Year 10/11 GCSE Computer Science Long Term Plan 2022-24

Vision statement

The discipline of Computer Science incorporates the required techniques and methods for solving problems and advancing knowledge, which requires a distinct way of thinking and working. The role of programming within Computer Science is to enable a practical exposition of the required skills and methodologies. It provides motivation and a context within which ideas can be investigated and tested. Computation thinking is core to this process. Students are afforded the opportunity to recognise aspects of computation in the world around us and apply tools and techniques from computing to both natural and artificial systems and processes. Computational thinking provides a powerful framework for studying computing and beyond that it allows students to tackle problems, breaking them into solvable elements and devising algorithms in order to solve them.

Building upon the knowledge and understanding gained, students are equipped to create programs, systems and a widening range of content. Computing ensures that students become digitally literate and are able to express themselves and develop ideas using information and communication technology. This will enable them to become confident and active participants in an increasingly technologically based working world.

Domains of knowledge	**Key concept**
1.1 Systems architecture 1.2 Memory and storage 1.3 Computer networks, connections and protocols 1.4 Network security 1.5 Systems software 1.6 Ethical, legal, cultural and environmental impacts of digital technology 2.1 Algorithms 2.2 Programming fundamentals 2.3 Producing robust programs 2.4 Boolean logic 2.5 Programming languages and Integrated Development Environments	**Unit 1** Understand the components that make up digital systems, and how they communicate with one another and with other systems Identify vulnerabilities to computer systems and how they can be prevented Understand the purpose and functionality of operating systems/ utility software Understand the impact technology can have on wider society including, ethical, legal, cultural and environmental issues. **Unit 2** Be able to apply the key principles of computational thinking – abstraction, decomposition and algorithmic thinking Be able to design create and refine algorithms Understand the processes involved with different searching and sorting algorithms Be able to apply and understand a range of programming fundamentals – such as selection and iteration Know how to produce robust programs through defensive design. Understand the characteristics of different levels of programming language, and how they are developed in an intergraded development environment.

Year 10

Autumn Term 1		Autumn Term 2		Spring Term 1	
Unit Title: System Architecture	Unit Length: 7 weeks	Unit Title: Memory and Storage	Unit Length : 7 Weeks	Unit Title: Algorithms and programming fundamentals	Unit Length: 7 Weeks
Domain: Key specialised knowledge associated with your subject 1.1 Systems architecture ● Architecture of the CPU ● Impacts on CPU performance ● Embedded Systems		**Domain: Key specialised knowledge associated with your subject** 1.2 Memory and storage ● Primary Storage methods ● Secondary storage methods ● Units of data storage ● Data storage methods ● Compression		**Domain: Key specialised knowledge associated with your subject** 2.1 Algorithms ● Computational thinking ● Designing, creating and refining algorithms ● Searching and sorting algorithms 2.2 Programming fundamentals ● Programming fundamentals ● Data types ● Additional programming techniques	
Key concepts: Taken from each domain The purpose of the CPU: The fetch-execute cycle Common CPU components and their function: ● ALU (Arithmetic Logic Unit) ● CU (Control Unit) ● Cache ● Registers Von Neumann architecture: ● MAR (Memory Address Register) ● MDR (Memory Data Register) ● Program Counter ● Accumulator		**Key concepts: Taken from each domain** 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 What is virtual memory 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 these characteristics:		**Key concepts: Taken from each domain** 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 Standard searching algorithms:	

How common characteristics of CPUs affect their performance: ● Clock speed ● Cache size ● Number of cores The purpose and characteristics of embedded systems Examples of embedded systems	● Capacity ● Speed ● Portability ● Durability ● Reliability ● Cost The units of data storage: ● Bit ● Nibble (4 bits) ● Byte (8 bits) ● Kilobyte (1,000 bytes or 1 KB) ● Megabyte (1,000 KB) ● Gigabyte (1,000 MB) ● Terabyte (1,000 GB) ● Petabyte (1,000 TB) How data needs to be converted into a binary format to be processed by a computer Data capacity and calculation of data capacity requirements How to convert positive denary whole numbers to binary numbers (up to and including 8 bits) and vice versa How to add two binary integers together (up to and including 8 bits) and explain overflow errors which may occur How to convert positive denary whole numbers into 2-digit hexadecimal numbers and vice versa How to convert binary integers to their hexadecimal equivalents and vice versa Binary shifts The need for compression Types of compression: ● Lossy ● Lossless	● Binary search ● Linear search Standard sorting algorithms: ● Bubble sort ● Merge sort ● Insertion sort 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
Relevant endpoints: What do you want students to know/demonstrate at the end of the unit?(Component Knowledge/Powerful Knowledge/Substantive/ Declarative) ✔ The purpose of the CPU and its components. A understanding of how these characteristics effect system performance ✔ The range of different embedded systems currently in society, what their typical characteristics	**Relevant endpoints: What do you want students to know/demonstrate at the end of the unit?(Component Knowledge/Powerful Knowledge/Substantive/ Declarative)** ✔ Why computers have primary storage, key characteristics of the components ✔ Why computers have secondary storage, the key characteristics of the storage mediums ✔ How computers use character sets to store information, and how they can convert between them ✔ What file compression is, and the techniques a user might employ	**Relevant endpoints: What do you want students to know/demonstrate at the end of the unit?(Component Knowledge/Powerful Knowledge/Substantive/ Declarative)** ✔ Understanding of the three main computational principles and how they are used to define and refine problems ✔ Complete, write or refine an algorithm ✔ Identify syntax/logic errors in code and suggest fixes ✔ Understanding and practical use of programming techniques in a high-level language within the classroom
Broken down and sequenced knowledge: ✔ What actions occur at each stage of the fetch-execute cycle ✔ The role/purpose of each component and what it manages, ✔ stores, or controls during the fetch-execute cycle ✔ The purpose of each register, what it stores (data or address) ✔ The difference between storing data and an address ✔ Understanding of each characteristic as listed ✔ The effects of changing any of the common characteristics on ✔ system performance, either individually or in combination	Broken down and sequenced knowledge: ✔ Why computers have primary storage ✔ How this usually consists of RAM and ROM ✔ Key characteristics of RAM and ROM ✔ Why virtual memory may be needed in a system ✔ How virtual memory works ✔ Transfer of data between RAM and HDD when RAM is filled ✔ Why computers have secondary storage ✔ Recognise a range of secondary storage devices/media	Broken down and sequenced knowledge (habit 4) ✔ Understanding of the three main computational principles and how they are used to define and refine problems ✔ Produce simple diagrams to show: The structure of a problem/ Subsections and their links to other subsections ✔ Complete, write or refine an algorithm using the techniques listed ✔ Identify syntax/logic errors in code and suggest fixes ✔ Create and use trace tables to follow an algorithm

✔ What embedded systems are ✔ Typical characteristics of embedded systems ✔ Familiarity with a range of different embedded systems	✔ Differences between each type of storage device/medium ✔ Compare advantages/disadvantages for each storage device ✔ Why data must be stored in binary format ✔ Familiarity with data units and moving between each ✔ Data storage devices have different fixed capacities ✔ Calculate required storage capacity for a given set of files ✔ Conversion between binary, denary and hexadecimal data sets ✔ Carry out a binary shift (both left and right) ✔ How characters are represented in binary ✔ How the number of characters stored is limited by the bits ✔ available ✔ The differences between and impact of each character set ✔ Understand how character sets are logically ordered, e.g. the code for 'B' will be one more than the code for 'A' ✔ Binary representation of ASCII in the exam will use 8 bits ✔ Common scenarios where compression may be needed ✔ Advantages and disadvantages of each type of compression ✔ Effects on the file for each type of compression	✔ Understand the main steps of each specified algorithm and apply them to a data set ✔ Understanding and practical use of programming techniques in a high-level language within the classroom ✔ Recognise and use logical operators ✔ Practical use of the data types in a high-level language within the classroom ✔ Ability to choose suitable data types for data in a given scenario ✔ Understand that data types may be temporarily changed through casting, and where this may be useful ✔ Practical use of the additional programming techniques in a high-level language within the classroom ✔ Ability to manipulate strings, including: Concatenation and slicing ✔ Use of 2D arrays to emulate database tables of a collection of fields, and records ✔ The use of functions ✔ The use of procedures ✔ Where to use functions and procedures effectively ✔ The use of local/ global variables within functions and procedures
Formal Formative assessment End of topic assessment Observation Live marking/ Multiple Choice Questions	**Formal Formative assessment** End of topic assessment Observation Live marking/ Multiple Choice Questions	**Formal Formative assessment** End of topic assessment Observation Live marking/ Multiple Choice Questions
Summative assessment	Summative assessment	Summative assessment

Year 10

Spring Term 2		Summer Term 1		Summer Term 2	
Unit Title: Algorithms and programming fundamentals	Unit Length: 5 Weeks	Unit Title: Computer networks, connections and protocols	Unit Length: 6 Weeks	Unit Title: Network security and system software	Unit Length : 7 Weeks
Domain: Key specialised knowledge associated with your subject 2.1 Algorithms ● Computational thinking ● Designing, creating and refining algorithms ● Searching and sorting algorithms 2.2 Programming fundamentals ● Programming fundamentals ● Data types ● Additional programming techniques		**Domain: Key specialised knowledge associated with your subject** 1.3 Computer networks, connections and protocols ● Computer networks and topologies ● Wired and wireless networks, protocols and layers		**Domain: Key specialised knowledge associated with your subject** 1.4 Network security ● Threats to computer systems and networks ● Identifying and preventing vulnerabilities 1.5 Systems software ● Operating systems ● Utility software	
Key concepts: Taken from each domain Principles of computational thinking: ● Abstraction ● Decomposition ● Algorithmic thinking Identify the inputs, processes, and outputs for a problem		**Key concepts: Taken from each domain** Types of network: ● LAN (Local Area Network) ● WAN (Wide Area Network) Factors that affect the performance of networks The different roles of computers in a client-server and a peer-to peer network ¨		**Key concepts: Taken from each domain** Forms of attack: ● Malware ● Social engineering, e.g. phishing, people as the 'weak point' ● Brute-force attacks ● Denial of service attacks ● Data interception and theft	

Structure diagrams Create, interpret, correct, complete, and refine algorithms using: ● Pseudocode ● Flowcharts ● Reference language/high-level programming language Identify common errors Trace tables Standard searching algorithms: ● Binary search ● Linear search Standard sorting algorithms: ● Bubble sort ● Merge sort ● Insertion sort 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	The hardware needed to connect stand-alone computers into a Local Area Network: ● Wireless access points ● Routers ● Switches ● NIC (Network Interface Controller/Card) ● Transmission media The Internet as a worldwide collection of computer networks: ● DNS (Domain Name Server) ● Hosting ● The Cloud ● Web servers and clients Star and Mesh network topologies Modes of connection: ● Wired - Ethernet ● Wireless - Wi-Fi, Bluetooth Encryption IP addressing and MAC addressing Standards Common protocols including: ● TCP/IP (Transmission Control Protocol/Internet Protocol) ● HTTP (Hyper Text Transfer Protocol) ● HTTPS (Hyper Text Transfer Protocol Secure) ● FTP (File Transfer Protocol) ● POP (Post Office Protocol) ● IMAP (Internet Message Access Protocol) ● SMTP (Simple Mail Transfer Protocol) The concept of layers	● The concept of SQL injection Common prevention methods: ● Penetration testing ● Anti-malware software ● Firewalls ● User access levels ● Passwords ● Encryption ● Physical security
Relevant endpoints: What do you want students to know/demonstrate at the end of the unit?(Component Knowledge/Powerful Knowledge/Substantive/ Declarative) ✔ Understanding of the three main computational principles and how they are used to define and refine problems ✔ Complete, write or refine an algorithm ✔ Identify syntax/logic errors in code and suggest fixes ✔ Understanding and practical use of programming techniques in a high-level language within the classroom	**Relevant endpoints: What do you want students to know/demonstrate at the end of the unit?(Component Knowledge/Powerful Knowledge/Substantive/ Declarative)** ✔ Sound understanding of the characteristics of different between types, the hardware used and the methods of communication they use ✔ Principles of network security.	**Relevant endpoints: What do you want students to know/demonstrate at the end of the unit?(Component Knowledge/Powerful Knowledge/Substantive/ Declarative)** ✔ Knowledge/principles of different threats posed to devices/systems ✔ Knowledge/principles of each prevention method: What each prevention method may limit/prevent and How it limits the attack ✔ Functions of an operating system
Broken down and sequenced knowledge	Broken down and sequenced knowledge	Broken down and sequenced knowledge

✔ Understanding of the three main computational principles and how they are used to define and refine problems ✔ Produce simple diagrams to show: The structure of a problem/ Subsections and their links to other subsections ✔ Complete, write or refine an algorithm using the techniques listed ✔ Identify syntax/logic errors in code and suggest fixes ✔ Create and use trace tables to follow an algorithm ✔ Understand the main steps of each specified algorithm and apply them to a data set ✔ Understanding and practical use of programming techniques in a high-level language within the classroom ✔ Recognise and use logical operators ✔ Practical use of the data types in a high-level language within the classroom ✔ Ability to choose suitable data types for data in a given scenario ✔ Understand that data types may be temporarily changed through casting, and where this may be useful ✔ Practical use of the additional programming techniques in a high-level language within the classroom ✔ Ability to manipulate strings, including: Concatenation and slicing ✔ Use of 2D arrays to emulate database tables of a collection of fields, and records ✔ The use of functions ✔ The use of procedures ✔ Where to use functions and procedures effectively The use of local/ global variables within functions and procedures	✔ The characteristics of LANs and WANs including common ✔ examples of each ✔ Understanding of different factors that can affect the performance ✔ of a network, e.g.: Number of devices connected and Bandwidth ✔ The tasks performed by each piece of hardware ✔ The concept of the Internet as a network of computer networks ✔ A Domain Name Service (DNS) is made up of multiple Domain Name Servers ✔ A DNS's role in the conversion of a URL to an IP address ✔ Concept of servers providing services (e.g. Web server, Web ✔ pages, File server file storage/retrieval) ✔ Concept of clients requesting/using services from a server ✔ The Cloud: remote service provision (e.g. storage, software processing) ✔ Advantages and disadvantages of the Cloud ✔ Advantages and disadvantages of the Star and Mesh topologies ✔ Apply understanding of networks to a given scenario ✔ Compare benefits and drawbacks of wired versus wireless ✔ connection ✔ Recommend one or more connections for a given scenario ✔ The principle of encryption to secure data across network ✔ connections ✔ IP addressing and the format of an IP address (IPv4 and IPv6) ✔ A MAC address is assigned to devices; its use within a network ✔ The principle of a standard to provide rules for areas of computing Standards allows hardware/software to interact across different manufacturers/producers ✔ The principle of a (communication) protocol as a set of rules for transferring data ✔ That different types of protocols are used for different purposes ✔ The basic principles of each protocol i.e. its purpose and key ✔ features ✔ How layers are used in protocols, and the benefits of using layers; for a teaching example, please refer to the 4-layer TCP/IP model	✔ Threats posed to devices/systems ✔ Knowledge/principles of each form of attack including: How the attack is used and The purpose of the attack ✔ Understanding of how to limit the threats posed ✔ Understanding of methods to remove vulnerabilities ✔ Knowledge/principles of each prevention method: What each prevention method may limit/prevent and How it limits the attack ✔ What each function of an operating system does ✔ Features of a user interface ✔ Memory management, e.g. the transfer of data between memory, and how this allows for multitasking ✔ Understand that: Data is transferred between devices and the processor - This process needs to be managed ✔ User management functions, e.g: Allocation of an account, Access rights, Security, etc. ✔ File management, and the key features, e.g.: Naming, Allocating to folders, Moving files, Saving, etc.
Formal Formative assessment End of topic assessment Observation Live marking/ Multiple Choice Questions	**Formal Formative assessment** End of topic assessment Observation Live marking/ Multiple Choice Questions	**Formal Formative assessment** End of topic assessment Observation Live marking/ Multiple Choice Questions
Summative assessment	Summative assessment	Summative assessment

Year 11					
Autumn Term 1		Autumn Term 2		Spring Term 1	
Unit Title: Impacts of digital technology/ Boolean Logic	Unit Length: 7 Weeks	Unit Title: Programming languages and Integrated Development Environments	Unit Length : 7 Weeks	Unit Title: Producing robust programs	Unit Length: 7 Weeks

<table>
<tr>
<td>Domain: Key specialised knowledge associated with your subject
1.6 Ethical, legal, cultural and environmental impact

2.4 Boolean logic</td>
<td>Domain: Key specialised knowledge associated with your subject
2.2 Programming fundamentals
● Programming fundamentals
● Data types
● Additional programming techniques

2.5 Programming languages and Integrated Development Environments

● Languages
● The Integrated Development Environment (IDE)</td>
<td>Domain: Key specialised knowledge associated with your subject
2.3 Producing robust programs
● Defensive design
● Testing</td>
</tr>
<tr>
<td>Key concepts: Taken from each domain
Impacts of digital technology on wider society including:
● Ethical issues
● Legal issues
● Cultural issues
● Environmental issues
● Privacy issues
Legislation relevant to Computer Science:
● The Data Protection Act 2018
● Computer Misuse Act 1990
● Copyright Designs and Patents Act 1988
Software licences (i.e. open source and proprietary)
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</td>
<td>Key concepts: Taken from each domain
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
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</td>
<td>Key concepts: Taken from each domain
Defensive design considerations:
● Anticipating misuse
● Authentication
Input validation
Maintainability:
● Use of sub programs
● Naming conventions
● Indentation
● Commenting
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</td>
</tr>
<tr>
<td>Relevant endpoints: What do you want students to know/demonstrate at the end of the unit?(Component Knowledge/Powerful Knowledge/Substantive/ Declarative)

✔ Knowledge of a variety of ethical, legal, cultural, environmental and privacy issues digital technology has brought and how this impacts on society</td>
<td>Relevant endpoints: What do you want students to know/demonstrate at the end of the unit?(Component Knowledge/Powerful Knowledge/Substantive/ Declarative)

✔ Practical use of the techniques in a high-level language within the classroom
✔ The differences between high- and low-level programming languages, including the need for translators
✔ Knowledge of the tools that an IDE provides, how each of the tools and facilities listed can be used to help a programmer develop a program</td>
<td>Relevant endpoints: What do you want students to know/demonstrate at the end of the unit?(Component Knowledge/Powerful Knowledge/Substantive/ Declarative)

✔ Understanding of the issues a programmer should consider to ensure that a program caters for all likely input values</td>
</tr>
<tr>
<td>Broken down and sequenced knowledge:
✔ Technology introduces ethical, legal, cultural, environmental and privacy issues</td>
<td>Broken down and sequenced knowledge:
✔ Practical use of the techniques in a high-level language within the classroom</td>
<td>Broken down and sequenced knowledge (habit 4)</td>
</tr>
</table>

✔ Knowledge of a variety of examples of digital technology and how this impacts on society ✔ An ability to discuss the impact of technology based around the issues listed ✔ The purpose of each piece of legislation and the specific actions it allows or prohibits ✔ The need to licence software and the purpose of a software licence ✔ Features of open source (providing access to the source code and the ability to change the software) ✔ Features of proprietary (no access to the source code, purchased commonly as off-the-shelf) ✔ Recommend a type of licence for a given scenario including benefits and drawbacks ✔ Knowledge of the truth tables for each logic gate ✔ Recognition of each gate symbol ✔ Understanding of how to create, complete or edit logic diagrams and truth tables for given scenarios ✔ Ability to work with more than one gate in a logic diagram	✔ Understanding of each technique ✔ Recognise and use the logical operators: ✔ Practical use of the data types in a high-level language within the classroom ✔ Ability to choose suitable data types for data in a given scenario ✔ Understand that data types may be temporarily changed through casting, and where this may be useful ✔ Practical use of the additional programming techniques in a high-level language within the classroom ✔ Ability to manipulate strings, including: Concatenation, Slicing, Arrays as fixed length or static structures ✔ Use of 2D arrays to emulate database tables of a collection of fields, and records ✔ The use of functions ✔ The use of procedures ✔ Where to use functions and procedures effectively ✔ The use of the following within functions and procedures: local variables/constants, global variables/constants, arrays (passing and returning) ✔ SQL commands: SELECT, FROM, WHERE ✔ Be able to create and use random numbers in a program ✔ The differences between high- and low-level programming languages ✔ The need for translators ✔ The differences, benefits and drawbacks of using a compiler or an interpreter ✔ Knowledge of the tools that an IDE provides ✔ How each of the tools and facilities listed can be used to help a programmer develop a program	✔ Understanding of the issues a programmer should consider to ensure that a program caters for all likely input values ✔ Understanding of how to deal with invalid data in a program ✔ Authentication to confirm the identity of a user ✔ Practical experience of designing input validation and simple authentication (e.g. username and password) ✔ Understand why commenting is useful and apply this appropriately ✔ The difference between testing modules of a program during development and testing the program at the end of production ✔ Syntax errors as errors which break the grammatical rules of the programming language and stop it from being run/translated ✔ Logic errors as errors which produce unexpected output ✔ Normal test data as data which should be accepted by a program without causing errors ✔ Boundary test data as data of the correct type which is on the very edge of being valid ✔ Invalid test data as data of the correct data type which should be rejected by a computer system ✔ Erroneous test data as data of the incorrect data type which should be rejected by a computer system ✔ Ability to identify suitable test data for a given scenario ✔ Ability to create/complete a test plan
Formal Formative assessment End of topic assessment Observation Live marking/ Multiple Choice Questions Filling in blank knowledge organisers	**Formal Formative assessment** End of topic assessment Observation Live marking/ Multiple Choice Questions	**Formal Formative assessment** End of topic assessment Observation Live marking/ Multiple Choice Questions
Summative assessment	**Summative assessment** Mock Paper Unit 1 and Unit 2	**Summative assessment** Modified mock paper unit 1 and 2 (In class)

Year 11	
Spring Term 2	
Unit Title: Revisiting Prior Knowledge Based on question level analysis of the mock exams/ summative assessments	Unit Length: 5 weeks