, MAC address, packet, protocol, layer, encryption, hosting, Cloud, Ethernet, frequency, channels, WAP. Internet, broadband, www, http, https, FTP, POP, IMAP, SMTP, TCP, IP addressing, domain name, DNS server	Component 1.4, 1.5 1.4 Wired and wireless networks 1.5 Network topologies, protocols and layers Students will have had some exposure to the basic concepts of computer networks at KS3, through the use of the school's local area network and their experience of using the Internet and connecting phones and tablets to WiFi at home and out-and-about. The aim of having this unit early on in Year 10 is to continue the transition to the academic side of the subject. Higher ability students will embrace the difficult concepts, while other students will definitely need to

	and Cloud services • describe the different layers in the TCP/IP protocol stack and the protocols used at each stage • explain the advantages of layering in this context • explain how Wi-Fi frequencies and channels affect connectivity and transmission • describe the different layers in the TCP/IP protocol stack and the protocols used at each stage • explain the advantages of layering in this context • explain how Wi-Fi frequencies and channels affect connectivity and transmission		re-visit the topic more than once.
Summer Half Term 1	**System Security AND** **System Software** **At the end of this Unit all students should be able to:** · list some of the threats posed to networks, including malware and phishing · explain briefly what is meant by phishing and how to keep data safe from phishing attacks · list precautions which can be taken to keep data safe from hackers including anti-malware software, firewalls, user access levels, passwords and encryption · list the functions of an operating system: user interface, memory management, multi-tasking, peripheral management, user and file management · explain briefly what is meant by memory management and multi-taski· describe briefly the purpose of encryption, defragmentation and data compression software · describe different types of user interface **Most students will be able to:** · describe briefly threats posed to networks including brute force attacks, denial of service attacks, data interception	Vocabulary malware, phishing, brute force attack, denial of service attack, data interception, SQL injection, network policy, penetration testing, network forensics, firewall, user access level operating system, user interface, memory management, multi-tasking, peripheral management, interrupt, defragmentation, data compression, symmetric encryption, asymmetric encryption, private key, public key, cypher text, plaintext, full backup, incremental backup	Component 1.6 and 1.7 1.6 System security 1.7 Systems software

	and theft, poor network policy · describe ways of identifying and preventing network vulnerabilities, including the use of passwords, encryption, penetration testing, network forensics and network policies · explain what is meant by a social engineering attack and give examples · explain what is meant by a Denial of Service attack and brute force attack		
Summer Half Term 2	**At the end of this Unit all students should be able to:** · List some ethical, legal, cultural or environmental issues in relation to a given scenario · List some privacy issues in relation to a given scenario · Choose from a given list, which Act is relevant to a particular scenario · List one attribute and advantage of open source software and proprietary software Most students will be able to: · Describe some ethical, legal, cultural and/or environmental issues in relation to a given scenario · Describe some privacy issues in relation to a given scenario · Describe the differences between open source and proprietary software and give advantages of each **Some students will be able to:** · List the clauses of the Data Protection Act and Computer Misuse Act and give examples of situations in which they are relevant · Evaluate the impact of and issues related to the use of computers in society		Component 1.8 1.8 Ethical, legal, cultural and environmental concerns

YEAR 11 Autumn Term 1	**Algorithms** **At the end of this Unit all students should be able to:** · state what is meant by an algorithm · state what is meant by abstraction · state what is meant by decomposition · state the sequence in which items in a sorted list will be examined in a linear and binary search · state the advantages and disadvantages of a linear and binary search · state an advantage of the merge sort and insertion sort over the bubble sort · show the state of a list after the first pass in a bubble sort · use a flowchart or pseudocode to define the steps in a simple algorithm · trace through a simple flow diagram or pseudocode algorithm to determine the output **Most students will be able to:** · explain how abstraction is used in a given scenario · explain how decomposition may be used in an algorithm for a given problem · explain how a binary search works · explain how a bubble sort works · show the state of a list at a given point in a bubble sort, merge sort or insertion sort · interpret, correct or complete a short algorithm **Some students will be able to:** · use pseudocode to define the steps in a complex algorithm · explain how a merge sort and an insertion sort work · correct or complete a complex algorithm	Vocabulary associated with this Unit, such as: abstraction, decomposition, algorithm, binary search, linear search, bubble sort, merge sort, insertion sort, pseudocode, flow diagram, trace table	Component 2.1 (2.2 covered Y10) 2.1 Algorithms Some of these concepts will have been introduced during programming and the completing the programming project in year 10. Approaching these topics from a theoretical perspective will be enhanced by having this practical experience.

Autumn Term 2	**Programs, Translators and Languages** **t the end of this Unit all students should be able to:** · Describe some simple validation checks that can be applied to data · Select test data that covers normal (typical), boundary (extreme) and erroneous data · Give examples of high-level and low-level languages · Give advantages of high-level languages over low-level languages · Explain the differences between a compiler, interpreter and assembler **Most students will be able to:** · Detect and correct errors in simple algorithms · Be able to justify the choice of test data · Give examples and reasons of when it might be appropriate to use a low-level language · Give examples of when it would be appropriate to use a compiler and interpreter **Some students will be able to:** · Write more complex authentication routines · Write robust programs that apply checks to data entered by the user	Vocabulary: compiler, interpreter, assembler, high level language, low level language, assembly language, source code, object code, bytecode, machine code, machine independence. validation, verification, authentication, syntax errors, logic errors, runtime errors, , trace table, dry run, valid data, invalid data, boundary data.	Component 2.3 and 2.4 2.3 Producing robust programs 2.5 Translators and facilities of languages

Spring Term 1	**Computational Logic And data representation** **At the end of this Unit all students should be able to:** · Recognise standard symbols used to represent NOT, AND OR, NAND, NOR and XOR logic gates · Draw truth tables for the above logic gates · Complete a trace table to trace through a simple algorithm **Most students will be able to:** · Recognise a logic gate from its truth table · Draw a logic circuit to solve a given problem **Some students will be able to:** · Draw a logic circuit to implement a given written logic statement	Binary, logic gate, NOT, AND, OR, NAND, NOR, XOR, truth table, logic circuit, logic statement	omponent 2.4 and 2.6 2.4 Computational logic 2.6 Data representation Students will have encountered binary in KS3 and during year 9. Now at the end of the course is a perfect time to revisit it as it will make more sense in the grand scheme of things! Then to complete the course by looking at representation of text, sound and images will finish the specification content. phew!