Curriculum Plan – Computing Department

Called as God's family,
we strive to achieve our personal best,
by living and learning in Christ.

Department Mission Statement -:

Computing is the doorway to the future and we know how important it is to ensure our young people can live in the digital world as responsible young people whilst also understanding how computers work. Throughout the curriculum, pupils will learn how to use a variety of software from Excel through to Photoshop, as well as learning the fundamentals of programming. Staring with Block based programming to complete text-based programming. Each year develops on the previous material to give a comprehensive computing education.

Key Stage 2

Knowledge Gained	Skills Developed
Pupils Should Have……………….. • An understanding of what algorithms are • Know how to keep personal information private • Recognise common uses of information technology beyond school • Understand that algorithms are implemented as programs on digital devices • Understand that programs execute by following precise and unambiguous instructions • Know how to use technology respectfully • Identify where to go for help and support when they have concerns about content or who to contact on the internet or in person or other online technologies	Pupils Should Have……………….. • Used technology purposefully to create digital content (Word, PowerPoint etc.) • Created simple programs • Used technology purposefully to store digital content (save files) • Use technology purposefully to retrieve digital content (load files) • Debug simple programs • Use logical reasoning to predict the behaviour of simple programs • Use technology purposefully to organise digital content (folders) • Use technology purposefully to manipulate digital content (editing) • Use technology safely

Curriculum Plan – Computing Department

Key Stage 3 Knowledge and Skills Requirement (What knowledge and skills do pupils need to gain by the end of year 9?)

Knowledge To Be Built	Skills To Be Developed
• Understand several key algorithms that reflect computational thinking [for example ones for sorting and searching] • Learn two or more programming languages, at least one of which is textual, to solve a variety of computational problems; • Understand appropriate use of data structures [for example, lists, tables or arrays]; design and develop modular programs that use procedures or functions • Understand simple Boolean logic [for example, AND, OR and NOT] and some of its uses in programming; understand how numbers can be represented in binary • Understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems • Understand how instructions are stored and executed within a computer system; • Understand how data of various types (including text, sounds and pictures) can be represented and manipulated digitally, in the form of binary digits • Understand a range of ways to use technology safely, respectfully, responsibly and securely, including protecting their online identity and privacy; recognise inappropriate content, contact and conduct and know how to report concerns.	• Design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems • use logical reasoning to compare the utility of alternative algorithms for the same problem • create data structures that use procedure/functions • carry out simple operations on binary numbers [for example, binary addition, and conversion between binary and decimal] • Use technology responsibly • Identify a range of ways to report concerns about contact • undertake creative projects that involve selecting, using, and combining multiple applications, preferably across a range of devices, to achieve challenging goals including collecting and analysing data and meeting the needs of known users • Create, re-use, revise and re-purpose digital artefacts for a given audience, with attention to trustworthiness, design and usability • Write programs that accomplish specific goals • Use sequence in programs • Work with various forms of input • Work with various forms of output • Use search technologies effectively • Use a variety of software to accomplish given goals • Collect information • Design and create content • Present information

Computer Science KS3 – Sequencing of Content, Knowledge and Skills

This document is designed to show how content across KS3 Computer Science implements the National Curriculum for Computing (Sept 2013), this also shows how units of work are sequenced (CS1, CS2, CS3 etc.) so that knowledge and skills are built upon across the whole of the Key Stage.

Content is divided into three strands:

- **Computing (computational thinking)**
- **Digital Literacy (use of computers & software)**
- **Creative (designing & creating)**

	KS3 Curriculum Content																	
National Curriculum for Computing KS3 Content	**Year 7 1 Lesson/week**						**Year 8 1.5 Lessons/week**					**Year 9 2 Lessons/week**						
	CS1 – Introduction to Computing	**IT1 – Catholic Life Presentations**	**IT2 – Spreadsheet Modelling**	**CS2 – Game Programming Concepts in Scratch**	**IT3 – E-Safety**	**IT4 – Data Handling**	**CS3 – Understanding Computers**	**IT5 – Website Design**	**CS4 – Computational Thinking with Flowol**	**CS5 –Programming Basics in Python**	**CS6 – Computer Crime & Cyber Security**	**CS7 – Programming Concepts**	**CS8 – Micro:bit Programming**	**IT6 – Graphics Design**	**CS9 – Networks**	**IT7 – Sound Editing**	**IT8 – Games Development**	**CS iDEA**
Design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems			✓	✓		✓			✓				✓				✓	
Understand several key algorithms that reflect computational thinking [for example, ones for sorting and searching]; use logical reasoning to compare the utility of alternative algorithms for the same problem			✓							✓		✓						

Use two or more programming languages, at least one of which is textual, to solve a variety of computational problems; make appropriate use of data structures [for example, lists, tables or arrays]; design and develop modular programs that use procedures or functions				✓				✓		✓		✓	✓				✓	
Understand simple Boolean logic [for example, AND, OR and NOT] and some of its uses in circuits and programming; understand how numbers can be represented in binary, and be able to carry out simple operations on binary numbers [for example, binary addition, and conversion between binary and decimal]				✓			✓											
Understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems						✓	✓	✓	✓				✓		✓			
Understand how instructions are stored and executed within a computer system; understand how data of various types (including text, sounds and pictures) can be represented and manipulated digitally, in the form of binary digits							✓			✓		✓		✓		✓		
Undertake creative projects that involve selecting, using, and combining multiple applications, preferably across a range of devices, to achieve challenging goals, including collecting and analysing data and meeting the needs of known user			✓			✓		✓						✓		✓	✓	
Create, re-use, revise and re-purpose digital artefacts for a given audience, with attention to trustworthiness, design and usability		✓			✓								✓	✓		✓		

Understand a range of ways to use technology safely, respectfully, responsibly and securely, including protecting their online identity and privacy; recognise inappropriate content, contact and conduct and know how to report concerns.	✓				✓						✓							✓

Computer Science KS3 to KS4 – Transition & Sequencing of Content, Knowledge and Skills

This document is designed to show how content transfers across from KS3 to KS4 Computer Science GCSE & KS4 Vocational Courses and continues in its implementation of the National Curriculum for Computing (Sept 2013), this also shows how units of work are sequenced (CS1, CS2, CS3 etc.) so that knowledge and skills are built upon across the whole of the curriculum.

- **Year 7 – 1 Hour 15mins per week**
- **Year 8 – 2 Hours per week**
- **Year 9 – 2 Hours 30mins per week**
- **GCSE – 2 Hours 30mins per week**

	KS4 Curriculum Content																			
KS3 Content	GCSE Computer Science OCR J277									KS4 Vocational IT OCR Creative iMeda J817										
	Fundamentals of algorithms & Computational Thinking	Programming Techniques	Fundamentals of data representation	Systems architecture	Computer networks & components	Cyber security	Relational databases and SQL	Ethical, legal and environmental impacts	Translators, Languages and IDEs	Pre-production skills	Creating digital graphics	User interface designs	Audience needs and purpose	Design principles	Project planning techniques	Data processing and modelling	Modern technologies	Communication	Data threats and cyber security	The wider implications of digital systems
CS1 – Introduction to Computing																		✓		
IT1 – Catholic Life Presentations											✓		✓	✓	✓					
IT2 – Spreadsheet Modelling			✓													✓				

Curriculum Plan – Computing Department

CS2 – Game Programming Concepts in Scratch	✓	✓								✓		✓								
IT3 – E-Safety						✓		✓										✓		✓
IT4 – Data Handling							✓									✓				
CS3 – Understanding Computers			✓	✓													✓			
IT5 – Website Design		✓										✓		✓						
CS4 – Computational Thinking with Flowol	✓															✓				
CS5 –Programming Basics in Python	✓	✓							✓											
CS6 – Computer Crime & Cyber Security						✓		✓											✓	✓
CS7 – Programming Concepts	✓	✓							✓					✓						
CS8 – Micro:bit Programming	✓	✓		✓					✓							✓				
IT6 – Graphics Design			✓								✓		✓	✓						
CS9 – Networks					✓			✓										✓		
IT7 – Sound Editing			✓							✓			✓	✓						
IT8 – Games Development	✓	✓								✓		✓								
CS iDEA						✓		✓											✓	✓

Key Stage 4 Knowledge and Skills Requirement (What knowledge and skills do pupils need to gain by the end of year 11?)

Knowledge To Be Built	Skills To Be Developed
○ understand and apply the fundamental principles and concepts of Computer Science, including abstraction, decomposition, logic, algorithms, and data representation ○ Control or simulate physical systems ○ Understand how computer networks can provide multiple services, such as the world wide web ○ Appreciate how search results are selected ○ Understand the opportunities computer networks offer for communication ○ understand the impacts of digital technology to the individual and to wider society ○ Identify a range of ways to report concerns about content ○ Recognize acceptable / unacceptable behaviour ○ 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	○ Debug programs that accomplish specific goals ○ Design and create program ○ Design programs that accomplish specific goals ○ Select a variety of software to accomplish given goals ○ Select, use and combine internet services ○ Analyse information ○ Evaluate information ○ Collect data ○ Present data ○ Use logical reasoning to detect and correct errors in programs ○ Use repetition in programs ○ Calculate required storage capacity for a given set of files ○ Calculate file sizes of sound, images and text files ○ Conversion of any number of the following ranges (Denary, Hexadecimal, binary) to another number base ○ Carry out a binary shifting

Curriculum Plan – Computing Department

Key Stage 5 Knowledge and Skills Requirement (What knowledge and skills do pupils need to gain by the end of year 13?)

Knowledge to Be Built	Skills To Be Developed
• Use selection in programs • Work with variables • Use logical reasoning to explain how some simple algorithms work • Use logical reasoning to detect and correct errors in algorithms • Understand computer networks including the internet • Appreciate how search results are ranked • Understand the opportunities computer networks offer for collaboration • Be discerning in evaluating digital content • Model state of real-world problems • Understand simple Boolean logic • Understand how numbers can be represented in binary • Understand the hardware components that make up computer systems • Understand how text can be represented digitally in the form of binary digits • Understand how pictures can be represented digitally in the form of binary digits • Undertake creative projects with challenging goals • Use multiple applications • Understand a range of ways to use technology respectfully • Know how to report concerns • Understand a range of ways to use technology safely	• Solve problems by decomposing them into smaller parts • Combine a variety of software to accomplish given goals • Select use and combine software on a range of digital devices • Analyse data • Evaluate data • Design and create systems • Use computational abstractions • Use a programming language to solve computational problems • Work with applications across a range of devices • Collect data • Recognise inappropriate content • Recognise inappropriate contact • Recognise inappropriate conduct • Reuse digital artefacts for a given audience • Attend to usability of digital artefacts

Curriculum Plan				
Year Group	**Scheme of Work**	**Knowledge Gained (Including How It Builds on Previous Knowledge Gained)**	**Skills Developed (Including How It Builds on Previous Skills Gained)**	**Assessment of knowledge and skills**
7	Introduction to Computing at Hagley	• How to log into school network • How to use email, haggle etc. • Able to discuss some design choices. • Know the difference between save-As and save • Know how to use Copy, Cut, Paste effectively and able to explain the differences • Understand how to use peer assessment	• Used simple skills in the software and explained a few of them • Able to explain some of the skills & design choices you have used. • Logged onto School Network, Haggle, Email • Create folders with suitable names • Save files with a suitable name in the correct folder • Download files from Haggle and save them with a suitable name in the correct folder • Search for information using a search engine • Considered appropriate colour schemes • Search, download and insert an image into a document or presentation • Considered appropriate colour schemes •	Each Unit is graded using the Hagley grading system from SA – 4B at the end of each Unit and at least once during the Topic. A 'Teacher Comment' and 'Area for Improvement' is given at the end of the Unit. Students also evaluate their work and record progress, while responding to both verbal and written feedback from both teacher and peer assessment, this is marked on their self-assessment sheets. End of topic tests are set or final pieces of work are teacher assessed. Homework is set and marked where necessary with a positive comment and given an Area for Improvement.

	Digital Literacy	• Use a Search Engine efficiently. • Know how to check the reliability and trustworthiness of a website • Know how to avoid copyright issues • Identify malware and preventive measures • Demonstrate efficient searching criteria. • Describe different virus types and how they infect a system • Explain the terms malware and hacking • Know the key components of a computer system	• Use Boolean operators to search • Describe specific cyber attacks • correctly identify trustworthy and untrustworthy websites using set criteria. • explain why certain websites are more reliable than others. • describe a wide range of threats and security measures.	Each Unit is graded using the Hagley grading system from SA – 4B at the end of each Unit and at least once during the Topic. A 'Teacher Comment' and 'Area for Improvement' is given at the end of the Unit. Students also evaluate their work and record progress, while responding to both verbal and written feedback from both teacher and peer assessment, this is marked on their self-assessment sheets. End of topic tests are set or final pieces of work are teacher assessed. Homework is set and marked where necessary with a positive comment and given an Area for Improvement.
	Spreadsheet Modelling	• Understand how a spreadsheet works • Learn how to enter text, data and formulae in a spreadsheet • Know the purpose of using spreadsheets • recognise the difference between data, text and formulae in a computer model	• use a model to predict an outcome • explain how rules govern a model • obtain information from a model and check this for plausibility • import and export data in appropriate formats	Each Unit is graded using the Hagley grading system from SA – 4B at the end of each Unit and at least once during the Topic. A 'Teacher

		and organise these so that the model is fit for purpose • know the mathematical symbols +-*/= • recognise how to use header and footers • understand why using formulae is more efficient	• understand when to use IT to solve a problem • format cells by changing the appearance in at least 3 different ways • create graphs and charts that show the information clearly. • Merge cells & wrap text correctly • Used the functions SUM, AVERAGE, MAX, and MIN • Sort & filter data appropriately • Created a formula linking data from two worksheets	Comment' and 'Area for Improvement' is given at the end of the Unit. Students also evaluate their work and record progress, while responding to both verbal and written feedback from both teacher and peer assessment, this is marked on their self-assessment sheets. End of topic tests are set or final pieces of work are teacher assessed. Homework is set and marked where necessary with a positive comment and given an Area for Improvement.
	Game Programming Concepts (via Scratch)	• Understanding the Scratch interface • how to make simple programs using Scratch • key terms (e.g. variables, random numbers) • how to save Scratch projects • how to draw new sprites and animate these with costumes. • Editing existing sprites • use the built-in backgrounds in Scratch	• Create a simple script that animates the sprite automatically • Create a script that allows the user to control the movement of the sprite using the arrow keys • Combine the different scripts to work together on the same sprite • Make the sprite reverse direction • Edit the stage using the paint editor facility within Scratch • Make the sprite react to the background	Each Unit is graded using the Hagley grading system from SA – 4B at the end of each Unit and at least once during the Topic. A 'Teacher Comment' and 'Area for Improvement' is given at the end of the Unit. Students also evaluate their work and record progress, while

		• concept of Boolean logic by combining conditions to determine certain outcomes. • Knowing that a game has different outcomes according to conditions. • tool for drawing patterns • the concept of subroutines • The use of strings as variables • use of Boolean NOT	• Import a new sprite from the ones available within scratch • Create and edit new costumes for an existing sprite • Create a simple script that moves the sprite automatically • Create variables to set the score and the countdown • Set a sprite to start in a certain position • Save work • Delete, resize and draw new sprites • Animate a sprite using costumes • Control the movement of a sprite using arrow keys • Design a maze on the stage • Make sprites interact with the background by using colours • Make sprites start in a pre-set starting position using coordinates • Make objects disappear and reappear in a random position • Create variables to set up scoring in the game • Make an autonomous sprite chase the sprite controlled by the player • Import sprites • Edit costumes of sprites given in Scratch • Animate sprites using scripts • Control the movement of sprites using the arrow keys	responding to both verbal and written feedback from both teacher and peer assessment, this is marked on their self-assessment sheets. End of topic tests are set or final pieces of work are teacher assessed. Homework is set and marked where necessary with a positive comment and given an Area for Improvement.

			• Create scripts that move sprites around automatically • Create interaction between sprites • Use coordinates and random numbers to set start position of sprites • Use variables and scripts to allow scoring and countdown • Use backgrounds built into Scratch • Make autonomous sprites • The following key terms – variable, operator, If…Else…, Repeat…Until… • How to give the user instructions at the beginning of the game • How to use broadcast to start the game • Greater Than and Less Than operators • Boolean logic as it applies to combining conditions necessary for game outcomes • How to combine different conditions using If…Else… to govern the game outcome • How to animate a sprite using costumes • How to control the movement of a sprite using arrow keys • How to design a maze on the stage • How to make sprites interact with the background by using colours	

			- The range of coordinates available on the stage in Scratch - How to make sprites start in a pre-set starting position using coordinates - How to make objects disappear and reappear in a random position using coordinates	

	Online Safety	• Identify safe sharing of content/images online • Describe and use strategies to protect their content/images online • Understand risks of friending strangers online • Recognise when to seek help with online dilemmas • Describe and use strategies in situations where they feel uncomfortable or unsafe online • Recognise that people they meet online may pretend to be someone else • Identify what cyberbullying is • Describe situations where they feel cyberbullied • Recognise when to seek help in cyberbullying situations • Understand how to use social media safely • Describe rules for online safety • Justification of chat room rules	• Identify what personal information is safe to put online • How to seek assistance if things go wrong online • use strategies in situations where they feel cyberbullied • Be a positive bystander in a cyberbullying situation • use social media safely	Each Unit is graded using the Hagley grading system from SA – 4B at the end of each Unit and at least once during the Topic. A 'Teacher Comment' and 'Area for Improvement' is given at the end of the Unit. Students also evaluate their work and record progress, while responding to both verbal and written feedback from both teacher and peer assessment, this is marked on their self-assessment sheets. End of topic tests are set or final pieces of work are teacher assessed. Homework is set and marked where necessary with a positive comment and given an Area for Improvement.
8	Understanding computers	• Distinguish between hardware and software • Identify input, output and storage devices • Name at least five pieces of software • Understand what happens at the "Process" stage • Explain what main memory is used for	• Suggest appropriate input and output devices for a given scenario • Draw a block diagram of the main components of a computer: input, processor, output & storage • Name the three stages in the Fetch Execute Cycle	Each Unit is graded using the Hagley grading system from SA – 5C at the end of each Unit and at least once during the Topic. A 'Teacher Comment' and 'Area for

		• Distinguish between main memory and permanent storage devices • Define Hz, MHz and GHz and state how these relate to the speed of the processor • Understand the difference between RAM and ROM • State why all data is represented in binary in a computer • Understand that a particular bit pattern may represent, for example, an instruction to do something, a letter, a number or a tiny piece of a graphical image • Define a Bit, Byte, Kb, Mb and Gb • Look up from a table the bit pattern for a given character • State how many different characters can be represented using 8 bits • State the typical capacities, strengths and weaknesses of different storage devices • Describe how data is stored on a CD • Describe how 0s and 1s are represented by pits and lands on a CD • Name three types of optical storage device	• Convert integers to binary numbers • Convert binary numbers to integers • Give examples of alphanumeric characters & special symbols that can be represented in ASCII • Show that a bit pattern can represent either a character or a decimal number • Add two binary numbers (each less than 7 binary digits) • Multiply a binary number by 2 • Identify a binary number as being odd or even	Improvement' is given at the end of the Unit. Students also evaluate their work and record progress, while responding to both verbal and written feedback from both teacher and peer assessment, this is marked on their self-assessment sheets. End of topic tests are set or final pieces of work are teacher assessed. Homework is set and marked where necessary with a positive comment and given an Area for Improvement.
	Computational Thinking Algorithms - Flowol	• Know a what a control system is and identify them from everyday life. • Understand How and why these systems are used • Explain about the impact they have on our lives and employment. • Know what an algorithm is	• write an algorithm as text and develop a graphical coded solution • To identify flowchart symbols and their purpose. • Use simple flowcharts which use sensors to control a system.	Each Unit is graded using the Hagley grading system from SA – 5C at the end of each Unit and at least once during the Topic. A 'Teacher Comment' and 'Area for

		• Understand why algorithms are used in computing • exploring the use of sensors to gather input data & affect outputs. • Know the symbols needed for a flowchart. • Understand How sensors are used to automate control systems • Know That sensors & motors can be combined into control systems. • How a control system can operate motors & that this can be combined with inputs from buttons & sensors. • Know that control systems carry out boring, monotonous jobs that people often do not like doing.	• Use a flowchart to control a cot mobile & use inputs (buttons) to turn on/off motors & lights; use a sensor input to turn on/off a nightlight. • Create a flowchart to control a set of traffic lights. • Use a flowchart to control one side of a car park barrier which uses sensors and motors to control a barrier. • Add a 'counting' step to the flowchart to control one side of a car park barrier which uses sensors and motors to control a barrier.	Improvement' is given at the end of the Unit. Students also evaluate their work and record progress, while responding to both verbal and written feedback from both teacher and peer assessment, this is marked on their self-assessment sheets. End of topic tests are set or final pieces of work are teacher assessed. Homework is set and marked where necessary with a positive comment and given an Area for Improvement.
	Programming Basics - Python	• Learn what Python is and some of the applications it is used for • Understand what a syntax error is and how to interpret an error message • Know the rules for variable names and use variables in a program • Understand the use and value of comments in a program • Understand the importance of using correct data types string, integer, float • Understand how to use assignment statements correctly • Understand and apply the principle of a binary search	• Run a simple Python program in Interactive mode using the input and print functions • Write, save and run a program in Script mode • Use comments in code • Perform arithmetic using the BIDMAS rule • Use the int, float and round functions • Write a program involving input, calculation and output • Use a while loop in a program	Each Unit is graded using the Hagley grading system from SA – 5C at the end of each Unit and at least once during the Topic. A 'Teacher Comment' and 'Area for Improvement' is given at the end of the Unit. Students also evaluate their work and record progress, while responding to both verbal and written feedback

		• Compare the efficiency of a binary search with a linear search • Compare alternative algorithms for a given problem • Understand how a binary search works • Learn to write algorithms in pseudocode • Review the difference between syntax errors, run-time errors and logic errors • Learn techniques for debugging programs	• Use an if statement within a while loop • Use a function to generate a random number • Use a linear search to find a number • Use selection statements if, else and elif in a program • Use indentation correctly to define a block of code	from both teacher and peer assessment, this is marked on their self-assessment sheets. End of topic tests are set or final pieces of work are teacher assessed. Homework is set and marked where necessary with a positive comment and given an Area for Improvement.
	Computer crime & cyber security	• Identify common types of computer crime • Look at examples of computer crime on the Internet • Learn about different types of email scam • Recognise the signs of fraudulent emails • Learn about the Computer Misuse Act – which makes certain activities illegal • Look at examples of computer misuse • Understand what is meant by hacking • Understand what is meant by malware • Learn ways to protect yourself from malware and hacking • Be aware of who might hold personal data about you • Discuss the need for various organisations to hold data about you • Be aware of the possibility of identity theft	• Know how to minimise the chance of identity theft • Be aware of who might hold personal data about you • Know how to minimise the chance of identity theft • Learn about the need to dispose of computer equipment in a responsible manner	Each Unit is graded using the Hagley grading system from SA – 5C at the end of each Unit and at least once during the Topic. A 'Teacher Comment' and 'Area for Improvement' is given at the end of the Unit. Students also evaluate their work and record progress, while responding to both verbal and written feedback from both teacher and peer assessment, this is marked on their self-assessment sheets. End of topic tests are set or final pieces of work are teacher assessed.

		• Discuss the need for various organisations to hold data about you • Be aware of the possibility of identity theft • Learn about some of the common health and safety problems associated with computer use • Learn ways of avoiding these problems • Learn about Health and Safety law		Homework is set and marked where necessary with a positive comment and given an Area for Improvement.
	Web Design	• Assess the effectiveness of existing website and enhance understanding of what makes a good website. • Record, develop and share ideas using templates given. • Understand how to create a website by more than one method. • Use and recognise HTML code • Use diagrams to represent the structure of a website. • Decide what information needs to go on website to plan layout. • Develop a website using appropriate text, images and table making sure attention is made to purpose and audience. • Make and use simple success criteria that ensures fitness for purpose. • Gather and use feedback to inform future work. • Reflect on their previous work and learning to improve their work.	• Identify different purposes/uses of websites • Explain what makes a good website. • Identify key features in websites. • create a simple website adding appropriate content. • Recognise some common HTML codes and understand how they are used. • Create diagram to show web pages and links. • Plan structure for homepage. • Use web-authorising software to create basic web page structure considering using appropriate colours, text and images within websites. • Create and save pages of website, homepage plus two other pages. • Save files in web design folder. • Add appropriate background colour and text.	Each Unit is graded using the Hagley grading system from SA – 5C at the end of each Unit and at least once during the Topic. A 'Teacher Comment' and 'Area for Improvement' is given at the end of the Unit. Students also evaluate their work and record progress, while responding to both verbal and written feedback from both teacher and peer assessment, this is marked on their self-assessment sheets. End of topic tests are set or final pieces of work are teacher assessed. Homework is set and marked where necessary

			• Add hyperlinks to link web pages together. • Add further content - images, text considering technical issues. • Devise/improve success criteria to evaluate a website. • Evaluate partners website. • Create screenshots of website and annotate to explain choices of design and content as well as future improvements	with a positive comment and given an Area for Improvement.
9	Programming Concepts	• Learn what Python is and some of the applications it is used for • Write, save and run a program in Script mode • Understand what a syntax error is and how to interpret an error message • Know the rules for variable names and use variables in a program • Understand the use and value of comments in a program • Understand the importance of using correct data types string, integer, float • Understand how to use assignment statements correctly • Review the difference between syntax errors, run-time errors and logic errors • Understand and apply the principle of a binary search • Compare the efficiency of a binary search with a linear search	• Run a simple Python program in Interactive mode using the input and print functions • Perform arithmetic using the BIDMAS rule • Use the int, float and round functions • Write a program involving input, calculation and output • Use selection statements if, else and elif in a program • Use indentation correctly to define a block of code • Learn to write algorithms in pseudocode • Learn techniques for debugging programs • Use a while loop in a program • Use an if statement within a while loop	Each Unit is graded using the Hagley grading system from SA – 6A at the end of each Unit and at least once during the Topic. A 'Teacher Comment' and 'Area for Improvement' is given at the end of the Unit. Students also evaluate their work and record progress, while responding to both verbal and written feedback from both teacher and peer assessment, this is marked on their self-assessment sheets. End of topic tests are set or final pieces of work are teacher assessed.

		• Compare alternative algorithms for a given problem • Understand how a binary search works	• Use a function to generate a random number • Use a linear search to find a number • Test a program	Homework is set and marked where necessary with a positive comment and given an Area for Improvement.
	Micro:bit programming	• Understand the purpose of the Micro:bit and what it can do • Create a 5 × 5 image and display it on the Micro:bit • Use the Micro:bit to develop understanding of problem solving and application documentation • Describe and explain some of the advantages and disadvantages of a scripting editor in comparison with the Blocks Editor	• Create a Guess the Number game. • Use the Micro:bit to develop understanding of creating programs • Create a basic application using the Python Editor with Micro:bit • Create a more complicated application using the Python Editor with Micro:bit	Each Unit is graded using the Hagley grading system from SA – 6A at the end of each Unit and at least once during the Topic. A 'Teacher Comment' and 'Area for Improvement' is given at the end of the Unit. Students also evaluate their work and record progress, while responding to both verbal and written feedback from both teacher and peer assessment, this is marked on their self-assessment sheets. End of topic tests are set or final pieces of work are teacher assessed. Homework is set and marked where necessary with a positive comment and given an Area for Improvement.
	Graphics Design	• Understand the difference between Vector and Bitmap images	• Able to use basic functions of graphics software	Each Unit is graded using the Hagley grading

		• understanding of technical considerations to produce effective and efficient digital communications • refine and combine different components of text, images from a range of sources • Understand the importance of Resolution • learn how to change the saturation, brightness and contrast in an image • learn the importance of white space in a poster or advertisement	• combine and manipulate objects in a graphics package to create an image • Crop, Layers and Spot Healing • manipulate objects in Photoshop • annotate the graphics they have created through the unit • Describe tools and techniques they have used	system from SA – 6A at the end of each Unit and at least once during the Topic. A 'Teacher Comment' and 'Area for Improvement' is given at the end of the Unit. Students also evaluate their work and record progress, while responding to both verbal and written feedback from both teacher and peer assessment, this is marked on their self-assessment sheets. End of topic tests are set or final pieces of work are teacher assessed. Homework is set and marked where necessary with a positive comment and given an Area for Improvement.
	Networks	• Learn what the Internet and World Wide Web are • Learn how web addresses are constructed • Learn what a protocol is and why one is needed for data communication • Understand how packet switching works • Learn what the Domain Name Server (DNS) does	• Calculate time taken to download files of different sizes at different bandwidths • Be able to give an example of each type of network • Identify three different network topologies – bus, ring and star • Be able to list advantages and disadvantages of each model	Each Unit is graded using the Hagley grading system from SA – 6A at the end of each Unit and at least once during the Topic. A 'Teacher Comment' and 'Area for Improvement' is given at the end of the Unit.

		• Learn the meaning and significance of bandwidth • Compare different types of cable, and the relative speeds of data transmission • Understand what is meant by buffering and why it is used • Understand the difference between LANs and WANs • Know what extra hardware is needed for a LAN to operate • Understand what constitutes a client-server network • Contrast a client-server network with a peer-to-peer network • Understand what is meant by cloud computing • To identify some of the types of data that need to be kept secure • To learn some of the ways in which data is kept secure • To learn how unauthorised people can break ciphers and read encrypted data	• List the main advantages of cloud computing • To use some classical encryption techniques	Students also evaluate their work and record progress, while responding to both verbal and written feedback from both teacher and peer assessment, this is marked on their self-assessment sheets. End of topic tests are set or final pieces of work are teacher assessed. Homework is set and marked where necessary with a positive comment and given an Area for Improvement.
	Sound Editing	• Learn about Audience and Purpose for a sound clip • Learn about how scripts help plan sound clips • Understand the importance of peer assessment and how it improves the final product • Know how to evaluate and reflect on the work that has been produced	• Create a sound file with multiple sound files. • Use sound effects • Mix sounds together to create a joined-up sound clip • Create a script to aid in the creation of a sound clip • Create a narration to add depth to the sound clip	Each Unit is graded using the Hagley grading system from SA – 6A at the end of each Unit and at least once during the Topic. A 'Teacher Comment' and 'Area for Improvement' is given at the end of the Unit.

				Students also evaluate their work and record progress, while responding to both verbal and written feedback from both teacher and peer assessment, this is marked on their self-assessment sheets. End of topic tests are set or final pieces of work are teacher assessed. Homework is set and marked where necessary with a positive comment and given an Area for Improvement.
	Idea.org	Split into 5 section: • Citizen - Digital awareness, safety & ethics • Worker - Tools & techniques which are useful in the workplace • Maker - Digital creativity & how to build & make in the digital world • Entrepreneur - How to originate ideas & bring them to life • Gamer - Learn gamification techniques & how to make games • Learn cyber security techniques and how hackers attack systems • Understand the use of data in our daily lives	• Complete ethics quizzes • Be able to shop safely online and have good online etiquette • Spot fake news • Apply data protection principles • Use the internet safely and report issues correctly • Set up social media accounts with good privacy settings • Use the cloud to save work	This is a professional qualification and the digital equivalent to the Duke of Edinburgh Award. The Duke of York Award is designed so that students can complete digital badges to earn their way to the Bronze & Silver Award. To achieve the Bronze Award, learners need to earn 250 points at Bronze level, including a

		- Know how to be a good digital citizen and learn how to manage the ethics behind decision making -		minimum of 50 points in each of the core categories of the curriculum: Citizen, Worker, Maker and Entrepreneur. To achieve the Silver Award, learners need to earn 400 points at Silver level - please see the Silver Award section on page 31 for more information.

GCSE Computer Science	**Component 1 – Computer Systems**	**1.1.1 Architecture of the CPU** • The purpose of the CPU: o The fetch-execute cycle • Common CPU components and their function: o ALU (Arithmetic Logic Unit) o CU (Control Unit) o Cache o Registers • Von Neumann architecture: o MAR (Memory Address Register) o MDR (Memory Data Register) o Program Counter o Accumulator **1.1.2 CPU performance** • How common characteristics of CPUs affect their performance: o Clock speed o Cache size o Number of cores **1.1.3 Embedded systems** • The purpose and characteristics of embedded systems • Examples of embedded systems • 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 **1.2.2 Secondary storage** • The need for secondary storage	o User management functions, e.g.: o Allocation of an account o Access rights o Security, etc. o File management, and the key features, e.g.: o Naming o Allocating to folders o Moving files o Saving, etc. o Produce simple diagrams to show: o The structure of a problem o Subsections and their links to other subsections o Complete, write or refine an algorithm using the techniques listed o Identify syntax/logic errors in code and suggest fixes o Create and use trace tables to follow an algorithm o Recognise and use the following operators: o Comparison operators ▪ == Equal to ▪ ! = Not equal to ▪ < Less than ▪ <= Less than or equal to ▪ > Greater than	Use of Exam board Assessment Objectives Continual assessment of Classwork/homework using 9-1 grades and marking criteria according to Hagley's homework policy. End of topic assessments throughout course, Attainment and PPG grades regularly given throughout the course.

	Component 2 - Computational thinking, algorithms and programming	• 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: ○ Capacity ○ Speed ○ Portability ○ Durability ○ Reliability ○ Cost **1.2.3 Units** • The units of data storage: ○ o 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 **1.2.4 Data storage** **Numbers**	■ >= Greater than or equal to ○ Arithmetic operators ■ + Addition ■ – Subtraction ■ * Multiplication ■ / Division ■ MOD Modulus ■ DIV Quotient ■ ^ Exponentiation (to the power) ○ Ability to choose suitable data types for data in a given scenario ○ Ability to manipulate strings, including: ○ Concatenation ○ Slicing ○ The use of functions ○ The use of procedures ○ The use of the following within functions and procedures: ○ local variables/constants ○ global variables/constants ○ arrays (passing and returning) ○ Be able to create and use random numbers in a program ○ Use of commenting ○ Testing data	

		• 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 **Characters** • The use of binary codes to represent characters • The term ‘character set’ • The relationship between the number of bits per character in a character set, and the number of characters which can be represented, e.g.: ○ ASCII ○ Unicode **Images** • How an image is represented as a series of pixels, represented in binary • Metadata • The effect of colour depth and resolution on: ○ The quality of the image ○ The size of an image file **Sound** • How sound can be sampled and stored in digital form	○ 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 ○ Use of other valid notation will be accepted within the examination, e.g. Using T/F for 1/0, or V for OR, etc. ○ Use of an IDE	

		• The effect of sample rate, duration and bit depth on: ○ The playback quality ○ The size of a sound file **1.2.5 Compression** • The need for compression • Types of compression: ○ Lossy ○ Lossless **1.3.1 Networks and topologies** • 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 • 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		

		1.3.2 Wired and wireless networks, protocols and layers • Modes of connection: o Wired o Ethernet o Wireless o Wi-Fi o Bluetooth • Encryption • IP addressing and MAC addressing • Standards • Common protocols including: o TCP/IP (Transmission Control Protocol/Internet Protocol) o HTTP (Hyper Text Transfer Protocol) o HTTPS (Hyper Text Transfer Protocol Secure) o FTP (File Transfer Protocol) o POP (Post Office Protocol) o IMAP (Internet Message Access Protocol) o SMTP (Simple Mail Transfer Protocol) • The concept of layers **1.4.1 Threats to computer systems and networks** • Forms of attack: o Malware o Social engineering, e.g. phishing, people as the ‘weak point’ o Brute-force attacks o Denial of service attacks o Data interception and theft o The concept of SQL injection		

		1.4.2 Identifying and preventing vulnerabilities • Common prevention methods: ○ Penetration testing ○ Anti-malware software ○ Firewalls ○ User access levels ○ Passwords ○ Encryption ○ Physical security **1.5.1 Operating systems** • The purpose and functionality of operating systems: ○ User interface ○ Memory management and multitasking ○ Peripheral management and drivers ○ User management ○ File management **1.5.2 Utility software** • The purpose and functionality of utility software • Utility system software: ○ Encryption software ○ Defragmentation ○ Data compression **1.6.1 Ethical, legal, cultural and environmental impact** • 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) **2.1.1 Computational thinking** • Principles of computational thinking: • Abstraction • Decomposition • Algorithmic thinking **2.1.2 Designing, creating and refining algorithms** • 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 **2.1.3 Searching and sorting algorithms** • Standard searching algorithms: ○ Binary search ○ Linear search • Standard sorting algorithms:		

		○ Bubble sort ○ Merge sort ○ Insertion sort **2.2.1 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 **2.2.2 Data types** • The use of data types: ○ Integer ○ Real ○ Boolean ○ Character and string ○ Casting **2.2.3 Additional programming techniques** • 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 **2.3.1 Defensive design** • Defensive design considerations: ○ Anticipating misuse ○ Authentication • Input validation • Maintainability: ○ Use of sub programs ○ Naming conventions ○ Indentation ○ Commenting **2.3.2 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 **2.4.1 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 **2.5.1 Languages** • Characteristics and purpose of different levels of programming language: o High-level languages o Low-level languages o The purpose of translators o The characteristics of a compiler and an interpreter **2.5.2 The Integrated Development Environment (IDE)** • Common tools and facilities available in an Integrated Development Environment (IDE): o Editors o Error diagnostics o Run-time environment Translators		

GCSE iMedia	**Unit R081: Pre-Production Skills**	○ The purpose, uses & content for: ○ mood boards (e.g. ideas and concepts for a new creative media product development, assisting the generation of ideas) ○ mind maps/spider diagrams (e.g. to show development routes and options for an idea, or component parts and resources needed for a creative media product) ○ visualisation diagrams ○ images ○ graphics ○ logos ○ text ○ storyboards (e.g. for use with video, animation) o number of scenes o scene content o timings o camera shots (e.g. close up, mid, long) o camera angles (e.g. over the shoulder, low angle, aerial) o camera movement (e.g. pan, tilt, zoom or using a track and dolly) o lighting (e.g. types, direction) o sound (e.g. dialogue, sound effects, ambient sound, music)	• interpret client requirements for pre-production (e.g. purpose, theme, style, genre, content) based on a specific brief (e.g. by client discussion, reviewing a written brief, script or specification) • identify timescales for production based on target audience and end user requirements • how to conduct and analyse research for a creative digital media product, i.e.: ○ using primary sources ○ using secondary sources • produce a work plan and production schedule to include: o tasks o activities o workflow o timescales o resources o milestones o contingencies. • the importance of identifying the target audience and how they can be categorised, i.e.: • gender • age • ethnicity • income • location • accessibility	Use of Exam board Assessment Objectives Continual assessment of Classwork/homework using 9-1 grades and marking criteria according to Hagley's homework policy. End of topic assessments throughout course, Attainment and PPG grades regularly given throughout the course.

		o locations (e.g. indoor studio or other room, outdoor) o camera type i.e. ▪ still camera ▪ video camera ▪ virtual camera (e.g. for animations, 3D modelling or computer games) • scripts (e.g. for a video production, voiceover, comic book or computer game) o set or location for the scene o direction (e.g. what happens in the scene, interaction) o shot type o camera movement o sounds (e.g. for actions or events) o characters o dialogue (e.g. intonation, loudness, emotion) o formatting and layout. • the hardware, techniques and software used for: o digitising paper-based documents	• how legislation applies to creative media production, i.e.: • data protection • privacy • defamation • certification and classification • use of copyrighted material and intellectual property. • create a: • mood board • mind map/spider diagram • visualisation diagram or sketch storyboard • analyse a script (e.g. scenes/locations, characters, resources and equipment needed). • the properties and limitations of file formats for still images • the properties and limitations of file formats for audio • the properties and limitations of file formats for moving images, i.e.: o video o animation • identify appropriate file formats needed to produce: ○ pre-production documents final products in line with client requirements.	

		o creating electronic pre-production documents • the health and safety considerations when creating digital media products (e.g. use of risk assessments, location recces, safe working practices) • legislation regarding any assets to be sourced, i.e.: o copyright o trademarks o intellectual property	• suitable naming conventions (e.g. version control, organisational requirements). • review a pre-production document (e.g. for format, style, clarity, suitability of content for the client and target audience) • identify areas for improvement in a pre-production document (e.g. colour schemes, content, additional scenes).	
	Unit R082: Creating a Digital Graphic	• why digital graphics are used (e.g. to entertain, to inform, to advertise, to promote, to educate) • how digital graphics are used (e.g. magazine covers, CD/DVD covers, adverts, web images and graphics, multimedia products, games) • types of digital graphics, i.e.: ○ bitmap/raster ○ vector • file formats, i.e.: o .tiff o .jpg o .png o .bmp o .gif o .pdf • The properties of digital graphics and their suitability for use in creating images, i.e.:	• interpret client requirements for a digital graphic based on a specific brief (e.g. by client discussion, reviewing a written brief, or specification) • understand target audience requirements for a digital graphic • produce a work plan for an original graphics creation; to include: ○ tasks ○ activities ○ workflow ○ timescales ○ resources ○ milestones ○ contingencies • produce a visualisation diagram for a digital graphic • identify the assets needed to create a digital graphic (e.g. photographs,	

		○ pixel dimensions ○ dpi resolution ○ quality ○ compression settings • how different purposes and audiences influence the design and layout of digital graphics (e.g. the use of colour, composition, white space and styles). • how to use version control when creating a digital graphic. • review a digital graphic against a specific brief • identify areas in a digital graphic for improvement and further development (e.g. cropping, rotating, brightness, contrast, levels, colour adjustment).	scanned images, library images, graphics, logos) • identify the resources needed to create a digital graphic (e.g. digital camera, internet, scanner, computer system and software). • how legislation (e.g. copyright, trademarks, logos, intellectual property use, permissions and implications of use) applies to images used in digital graphics, whether sourced or created. • source assets identified for use in a digital graphic, i.e.: ○ images ○ graphics • create assets identified for use in a digital graphic, i.e.: o images o graphics • ensure the technical compatibility of assets with the final graphic (e.g. pixel dimensions, dpi resolution) • create a digital graphic using a range of tools and techniques within the image editing software application (e.g. cropping, rotating, brightness, contrast, colour adjustment) • save a digital graphic in a format appropriate to the software being used	

			• export the digital graphic using appropriate formats and properties for ○ print use ○ web use ○ multimedia use.	
	Unit R085: Creating a Multipage Website	• the purpose and component features of multipage websites in the public domain • the devices used to access web pages i.e.: ○ laptops and personal computers ○ mobile devices and smartphones ○ tablets ○ games consoles ○ digital television • the methods of internet connection i.e.: ○ wired broadband ○ wi-fi (e.g. private local area networks, public hotspots) ○ wireless broadband (e.g. 3G, HSDPA, 3GPP, LTE). • review a multipage website against a specific brief • identify areas for improvement and further development of a multipage website (e.g. text, graphics, moving images, embedded content).	• interpret client requirements for a multipage website (e.g. to inform, entertain, promote or sell products and/or services), based on a specific brief (e.g. by client discussion, reviewing a written brief, or specification) • understand target audience requirements for a multipage website • produce a work plan for the creation of a multipage website, to include: ○ tasks ○ activities ○ workflow ○ timescales ○ resources ○ milestones ○ contingencies • create a site map with navigation links • produce a visualisation diagram for a web page identifying the house style	

			<ul><li>identify the assets needed to create a multipage website (e.g. backgrounds, banners, buttons, shapes, text, fonts)</li><li>identify the resources needed to create and publish a multipage website (e.g. internet access, web server, domain name, computer system and software)</li><li>prepare assets for use in web pages</li><li>create and maintain a test plan to test a multipage website during production.</li><li>how legislation (e.g. copyright, trademarks, intellectual property use, permissions and implications of use) applies to assets used in multipage websites (e.g. images, graphics, corporate logos, music and video clips), whether sourced or created.</li><li>create suitable folder structures to organise and save web pages and asset files using appropriate naming conventions</li><li>source and import assets (e.g. graphics, image, texture, sound, video, animation, text)</li><li>create a suitable master page as a template for a multipage website</li></ul>	

			• use a range of tools and techniques in web authoring software to create a multipage website • insert assets into web pages to create planned layouts (e.g. text, lists, tables, graphics, moving images, embedded content) • create a navigation system (e.g. using a navigation bar, buttons, hyperlinks) • save a multipage website in a format appropriate to the software being used • publish a multipage website to a location appropriate to client requirements. • how to use version control when creating multipage websites	
	Unit R089: Creating a Video Sequence	• the sectors and uses of digital video products, i.e.: ○ commercial contexts (e.g. public information films, multimedia products, advertising) ○ entertainment (e.g. film, television, websites, computer games) ○ business (e.g. information, promotion) education (e.g. tutorials) • video file formats, i.e.: ○ avi ○ mp4	• produce a work plan for a digital sound sequence to include: ○ video footage recording tasks ○ post-production tasks ○ activities ○ resources ○ timescales ○ workflow ○ resources ○ milestones ○ contingencies • produce a storyboard to include: ○ angles	

		○ wmv ○ mov ○ flv • the properties of digital video, i.e.: ○ resolution (e.g. 576, 640, 720, 1440) ○ format (e.g. PAL, NTSC, HD) aspect ratio (e.g. 4:3, 16:9). • interpret client requirements for a digital video sequence (e.g. journalism, documentary, film teaser trailer) based on a specific brief (e.g. by client discussion, reviewing a written brief, or specification) • understand target audience requirements for a digital video sequence • identify appropriate equipment and software to be used in the digital video sequence (e.g. Camera type, lighting, tripods, software applications, connection equipment). • how legislation (e.g. copyright, trademarks, intellectual property use, permissions and implications of use) applies to the use of video footage, whether sourced or recorded. • how to use version control when creating digital video sequences • considerations when exporting different file formats and file sizes (e.g. compression, optimisation, codecs, bit rate, compatibility).	○ sequence ○ timings for each shot • produce a shooting script to include: ○ angles ○ sequence ○ timings for each shot • use a range of camera techniques to record original digital video footage, i.e.: ○ camera shots (e.g. long, medium, close-up) ○ camera movement (e.g. pan/tilt, tracking, dolly) ○ changing camera settings lighting • source additional footage and other assets (e.g. static frames and graphics, motion graphics, background music, narrated voiceover) for use in a digital video sequence • identify appropriate original recorded footage for use in a digital video sequence • import original recorded footage and assets into video editing software recognising any limitations of the software • use software features to produce, edit and enhance a video sequence (e.g. splitting, trimming and cutting	

		- review a digital video sequence against a specific brief - identify areas for improvement and further development of a digital video sequence.	tracks, layering with multiple tracks, sound editing, adding transitions, titles and credits) - save a digital video sequence file in a high-quality format appropriate to the software being used - export a digital video sequence in a file format appropriate to client requirements (e.g. .avi, .mp4, .wmv, .flv, .3GP). -	
A - Level Computer Science	**Computer systems (Component 01)**	The Arithmetic and Logic Unit; ALU, Control Unit and Registers - (Program Counter; PC, Accumulator; ACC, Memory Address Register; MAR, Memory Data Register; MDR, Current Instruction Register; CIR). - Buses: data, address and control: how this relates to assembly language programs. - The Fetch-Decode-Execute Cycle, including its effects on registers. - The factors affecting the performance of the CPU: ○ clock speed ○ number of cores ○ cache - The use of pipelining in a processor to improve efficiency. - Von Neumann, Harvard and contemporary processor architecture. - The differences between and uses of CISC and RISC processors.	- Represent positive integers in binary. - Use of sign and magnitude and two's complement to represent negative numbers in binary. - Addition and subtraction of binary integers. - Represent positive integers in hexadecimal. - Convert positive integers between binary hexadecimal and denary. - Representation and normalisation of floating-point numbers in binary. - Floating point arithmetic, positive and negative numbers, addition and subtraction. - Bitwise manipulation and masks: shifts, combining with AND, OR, and XOR.	Use of Exam board Assessment Objectives Continual assessment of Classwork/homework using A*- E grades and marking criteria according to Hagley's homework policy. End of topic assessments throughout course, Attainment and PPG grades regularly given throughout the course.

		• GPUs and their uses (including those not related to graphics). • Multicore and Parallel systems. • How different input, output and storage devices can be applied to the solution of different problems. • The uses of magnetic, flash and optical storage devices • RAM and ROM. • Virtual storage. • The need for, function and purpose of operating systems. • Memory Management • paging • segmentation • virtual memory • Interrupts, the role of interrupts and Interrupt Service Routines (ISR), role within the Fetch-Decode-Execute Cycle. • Scheduling: • round robin • first come first served • multi-level feedback queues • shortest job first • shortest remaining time. • Operating systems • Distributed • Embedded • multi-tasking • multi-user • Real Time • BIOS	• How character sets (ASCII and UNICODE) are used to represent text. • Arrays (of up to 3 dimensions) • Records • Lists • tuples. • Structures to store data: • linked lists • graph (directed and undirected) • stack, queue • tree • binary search tree • hash table. • How to create, traverse, add data to and remove data from the data structures mentioned above. • Define problems using Boolean logic • Manipulate Boolean expressions, including the use of Karnaugh maps to simplify Boolean expressions • Use the following rules to derive or simplify statements in Boolean algebra: ○ De Morgan's Laws ○ Distribution ○ Association ○ Commutation ○ double negation • Using logic gate diagrams and truth tables	

		• Device drivers. • Virtual machines, any instance where software is used to take on the function of a machine, including executing intermediate code or running an operating system within another • The nature of applications, justifying suitable applications for a specific purpose. • Utilities • Open source vs closed source • Translators: Interpreters, compilers and assemblers. • Stages of compilation ○ lexical analysis ○ syntax analysis ○ code generation ○ optimisation • Linkers and loaders and use of libraries. • Development methods: ○ waterfall lifecycle ○ agile methodologies ○ extreme programming ○ the spiral model ○ rapid application development. • The relative merits and drawbacks of different methodologies and when they might be used. • Writing and following algorithms. • Need for and characteristics of a variety of programming paradigms	• The logic associated with D type flip flops, half and full adders.	

		• Procedural languages, event driven, visual and mark up • Assembly language (including following and writing simple programs with the Little Man Computer instruction set). • Modes of addressing memory • immediate • direct • indirect • indexed • Object-oriented languages with an understanding of: ○ classes ○ objects ○ methods ○ attributes ○ inheritance ○ encapsulation ○ polymorphism. • Lossy vs Lossless compression. • Run length encoding and dictionary coding for lossless compression. • Symmetric and asymmetric encryption. • Different uses of hashing. • Relational database • flat file • primary key • foreign key • secondary key • entity relationship modelling • normalisation • indexing		

		• Methods of capturing, selecting, managing and exchanging data. • Normalisation to 3NF • SQL • Referential integrity • Transaction processing • ACID (Atomicity, Consistency, Isolation, Durability) • record locking • redundancy • Characteristics of networks and the importance of protocols and standards. • The internet structure: • • The TCP/IP Stack. • • DNS • • Protocol layering. • • LANs and WANs. • • Packet and circuit switching. • Network security and threats, use of: • firewalls • proxies • encryption. • Network hardware • Client-server and peer to peer • HTML • CSS • JavaScript. • Search engine indexing. • PageRank algorithm. • Server and client-side processing. • Primitive data types ○ Integer		

		○ real/floating point ○ character ○ string ○ Boolean. • The Data Protection Act 1998. • The Computer Misuse Act 1990 • The Copyright Design and Patents Act 1988 • The Regulation of Investigatory Powers Act 2000. • The individual moral, social, ethical and cultural opportunities and risks of digital technology: ○ Computers in the workforce. ○ Automated decision making. ○ Artificial intelligence. ○ Environmental effects. ○ Censorship and the Internet. ○ Monitor behaviour. ○ Analyse personal information. ○ Piracy and offensive communications. ○ Layout, colour paradigms and character sets.		
	Algorithms and programming (Component 02)	• The nature of abstraction • The need for abstraction • The differences between an abstraction and reality • Devise an abstract model for a variety of situations. (pseudocode & flowcharts) • Identify the inputs and outputs for a given situation.	• Programming constructs: sequence, iteration, branching, counts. • Recursion, how it can be used and compares to an iterative approach. • Global and local variables. • Constants • Mathematical Operators: ○ + ○ -	

		• Determine the preconditions for devising a solution to a problem. • The nature, benefits and drawbacks of caching. • The need for reusable program components. (modules) • Identify the components of a problem • Identify the components of a solution to a problem • Determine the order of the steps needed to solve a problem. • Identify sub-procedures necessary to solve a problem. • Identify the points in a solution where a decision must be taken. • Determine the logical conditions that affect the outcome of a decision. • Determine how decisions affect flow through a program. • Determine the parts of a problem that can be tackled at the same time • Outline the benefits and trade-offs that might result from concurrent processing in a particular situation. • Learners should apply their knowledge of: ○ backtracking ○ data mining ○ heuristics ○ performance modelling ○ pipelining ○ visualisation to solve problems.	○ * ○ / ○ = ○ == ○ > ○ >= ○ < ○ <= ○ % • Logical Operators: ○ AND ○ OR ○ NOT ○ XOR • Modularity, functions and procedures, parameter passing by value and by reference. • Use of an IDE to develop/debug a program. • Self-documenting identifiers, annotation and commenting • Use of object-oriented techniques. • Features that make a problem solvable by computational methods. • Problem recognition. • Problem decomposition. • Use of divide and conquer. • Use of abstraction. • Measures and methods to determine the efficiency of different algorithms, Big O notation: ○ constant	

		• Analysis and design of algorithms for a given situation • The suitability of different algorithms for a given task and data set, in terms of execution time and space.	○ linear ○ polynomial ○ exponential ○ logarithmic complexity • Comparison of the complexity of algorithms • Algorithms for the main data structures ○ stacks ○ queues ○ trees ○ linked lists ○ depth-first (post-order) ○ breadth-first traversal • Standard algorithms: ○ bubble sort ○ insertion sort ○ merge sort ○ quick sort ○ Dijkstra's shortest path algorithm ○ A* algorithm ○ binary search linear search	
	Programming Project (Component 03)	• Describe and justify the features that make the problem solvable by computational methods. • Explain why the problem is amenable to a computational approach. • Identify and describe those who will have an interest in the solution explaining how the solution is appropriate to their needs (this may be named individuals, groups or	• Specify and justify the solution requirements including hardware and software configuration (if appropriate). • Identify and justify measurable success criteria for the proposed solution. • Break down the problem into smaller parts suitable for	

		persona that describes the target end user). • Research the problem and solutions to similar problems to identify and justify suitable approaches to a solution. (feasibility study) • Describe the essential features of a computational solution explaining these choices. • Explain the limitations of the proposed solution. • Provide annotated evidence of each stage of the iterative development process justifying any decision made • Provide annotated evidence of prototype solutions justifying any decision made. • Provide annotated evidence for testing at each stage justifying the reason for the test. • Provide annotated evidence of any remedial actions taken justifying the decision made. • Provide annotated evidence of testing the solution of robustness at the end of the development process. • Provide annotated evidence of usability testing (user feedback). • Use the test evidence from the development and post development process to evaluate the solution against the success criteria from the analysis.	computational solutions justifying any decisions made. • Explain and justify the structure of the solution. • Describe the parts of the solution using algorithms justifying how these algorithms form a complete solution to the problem • Describe usability features to be included in the solution. • Identify key variables / data structures / classes justifying choices and any necessary validation. • Identify the test data to be used during the iterative development and post development phases and justify the choice of this test data.	

		• Provide annotated evidence of the usability features from the design, commenting on their effectiveness. • Discuss the maintainability of the solution. • Discuss potential further development of the solution.		
A - Level IT	**Unit 1 – Fundamentals of IT**	• input devices • output devices • communications devices • benefits (e.g. integrated devices make portable devices simpler to use) • limitations (e.g. voice recognition performs poorly in noisy environments) • uses (e.g. membrane keyboard could be used in harsh physical environments) • processors • motherboards • storage (i.e. hard drive, solid state, flash, internal, removable, SAS, SCSI, portable, Cloud) • ports (i.e. USB, Firewire, SATA, Network, Fibre Channel) • memory (i.e. RAM, ROM, cache) •expansion cards (i.e. sound, network, graphics, storage controller, fibre channel) •power supplies •characteristics •purpose •desktop/server •tablet/hybrid •smartphone	• interpersonal skills (i.e. eye contact, body language) • questioning techniques • verbal (i.e. meetings, telephone, group discussions) • written (i.e. reports, letters, emails, social networking) • non-verbal (i.e. body language) • barriers (i.e. language, distraction, noise, lack of concentration) • appropriate use of language (i.e. formal, informal, technical, non-technical) • presentation software • word processing • email • web • blogs/vlogs • instant messaging • use • self-motivation • leadership • respect • dependability • punctuality • problem solving	Use of Exam board Assessment Objectives Continual assessment of Classwork/homework using A*- E grades and marking criteria according to Hagley's homework policy. End of topic assessments throughout course, Attainment and PPG grades regularly given throughout the course.

		•embedded system/Internet of Things (e.g. cars, home appliances, etc.) •mainframe •quantum •uses (e.g. tablet device can be used when travelling due to physical properties) •benefits (e.g. desktop computer can have a large screen which can improve productivity) •limitations (e.g. mainframes can be expensive to purchase and maintain) •copper •fibre •wireless technologies (i.e. Bluetooth, Wi-Fi, microwave, infrared, laser, Satellite, GSM, 3G/4Gand future technologies) •characteristics •purpose •hub •switch •router •modem •wireless access point •combined/hybrid devices •characteristics •purpose and use •identifying hardware faults •troubleshooting tools •documentation/fault management •bit, nibble, byte •metric (i.e. kilo, mega, giga, tera, peta) •binary (i.e. kibi, mebi, gibi, tebi, pebi)	• determination • independence • time management • team working • written numerical and verbal skills • planning and organisation skills • dress (i.e. appropriate clothing depending on situation) • presentation (i.e. personal grooming, appearance etc.) • attitude (i.e. can-do attitude, responsive)	

		•comparison in sizes between metric and binary measurements. e.g. 1 kilobyte = 1000 bytes vs1024 bytes •binary •decimal •hexadecimal •converting between binary, decimal and hexadecimal • open source • closed source • off the shelf • bespoke • shareware • freeware • embedded • characteristics • use • productivity software (i.e. word processor, spreadsheet, database, email) • development tools (i.e. compiler, debugger, translator, integrated design environment) • business software (i.e. MIS, multimedia, collaboration, project management, manufacturing, CAD/CAM, publishing, expert systems, healthcare) • purpose • advantages and disadvantages • single user/multiuser • single processor/multiprocessor • off the shelf/open source/bespoke • Functions • Benefits and limitations •SMS		

		•email •messaging software •social networking/social media •VoIP •personal assistants (e.g. Siri, Cortana) •teleconference •video conference •cellular/satellite •instant messaging •characteristics •purpose •advantages and disadvantages •common faults (i.e. unexpected software behaviour, software freeze, unexpected rebooting) •troubleshooting tools to investigate a problem (i.e. Logs, installable tools, baselines) •documentation (i.e. types of documentation) •popular protocols ○ IP (Internet Protocol) ○ TCP (Transmission Control Protocol) ○ UDP (User Data Protocol) ○ SMTP (Simple Mail Transfer Protocol) ○ FTP (File Transfer Protocol) ○ HTTP (Hyper Text Transfer Protocol) ○ SNMP (Simple Network Management Protocol) ○ ICMP (Internet Control Message Protocol) ○ POP (Post Office Protocol) • features • purpose • common usage scenarios		

		• file/print • application • database • web • mail • hypervisor • server • client • storage • cloud • hybrid • benefits and limitations • peer to peer • client server (i.e. DNS) • bus/star/ring/mesh • addressing (i.e. default gateway, IP address, subnet mask) • diagrammatical representation • linking to given context • LAN (i.e. Ethernet, Token Ring) • WAN (i.e. ADSL, leased line, ISDN) • MAN • voice (i.e. PSTN, cellular) • satellite (i.e. voice, data) • characteristics • purpose • MIS (Management Information System) • CRM (Customer Relationship Management) • SOP (Sales Ordering Process, Standard Operating Procedures) • helpdesk • purpose • benefits and limitations		

		• Network manager • IT technician • Programmer • Web designer • Animator • Key skills required for each (i.e. technical and non-technical) •whistle blowing •disability/gender/sexuality discrimination •use of information •codes of practice •staying safe online •bias •security of information •health and safety •disaster planning and recovery •organisational policies (i.e. acceptable use policy, code of conduct, etc.) •change management •scale of change: ○ drivers (i.e. change in business practice, legislation, competition) ○ needs (i.e. improved networking, remote access for employees) • phishing • hacking • virus • Trojan • interception • eavesdropping • data theft • social engineering • locks		

		• biometrics • RFID • tokens • privacy screens • shredding • characteristics • anti-virus • firewalls • anti-spyware • username/passwords • permissions • encryption • characteristics • legislation • overwrite data • electromagnetic wipe • physical destruction		
	Unit 2 – Global information systems	• categories of holders (individual citizens, businesses, educational institutions, governments, charities, healthcare services and community organisations) • location (e.g. developing country, developed country, urban, rural, home, workplace) • comparison of technologies available and access issues across the global divide (e.g. between developed and developing countries) • paper (e.g. forms, handwritten notes, maps, telephone directories) • optical media (e.g. CD and DVD) • magnetic media (e.g. magnetic hard drives and tapes)	• characteristics (e.g. valid, bias, reliable, comparable) • importance of good quality information to stakeholders (e.g. innovation, agility, improved strategic decision making and focus) • consequences of poor-quality information on stakeholders (e.g. misinformation, reputational damage) • collecting, storing and retrieving (e.g. adding a new member to a cycling club membership database) • manipulating and processing (e.g. producing a graph from a table of data) • analysing (e.g. looking for patterns in annual rainfall in an area)	

		• solid state media (e.g. SSD hard drives, memory cards) • characteristics • purpose • advantages and disadvantages • handheld device (e.g. small tablet, smart phone, wearable device, eBook readers) • portable devices (e.g. laptop, large tablet) • fixed devices (e.g. desktop computer, smart TV, games consoles) • shared devices (e.g. database server, data centre, cloud storage devices) • characteristics • purpose • advantages and disadvantages • a network of interconnected networks, spanning the world • internet connections ○ type (e.g. copper-cable, optical-fibre, satellite, microwave, mobile data networks) ○ characteristics (e.g. speed, range (distance), storage capacity) • types of networks that use www software: ○ internet (e.g. public, open access) ○ intranet (e.g. private, closed access) ○ extranet (e.g. private, part shared access) • comparison of networks (e.g. suitability for given uses, issues related to access to the network) • characteristics of networks • purpose of networks	• securing (e.g. storing patient records on an encrypted hard drive) • transmitting (e.g. posting a printed school report to a parent) • impact on individuals and organisations (e.g. additional costs associated with keeping sensitive information secure) • data-raw, unorganised facts that needs to be processed information-data which is processed, organised and structured into a meaningful context. • communication (e.g. to send an email to a relation living overseas) • education and training (e.g. by a student to check their current grades on a hand-written feedback sheet from their teacher) • entertainment (e.g. to read a film review in a magazine) • planning (e.g. to use a shared electronic diary to arrange meeting dates) • financial (e.g. to use a bank statement to help plan saving for a holiday) • research (e.g. to look up a recipe online) • location dependent (e.g. to search for emergency dental care when on holiday) • benefits and limitations	

		• webpages (static and dynamic) • blogs • podcasts • streamed audio and video (e.g. internet radio, catch-up TV) • social media channels (e.g. Twitter, LinkedIn, discussion boards) • document stores (upload and download) • RSS feeds: ○ purpose ○ accessibility • for individuals (e.g. speed of personal communication, easy access to large amounts of information for research, access to internet banking 24/7) • for organisations (e.g. share large amounts of information quickly between different countries; charity websites accepting donations 24/7) • for individuals (e.g. potential for identity theft, cost of data connection) • for organisations (e.g. threats caused by malicious attacks, cost of maintaining websites and data stores) • text (different character sets, e.g. Western, Cyrillic, Arabic, etc.) • graphic (e.g. logo, photograph, diagram) • video (e.g. instructions on how to carry out a software update, live broadcast of a music festival) • animated graphic (e.g. pop-up book character, operation of the human heart) • audio (e.g. spoken instructions, music track)	• knowledge management and creation (e.g. to create an accurate model of key markets) • management information systems (MIS) (e.g. to monitor staff training in a hospital; the location and contact details of each charity worker in a disaster area; personnel record of all staff) • marketing, promotion and sales (e.g. to identify patterns or trends in sales figures) • financial analysis and modelling (e.g. to determine the top selling products, cash flow each week over a year) • contact management (e.g. to keep track of appointments at a doctor's surgery) • decision making (e.g. to decide the number of tents to be sent to a disaster area by a charity; the percentage of faulty items made each month by a manufacturer) • internal and external communication (e.g. to inform all staff of office closures over the Christmas period) • big data, i.e.: • any data that is either too large or too complex for traditional data analysis techniques to be used, e.g. the annual web clicks of a major online retailer,	

		• numerical (e.g. profit, date and time) • Braille text (e.g. written report printed on a Braille printer) • tactile images (e.g. NASA's Hubble Space Telescope images converted into tactile images for people who cannot explore the images by sight) • subtitles (e.g. translated speech for a film in a foreign language) • Boolean (e.g. yes or no answer on a form) • tables and spreadsheets (e.g. simple database tables and spreadsheets) • charts and graphs (e.g. identifying trends, making comparisons) • sensitive/non-sensitive • private/public • personal/business • confidential/classified • partially anonymised/completely anonymised • impacts on different stakeholders • benefits and limitations • identify the need (e.g. what information is needed? what do we want to find out?) • define scope (e.g. content, detail, timescales, constraints) • identify potential sources (e.g. sales figures, customer surveys) • source and select information (e.g. determine accuracy and reliability of sources, selecting the best) • select the most appropriate tools (e.g. charts, graphs, regression, trend analysis)	health data on the population of an entire country • connectivity rules for drawing Level 1 DFDs: ○ at least one input or output for each external entity ○ data flows only in one direction ○ every data flow is labelled ○ every data flow connects to at least one process ○ at least one input data flow and/or at least one output data flow for each process ○ impacts affecting the flow of information in information systems	

		• process and analyse data (e.g. set up a spreadsheet to produce a graph) • record and store information (e.g. write a report based on the results of the processing) • share results (e.g. send the report to stakeholders) • data tables (e.g. a database table of patients) • visualisation of data (e.g. a pie chart showing sales of five leading trainers) • trend and pattern identification (e.g. a line graph of last year's sales per month) • data cleaning (e.g. removing customers who have not made a purchase in the last two years) • geographic information system/location mapping (e.g. tracking the movement of shipping containers around the world) • open systems/closed systems • characteristics • benefits and limitations • current UK legislation and regulation: • Data Protection Act (DPA) 1998 • Regulation of Investigatory Powers Act (RIPA) 2000 • Protection of Freedoms Act 2012 • Privacy and Electronic Communications Regulations 2003 (amended 2011) • Freedom of Information Act 2000 • Computer Misuse Act 1990 • Information Commissioner's Office (ICO) codes of practice		

		• Copyright, Designs and Patents Act 1988 • Equality Act (EQA) 2011 • impact and consequences of UK legislation and regulation on organisations operating in the UK and the way they handle information and individuals’ personal data • actions that can be taken by organisations to comply with legislation and regulatory requirements • regulation relating to data protection outside the UK (e.g. USA, France, Far East and Africa) • comparison between data protection legislation and regulation in different countries (e.g. similar legislation in many countries, but not all) • UN Convention on the Rights of Persons with Disabilities (UNCRPD): ○ (e.g. the UNCRPD specifically recognises (under articles 9 and 21) that access to information, communications and services, including the internet, is a human right) • global requirements on organisations and individuals • United Nations Climate Change Summits • UK Government policy (e.g. Greening Government ICT Strategy (2011)) • reducing carbon footprint • purpose (e.g. sustainability)		

		• benefits (e.g. enhanced brand image, reduced energy costs) • internal source (e.g. internal financial reports, market analysis) • external source (e.g. supplier price lists, financial report from a third party) • primary data (e.g. reports direct from employees, foot measurements taken in a shoe shop) • secondary data (e.g. survey results received from a market research organisation, interest rate charged on a loan from a bank) • qualitative data (e.g. the colour of products, the names of employees) • quantitative data (e.g. expiry date of medicines, the number of staff working in an organisation) • purpose • external entities • processes • data stores • data flows • standard symbols used • confidentiality – information can only be accessed by individuals, groups or processes authorised to do so • integrity – information is maintained, so that it is up to date, accurate, complete and fit for purpose • availability – information is always available to and usable by the individuals, groups or processes that need to use it		

		• unauthorised or unintended access to data (e.g. espionage, poor information security policy) • accidental loss of data (e.g. human error, equipment failure) • intentional destruction of data (e.g. computer virus, targeted malicious attack) • intentional tampering with data (e.g. fraudulent activity, hacking) • loss of intellectual property • loss of service and access • failure in security of confidential information • loss of information belonging to a third party • loss of reputation • threat to national security • recent cases of failures of information security • Policies, e.g.: ○ staff access rights to information ○ responsibilities of staff for security of information ○ disaster recovery ○ information security risk assessment ○ effectiveness of protection measures ○ training of staff to handle information • locks, keypads and biometrics used on: ○ workstations ○ server room access • placing computers above known flood levels • backup systems in other locations • security staff		

		• shredding old paper-based records • tiered levels of access to data • firewalls (hardware and software) • anti-malware applications • obfuscation • encryption of data at rest • encryption of data in transit • password protection		
	Unit 3 – Cyber Security	• confidentiality • integrity • availability • unauthorised access including hacking, escalation of privileges • information disclosure including personal information, government information • modification of data • inaccessible data including account lockout, denial of service • destruction including using malware, deliberate erasure • theft including identity, finance, military secrets • the need to protect personal data (e.g. health, financial, national insurance) • the need to protect an organisation's data (e.g. financial, research, development plans) • the need to protect a state's data (e.g. economic data, national security) • vulnerabilities o system attacks o physical threats o environmental • accidental	• intrusion detection systems (IDS) including network intrusion detection systems (NIDS), host intrusion detection systems (HIDS), distributed intrusion detection system (DIDS), anomaly-based, signature-based, honeypots • intrusion prevention systems (IPS) • emerging technologies • effectiveness • physical including biometric access, swipe cards, alarms • hardware including cable locks, safes • software including firewalls, anti-malware, operating system updates, patch management • data including in use, at rest, in-transit, in the cloud • encryption including disks, databases, files, removable media, mobile devices • cryptography • devices including. hard drives, external drives, USBs • procedures including access management, data backup, remote working, device management, user	

		• intentional • organised crime • state sponsored • hacktivist • cyber-criminal • insider • script kiddie • vulnerability broker • scammers • phishers • cyber-terrorists • characteristics including age, location, social group • espionage • righting perceived wrongs • publicity • fraud • score settling • public good • thrill • income generation • people • organisations • equipment • information • methods that can be used during an attack • global problem, individuals, organisations and states • loss including confidentiality, integrity, availability, data, finance, business, identity, reputation, customer confidence	accounts and permissions, awareness and training • emerging technologies • characteristics • know responsibilities • know who to contact • know procedures • know the extent of the incident • contain the incident • eradicate the incident • reduce the impact of the incident • recover from the incident • confirm the system is functioning normally • incident title and date of incident • target of the incident • incident category, i.e.: o critical o significant o minor o negligible • description of the incident • type of attacker(s) • purpose of incident • techniques used by the attacker(s) • capability of attacker(s) • impact of the incident on business, data, recovery time • cost of the incident • responses needed • future management o review (of incident)	

		• disruption including people's lives, business, industry, transport, industry, the media, utilities • safety including identity theft, oil installations, traffic control • ethical • legal • operational • implications for stakeholders • identify assets and analyse risks • mitigate risks by: o testing for potential vulnerabilities • monitoring and controlling systems • protect vulnerabilities • cost/benefit • vulnerability testing including penetration testing, fuzzing, security functionality, sandboxing	o evaluation to include identification of trends o update of documentation, key information, procedures and controls o recommendations of changes	
	Unit 6 – Application Design	• requirements analysis • design • implementation/coding • testing • deployment • maintenance • waterfall model • iterative model • agile development model • rapid application development (RAD) model • spiral model • prototype model • client and user interviews e.g. o closed and open questions o leading questions	• functional requirements (e.g. use case diagrams) • processing and data handling (e.g. flowcharts, data flow diagrams, class diagrams, object diagrams, entity relationship diagrams) • user interface designs (e.g. wireframe diagrams and graphical mock-ups) • standard algorithms or processes • modularisation • cross-platform standards • standard protocols • standard interface widgets (appearance of buttons, dropdown menus, hyperlinks)	

		o funnelling o structure to interviews o allowing thinking time for respondents o encouraging further detail or thought • observation of tasks • analysis of existing documents and systems • functional requirements, e.g.: o what the application should do o data and information collected and used in the existing approach o data and information to be collected and used in the new application o functions or processing that the application should perform o outputs from the application o core functional requirements o optional functional requirements o user interface requirements including accessibility requirements • Functional requirements may be divided into core requirements and optional requirements. • Constraints, e.g.: o hardware or platform constraints o software constraints o development constraints e.g. development software • limitations (e.g. scope of solution, aspects that will not be developed)	• common user interface layouts, icons and labels throughout application • automation • operational efficiency • cost-effectiveness • globalisation • improved communication • customisation and adaptability • increased markets • ease of access for customers • new marketing opportunities • customer or user information • real-time information • new employment • financial cost • changeover costs and risks • training needs • lack of job security and job losses • security issues • privacy issues • potential customer concerns • loss of personal contact • what is the proposed design solution? • who would be interested in it? • why is it a valuable idea? • what makes it effective? • courtesy • speak clearly and concisely • be aware of body language • accurate spelling, punctuation and grammar • engage the audience • be honest	

		•different ways to address the identified user need • technological requirements • economic or financial costs of development and potential benefits • legal issues • operational impact • scheduling and resources (e.g. time scale for development, resources needed for development)	• be positive • anticipating likely questions • giving a positive response • seeking clarification where necessary • recognising improvements and responding in a way that suggests how these can be incorporated • purpose of prototyping • features of prototypes • interviewing and questioning techniques • development formats • meeting core requirements and any optional requirements • effectiveness (e.g. how well the design meets each requirement) • usability (e.g. how easy it is to carry out actions, readability and clarity of displays or output to user, navigability) • learnability (e.g. how easy it is to learn how to use the application, clarity of the function of different components or elements) • identify distinct points in feedback • identify required changes • identification and implementation of improvements based on feedback	

	Unit 15 – Games Design & Prototyping	• purpose of a game • audience of a game • common features in games • first/third person • player characters • non-player characters • sprites • achievement attainment • maintaining a player’s interest • player interaction/controls • player immersion • progression • competition (e.g. high score) • types of prototyping e.g. visual/representational (non-working) and proof of concept (working) • testing concepts • gauging player interest • skill level required • gauging difficulty of achievement • clarity of interface • clarity of understanding of gameplay/goals • ensures the design does what it is supposed to do • helps to identify and address problems at an early stage • gives the client and end-user an appreciation of the final product • allows developer to explore ideas and obtain feedback with the client and end-user • navigation • scoring • movement	Games Development Environment e.g. o Gamemaker o Godot o Defold • variables/constants • strings • inputs • outputs • sequence • selection • iteration (e.g. counting/conditional) • subroutines (e.g. functions/procedures) • conditions • counting • totalling • data structures (e.g. arrays/lists) • file handling • maintainable code • libraries • test plans • test data • black box • white box • alpha • beta • user testing • choose a suitable method of presentation (e.g. live demonstration, use of presentation software, report) • plan a presentation to incorporate: o comparison of game concept against requirements	

		• interaction/controls • conveying information • sound • levels • enemies • problem solving • layout • colour palette • text styles • sound • stage/scene • actions (e.g. menus/buttons) • perspective (e.g. 2D/3D) • bitmaps • wireframe • requirement specification • design specification • project plan • system flowchart • top-down • JSP • clear definition of objectives of game • flow chart showing the 'flow' of the game through single or multiple layers with single or multiple players	o demonstration of functionality o demonstration of interactivity o demonstration of responsive design o justification of design choices o presents the solution to the client • has it met the requirements? • does it reflect the design? • is it suitable for the identified audience and purpose? • suggest recommendations for full game concept • justify the continued development of the full game concept	

For full curriculum overviews & complete Schemes of Work and student versions please see Haggle.