Pearson BTEC

L3

Pearson Level 3 Alternative Academic Qualification
BTEC National in

Computing (Extended Certificate)

Planning and Teaching Guide

First teaching from September 2025

First certification from 2027

Qualification Number: 610/3963/9

About Pearson

We are the world's leading learning company operating in countries all around the world. We provide content, assessment and digital services to students, educational institutions, employers, governments and other partners globally. We are committed to helping equip students with the skills they need to enhance their employability prospects and to succeed in the changing world of work. We believe that wherever learning flourishes so do people.

References to third-party material made in this Planning and Teaching Guide are made in good faith. Pearson does not endorse, approve or accept responsibility for the content of materials, which may be subject to change, or any opinions expressed therein. (Material may include textbooks, journals, magazines and other publications and websites.)

All information in this Guide is correct at time of publication.

Contents

1. Introduction

This Planning and Teaching Guide complements your Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Extended Certificate) specification, Pearson Set Assignment Briefs (PSABs), Sample Assessment Materials (SAMs) and the Pearson BTEC Level 3 National Alternative Academic Qualification Administrative Support Guide. This Planning and Teaching Guide provides:

- an overview of dates and deadlines for key events and activities relevant to qualification delivery – from registration to assessment and review of marking – throughout the academic year
- suggestions for planning and delivering your course including induction and unit sequencing
- creative and realistic teaching and learning ideas as well as links to resources for each unit to support and inspire you in creating a dynamic learning environment to keep your students engaged and motivated to learn.
- wider delivery support such as guidance on study programme planning and descriptions and links to qualification resources and materials.

The guide was designed and written in collaboration with current practitioners to ensure that the planning and delivery suggestions and teaching and learning ideas are feasible, pedagogically sound and appropriate for the vocational area and the purpose of the qualification.

We recognise that delivery contexts will vary from one centre to the next and that practitioners are the best decision-makers for what works best for them and their students. Therefore, teachers can tailor the suggestions and ideas proposed in this guide to meet the specific needs of their students and the available resources in their centre. There are, however, requirements that have to be met in relation to assessment plans and to teaching and learning preceding assessment, which will be clarified/covered in this guide.

We hope you will find this guidance relevant and useful, and you enjoy teaching this this qualification!!

What's new

When creating these BTEC Nationals, in addition to ensuring the sector technical content was current and up-to date, we have also focused on developing the skills and personal attributes students need to navigate the future. We have worked with many higher education providers, professional bodies, colleges and schools to ensure these qualifications also meet their needs. Employers are looking for future employees with a thorough grounding in the latest industry requirements and work-ready skills such as critical thinking and problem solving. Higher education needs students who have experience of research, extended writing and meeting deadlines to be successful on their undergraduate programmes.

We have addressed these requirements by:

- Facilitating and guiding the development of transferable skills through the design and delivery of the qualifications, using a holistic and practical framework which is based on recent research into the most critical skills needed to navigate the future. A Transferable Skills framework has been used to embed transferable skills in the qualifications where they naturally occur and to signpost opportunities for delivery and development as a part of the wider BTEC learning experience. Please refer to the BTEC Transferable Skills Guide for Teachers for further information on this framework, its relevance and how it has been implemented in the qualifications.
- Supporting the delivery of Sustainability Education and development of Digital Skills naturally through the content design of the qualifications. Mapping is provided in the specification to identify where these opportunities for teaching and learning exist.
- Updating sector-specific content to ensure it is current, relevant and future-facing.
- Implementing a consistent approach to assessment with a balanced combination of internal and external assessments to better engage students, make the qualifications more accessible for them and more manageable for centres to deliver.

We are providing a wealth of support, both resources and people, to help ensure that you and your students have the best possible experience during their course. Please see the section on Pearson Qualification Support and resources on page 126 for details of the available resources and support with links to access these.

Notes:

The qualification specification provides the content that must be taught and what must be assessed. This planning and teaching guide provides suggestions and ideas for how the content could be delivered. The suggestions given in this guide link with the Pearson Set Assignment Briefs provided by Pearson, which are mandatory for internal assessment and cannot be amended or contextualised by centres.

2. BTEC Calendar of Key Dates

Each academic year there are some key dates and deadlines in the delivery of BTEC qualifications that teachers need to be aware of, and act on appropriately, to ensure:

- the smooth running of student registration, assessment and the quality assurance process, and
- effective timetable planning to fully prepare students for assessments and ensuring timely completion of administrative tasks.

Here is an overview of the key dates and deadlines for this qualification.

The specific date for each activity or event will vary each academic year and so only the month is provided. For the specific dates for the current academic year, please go to our webpage: https://qualifications.pearson.com/en/qualifications/btec-nationals/computing-aaq.html

Month	General related dates	Internal Assessment related dates	External Assessment related dates
September	Student registration		
October		Lead IV registered and completion of team standardisation	Entry deadline for January external assessment
November	Late student registration fee		
December	Late student registration fee Deletion deadline: delete student registrations for any leaner withdrawn from the qualification		
January		Standards Verification Window opens	January External Assessment Series
February			

Month	General related dates	Internal Assessment related dates	External Assessment related dates
March			Restricted release of results to centres Release of results to students Entry deadline for Summer external assessments Review of Marking
April			Review of Marking
May		Standards Verification for first sample closes	Summer External Assessment
June		Standards Verification for second sample closes	
July	Deadline for full qualification claim for summer certification		
August			Restricted release of results to centres Release of results to students Review of marking

3. Planning the Delivery of your Course

Planning your course ensures a coherent and logical approach to teaching that helps students to connect concepts effectively and build their knowledge progressively.

Effective assessment planning is also essential to allow for timely evaluation of student progress and adjustment of teaching strategies or interventions as needed.

This section offers recommended approaches to support practitioners with planning and implementation of this qualification

Induction

Students

An induction period at the start of the course is recommended to help students understand and prepare for the demands of their chosen course, as well as familiarise them with the BTEC ethos and methodology. This induction aims to not only equip students with the necessary knowledge and skills but also to create a welcoming environment where they feel safe, supported and gain a sense of belonging as they begin their course in a new setting.

Centres will have their own induction programmes, and to support this, Pearson have provided a range of adaptable resources that can be integrated into this existing programmes. These resources cover areas such as welcome activities and information to include in the induction, with supporting slides. As we believe that every opportunity should be taken to develop transferable skills across the wider BTEC learning experience, we have also provided guidance on which transferable skills could be delivered as a part of the induction process including Managing Own Learning, Continuous Learning, goal setting and personal strength and resilience. The resources are designed to help students develop the relevant transferable skills through learning how to manage their course workload, completing their assessments successfully and meeting deadlines whilst also building their confidence and ability to thrive on their BTEC journey.

Tutors/Teachers

In addition to the annual standardisation training that all BTEC teaching staff are required to complete at the beginning of each academic year using the Pearson provided materials, an induction period for new tutors is also recommended. This will help new tutors familiarise themselves with the specific demands and expectations of the BTEC curriculum, equipping them with the necessary knowledge and skills to effectively plan and support their students from the outset.

Overview of Assessment Availability

Internal Assessments

Pearson Set Assignments (PSABs) are provided by Pearson for all internally assessed units and must only be used for summative assessment.

These are available for the lifetime of the qualification and are accessible through our website. Teachers with a Pearson online account can log in through the sign-in portal to access them. Any teacher with students registered for this qualification can create a Pearson online account.

The PSAB's for computing are expected to be changed by the centre each year and guidance on the PSAB's will be availble through our website

External Assessments

External assessments are available in two series each academic year as shown below:

Dates	Jan	Mar	May/June	Aug
Assessment	External Assessments Series 1 *Not available in Jan 2026	External assessment Series 1 Results	External assessment Series 2	External Assessment Series 2 Results

Delivery and Assessment Planning

Clear unit planning and understanding is essential for a successful qualification delivery. This helps students to build on prior learning and reinforce concepts to develop a deeper understanding of the unit content and progressively develop their knowledge, understanding and skills throughout the course delivery.

We have produced a sample delivery plan showing how the Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Extended Certificate) **could** be delivered over **two** years, highlighting ordering of units and assessment milestones.

This plan is intended to be used as guidance.

Key

Del = Unit content delivery

PSAB = Pearson Set Assignment Brief

Rev = Revision for External assessment

Ext = External assessment

Resit Ext = Resit External assessment opportunity

Sequence of delivery

Year One

Unit	Unit Title	GLH	Assessment method	Term 1	January exam series	Term 2	Term 3	Summer exam series
1	Programming Fundamentals	120	Ext	Del (Topic A & B)		Del (Topic C & D)	Rev	Ext
3	Human-Computer Interaction	60	Int	Del (Learning Aim A)		Del (Learning Aim B & C)	PSAB	

Unit 1 Programming Fundamentals: This unit is an externally assessed mandatory unit that lays the foundation for the entire course. This unit provides essential underpinning knowledge of computing facts, terms, standards, concepts, technologies, and processes.

It also supports learning in other units such as Topic C1 introduces students to different methods for expressing algorithms and solutions, which can later be applied in Unit 3, Topic B2 when creating design documentation for a Human-Computer Interaction (HCI) solution. Additionally, students can draw on content in Unit 4, applying concepts in Topic B1 to develop a computer program to solve a problem. As this the first external assessment that students will sit, assessment is planned for the summer exam season. This is to allow centres sufficient time to teach the unit content and allow students to develop their knowledge of the command words that are used. Following this approach will allow students to take advantage of a resit opportunity in the second year of study if needed.

Unit 3 Human-Computer Interaction: This unit is internally assessed via a 20-hour PSAB which is marked by centres and verified by Pearson. Students will explore the core principles of human-computer interaction (HCI), focusing on how to design effective interfaces that enable seamless user interaction with programs.

Students will draw on the knowledge, understanding, and skills developed in Unit 1. This is because students gained an understanding of various programming fundamentals used to create programs. This unit builds on that foundation, helping students understand how to develop effective HCI to facilitate user interaction with programs. Students will also be able to directly apply specific topics from Unit 1. For example, they can utilise their knowledge of Topic A2: Fundamentals of Data and Logic, Topic C1: Problem Solving, Topic C2: Developing High-Quality Computer Programs, and Topic D: Issues Relating to Developing Computer Programs. This knowledge will support them in designing and developing an HCI that effectively meets the requirements of a given brief.

Year Two

Unit	Unit Title	GLH	Assessment method	Term 1	January exam series	Term 2	Term 3	Summer exam series
1	Programming Fundamentals	120	Ext		Resit Ext			
2	Computer Network Security and Encryption	120	Ext	Del (Topic A, B, C & D)	Ext			Resit Ext
4	Practical Programming	60	Int			Del (Learning Aim A & B)	PSAB (36 hours)	

Unit 2 Computer Network Security and Encryption: This mandatory unit is externally assessed and equips students with the knowledge and skills to understand and apply computer network security and encryption techniques. In Unit 1, students gained an understanding of key programming concepts used to create programs. Building on this, Unit 2 will enable students to identify and address the various risks that must be considered and mitigated during program development.

Students will also be able to build on their familiarity with command words used in the external assessment for unit 1 such as such as "complete," "describe," "develop," and "evaluate," as these remain consistent. The assessment for this unit is planned for the January window, allowing students the option to resit during the summer exam series if needed.

Unit 4 Practical Programming: This unit is internally assessed through a 36-hour PSAB, which will be marked by centres and verified by Pearson. Students will explore the principles of computer science in relation to software development concepts and will adopt a systematic and methodical approach to managing the development of a software solution to a given problem.

Students will draw upon their knowledge from across all other units. For instance, they will be able to apply many of the facts, terms, standards, concepts, technologies, and processes from Unit 1 to their own software solution. From Unit 2, students will utilise their understanding to identify and mitigate potential security risks. Finally, knowledge from Unit 3 will enable students to incorporate the fundamental concepts of human-computer interaction into their program.

Centres may deliver the qualification over a one-year period if required to provide flexibility to meet student or centre qualification planning needs.

4. Qualification Unit Delivery Guides

This section contains support for delivery of all the units in this qualification. The focus of these guides is on structuring and supporting the teaching and learning process. You will find ideas for activities and guidance on how best to use the activities to develop students understanding of the topics in each unit. This section also includes activities and information on how to deliver transferable skills which are embedded or signposted in the qualification.

Unit 1: Programming Fundamentals

Unit overview

Unit 1: Programming Fundamentals	
Assessment type: External	
Content Area	**Topics**
A: Introduction to programming, logic and number	A1 Number systems used in computers A2 Fundamentals of data and logic A3 Program structure
B: Extending program functionality	B1 Data structures B2 Built-in functions
C: Developing programs to solve problems and specific requirements	C1 Problem solving C2 Developing high-quality computer programs
D: Issues relating to developing computer programs	
Assessment overview The unit will be assessed through one examination of 90 marks lasting 2 hours and 30 minutes. Students will be assessed through a number of short- and long-answer questions. The questions will assess knowledge and understanding of programming concepts and how they are used to solve problems. The assessment availability is twice a year in January and May/June. The first assessment availability is May/June 2026. Sample assessment materials will be available to help centres prepare students for assessment	

Common student misconceptions

Below are some common misconceptions related to the content of this unit by students and ideas for how you can help your students to avoid and overcome these.

What is the misconception?	How to help students overcome it
Students often think that data validation is performed on data to ensure that it is correct, rather than a check to ensure that the data is reasonable, sensible, and within set boundaries.	Model validation rules on data to demonstrate how data can be incorrect while still meeting the validation criteria. For example, a presence check will verify that data has been entered, but it does not ensure that the data is correct. A format check can confirm that a date of birth is structured correctly but does not validate its accuracy.
Students often confuse recursion with iteration, thinking they are the same concept. However, recursion is a method where a function calls itself and iteration involves using loops to repeatedly execute a block of code until a specified condition is met.	Demonstrate tracing both types of algorithms to compare a simple recursive function, such as calculating the Fibonacci sequence, with its iterative counterpart. As you trace the execution of both functions, students can observe how the recursive function calls itself multiple times and builds a call stack, while the iterative function uses a loop to achieve the same result.
Students may struggle with converting between the different data types, especially understanding when to use int(), float(), or str() with input().	Create a hands-on exercise where students collect user input, display its type, and convert it using int(), float(), and str(). Show examples of inputs that need to be converted for calculations. Emphasise the common error of trying to perform mathematical operations on strings without converting them first.
Students may find understanding the flow of conditional statements and how elseif/elif and else are used confusing.	Use flowcharts to represent how conditions are checked visually. Provide exercises where students write multiple conditions and predict the output based on different inputs. Gradually introduce match by showing how it can simplify complex if-elseif/elif chains.
Students often mix up while and for loops, not understanding when to use each.	Compare while and for with specific examples such as using for loops for a known number of iterations and while loops for conditions that may not have a set endpoint. Create practical tasks, such as using a for loop to print a list and a while loop to get user input until they type "quit."

What is the misconception?	How to help students overcome it
Students may not understand the differences between random(), randint(), and sample() and what they do.	Run a class activity where each student “rolls a dice” using randint(), generates a random decimal with random(), and picks random names from a list with sample().
Students may not know when to use import on its own or from … import and might struggle with accessing functions correctly.	Provide a simple example where they use import math to call math.sqrt() and compare it to from math import sqrt to call sqrt() directly. This will illustrate how each form changes how they access functions from libraries.
Students might not understand the difference between variables defined inside functions (local) and those defined outside (global), leading to confusion when accessing or modifying variables.	Provide examples that illustrate variable scope. Start with a simple program where a variable is defined both globally and within a function. Show how changes inside the function don't affect the global variable unless global is explicitly used.

Learning Activities and Resources

This section offers a starting point for delivering the unit by outlining a logical sequence through the unit topics and suggesting practical activities and teacher guidance for covering the main areas of content during guided learning time. Transferable skills are integrated into various activities, with those embedded in a unit indicated by an acronym in square brackets. The acronym combines the letters from the broad skill area and the specific transferable skill, e.g., **[IS-WC]**.

Please note that the activities provided below are suggestions and not mandatory.

Learning Topic	Activities and guidance for unit content delivery	Resources
A1 Number systems used in computers	• **Whole class and individual activity – Positive binary and denary** o Tell the class the difference between the binary and denary number systems and why they are needed. Model how to convert positive denary numbers to binary and vice versa. o Give students consolidation questions requiring them to practise converting between binary and denary number systems. Students can use online binary calculators to check their answers. • **Whole class and individual activity – Negative binary** o Model to the class how to represent negative numbers. First, show students how to do this using one's complement and then two's complement. o Give students consolidation questions requiring them to practise converting negative denary numbers into one's and two's complement. • **Whole class and individual activity – Floating-point binary** o Once students are familiar with binary and denary number systems, including negative binary numbers, introduce students to floating-point binary. Tell students why floating-point numbers are used and the difference between the mantissa and exponent. o Next, model how to convert positive and negative denary numbers into floating-point binary numbers.	Online binary calculator – Rapid Tables – rapidtables.com **Calculator.net** – calculator.net **BBC Bitesize** – Notes on binary and denary number systems, – bbc.co.uk/bitesize **ADA Computer Science** – Notes on negative binary numbers (one's and two's complement) – adacomputerscience.org **ADA Computer Science** – Notes on floating-point binary numbers – dacomputerscience.org **Uobabylon** – Notes on binary arithmetic – uobabylon.edu Notes on binary shifts, e.g. ADA Computer Science – adacomputerscience.org

	o Discuss the impacts of increasing the number of bits in both the mantissa and exponent. o Give students consolidation questions requiring them to practise converting positive and negative denary numbers into floating-point binary. - **Whole class and individual activity – Binary Arithmetic and overflow errors** o Tell students that the role of the CPU is to complete calculations. Basic calculations include addition, subtraction, division, and multiplication. o Model to the class how to carry out these operations. While doing this, model how overflow errors can occur, what these are, what happens when they occur and how they can be avoided o Next, give students consolidation questions requiring them to practise calculations. Students can use online binary calculators to check their answers. - **Whole class and individual activity – Binary shifts** o Model to the class how to carry out left and right binary shifts. Ask students to identify what has happened to the numbers once the shifts have taken place and if they can identify the purpose of each one. Again, model how overflow errors can occur and reinforce what they are, the problems they can cause, and how they can be avoided. o Give students consolidation questions requiring them to practise carrying out left and right binary shifts on different binary numbers and different shift amounts.	
A2 Fundamentals of data and logic	**Note:** Students will be expected to write, interpret and debug code and algorithms using the programming language Python 3 version 3 .10 (or higher). - **Whole class teaching and learning – Introduction to Python programming** o Ask students what is meant by the term programming code. Ask them what programming languages they have used before and what sorts of programs they have developed.	**Python tutorials**, e.g. Python – python.org **PythonByteSize** – pythonbytesize.com **Practical python activities**, e.g. W3Schools – w3schools.com

	o Next, tell students that in this unit they will be learning how to use the Python programming language and demonstrate the use of basic Python techniques. Demonstrate how to set up variables and constants and assign data types to them. Model good practice when naming variables and constants and discuss the use of suitable identifier names. o Next, demonstrate how to use the input() function to receive user input and assign this to a variable and then demonstrate the print() function to display output in Python. • **Paired activity – Python basics consolidation exercises** o Next, give students practical exercises that allow them to consolidate their understanding of variables, constants, data types, input, and output. For example, students could work through the first few Python tutorials at W3Schools.com (e.g., home, introduction, getting started, syntax, variables, and data types). • **Whole class activity and individual activity – Mathematical, relational and Boolean operators** o Tell students the different types of operators (e.g., mathematical, relational, and Boolean). For each type of operator, cover the operators that students are expected to learn as listed in the specification. o Next, demonstrate how to use these operators in Python code. Show examples of how these operators work with various data types, including integers, floats, and strings. Emphasise the importance of using the correct data types with each operator to avoid errors. For instance, demonstrate what happens if you try to add a number to a string, and explain why Python raises an error in such cases. • **Paired activity – Python operators consolidation exercises** o Next, give students practical exercises for them to practise using the different operators. For example, students can be given code that uses incorrect operators and could be tasked to find and correct them. Students could also, for example, work through the tutorials at W3Schools.com.	**Geeks For Geeks** – Notes on Python data types – geeksforgeeks.org

A3 Program structure	• **Whole class teaching and learning – Programming constructs** o Tell students what is meant by the term construct and explain the difference between sequence, selection (branching), and iteration. Ask students how they have used these constructs before and why they are used in programming code. • **Whole class and individual activity – Selection (Branching)** o Reinforce what is meant by the term selection and demonstrate how to use IF statements in Python. Start by creating a single IF statement with a simple condition, then build this up to use an ELSE statement and then an ELIF statement. o Next, demonstrate the use of switch/case (match) and discuss the similarities and differences with IF statements. o Give students practical exercises for practising IF statements and switch/case. For example, students could work through the tutorials at W3Schools.com. • **Whole class and individual activity – Iteration** o Reinforce what is meant by the term iteration and demonstrate how to use FOR loops in Python. Next, demonstrate how to use WHILE loops in Python and discuss the similarities and differences between them. Discuss the types of problems suited to each type of loop. o Give students practical exercises for practising FOR loops and WHILE loops. For example, students could work through the tutorials at W3Schools.com. • **Whole class and individual activity – Functions** o Show students an example of a program containing lots of repeated code, such as a basic calculator where each mathematical operation (addition, subtraction, multiplication, and division) is written out multiple times in different parts of the code, rather than using functions to avoid repetition. Discuss the problems this can cause and possible solutions. o Next, introduce students to the idea of a function and demonstrate how to write functions in Python. Explain how to define functions, call functions, define parameters, pass parameters, and return values.	**Python** – Python tutorials, – python.org **PythonByteSize** – pythonbytesize.com **W3Schools** – Practical python activities – w3schools.com **Geeks For Geeks** – Notes on selection statements – geeksforgeeks.org Notes on iteration statements, e.g. Geeks For Geeks – geeksforgeeks.org Notes on Python functions, e.g. Geeks For Geeks – geeksforgeeks.org Notes on recursion, e.g. Geeks For Geeks – geeksforgeeks.org

	o Give students practical exercises for practising writing functions. For example, students could work through the tutorials at W3Schools.com. • **Paired activity – Local and global variables** o Ask students to research the difference between local and global variables. For each, students should research what they are, how they are defined, the benefits, the potential issues, and how to mitigate/reduce the problems caused by global variables. Students should then share their answers with the rest of the class. o Next, give students programming code that contains issues created by using global variables. Ask students to find the problems and provide a fix for them. • **Whole class and individual activity – Recursion** o Remind students what is meant by the term iteration. Provide some iterative code (e.g., to calculate factorials) and trace the code through. Reinforce how the code is repeated, and the variables are reused for each repetition. o Next, show students the code to solve the same problem using recursion. Ask students to identify how the code differs from iteration and trace the algorithm through and reinforce how the function is called from within itself, and a copy of the variables is made for each recursive call. • **Paired activity – Benefits and drawbacks of recursion** o Provide students with code that uses iteration and ask them to rewrite it using recursion, and vice versa. o Ask students to research the benefits and drawbacks of recursion. Students should then share their answers with the rest of the class.	

B1 Data structures	• **Whole class teaching and learning – Introduction to data structures** o Show students some code containing various variables, each storing a number. Ask students to identify which variables could be grouped to simplify the code and make it more manageable. o Next, introduce students to the meaning of a data structure and how they can be manipulated using indexes. • **Whole class teaching and learning – Arrays** o Explain what is meant by the term array and discuss different characteristics of arrays, including fixed vs dynamic arrays and single vs mixed data types. Next, show how to write a single-dimensional array in Python. Then demonstrate how to write a two-dimensional array and discuss the similarities and differences between them. Discuss different uses of arrays and what sorts of problems they can solve. o Give students practical exercises to practise using single-dimensional and two-dimensional arrays, including adding, updating, and removing data from them. For example, students could work through the tutorials at W3Schools.com. • **Whole class teaching and learning – Lists and Tuples** o Ask students to discuss examples of when the contents of a data structure may need to remain fixed (e.g., to store the days of the week) and when the contents may need to change (e.g., when updating a quiz score). o Reinforce what is meant by the term data structure and demonstrate how to use lists in Python. Next, demonstrate how to use tuples and discuss the similarities and differences between them. Discuss the types of problems that each of these is suited to. o Give students practical exercises to practise using lists and tuples, including adding, updating, and removing data from them. For example, students could work through the tutorials at W3Schools.com.	Python tutorials, e.g. Python – python.org PythonByteSize – pythonbytesize.com Practical python activities, e.g. W3Schools – w3schools.com Notes on arrays, lists and tuples, e.g. DevOps School – devopsschool.com Notes on dictionaries, e.g. Geeks For Geeks – geeksforgeeks.org Notes on stacks, e.g. ADA Computer Science – adacomputerscience.org Notes on queues, e.g. ADA Computer Science – adacomputerscience.org

	• **Whole class teaching and learning – Dictionaries** o Start by asking students how they store contact details for friends and family. Discuss the importance of linking names and phone numbers together. o Next, demonstrate how to use dictionaries in Python and discuss the types of problems they are suited to. o Give students practical exercises to practise using dictionaries, including adding, updating, and removing data. For example, students could work through the tutorials at W3Schools.com. • **Whole class teaching and learning – Stacks and queues introduction** o Use real-world scenarios to illustrate the concept of a stack data structure. For example, ask students to imagine they are in a restaurant and consider which plate they would take first. Emphasise that the last plate added is the first to be removed, representing the LIFO (Last In, First Out) principle. o Next, use real-world scenarios to illustrate the concept of a queue data structure. For example, ask students to imagine they are waiting in line to buy a ticket, with the line getting longer as people join and leave. Ask students what happens if someone arrives late and where they go. Highlight how this relates to the FIFO (First In, First Out) principle. o Demonstrate how stacks and queues can be implemented in Python using associated built-in functions. • **Paired activity – Stacks and queues reinforcement activities** o Explain to students that stacks and queues can be implemented using arrays/lists and that these can be manipulated with pointer values. Discuss the steps involved when adding and removing data from both stacks and queues. For example, before data is added to a stack, it checks if the stack is full; if not, it will increment the pointer and add the data, otherwise, it will output a message stating the stack is full.	

	○ Next, ask students to create a program to simulate a stack using two functions. The first function should add data to the stack or output a suitable message if the stack is full. The second function should remove data or output a suitable message if the stack is empty. ○ Then ask students to create another program to simulate a queue using two functions to add and remove data.	
B2 Built-in functions	**Note:** Students will be expected to write, interpret and debug code and algorithms using the programming language Python 3 version 3 .10 (or higher). Appendix 1 of the specification provides a list of key functions, methods and libraries that students should be able to use. • **Whole class teaching and learning – Built-in functions** ○ As a class, write a piece of code that uses a list to hold 10 items. Then extend the code to output the minimum and maximum value in the list. ○ Next, explain what is meant by the term built-in function and demonstrate how to use the min() and max() built-in functions to perform the same task for the code written. Discuss the benefits and drawbacks of using built-in functions. • **Paired activity – Introduction to numerical functions** ○ Provide students with a list of numerical functions as stated in Appendix 1 of the specification (e.g., random, randint, uniform, sample, range, round, etc.). Ask students to research each function and write down two example uses of each. Students should then share their findings with the rest of the class. • **Individual activity – Using numerical functions in Python** ○ Give students practical exercises that allow them to use the numerical functions listed in Appendix 1 of the specification. For example, students could be given a list of football scores and asked to calculate statistics using min, max, mean, and count. Students could also create a two-player game involving random dice rolls, where they use randint to determine outcomes and apply max to identify the winner, while tracking scores with count to see how many rounds each player wins.	Python tutorials, e.g. Python – python.org PythonByteSize – pythonbytesize.com Practical python activities, e.g. W3Schools – w3schools.com Notes on text file handling, e.g. Geeks For Geeks – geeksforgeeks.org

	• **Paired activity – Introduction to string handling functions** o Provide students with a list of string handling functions as stated in Appendix 1 of the specification (e.g., isupper, islower, upper, lower, etc.). Ask students to research each function and write down two example uses of each. Students should then share their findings with the rest of the class. • **Individual activity – Using string handling functions in Python** o Provide students with practical exercises that allow them to use the string handling functions listed in Appendix 1 of the specification. For example, students could add validation rules to some simple user input using string handling functions. They could use isalpha to ensure the name input contains letters only, isdigit to verify that the telephone number contains numbers only, upper to format postcode letters to uppercase, or split to separate the country code from the mobile phone number. • **Whole class teaching and learning – Introduction to text file handling** o Discuss what happens to all data stored within variables and data structures once a program ends. Discuss scenarios where data needs to be saved beyond the program's execution. o Explain that one common method of storing data permanently is using a text file. Discuss what is meant by the term delimiter and provide common examples, such as commas and whitespaces. o Outline the stages involved in working with text files, including opening the file, reading/writing data to/from the file, and then closing the file. • **Individual activity – Using text file handling in Python** o Provide students with practical exercises that allow them to use the text file commands listed in Appendix 1 of the specification. For example, students could create a program that allows teachers to log student grades. They could write a function to enter a list of student names and grades, then save this data to a text file. Students could then create additional functions to update test scores and read data from the text file, displaying the average score or listing students who achieved a specific score.	

C1 Problem solving	- **Individual activity – Computational thinking methods** o Ask students to research what is meant by the term computational thinking methods and the meaning of decomposition, pattern recognition, abstraction, and algorithmic design. o Provide students with a scenario and ask them to apply different computational thinking methods to it. For example, a banking application that allows users to create accounts, deposit and withdraw money, and check their balances, or a recipe management system that enables users to store, search, and manage their recipes. - **Whole class and paired activity – Algorithms** o Explain the term algorithm and tell students that there are different ways to express algorithms, such as pseudocode, flowcharts, and program code. o Show students the different flowchart symbols as listed in Appendix 2 of the specification and demonstrate how to draw simple flowcharts, modelling good practices and flowchart conventions. o Next, provide students with problems and ask them to solve these by drawing a flowchart. Begin with simple exercises by asking students to draw flowcharts in sequence. Then, move to flowcharts that use selection and iteration. Once students have understood these concepts, ask them to draw flowcharts that include subprograms. o Then, ask students to write pseudocode/program code for each of their algorithms, and discuss the benefits and drawbacks of each method for representing algorithms. - **Whole class and individual activity – Interpreting algorithms** o Explain what is meant by the term trace table. Show an algorithm to students and demonstrate how to create a trace table with suitable headings. Then trace the algorithm, showing students how to determine its purpose. o Give students additional algorithms and ask them to create and complete a trace table for each algorithm.	Notes on computational thinking methods, e.g. ADA Computer Science – adacomputerscience.org Flowchart software, e.g. Lucidchart – lucidchart.com Notes on linear and binary searches, e.g. Geeks For Geeks – geeksforgeeks.org Notes on bubble and quick sorts, e.g. Medium – medium.com

	• **Whole class and individual activity – Search algorithms** o Discuss practical examples of when a program may need to search for data. For example, a banking application may need to search for a specific user's account information, or an e-commerce website may search its product database to display items matching a user’s search query. o Provide students with a list of unsorted numbers (e.g., 9, 6, 4, 1, 5, 7, 10, 8, 2, 3) and demonstrate the steps a linear search would take to find the value 5. Then, provide the same numbers in sorted order and demonstrate the steps a binary search would take to find the value 5. Discuss the differences between these approaches and the potential benefits and drawbacks of each, especially when searching through large lists. o Give students Python code representing each search algorithm and ask them to interpret and/or debug it. For example, students could be given code with errors to fix or steps missing to complete. Alternatively, students could be asked to comment on the code to show their understanding of each line. • **Whole class and individual activity – Sort algorithms** o Discuss practical examples of when a program may need to sort data. For example, a library management system may need to sort books by title, author, or publication date to make it easier for users to locate specific titles. o Provide students with a list of unsorted numbers (e.g., 9, 6, 4, 1, 5) and demonstrate the steps a bubble sort would take to arrange the values in ascending order. Then demonstrate the steps a quick sort would take to order the same values. Discuss the differences between these sorting approaches and the potential benefits and drawbacks of each, particularly when dealing with large lists.	

	o Give students Python code representing each sorting algorithm and ask them to interpret and/or debug it. For example, students could be given code with errors to fix or missing steps to complete. Alternatively, students could be asked to add comments to the code to show their understanding of each line.	
C2 Developing high-quality computer programs	• **Whole class and individual activity – Program maintainability** o Discuss practical examples of when program code may need to be changed or maintained, such as to comply with new legislation, fix security vulnerabilities, or meet new user needs. o Next, show students some code that does not follow typical programming conventions, for example, code with inconsistent styles, inconsistent naming conventions, poor identifier names, limited or poor code comments, or repeated code. Discuss the implications of maintaining this code and techniques to ensure code is readable and maintainable. o Then, allow students to implement the techniques discussed to improve code quality and maintainability. • **Paired activity – Data validation** o Explain the term data validation to students. Then display different data validation methods listed in the specification and discuss their purposes. o Show examples of user input forms, which could be electronic or paper based. Then discuss how data validation ensures that data is sensible and reasonable. o Ask students to write code that creates validation rules for data. They can use Python built-in functions or implement custom validation logic. For instance, students could create presence checks, type checks, and format checks for fields like postcodes. o Then, discuss actions that follow a validation check, such as providing user feedback, and ask students to implement these actions in their code.	Notes on code maintainability, e.g. Coveros – coveros.com Notes on test data, e.g. ADA Computer Science – adacomputerscience.org

	• **Small group activity – Providing meaningful user interactions** o Discuss the importance of user interaction in applications, emphasising how meaningful interactions can enhance user experience, make software more intuitive, and build user trust. o Present examples of good and bad user interactions from existing applications, such as chatbots, forms, and menus. o Divide students into small groups and ask them to design a chatbot on a topic of their choice. Ask students to create a flowchart or storyboard outlining how users will interact with the chatbot, including welcome messages, user options, potential questions and responses, and error handling for unrecognised input. Then, allow students to code their chatbots based on their designs, ensuring they implement meaningful interactions. • **Individual activity – Purpose of testing** o Explain what is meant by testing and why it is crucial, highlighting that testing ensures a program function's correctly and produces the expected outcomes. o Ask students to research past software or ICT system failures, focusing on what went wrong and the consequences of insufficient testing. Students should then share their findings with the class. • **Whole class and individual activity – Test data** o Explain test data and the different types of test data, such as valid, invalid, extreme, and erroneous. Discuss the importance of testing a program with various types of test data and combinations. o Show students some user requirements and programming code. Ask them to think of as many pieces of test data as possible, covering different types. For example, they could include valid data that meets all requirements, invalid data that violates specific rules, boundary data that tests value limits, and extreme data that pushes the program to its limits. Encourage students to consider edge cases and think about how the code should handle unexpected inputs, such as empty strings, special characters, or out-of-range numbers.	

D Issues relating to developing computer programs	- **Whole class and individual activity – Third-party code** - Discuss what is meant by third-party code and why programmers may choose to use code written by others. Provide examples of different types of third-party code, such as open-source code, Application Programming Interfaces (APIs), closed-source or proprietary APIs, and code generated by AI. - Ask students to consider factors that should be considered when using third-party code, including compatibility, costs, licensing issues, legal concerns, and contractual obligations. For each factor, students should provide an example of how it could impact a project. For instance, they might examine how compatibility issues could arise if third-party code is not updated for newer software versions or discuss how licensing restrictions might limit code modifications. They could also reflect on the impact of legal and contractual requirements on long-term maintenance and ownership. Students should then share their findings with the class. - **Group activity – Considering diversity and inclusion when developing programs** - Introduce students to the Web Content Accessibility Guidelines (WCAG), covering the principles of perceivable, operable, understandable, and robust (POUR). Give a brief overview of each principle and ask students to think of any websites they have used that meet or fail to meet these principles. Discuss the impact of accessibility issues on users. - Divide students into small groups and assign each group a website (or webpage) to evaluate against WCAG principles. Ask them to identify at least one improvement that could enhance the page's accessibility, focusing on any of the POUR principles. Each group should then share their findings with the class, explaining how their recommended change would make the site more accessible.	Notes on WCAG principles, e.g. W3C – w3.org Notes on data set bias, e.g. The Beautiful Truth – thebeautifultruth.org

	• **Whole class teaching and learning – Considering potential bias in data sets** o Discuss the potential for bias in data sets and introduce students to different types of bias: sampling, measurement, response, analysis, and reporting bias. o Use practical examples, such as biased sample data in facial recognition technology (e.g., training data that primarily includes one demographic group). Emphasise how biased data can lead to unfair or inaccurate outcomes in software, especially in areas like predictive algorithms and machine learning models. o Discuss the types of bias that might arise when creating an app to monitor user activity levels and explore ways to reduce bias in such applications. • **Small group activity – Problems associated with developing computer programs** o Tell students that certain common problems are often encountered in software development. Split the class into four groups and assign each group a specific problem to research and present. *Group 1*: Research *compatibility issues*, including how compatibility can impact program performance across different systems and ways to mitigate this problem. *Group 2*: Investigate the *digital divide* in relation to access to and use of software, focusing on accessibility issues and barriers created by software design. *Group 3*: Explore *security issues* related to data threats, discussing common vulnerabilities and protective measures. *Group 4*: Research the *impacts of development time* on new programs, including how delays can affect project success and strategies for efficient project management. o Each group should explore what their assigned problem entails and suggest methods for reducing the associated risks by referring to the relevant content in the specification. Each group will then present their findings to the class.	

Delivering signposted transferable skills

Signposted transferable skills are not mandatory for the delivery of the unit, and it is therefore your decision to deliver these skills as a part of the qualification. Below we have provided some ideas of teaching and learning activities that you could use to deliver these skills if you chose to.

Transferable skills	Ideas for delivery
SP – PS Problem solving	There are lots of opportunities for students to demonstrate problem solving. For example: • Students can break down complex problems into smaller, manageable tasks – This allows them to approach each part individually and make the problem less overwhelming, showing their ability to simplify and structure tasks logically. • Students can use debugging techniques to find and fix errors – By testing their code and identifying issues, they show resilience and persistence in troubleshooting, which is essential for solving programming problems effectively. • Students can research and apply relevant algorithms – Choosing appropriate algorithms to solve specific problems demonstrates their understanding of programming concepts and their ability to apply theoretical knowledge practically. • Students can optimise code to improve efficiency – Refining their code to make it run faster or use fewer resources shows their ability to improve solutions, making them more practical and efficient. Students can write pseudocode or flowcharts before coding – Planning their approach with pseudocode or flowcharts helps them structure their thought process, making it easier to translate ideas into code. This shows they can organise their problem-solving steps clearly.

Resources

This section has been created to provide a range of links and resources that are publicly available that you might find helpful in supporting your teaching and delivery of this unit in the qualification. We leave it to you, as a professional educator, to decide if any of these resources are right for you and your students, and how best to use them.

Pearson is not responsible for the content of any external internet sites. It is essential that you preview each website before using it to ensure the URL is still accurate, relevant, and appropriate. We'd also suggest that you bookmark useful websites and consider enabling students to access them through the school/college intranet.

Websites

PythonByteSize – A range of Python demonstration tutorial videos.
https://www.pythonbytesize.com/detailed-videos.html

The Python Tutorial – An overview of the basic concepts and features of the Python language.
https://docs.python.org/3/tutorial/index.html

W3Schools – A Python code editor and a range of Python tutorials and exercises.
https://www.w3schools.com/python/default.asp

Textbooks

Matthes, E., Python Crash Course, 3rd Edition: A Hands-On, Project-Based Introduction to Programming, 2023 (ISBN 1718502702)

Shovic, J., Simpson, A., Python All-in-One For Dummies (For Dummies: Learning Made Easy), 2024 (ISBN 1394236158)

Pearson paid resources also available

- Pearson Student book
- ActiveBook (a digital version of the Student Book, via ActiveLearn Digital Service)
- Digital Teacher Pack (via ActiveLearn Digital Service)

Unit 2: Computer Network Security and Encryption

Unit overview

Unit 2: Computer Network Security and Encryption	
Assessment type: External	
Content Area	**Topics**
A: Computer networks	A1 The main challenges of computer network security, with reference to the threats included in Section B A2 The main components of computer networks A3 Software components of computer networks
B: Network security	B1 The main threats to data and computer networks as described in Section A B2 Common methods of attacking data and computer networks B3 Methods of defending against attacks on data and computer networks B4 Legal issues related to computer network security and encryption
C: Encryption	C1 Methods and techniques for data encryption
D: Evaluating cyber security and encryption solutions	D1 Appropriate cyber security and encryption methods and techniques to secure data transmission and storage on a network D2 Suitable cyber security and encryption solutions for different types of networks based on analysis of the requirements
Assessment overview The unit will be assessed through one examination of 90 marks lasting 2 hours and 15 minutes. Students will be assessed through a number of short- and long-answer questions. The questions will assess students' understanding of computer networks, security threats and the methods used to counter them and the use of encryption to safeguard data. Students will relate their understanding to different contexts. Examination questions will use the following contexts: education, essential infrastructure, tech and software, industrial settings. The assessment availability is twice a year in January and May/June. The first assessment availability is May/June 2026. Sample assessment materials will be available to help centres prepare students for assessment	

Common student misconceptions

Below are some common misconceptions related to the content of this unit by students and ideas for how you can help your students to avoid and overcome these.

What is the misconception?	How to help students overcome it
Encryption alone makes data completely secure.	Demonstrate how encrypted data can still be vulnerable through poor key management or implementation flaws; Show real-world examples where encrypted systems were compromised; Teach the importance of comprehensive security approaches.
Network protocols like HTTPS and SSH are always secure.	Explain how protocols can be misconfigured or have vulnerabilities; Show examples of protocol downgrade attacks; Demonstrate importance of keeping protocols updated.
Password complexity is more important than length.	Use practical demonstrations of password cracking; Show mathematical probability calculations; Have students test different passwords against strength meters.
VPNs provide complete anonymity online.	Explain what VPNs actually protect against and their limitations; Show how VPN providers can still log data; Demonstrate other ways tracking can occur even with VPNs.
Firewalls block all malicious traffic and prevent malware.	Show how malware can use allowed ports/protocols; Demonstrate bypass techniques; Explain why layered security is necessary and why anti-malware software needs to be installed in addition to firewall software.
Public networks are safe if using HTTPS.	Demonstrate man-in-the-middle attacks; Show DNS hijacking examples; Explain various public network vulnerabilities.
MAC addresses can't be spoofed.	Show practical examples of MAC spoofing; Demonstrate why MAC filtering isn't sufficient security; Explain proper network access controls.
Updates only add features and aren't security critical.	Share examples of major breaches due to missing patches; Demonstrate vulnerability exploitation; Show how patches address security flaws.

Learning Activities and Resources

This section offers a starting point for delivering the unit by outlining a logical sequence through the unit topics and suggesting practical activities and teacher guidance for covering the main areas of content during guided learning time. Transferable skills are integrated into various activities, with those embedded in a unit indicated by an acronym in square brackets. The acronym combines the letters from the broad skill area and the specific transferable skill, e.g., **[IS-WC]**.

Please note that the activities provided below are suggestions and not mandatory.

Learning Topic	Activities and guidance for unit content delivery	Resources
A: Computer networks		
A1 The main challenges of computer network security A1.1 Network scale	• **Whole class instruction – Introduction to network security challenges** o Introduce the key challenges in securing different types of computer networks, touching on various network scales (personal, organisational, national, and global). o Discuss how each scale faces unique security considerations, with examples of threats from malware, DoS/DDoS, hacking, and social engineering attacks. o Use visual aids, such as infographics and concept maps, to illustrate the potential impact of these threats. • **Small group activity – Threat assessment across network scales** o In small groups, students will receive case studies representing different network scales (e.g., personal network vs. national network). Their task is to: Identify the key threats based on the network scale (e.g., insider threats, zero-day vulnerabilities). Outline the security risks and possible consequences for each case study. Each group will present their findings, helping classmates understand how threats vary across scales.	This resource outlines various network security threats, including malware, DoS/DDoS attacks, and social engineering, offering examples and preventive measures. https://www.enterprisenetworkingplanet.com/security/network-security-threats/ This resource discusses top network security threats and provides defences for each, emphasising the importance of security solutions. https://www.esecurityplanet.com/networks/network-security-threats/

	• **Pair activity – Compare and contrast network types** o Students work in pairs to create a Venn diagram that compares two network types (e.g., LAN and WAN) based on their security-related characteristics and potential vulnerabilities (e.g., malware risks, hacking susceptibility). o Pairs will present their Venn diagrams to the class and discuss which network types are more vulnerable to specific threats.	
A1.2 Security considerations when implementing topologies	• **Whole class instruction teaching and learning – Introduction** o Discuss the concept of network topologies and their implications for security. Use a class mind map to identify the star, mesh, hybrid, and ad-hoc topologies. o Highlight the security strengths and weaknesses associated with each, such as points of failure, scalability, and data interception risks. • **Small group activity – Topology comparison exercise** o Groups will analyse different scenarios and decide the most secure topology to implement. Each group will justify their choice by: Listing the security advantages and disadvantages of the topology. Identifying potential threats to the topology. Recommending additional security measures. • **Pair activity – Diagram creation and analysis** o Pairs will create detailed diagrams of one assigned topology (star, mesh, hybrid, or ad-hoc). o They will annotate the diagrams with security features and points of vulnerability. o Pairs will then swap diagrams with another pair for critique and discussion.	This resource discusses the strengths and weaknesses associated with different network topologies, such as points of failure, scalability, and data interception risks. https://www.networkstraining.com/compare-and-contrast-network-topologies/ This NIST publication provides an overview of OT systems, typical topologies, common threats, vulnerabilities, and recommended security countermeasures. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r3.pdf

A1.3 Architectures	• **Whole class instruction teaching and learning – Introduction** o Introduce the concept of network architectures by contrasting client-server and peer-to-peer models. Use a class mind map to explore the security implications of each, such as centralisation risks in client-server and the lack of centralised control in peer-to-peer networks. o Discuss common threats like data interception, unauthorised access, and malware propagation. • **Small group activity – Risk analysis of architectures** o Assign each group one of the two architectures (client-server or peer-to-peer). Their task is to: Identify and outline potential security risks. Discuss how these risks could impact an organisation. Propose specific security measures to mitigate the identified risks. o Groups will present their findings to the class. • **Pair activity – Threat scenario evaluation** o Provide pairs with a scenario involving either a client-server or peer-to-peer architecture (e.g., a data breach, malware spread, or insider attack). o Pairs will analyse the scenario and write a brief explanation of: How the architecture contributed to the issue. What steps could have been taken to prevent the problem. The strengths and weaknesses of the chosen architecture in responding to such a threat.	This guide provides an overview of client-server and peer-to-peer networks. https://www.bbc.co.uk/bitesize/guides/zvspfcw/revision/4 This survey provides insights into security challenges within peer-to-peer networks, contrasting them with client-server models. https://www.cse.wustl.edu/~jain/cse571-07/ftp/p2p/

A1.4 Connection media	• **Whole class instruction teaching and learning – Introduction** o Introduce the concept of connection media, dividing it into wired (Ethernet, USB, optical fibre) and wireless (Wi-Fi, Bluetooth, cellular (4G/5G), Zigbee). o Discuss their security characteristics, focusing on potential vulnerabilities like signal interception in wireless media or physical tampering with wired connections. • **Small group activity – Media vulnerability exploration** o Each group will analyse one type of connection media (e.g., Ethernet, Wi-Fi, Zigbee) to: Identify potential security threats specific to their assigned media. Outline the implications of these threats for organisations or users. Recommend measures to enhance security. o Groups will rotate and add to other groups' analyses to ensure an exploration of all media types. • **Pair activity – Comparison chart creation** o Pairs will collaborate to create a comparison table that evaluates wired and wireless media on: Security risks. Cost-effectiveness. Suitability for specific environments (e.g., corporate, industrial, IoT). o The table will include examples of where each media type is commonly used.	This article discusses security risks associated with wireless peripheral devices and suggests remediation strategies. https://www.cyberdefensemagazine.com/wireless-peripheral-devices-security-risk-exploits-and-remediation/ This article outlines significant security risks associated with wireless networks and discusses mitigation strategies. https://www.portnox.com/cybersecurity-101/wireless-network-security-risks/ This article discusses vulnerabilities in Ethernet VLAN stacking that could allow attackers to launch denial-of-service and man-in-the-middle attacks. https://www.bleepingcomputer.com/news/security/ethernet-vlan-stacking-flaws-let-hackers-launch-dos-mitm-attacks/

A1.5 Network types	• **Whole class instruction teaching and learning – Introduction** o Introduce the concept of network types, including LAN, WAN, PAN, VPN, wireless networks, intranet, extranet, cloud networks, and IoT networks. o Use a collaborative mind map to outline the primary characteristics and purposes of each type. o Discuss the unique security challenges each network type will face, such as unauthorised access in cloud networks or vulnerabilities in IoT networks due to poor device security. • **Small group activity – Network type risk analysis** o Assign each group a network type to investigate (e.g., LAN, WAN, IoT). Groups will: Identify potential security threats for their assigned network type. Discuss how these threats impact data confidentiality, integrity, and availability. Recommend security practices to address these risks. o Each group will present their findings to the class. • **Pair activity – Case study review** o Provide pairs with a case study related to a security issue in a specific network type, such as a data breach in a cloud network or a malware attack in a VPN. Pairs will: Analyse the case to identify the vulnerabilities exploited. Summarise the security measures that could have prevented the issue. Share their conclusions with another pair for peer review.	This article outlines best practices for securing Wide Area Networks and Local Area Networks, addressing common threats and mitigation techniques. https://www.layer8packet.io/home/network-security-best-practices-for-wan-and-lan-protecting-your-infrastructure This case study examines a ransomware attack initiated through a VPN vulnerability, highlighting the importance of regular patching. https://www.cfc.com/en-gb/knowledge/resources/case-studies/cyber-claims-case-study-vulnerable-vpn/

A1.6 Security implications of common network features	• **Whole class instruction teaching and learning – Introduction** o Introduce common network features such as data sharing and storage, resource sharing, communication, creation of larger systems, and performance (latency and bandwidth). o Discuss how these features can create security vulnerabilities. o Use a collaborative class mind map to explore examples of how these features might be exploited, such as eavesdropping during communication or data breaches in shared storage systems. • **Small group activity – Feature vulnerability analysis** o Assign each group a network feature (e.g., resource sharing or latency performance). Groups will: Identify the key security risks associated with their assigned feature. Explain the potential impact on an organisation if these risks are exploited. Suggest security strategies to mitigate these vulnerabilities. o Groups will share their analyses through a gallery walk, where other students can leave questions or comments on their work. • **Pair activity – Case scenario evaluation** o Provide pairs with a network scenario highlighting one or more common features (e.g., a collaborative workspace with shared data storage). Pairs will: Analyse the scenario to identify which network features are at risk. Propose a set of specific security measures to address the vulnerabilities. Present their findings to the class in a concise oral summary.	This article examines current network security threats, including social engineering attacks like vishing, and discusses their implication https://www.checkpoint.com/cyber-hub/network-security/what-is-network-security/top-network-security-issues-threats-and-concerns/ This article explores the security risks associated with information sharing in networks and suggests strategies to mitigate them. https://radware.com/blog/security/security-risks-network-perimeter/

A2.1 Memory types	- **Whole class instruction teaching and learning – Introduction** o Introduce memory types used in computer networks: RAM, ROM, EPROM, and EEPROM. o Use a class concept map to outline their characteristics, purposes, and roles in network components. o Discuss the security implications, such as unauthorised access to ROM or data recovery from volatile RAM and explain how different types are utilised in securing and operating network systems. - **Pair activity – Practical memory role analysis** o Provide pairs with a practical scenario, such as a networked device (e.g., a router or IoT device) that relies on specific memory types. Pairs will: Analyse the role of the assigned memory type in the device. Explain potential security risks and how to mitigate them. Share findings in a quick report or verbal summary. - **Individual activity – Annotated diagram** o Students will create an annotated diagram that illustrates the relationships and roles of the four memory types within a networked device. They will label security features, potential risks, and describe how the memory types interact to support the device's operation.	This article provides an overview of RAM and ROM, detailing their characteristics, purposes, and roles in computer systems. https://www.geeksforgeeks.org/random-access-memory-ram-and-read-only-memory-rom/ This article discusses the security features of EEPROM, highlighting its non-volatile nature and applications in embedded systems. https://nexusindustrialmemory.com/eeprom-and-security-harnessing-its-potential-for-robust-data-protection/ This article provides detailed information on different types of ROM, including their characteristics and applications. https://studyelectrical.com/2017/06/different-types-of-rom-prom-eprom.html
A2.2 Storage types A2.2.1 primary and secondary storage A2.2.2 internal storage A2.2.3 removable or external storage	- **Whole class instruction teaching and learning – Introduction** o Introduce storage types, including primary storage (RAM, cache), secondary storage (hard drives), and removable/external storage (USB flash drives, external drives, optical media). o Use a flow diagram to categorise these storage types and outline their purposes. o Discuss the security implications, such as data theft from removable storage, or malware on external drives, and introduce measures like encryption or access control.	SSD vs Hard Drive Vs Hybrid Drive https://www.youtube.com/watch?v=1cyMTl_QXSc RAID Animated https://www.youtube.com/watch?v=U-OCdTeZLac

	• **Small group activity – Storage type risk assessment** o Assign each group one storage type (e.g., primary, secondary, or external/removable storage). Groups will: Explore the role and typical use of their assigned storage type in a network. Identify potential security vulnerabilities, such as data interception or unauthorised access. Propose strategies for mitigating these vulnerabilities. o Each group will present their findings using a flowchart or diagram. • **Pair activity – Case study analysis** o Provide pairs with a case study where storage security was compromised (e.g., data loss due to an unencrypted USB drive). Pairs will: Identify which type of storage was involved. Analyse how the breach occurred and its consequences. Suggest practical measures to prevent similar incidents.	
A2 The main components of computer networks A2.2 Storage types A2.2.4 data back-up and recovery systems	• **Whole class instruction teaching and learning – Introduction** o Introduce data backup and recovery systems, including RAID arrays, NAS, and SAN. o Use a diagram to illustrate how these systems function and their importance in maintaining data integrity and availability. o Discuss the role of these systems in network security, such as preventing data loss during hardware failures or attacks and highlight potential vulnerabilities like unauthorised access or misconfigured systems. • **Small group activity – Backup system evaluation** o Assign each group one backup and recovery system (RAID, NAS, or SAN). Groups will: Describe how their assigned system works and its primary use cases. Identify potential security vulnerabilities (e.g., RAID failures, NAS hacking, or SAN misconfigurations). Propose strategies to secure the system and improve reliability. o Groups will present their findings as a short verbal presentation or a flowchart.	NAS vs SAN – Network Attached Storage vs Storage Area Network https://www.youtube.com/watch?v=3yZDDr0JKVc What is RAID Parity? https://www.youtube.com/watch?v=BjuBloMHhKk

	• **Pair activity – Disaster recovery scenario** o Provide pairs with a network disaster scenario, such as a server crash or ransomware attack. Pairs will: Decide which backup system (RAID, NAS, or SAN) would be most suitable for recovery. Explain how the system would be utilised to restore operations. Suggest additional measures to strengthen the organisation's disaster recovery plan. • **Individual activity – System comparison infographic** o Students will create an infographic comparing RAID, NAS, and SAN based on: Functionality. Security features. Cost-effectiveness. Common use cases. o The infographic will highlight strengths, weaknesses, and practical applications of each system.	

A2.2 Storage types A2.2.5 cloud storage	- **Whole class instruction teaching and learning – Introduction** o Introduce cloud storage, explaining its functionality, advantages (e.g., accessibility, scalability), and security challenges (e.g., data breaches, unauthorised access). o Use a cause-and-effect diagram to explore how cloud storage works and potential threats, such as shared vulnerabilities in multi-tenant environments or weak access controls. - **Small group activity – Cloud storage threat analysis** o Divide the class into groups, assigning each a specific cloud storage security risk, such as: Data breaches. Insider threats. Insufficient access control. Misconfigured servers. o Groups will: Research their assigned risk. Identify real-world examples. Propose solutions to mitigate these risks. o Groups will present their findings using a process infographic. - **Pair activity – Use case evaluation** o Provide pairs with cloud storage use cases (e.g., a small business, a multinational corporation, a personal user). Pairs will: Assess the benefits and risks of cloud storage for their assigned use case. Recommend security measures tailored to the scenario, such as encryption or multi-factor authentication. Share their insights through a short verbal summary.	This resource provides best practices for securing cloud storage, such as implementing encryption and access controls, to ensure compliance with data protection regulations. https://forgeahead.io/2024/09/02/cloud-storage-security-compliance-tips/ This article discusses various security risks associated with cloud computing, including compliance challenges and insider threats, and suggests preventive measures. https://www.sentinelone.com/cybersecurity-101/cloud-security/security-risks-of-cloud-computing/ Cloud computing explained https://www.youtube.com/watch?v=_a6us8kaq0g

A2.3 Network devices that have user-controllable security settings	- **Whole class instruction teaching and learning – Introduction** o Introduce the concept of network devices with user-controllable security settings. o Use a collaborative concept map to explore how devices such as PCs, servers, WAPs, switches, routers, modems, bridges, firewalls, and mobile devices contribute to network operations. o Highlight key security settings, such as password protection, firewalls, and firmware updates, and discuss their importance in preventing threats like unauthorised access or data breaches. - **Small group activity – Device security analysis** o Assign each group one or two devices (e.g., WAP, firewall, router). Groups will: Research the security settings available for their assigned device(s). Identify the potential security risks if these settings are not configured correctly. Recommend best practices for configuring the device securely. o Groups will present their findings using annotated diagrams of the devices. - **Pair activity – Security scenario walkthrough** o Provide pairs with a scenario involving a network security issue, such as unauthorised access to a WAP or a misconfigured router leading to a data breach. Pairs will: Identify the device(s) involved in the scenario. Suggest how the security settings could be adjusted to resolve or prevent the issue. Share their solutions in a brief oral presentation or a written summary. - **Individual activity – Troubleshooting guide** o Students will create a troubleshooting guide for securing one network device of their choice. The guide should include: Common security vulnerabilities of the device. Step-by-step instructions for configuring key security settings. Tips for ongoing security maintenance (e.g., firmware updates, regular audits).	CISA's Securing Network Infrastructure Devices: Recommends best practices for hardening network devices, ensuring they are safeguarded against potential threats. https://www.cisa.gov/news-events/news/securing-network-infrastructure-devices National Cyber Security Centre (NCSC) Device Security Guidance: Offers detailed instructions on securely configuring various platforms, aiding in the management of organisational devices. https://www.ncsc.gov.uk/collection/device-security-guidance/platform-guides

A3 Software components of computer networks A3.1 Types of Operating System (OS) A3.1.1 server/ network A3.1.2 multi-functional/desktop A3.1.3 mobile A3.1.4 real time A3.1.5 modes	• **Whole class instruction teaching and learning – Introduction** o Explain the various types of operating systems, including server/network, multi-functional/desktop, mobile, and real-time operating systems. o Use visual aids like diagrams to highlight the differences in their architecture, purposes, and use cases. o Provide a brief demonstration of how kernel and user modes operate within an OS. • **Small group activity – Operating system investigation** o Each group will investigate a specific type of operating system (e.g., server/network, mobile, real-time). Their task is to: Identify the key features and functions of their assigned OS. Explore its real-world applications and examples of its usage. Analyse its advantages and disadvantages in its domain. o Groups will create a mind map of their findings and share it with the class. • **Pair activity – Compare and contrast OS types** o Pairs will use a Venn diagram to compare and contrast two operating systems, such as multi-functional/desktop and mobile. o They will focus on aspects like user interface, performance, scalability, and typical users. o Pairs will then present their diagrams and explain the overlaps and differences.	Provides an overview of different operating systems, including batch, time-sharing, distributed, real-time, and network operating systems. https://www.geeksforgeeks.org/types-of-operating-systems/ Provides insights into user mode and kernel mode operations within an OS, detailing their functions and significance. https://www.tutorchase.com/answers/a-level/computer-science/what-are-the-different-modes-of-operation-in-an-operating-system Gives explanations of batch, distributed, real-time, and other operating systems, along with their applications. https://www.studysmarter.co.uk/explanations/computer-science/computer-systems/types-of-operating-systems/
A3.2 Utility software A3.2.1 backup software A3.2.2 compression tools A3.2.3 disk management tools A3.2.4 file managers	• **Whole class instruction teaching and learning – Introduction** o Introduce the various types of operating systems (server/network, multi-functional/desktop, mobile, real-time) and their modes (kernel and user). o Use a visual hierarchy diagram to categorise these OS types and highlight their roles in network functionality. o Discuss key security features and vulnerabilities associated with each, such as user authentication in desktops or real-time response requirements in embedded systems.	Provides an overview of various utility software types, including backup software, compression tools, and disk management utilities. https://www.bbc.co.uk/bitesize/guides/zmqw7p3/revision/4

A3.2.5 network utilities and tools A3.2.6 package managers A3.2.7 device drivers	• **Small group activity – OS type evaluation** o Assign each group a specific OS type (e.g., server, mobile, real-time). Groups will: Research the primary functions and characteristics of their assigned OS type. Identify its common security risks (e.g., data theft, unauthorised access, malware). Recommend security measures to mitigate these risks. o Groups will present their findings using a concept map or flowchart. • **Individual activity – Comparative table** o Students will create a comparative table that examines the security features and challenges of the four OS types. The table should include: Key characteristics. Typical use cases (e.g., mobile OS for smartphones, real-time OS for industrial control). Security vulnerabilities and mitigation strategies.	Explains different utility software categories such as backup software, compression tools, and disk management utilities, along with their functions and examples. https://www.geeksforgeeks.org/utility-software/ Offers insights into utility software types like backup software, compression tools, and disk management utilities, highlighting their roles in system maintenance. https://www.studysmarter.co.uk/explanations/computer-science/computer-systems/utility-software/

B: Network security		
B1 The main threats to data and computer networks B1.1 Malware	• **Whole class instruction teaching and learning – Introduction** o Explain the concept of malware, including viruses, spyware, and adware. o Use visual aids such as infographics or mind maps to illustrate how malware infiltrates systems, spreads, and affects network operations. o Include examples of real-world malware attacks and their impact on organisations. • **Small group activity – Analysing malware cases** o Divide students into groups and provide each group with a case study of a malware incident. Their task is to: Identify the type of malware involved. Describe the attack's entry point and spread mechanism. Analyse the impact on the organisation. o Groups will present their findings to the class. • **Pair activity – Comparing malware types** o Each pair of students will create a comparison table outlining the characteristics, methods of infection, and potential damage caused by viruses, spyware, and adware. o Pairs will then discuss similarities and differences with another pair.	Kaspersky: Types of Malware Provides an overview of various malware types, including viruses, spyware, and adware, detailing their characteristics and methods of infection. https://www.kaspersky.com/resource-center/threats/types-of-malware CrowdStrike: 12 Types of Malware Offers insights into various malware categories, including viruses, spyware, and adware, with examples of real-world incidents. https://www.crowdstrike.com/en-us/cybersecurity-101/malware/types-of-malware/

B1.2 Denial-of-service (DoS) and distributed denial of service (DDoS)	- **Whole class instruction teaching and learning – Introduction** o Introduce the concepts of DoS and DDoS attacks. o Use a real-world example of a major DDoS attack to explain how such attacks overwhelm network resources, rendering services inaccessible. o Highlight the differences between DoS and DDoS attacks in terms of scale, origin, and execution. - **Pair activity – Attack and defence brainstorm** o In pairs, students will brainstorm and list techniques used to launch DoS/DDoS attacks (e.g., botnets, amplification attacks) and methods to defend against them (e.g., firewalls, load balancing). Each pair will discuss which methods they think are most effective and why. - **Individual activity – Essay on mitigation strategies** o Students will write a short essay explaining their understanding of how organisations can prevent or mitigate the effects of DoS/DDoS attacks. o They should include examples of tools or practices that can be used to detect and respond to these attacks effectively.	Cloudflare: What is a DDoS Attack? Provides an overview of DDoS attacks, explaining how they disrupt normal traffic to a web property. https://www.cloudflare.com/en-gb/learning/ddos/what-is-a-ddos-attack/ Highlights the differences between DoS and DDoS attacks in terms of scale, origin, and execution. https://www.fortinet.com/resources/cyberglossary/dos-vs-ddos Highlights the differences between DoS and DDoS attacks in terms of scale, origin, and execution. https://www.cisa.gov/sites/default/files/2024-03/understanding-and-responding-to-distributed-denial-of-service-attacks_508c.pdf
B1.3 Hacking, unauthorised access	- **Whole class instruction teaching and learning – Introduction** o Explain the concept of hacking and unauthorised access, emphasising different types of hackers. o Highlight how unauthorised access is achieved through methods such as phishing, credential stuffing, and exploiting weak passwords. o Use case studies to show the real-world impacts of hacking.	Bullet Proof: Types of Hackers Provides an overview of different hacker types explaining their roles and motivations. https://www.bulletproof.co.uk/blog/different-types-of-hackers

	• **Small group activity – Creating a security breach scenario** o Students will work in groups to design a fictional scenario where a hacker gains unauthorised access to a network. They must detail: The hacker's method of entry. The consequences of the breach. Preventative measures that could have stopped the breach. o Groups will present their scenarios to the class. • **Pair activity – Analysing hacker tools** o Pairs will research a specific hacking tool (e.g., keyloggers, password crackers, or remote access trojans). o They will describe how the tool is used, its potential impacts, and ways to defend against it. Each pair will briefly share their findings with the class.	
B1.4 Social engineering attacks	• **Whole class instruction teaching and learning – Introduction** o Introduce the concept of social engineering attacks and how they exploit human psychology to gain unauthorised access to information or systems. o Discuss examples such as phishing, pretexting, baiting, and tailgating. o Use real-life cases to demonstrate the impact of these attacks on organisations and individuals. **Small group activity – Scenario role-play** o Groups will act out scenarios of social engineering attacks, with some members playing the role of attackers and others as victims. o Scenarios may include phishing emails, phone scams, or physical tailgating. o Each group will present their role-play to the class, emphasising what could have been done to avoid falling victim to the attack. • **Pair activity – Recognising phishing attempts** o Provide pairs with a mix of genuine and fake emails or messages. o Their task is to analyse each one and identify which are phishing attempts, justifying their decisions. Afterward, pairs will discuss tips for recognising and avoiding phishing attacks.	Vishing hacking challenge at Defcon. https://www.youtube.com/watch?v=fHhNWAKw0bY

B1.5 Insider threats	• **Whole class instruction teaching and learning – Introduction** o Discuss the concept of insider threats and their potential impact on network security. o Use case studies to highlight instances of insider threats, such as sabotage, accidental errors, or data theft. o Facilitate a class mind map activity on the board, where students contribute examples and implications of insider threats. • **Small group activity – Insider threat scenario analysis** o Each group is given a scenario depicting an insider threat, such as data theft by a disgruntled employee or unintentional data leaks. Tasks: Identify the threat. Analyse its potential consequences. Propose strategies to mitigate the threat. o Groups present their findings to the class. • **Pair activity – Safeguarding measures brainstorm** o Pairs brainstorm preventive measures to combat insider threats, focusing on topics such as access controls, employee training, and monitoring systems. o Each pair creates a concept map of their ideas and shares it with the class.	Human Error's Guide to Keeping Security Simple – Mimecast Security Awareness Training https://www.youtube.com/watch?v=GVT5IgmA6WE&pp=ygUIbWltZWNhc3Q%3D

B1.6 Zero-day vulnerabilities	• **Whole class instruction teaching and learning – Introduction** o Introduce zero-day vulnerabilities by explaining their nature and why they are challenging to detect and mitigate. o Use examples of high-profile attacks exploiting zero-day vulnerabilities, such as major data breaches. o Guide the class in creating a timeline infographic showcasing the lifecycle of a zero-day attack, from discovery to patch deployment. • **Small group activity – Zero-day threat analysis** o Groups are given a brief case study of a zero-day exploit. Tasks: Identify the exploited vulnerability. Analyse how the attack was executed. Recommend proactive defence measures organisations could take. o Groups share their findings in a carousel brainstorming session. • **Individual activity – Expert research** o Students individually explore a specific real-world zero-day vulnerability. o They create a digital story summarising the exploit, its impact, and lessons learned from the incident.	CSO Online: Zero Days Explained Provides an overview of zero-day vulnerabilities, including their definition, examples, and defence strategies. https://www.csoonline.com/article/565704/zero-days-explained-how-unknown-vulnerabilities-become-gateways-for-attackers.html SoftwareLab: The 10 Worst Zero-Day Exploit Attacks Ever Discusses some of the most significant zero-day attacks in history, detailing how they were executed and their impacts. https://softwarelab.org/blog/zero-day-exploit-examples/

B2 Common methods of attacking data and computer networks B2.1 Eavesdropping B2.2 Interception B2.6 Man-in-the-middle	• **Whole class instruction teaching and learning – Introduction** o Explain the principles behind eavesdropping, interception, and man-in-the-middle attacks, highlighting their differences and impacts on network security. o Illustrate these attacks with real-world scenarios, such as compromised email communications or intercepted financial transactions. o Create a class mind map to explore potential vulnerabilities in networks that could lead to these attacks and ways to safeguard against them. • **Whole class activity – Use a human network simulation** o Assign roles (e.g., sender, receiver, attacker, observer) to students, creating a “human network” to physically simulate data transmission. o During the simulation, introduce an “eavesdropper” and a “man-in-the-middle” attacker who intercept data. o Debrief as a class to discuss how these attacks compromise network communication.	Fortinet: What Are Eavesdropping Attacks? Provides an overview of eavesdropping attacks, explaining how attackers intercept data transmitted between devices and the implications for network security. https://www.fortinet.com/resources/cyberglossary/eavesdropping IBM: What Is a Man-in-the-Middle (MITM) Attack? Explains man-in-the-middle attacks, detailing how attackers position themselves between two parties to intercept or alter communications, with real-world examples. https://www.ibm.com/think/topics/man-in-the-middle

B2.3 Replay B2.4 Address Resolution Protocol (ARP) spoofing B2.5 Domain Name System (DNS) poisoning	• **Whole class teaching and learning – Introduction** o Explain the methods of attacking data and computer networks, focusing on replay attacks, ARP spoofing, and DNS poisoning. o Use a class mind map to visually organise the connections between these attack methods, their mechanisms, and their impacts on network security. o Provide real-world examples to illustrate each attack type. • **Small group activity – Identifying attack vectors** o Assign each group one of the attack methods (replay, ARP spoofing, DNS poisoning). Their tasks are to: Analyse how the assigned attack operates. Identify potential vulnerabilities in a network that could be exploited by the attack. Suggest possible defensive measures. o Groups will create a process infographic summarising their findings and present it to the class. • **Individual activity – Writing a mitigation guide** o Students will write a step-by-step guide on defending against one of the three attack types. o They should incorporate specific tools or strategies (e.g., using ARP inspection for ARP spoofing) and reference network security principles. o This will be submitted as an individual written product.	Kaspersky: What is a Replay Attack and How to Prevent it Provides an overview of replay attacks, explaining how attackers capture and retransmit data to deceive systems, along with prevention strategies. https://www.kaspersky.com/resource-center/definitions/replay-attack GeeksforGeeks: What is ARP Spoofing Attack? Explains ARP spoofing attacks, detailing how attackers associate their MAC address with the IP address of another device to intercept data. https://www.geeksforgeeks.org/what-is-arp-spoofing-attack/

B2.7 Crypto jacking B2.8 Structured Query Language (SQL) injection B2.9 Buffer overflows	• **Whole class teaching and learning – Introduction** o Introduce the concepts of crypto-jacking, SQL injection, and buffer overflow attacks. o Use a multimedia presentation with visual aids to demonstrate how each attack works, including animated diagrams showing the impact on system resources and data integrity. o Highlight real-world incidents to contextualise these attacks. • **Small group activity – Analysing attack scenarios** o Provide each group with a case study of an attack involving either crypto-jacking, SQL injection, or buffer overflow. Their tasks are to: Analyse the incident and identify how the attack was executed. Discuss the vulnerabilities that allowed the attack to succeed. Recommend ways the attack could have been mitigated. o Each group will design a concept map showing their analysis and present it to the class. • **Pair activity – Creating a defence strategy** o Pairs will select one attack type and create a checklist of actions to prevent it. For example: For crypto jacking, suggest implementing resource monitoring tools or browser extensions. For SQL injection, include using prepared statements and input validation. For buffer overflows, propose strategies like bounds checking or using modern languages with inbuilt memory management. o Pairs will share their checklist with the class for peer feedback.	Ensign InfoSecurity: Cryptojacking Explained Provides an overview of crypto-jacking, detailing how attackers exploit systems to mine cryptocurrencies without authorisation. https://www.ensigninfosecurity.com/cybersecurity-101/what-is-cryptojacking SoftwareLab: SQL Injection Examples Discusses notable SQL injection attacks, such as the 2015 TalkTalk breach, illustrating the impact of these vulnerabilities. https://softwarelab.org/blog/sql-injection-examples/

B3 Methods of defending against attacks on data and computer networks B3.1 Security settings on network components	• **Whole class teaching and learning – Introduction** o Introduce the concept of securing network components, emphasising the importance of configuring security settings on devices like PCs, servers, WAPs, switches, routers, modems, bridges, firewalls, and mobile devices. o Use a class mind map to show how these devices interact within a network and the potential vulnerabilities each can have. Include examples of typical security settings, such as enabling firewalls, updating firmware, and configuring access control lists (ACLs). • **Pair activity – Security settings troubleshooting** o Provide pairs with a scenario involving a misconfigured network component (e.g., an open router port or disabled firewall). o Their task is to diagnose the issue, outline the potential risks, and create a plan to reconfigure the settings correctly. o Pairs will present their findings to the class. • **Individual activity – Creating a security checklist** o Each student will create a checklist for securing network components, focusing on at least three devices (e.g., PC, firewall, and WAP). o They should include steps like enabling encryption, setting strong admin passwords, and monitoring for updates. o This checklist will be submitted as an individual product.	Auvik: Network Device Security – Guide and Best Practices Provides comprehensive best practices for securing network devices, including configuration management and access controls. https://www.auvik.com/franklyit/blog/network-device-security/ DNSstuff: Network Device Security – Guide + Recommended Software Offers insights into securing various network components and suggests tools to enhance network security. https://www.dnsstuff.com/network-device-security

B3.2 Security features of operating systems B3.2.1 Dynamic Host Configuration Protocol (DHCP) B3.2.4 file management B3.2.5 network policies B3.2.6 patch and update management B3.2.7 trusted software B3.2.8 logging and auditing	• **Whole class teaching and learning – Introduction** o Explain the importance of operating system security features such as DHCP configuration, file management, network policies, patch and update management, trusted software, and logging and auditing. o Use a collaborative class mind map to categorise these features into prevention, detection, and response strategies. o Highlight real-world examples of vulnerabilities caused by poor OS management and how these features help mitigate them. • **Small group activity – Securing operating systems** o Assign each group one OS security feature (e.g., DHCP, file management, network policies). Their tasks are to: Research how the feature contributes to overall system security. Identify potential weaknesses or misconfigurations in the feature. Develop a simple action plan to implement or enhance the feature in an organisational network. o Groups will present their action plans using a flow diagram. • **Pair activity – Auditing and logging analysis** o Provide pairs with a sample system log (mock data). Their tasks are to: Analyse the log for suspicious activity (e.g., unauthorised login attempts, unexpected file changes). Suggest actions that could be taken based on the log's information. Create a checklist for effective logging and auditing practices. o Pairs will share their analysis with the class. • **Individual activity – Writing a network policy draft** o Each student will draft a network policy document focusing on one area (e.g., patch management, trusted software usage, or file access control). o The policy should outline rules, responsibilities, and procedures for maintaining system security. o This will be submitted as an individual written product.	NCSC: Logging and Protective Monitoring Provides guidance on effective logging and monitoring practices to detect and respond to security incidents. https://www.ncsc.gov.uk/collection/device-security-guidance/managing-deployed-devices/logging-and-protective-monitoring NIST: Guide to Enterprise Patch Management Planning Offers comprehensive strategies for planning and implementing patch management to maintain system security. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-40r4.pdf

B3.2 Security features of operating systems B3.2.2 authentication and access control B3.2.3 user management	• **Whole class teaching and learning – Introduction** o Introduce the security features of authentication and access control, including passwords, biometrics, and multi-factor authentication (MFA), as well as user management. o Use an interactive class demonstration to show how these measures protect against unauthorised access. o For example, simulate an MFA login process and compare it to single-factor authentication. o Discuss the role of user management in assigning permissions and limiting access to sensitive data. • **Small group activity – Comparing authentication methods** o Assign each group a specific authentication method (passwords, biometrics, MFA). Their tasks are to: Research the strengths and weaknesses of their assigned method. Discuss how it could be implemented in a workplace setting. Create a comparison table evaluating their method against others. o Groups will present their comparison tables, and the class will consolidate findings into a master table. • **Pair activity – Designing a user management plan** o Provide pairs with a scenario where a network administrator must create a user management plan for a medium-sized organisation. Their tasks are to: Design a structure for user accounts (e.g., admin, standard users, guest accounts). Outline best practices for managing permissions and account lifecycle (e.g., adding/removing users, monitoring usage). Develop a checklist of actions to ensure user management aligns with security policies. o Pairs will share their plans with the class for feedback.	Techlocity: Biometrics vs Passwords – Understanding Authentication Methods Explores the differences between biometric authentication and traditional passwords, highlighting security implications and user convenience. https://www.techlocity.com/blog/biometrics-vs-passwords CERTAURI: Understanding the IAM User Lifecycle – A Comprehensive Guide Discusses Identity and Access Management (IAM) user lifecycle, including onboarding, managing changes, and offboarding, emphasising security and compliance. https://www.certauri.com/understanding-the-iam-user-lifecycle-a-comprehensive-guide/ DNSstuff: Best Practices for User Account Management in Companies

	• **Individual activity – Creating a security awareness guide** o Each student will create a user-friendly guide on authentication and access control for non-technical employees. o The guide should include tips for creating strong passwords, examples of MFA in use, and the importance of secure login practices.	Offers best practices for managing user accounts, focusing on avoiding privilege creep, protecting data, and updating directories as staff and access needs change. https://www.dnsstuff.com/best-practices-for-user-account-management-in-company
B3.3 Security features of utilities and applications B3.3.1 antivirus/ anti-malware	• **Whole class teaching and learning – Introduction** o Introduce the role of antivirus and anti-malware software in defending against cyber threats. Use a multimedia presentation to explain how these tools detect, prevent, and remove malicious software. o Highlight key features like real-time scanning, signature-based detection, and heuristic analysis. o Provide examples of how antivirus software has mitigated major malware attacks. • **Small group activity – Evaluating antivirus solutions** o Divide students into small groups and assign each group a popular antivirus or anti-malware software to evaluate. Their tasks are to: Research the features, strengths, and weaknesses of the assigned software. Compare it to another solution in terms of detection methods, system impact, and usability. Summarise their findings in a decision matrix infographic. o Groups will present their decision matrices to the class. • **Pair activity – Analysing malware behaviour** o Provide pairs with a case study of a malware attack (e.g., a ransomware or spyware incident). Their tasks are to: Analyse how the malware infiltrated the system and spread. Identify how antivirus software could have detected and mitigated the attack. Create a flow diagram showing the lifecycle of the malware and the intervention points.	PCInsider: 10 Malware Detection Techniques Used by Antivirus Provides an overview of various malware detection methods, including signature-based detection and heuristic analysis. https://www.thepcinsider.com/malware-detection-techniques-how-antivirus-works/

	o Pairs will share their diagrams with another pair for peer review.	
B3.3 Security features of utilities and applications B3.3.2 backup and recovery tools B3.3.4 encryption tools	• **Whole class teaching and learning – Introduction** o Explain the importance of backup and recovery tools and encryption tools in securing data. o Use real-world examples, such as data loss incidents or breaches due to unencrypted information, to illustrate their critical roles. o Demonstrate how backups ensure data recovery after an incident and how encryption secures sensitive information during storage and transmission. • **Small group activity – Analysing backup and encryption strategies** o Assign each group a scenario (e.g., a ransomware attack, accidental file deletion, or data transmission over a public network). Their tasks are to: Evaluate how backup tools or encryption tools could mitigate the risk. Identify specific tools or techniques suitable for the scenario (e.g., full disk encryption, cloud backups). Create a process diagram showing how the chosen solution would work in practice. o Groups will present their diagrams to the class, explaining their reasoning. • **Pair activity – Hands-on encryption exercise** o Provide pairs with a simple example of plaintext data. Their tasks are to: Use an encryption method (e.g., substitution cipher or ROT13) to encrypt the data. Exchange encrypted data with another pair and attempt to decrypt it. Reflect on the importance of using secure and complex encryption techniques in real-world applications. o Pairs will share their experiences with the class.	Morgan Stanley: Data Backups – Its Importance for Cybersecurity Discusses the critical role of data backups in preventing data loss and ensuring cybersecurity. https://www.morganstanley.com/articles/data-backup-importance-cybersecurity/ Stronghold Data: 4 Real-Life Examples of Data Loss Presents case studies highlighting the consequences of inadequate backup strategies and the importance of data restoration plans. https://strongholddata.com/4-real-life-examples-of-data-loss/

	• **Individual activity – Designing a data protection plan** o Each student will design a data protection plan for a small business, incorporating backup schedules and encryption practices. The plan should include: Recommendations for backup frequency and storage (local, offsite, or cloud). A strategy for encrypting sensitive customer and business data. Tools or software to implement the plan effectively. o This plan will be submitted as an individual written product.	
B3.3 Security features of utilities and applications: B3.3.3 firewall B3.3.8 intrusion detection systems (IDS) B3.3.9 intrusion prevention systems (IPS)	• **Whole class teaching and learning – Introduction** o Introduce firewalls, intrusion detection systems (IDS), and intrusion prevention systems (IPS) as critical tools for network security. o Use a class diagram to visually compare and contrast their roles: firewalls as gatekeepers, IDS as monitors, and IPS as proactive defenders. o Provide examples of how these tools function together to protect networks from threats. • **Small group activity – System protection strategy** o Assign each group a specific tool (firewall, IDS, or IPS) to explore. Their tasks are to: Investigate how their assigned tool works to defend a network. Identify common threats it addresses. Develop a use case demonstrating its role in a layered security strategy. o Groups will create a cause-and-effect diagram to illustrate how the tool reacts to an attempted attack and present it to the class. • **Pair activity – Analysing log data** o Provide pairs with sample log data (mock or anonymised) from an IDS or firewall. Their tasks are to: Analyse the logs for suspicious activity (e.g., repeated failed login attempts, unusual traffic spikes). Suggest how an IPS or firewall configuration could prevent such threats in the future.	Palo Alto Networks: Firewall vs. IDS vs. IPS – Understanding the Differences Provides a comparative analysis of firewalls, IDS, and IPS, detailing their distinct roles in network security. https://www.paloaltonetworks.com/cyberpedia/firewall-vs-ids-vs-ips

	Draft a short report summarising their analysis and recommendations. o Pairs will exchange their reports with another pair for peer review. • **Individual activity – Firewall configuration plan** o Each student will create a firewall configuration plan for a small organisation. The plan should include: Rules for allowing or blocking traffic based on IP address, ports, and protocols. Guidelines for regular updates and monitoring. Integration with IDS or IPS for enhanced security. o This configuration plan will be submitted as an individual written product.	
B3.3 Security features of utilities and applications B3.3.5 sandbox B3.3.6 virtual machines B3.3.7 network segmentation tools	• **Whole class teaching and learning – Introduction** o Introduce sandboxing, virtual machines (VMs), and network segmentation tools as advanced methods for securing systems and networks. o Use a collaborative class mind map to connect these tools to their specific functions: sandboxes for isolating threats, VMs for creating secure testing environments, and network segmentation for limiting attack spread. o Demonstrate each tool's purpose with a real-world example, such as using a VM to test suspicious software. • **Pair activity – Sandbox and VM hands-on simulation** o Provide pairs with a simple task: Use a sandbox or VM to isolate and test a mock piece of software (provide predefined parameters and tools for the exercise). Reflect on how isolation prevented the software from impacting the rest of the system. Discuss how this method could be scaled in a corporate environment. o Pairs will summarise their findings in a concept map and share with the class. • **Individual activity – network segmentation plan** o Each student will create a network segmentation plan for a medium-sized business. The plan should include: Segments for different departments or services (e.g., HR, finance, guest Wi-Fi).	Fortinet: What Is Sandboxing? Provides an overview of sandboxing, explaining how it isolates code in a controlled environment to analyse behaviour without risking the host system. https://www.fortinet.com/resources/cyberglossary/what-is-sandboxing

	Access control policies for each segment. Methods for monitoring traffic between segments to detect anomalies. o This segmentation plan will be submitted as an individual written product.	
B3.3.10 penetration testing tools	• **Whole class teaching and learning – Introduction** o Introduce penetration testing as a proactive method to identify and address security weaknesses. o Use a multimedia presentation to outline the stages of penetration testing: reconnaissance, scanning, vulnerability assessment, exploitation, and reporting. Highlight how penetration testing tools are used at each stage and discuss their significance in a layered security strategy. o Small group activity – Analysing penetration testing phases o Divide students into groups and assign each group one phase of penetration testing (reconnaissance, scanning, vulnerability assessment, exploitation, or reporting). Their tasks are to: Research the purpose and techniques of their assigned phase. Identify tools commonly used for their phase (e.g., Nmap for scanning, Metasploit for exploitation). Create a flowchart showing how the phase integrates with the entire penetration testing process. o Groups will present their flowcharts to the class for discussion. • **Pair activity – Tool research and evaluation** o Provide pairs with a penetration testing tool to evaluate (e.g., Wireshark, Nessus, Metasploit). Their tasks are to: Research the tool's features, strengths, and limitations. Create a comparison table showing how the tool performs in different penetration testing phases. Suggest scenarios where the tool would be most effective. o Pairs will share their findings with another pair for peer feedback.	eSecurity Planet: Penetration Testing Phases & Steps Explained Explains each phase of penetration testing and discusses tools like Metasploit used during the exploitation stage. https://www.esecurityplanet.com/networks/penetration-testing-phases/ The Knowledge Academy: Penetration Testing Phases: A Roadmap Provides insights into the tools used in each phase of penetration testing, such as reconnaissance tools like Maltego and vulnerability scanners. https://www.theknowledgeacademy.com/blog/penetration-testing-phases/

B3.3.11 network monitoring tools	• **Whole class teaching and learning – Introduction** o Explain the purpose and importance of network monitoring tools, including packet analysers, network mappers, traffic analysers, and network protocol analysers. o Use visual aids to show how these tools work together to detect anomalies, monitor performance, and identify potential threats. o Demonstrate their real-world applications, such as detecting unauthorised access or unusual traffic patterns. • **Small group activity – Exploring network monitoring tools** o Assign each group one type of network monitoring tool (e.g., packet analyser, network mapper). Their tasks are to: Research how their tool operates and its key features. Identify specific use cases, such as detecting a denial-of-service attack or mapping a network's topology. Create an infographic explaining how their tool contributes to network security. o Groups will present their infographics in a gallery walk for peer feedback.	Wireshark: What is a Packet Analyser? Explains packet analysers, their importance, and practical applications in network management and troubleshooting. https://www.wireshark.org/docs/wsug_html_chunked/ChapterIntroduction.html Nmap: The Network Mapper Provides an overview of Nmap's capabilities in scanning and mapping networks, with real-world use cases. https://nmap.org

B3.4 Security policies B3.4.1 CIA triad	• **Whole class teaching and learning – Introduction** o Introduce the CIA triad as the foundation of cybersecurity: confidentiality, integrity, and availability. Use a diagram to illustrate how these three principles intersect and complement each other. o Discuss real-world examples of how breaches in each area (e.g. data leaks for confidentiality, tampered records for integrity, or server downtime for availability) can impact organisations. • **Small group activity – Analysing CIA triad in action** o Divide the class into groups and assign each group a scenario that involves one or more aspects of the CIA triad (e.g., a ransomware attack targeting availability, a phishing attack compromising confidentiality). Their tasks are to: Identify which aspect(s) of the CIA triad are affected. Discuss the potential impact of the breach on the organisation and stakeholders. Propose mitigation strategies aligned with the principles of the CIA triad. o Groups will create a cause-and-effect diagram to present their findings to the class. • **Pair activity – Securing the triad** o Provide pairs with a mock business environment (e.g., an e-commerce site or healthcare system). Their tasks are to: Analyse potential threats to confidentiality, integrity, and availability within the environment. Develop a brief action plan addressing how the triad can be upheld using security policies and practices. Create a visual representation (e.g., a cycle diagram) showing how the triad is protected in their scenario. o Pairs will share their diagrams with the class.	Introduction to the CIA Triad: This article provides a clear explanation of confidentiality, integrity, and availability, serving as a solid foundation for understanding the CIA triad. https://www.geeksforgeeks.org/the-cia-triad-in-cryptography Real-World Applications of the CIA Triad: This resource discusses how the CIA triad principles apply in various scenarios, offering practical examples that can enrich classroom discussions. https://blog.netwrix.com/2019/03/26/the-cia-triad-and-its-real-world-application/ The CIA Triad: A Principled Framework for Defining Infosec Policies: This article explores how the CIA triad serves as a foundational model for developing effective information security policies. https://www.csoonline.com/article/568917/the-cia-triad-definition-components-and-examples.html

B3.4 Security policies B3.4.2 enforcing security policies and best practices B3.4.3 user training policies	• **Whole class teaching and learning – Introduction** o Explain the role of security policies and best practices in maintaining organisational cybersecurity. o Discuss the importance of enforcing these policies consistently and highlight user training as a critical component. o Use a case study of a successful or failed enforcement of security policies to emphasise the consequences. o Cover key training topics such as privacy, recognising phishing attacks, creating strong passwords, and reporting suspicious activities. • **Small group activity – Policy enforcement simulation** o Assign each group a scenario where security policy enforcement is critical (e.g., an organisation rolling out multi-factor authentication or implementing password policies). Their tasks are to: Identify challenges in enforcing the policy. Develop a strategy to address these challenges, including monitoring and compliance measures. Create a flowchart outlining the steps to enforce the policy effectively. o Groups will present their flowcharts to the class. • **Pair activity – Designing a user training module** o Provide pairs with one aspect of user training (e.g., phishing awareness, creating strong passwords). Their tasks are to: Design a 5-minute training session for employees on their topic. Include a visual aid (e.g., a poster or infographic) summarising key points. Prepare a quick quiz to assess understanding after the training. o Pairs will share their training modules with the class for feedback. • **Individual activity – Policy enforcement plan** o Each student will draft a policy enforcement plan for a medium-sized organisation. The plan should include: Steps to implement and monitor adherence to a specific security policy (e.g., BYOD policy or password updates).	The Importance of User Acceptance in Cybersecurity Policies: This article discusses how user acceptance is vital for the successful implementation of cybersecurity policies. https://cyberconiq.com/blog/the-importance-of-user-acceptance-in-cybersecurity-policies/ Cybersecurity Policy Enforcement: Strategies for Success: This article provides strategies for effectively enforcing cybersecurity policies within organisations. https://www.trustedsec.com/blog/cybersecurity-policy-enforcement-strategies-for-success BYOD Policy Best Practices Guide: This resource provides recommendations for launching a BYOD policy, addressing security, compliance, and employee well-being. https://uk.indeed.com/hire/c/info/byod-policy-best-practices-guide

	Methods for tracking compliance and addressing non-compliance. Recommendations for integrating user training to support the policy. o This plan will be submitted as an individual written product.	
B3.4 Security policies B3.4.4 risk assessments B3.4.5 cyber-security plans	• **Whole class teaching and learning – Introduction** o Introduce the concepts of risk assessments and cybersecurity plans as proactive measures for safeguarding networks. o Use a visual presentation to explain the steps involved in conducting a risk assessment (identification, analysis, evaluation, and treatment) and how these inform the creation of cybersecurity plans. o Include examples of risks (e.g., unauthorised access, malware) and how a plan addresses them. • **Small group activity – Conducting a risk assessment** o Assign groups a fictional organisation (e.g., a retail store, hospital, or school). Their tasks are to: Identify potential risks to the organisation's network. Prioritise the risks based on likelihood and impact. Develop a risk matrix to visualise and categorise the risks. o Groups will share their matrices and discuss their findings with the class. • **Pair activity – Developing a cybersecurity plan** o Provide pairs with a hypothetical scenario where a cybersecurity plan is needed (e.g., protecting an online database or implementing remote work policies). Their tasks are to: Draft a basic cybersecurity plan addressing key areas like access control, monitoring, incident response, and recovery. Include a summary of actions to mitigate high-priority risks identified in the scenario. Present their plans as a one-page action sheet. o Pairs will exchange and review each other's plans for completeness and clarity.	Cyber Security Risk Assessment: Step-by-Step Process: This article provides a detailed guide on conducting a cybersecurity risk assessment, covering the importance of risk assessment, common threats, best practices, and a checklist for businesses. https://www.sentinelone.com/cybersecurity-101/cybersecurity/cyber-security-risk-assessment/ An Effective Cyber Security Plan: A Guide: This article discusses the essential components of a cybersecurity plan, emphasising the importance of aligning security measures with business goals and regulatory compliance. https://cyberexperts.com/cyber-security-plan/

B3.4 Security policies B3.4.6 disaster recovery plans B3.4.7 incident response plan B3.4.8 backup plans	• **Whole class teaching and learning – Introduction** o Explain the importance of disaster recovery plans, incident response plans, and backup plans in maintaining organisational resilience. o Use a timeline infographic to show the sequence of actions in preparing for and responding to incidents. o Highlight key differences: disaster recovery focuses on restoring normal operations after major events, incident response focuses on immediate containment and mitigation, and backup plans ensure data availability. • **Small group activity – disaster recovery simulation** o Provide groups with a disaster scenario (e.g., a ransomware attack, server room flooding, or power outage). Their tasks are to: Develop a disaster recovery plan detailing key steps for restoring operations. Identify stakeholders responsible for implementing the plan. o Create a process map to illustrate the plan's execution from disaster detection to full recovery. • **Individual activity – Backup strategy** o Each student will design a backup strategy for a small business. The strategy should include: Types of data to be backed up (e.g., financial records, customer data). Backup frequency and storage locations (e.g., cloud, offsite, or local). Methods for testing the reliability of backups and procedures for data restoration.	Developing a Cybersecurity Disaster Recovery Plan: This article outlines steps to create a disaster recovery plan, including risk assessment, strategy development, and testing procedures. https://cybersguards.com/disaster-recovery-plan/ Response and Recovery Planning Principles: This UK Government resource discusses the importance of well-defined incident management processes to ensure continuity of essential functions during system failures. https://www.security.gov.uk/policy-and-guidance/government-cyber-security-policy-handbook/principle-d1-response-and-recovery-planning/

B3.4 Security policies B3.4.9 Bring Your Own Device (BYOD) policy B3.4.10 remote work policies B3.4.11 physical security policies B3.4.12 third-party security policies.	• **Whole class teaching and learning – Introduction** o Introduce the challenges and solutions associated with BYOD policies, remote work, physical security, and third-party security. o Use examples to illustrate potential vulnerabilities in each area, such as unsecure devices in BYOD, remote work network threats, physical breaches in server rooms, and risks posed by third-party vendors. o Show how well-defined policies can mitigate these risks. • **Small group activity – Policy design workshop** o Assign each group one type of policy to design (BYOD, remote work, physical security, or third-party security). Their tasks are to: Identify specific risks related to their assigned policy type. Draft a policy addressing these risks, including rules, procedures, and enforcement mechanisms. Present their policy using a decision matrix infographic showing the risks and mitigations. o Groups will share their policies with the class and discuss their rationale. • **Pair activity – Analysing policy implementation** o Provide pairs with a scenario involving weak implementation of one of the policies (e.g., an employee using an unsecured personal device under a BYOD program). Their tasks are to: Analyse the weaknesses in the existing policy or its enforcement. Propose practical solutions to improve implementation. Summarise their findings in a cause-and-effect diagram. o Pairs will exchange diagrams with another pair for peer feedback.	Bring Your Own Device (BYOD) Guidance: The Information Commissioner's Office provides comprehensive guidance on implementing BYOD policies, including considerations for data protection and security measures. https://ico.org.uk/media/for-organisations/documents/1563/ico_bring_your_own_device_byod_guidance.pdf Remote Working Security Policy: This sample policy outlines security measures and best practices for remote working, covering aspects such as acceptable use, information systems security, and employee responsibilities https://cdn2.hubspot.net/hubfs/4071802/Remote_Working_Policy.pdf

	• **Individual activity – Organisational security policy portfolio** o Each student will create a security policy portfolio for a medium-sized organisation. The portfolio should include: A BYOD policy outlining device requirements and security measures. A remote work policy covering network access, device usage, and secure communication. Physical security measures, such as access controls and surveillance. o A third-party security policy addressing vetting, contracts, and compliance monitoring.	Physical Security Standards: The UK Government provides guidelines on physical security measures to protect organisational assets, including access controls, surveillance, and environmental controls. https://www.gov.uk/government/publications/government-security-classifications/guidance-14-working-remotely-at-official-and-secret-html
B4 Legal issues related to computer network security and encryption B4.1 General Data Protection Regulation (GDPR) B4.2 Legislation relating to computer misuse	• **Whole class teaching and learning – Introduction** o Introduce the legal framework governing computer network security in the UK, focusing on GDPR and computer misuse legislation. o Use a presentation to explain GDPR's requirements for data protection and privacy rights, including its key principles (e.g., data minimisation, consent, and accountability). o Follow with an overview of laws targeting unauthorised access, modification, and misuse of computer systems, such as the Computer Misuse Act 1990. • **Small group activity – GDPR in practice** o Assign groups a case study involving a GDPR violation (e.g., a company fined for a data breach). Their tasks are to: Identify the GDPR principles violated in the case. Discuss the consequences for the organisation and affected individuals. Propose measures the organisation could have taken to remain compliant. o Groups will present their findings as a comparison infographic showing the violations and corrective actions.	Guide to the Data Protection Principles: The Information Commissioner's Office (ICO) provides an overview of the seven key principles of the UK GDPR, essential for understanding data protection requirements. https://ico.org.uk/for-organisations/uk-gdpr-guidance-and-resources/data-protection-principles/ Computer Misuse Act Guidance: This document provides detailed guidance on prosecuting offences under the Computer Misuse Act 1990, covering unauthorised access and other related crimes. https://www.cps.gov.uk/legal-guidance/computer-misuse-act

	• **Pair activity – Analysing computer misuse scenarios** o Provide pairs with a scenario involving potential computer misuse (e.g., an employee sharing sensitive company data or unauthorised access to systems). Their tasks are to: Identify which aspects of the Computer Misuse Act may apply. Discuss the potential legal consequences for the perpetrator and the organisation. Draft a short incident report summarising the case and preventive recommendations. o Pairs will share their reports with the class for discussion. • **Individual activity – Legal compliance guide** o Each student will create a concise guide for an organisation on ensuring legal compliance with GDPR and computer misuse legislation. The guide should include: Key GDPR requirements and examples of best practices (e.g., data encryption, access control). Steps to prevent violations of computer misuse laws, such as employee training and access management. o A checklist for periodic audits to maintain compliance.	Cybercrime Prosecution Guidance: This resource offers insights into prosecuting cybercrimes, including offences under the Computer Misuse Act and data protection violations. https://www.cps.gov.uk/legal-guidance/cybercrime-prosecution-guidance

C: Encryption		
C1 Methods and techniques for data encryption C1.1 Data communication, the role of data packets in transmitting data over a network C1.1.1 contents and structure of a data packet C1.1.2 the purpose and function of each component of the data packet	• **Whole class instruction teaching and learning – Introduction** o Introduce the concept of data packets in network communication. o Use an infographic to visually represent the structure and components of a data packet. o Highlight the purpose of each component and how packets facilitate data transmission over networks. o Encourage students to analyse why packet switching is efficient and discuss its role in reliable data transfer. • **Small group activity – Dissecting a data packet** o In groups, students will examine a sample data packet's structure. Each group will: Identify and describe each component (header, payload, trailer). Discuss the role of each component in ensuring data integrity and proper delivery. Present findings to the class using a flow diagram. • **Pair activity – Packet switching simulation** o Students will simulate packet switching by creating a physical model: Use cards to represent data packets with sections labelled as header, payload, and trailer. Pairs will simulate the transmission of packets through a “network” with interruptions (e.g., rerouting). Reflect on how reordering occurs and the role of acknowledgments and retransmissions. • **Individual activity – Network protocol research** o Students will independently create a comparison table for key network protocols (TCP, UDP, HTTP, HTTPS). The table will include: Protocol name Purpose Security implications Examples of use.	Introduction to IPv4 Datagram Header: This resource explains the structure and fields of the IPv4 header, detailing each component's purpose in data transmission. https://www.geeksforgeeks.org/introduction-and-ipv4-datagram-header/

C1.1 Data communication, the role of data packets in transmitting data over a network C1.1.3 the concept of packet switching	• **Whole class instruction teaching and learning – Introduction** o Introduce the concept of packet switching by presenting a timeline infographic that traces the evolution of data communication methods, leading to the development of packet switching. o Discuss its relevance in modern networks and use examples like video streaming or VoIP calls to illustrate its practical applications. • **Small group activity – Network design challenge** o Groups will design a simple network model demonstrating packet switching: Use diagrams to depict nodes, packets, and multiple paths. Include labels to show how packets are routed, rerouted, and reassembled. Present their models and explain the decision-making process for packet routing in their designs. • **Pair activity – Analysing real-world systems** o Pairs will research and prepare a poster on how packet switching is implemented in a real-world system, such as the internet or a mobile network. The poster must include: A labelled diagram of packet flow Examples of packet-switching protocols used Security considerations in packet switching • **Individual activity – Packet switching comic strip** o Students will create a comic strip that narrates the journey of a data packet through a network using packet switching. Include: Key components (e.g., nodes, routers) Obstacles (e.g., dropped packets, congestion) Resolution (e.g., retransmission, reassembly)	Packet Switching and Delays in Computer Networks: This article discusses packet switching, highlighting its efficiency in data transmission and its role in reliable data transfer. https://www.geeksforgeeks.org/packet-switching-and-delays-in-computer-network/

C1.2 IP addresses (IPV4 and IPV6) and Media Access Control (MAC) addresses and their purpose	• **Whole class instruction teaching and learning – Introduction** o Explain the differences between IPv4, IPv6, and MAC addresses using a hierarchical diagram. o Highlight the structure of each type of address, their purpose in network communication, and their roles in identifying devices and routing data. o Provide examples of when each address type is used (e.g., IPv6 for modern networks due to address space). • **Small group activity – Address comparison exercise** o Groups will compare IPv4, IPv6, and MAC addresses using a comparison table. They will: List key features (e.g., address length, format, examples). Discuss scenarios where each address type is most effective. Present a short summary of their comparison using an infographic. • **Pair activity – Decoding and identifying addresses** o Each pair will analyse a set of sample addresses (IPv4, IPv6, MAC). Their tasks will include: Identifying the address type and explaining its format. Explaining the purpose of the address in a network communication scenario. Discussing any security considerations tied to the use of these addresses (e.g., spoofing). • **Individual activity – Design a learning aid** o Students will create a reference sheet summarising: IPv4 structure and examples IPv6 structure and examples MAC address format and its use in devices. o Include diagrams or flowcharts to visually represent how these addresses are used in a network.	IP Addresses (IPv4, IPv6), MAC Addresses & URLs: This article provides an overview of IPv4, IPv6, and MAC addresses, explaining their formats and purposes in network communication. https://www.101computing.net/ip-addresses-ipv4-ipv6-mac-addresses-urls/ Difference Between MAC Address and IP Address: This article explains the distinctions between MAC addresses and IP addresses, highlighting their functions and significance in networking. https://www.geeksforgeeks.org/difference-between-mac-address-and-ip-address/

C1.3 The purpose and function of common network protocols and implications for security	• **Whole class instruction teaching and learning – Introduction** o Provide an overview of common network protocols and their roles in communication and data transfer. o Use a process diagram to categorise protocols by function (e.g., data transfer, email communication, web browsing). o Discuss the security implications of each, such as the transition from HTTP to HTTPS and the importance of protocols like TLS for secure communication. • **Small group activity – Protocol case studies** o Assign each group a specific protocol (e.g., HTTPS, FTP, SMTP). Groups will: Research their assigned protocol's function, common use cases, and potential vulnerabilities. Create a system diagram showing how their protocol fits into network communication. Present findings to the class, including one real-world example of the protocol in use. • **Individual activity – Protocol comparison chart** o Students will independently complete a comparison chart that includes: Protocol name Purpose Secure vs. non-secure versions (e.g., HTTP vs. HTTPS) Examples of usage scenarios Security implications	16 Most Common Network Protocols You Should Know: This resource discusses essential network protocols, explaining their roles in communication and highlighting potential security concerns. https://www.auvik.com/franklyit/blog/common-network-protocols/ Types of Internet Protocols: This resource discusses different internet protocols such as SMTP, FTP, and DNS, elaborating on their roles in network communication and associated security considerations. https://www.geeksforgeeks.org/types-of-internet-protocols/

C1.4 Symmetric and asymmetric encryption	- **Whole class instruction teaching and learning – Introduction** o Introduce the concepts of symmetric and asymmetric encryption. o Use a visual comparison diagram to highlight key differences such as key usage, encryption speed, and typical applications. Include real-world examples (e.g., AES for symmetric, RSA for asymmetric). o Demonstrate how asymmetric encryption facilitates secure key exchange in symmetric encryption systems. - **Small group activity – Encryption scenarios** o Assign each group a scenario where encryption is required (e.g., securing a database, sending an email, encrypting a file). Each group will: Decide whether symmetric or asymmetric encryption is more suitable. Justify their choice by outlining the advantages and limitations of their selected method. Present their solution using a process flow diagram. - **Pair activity – Key exchange simulation** o Pairs will simulate a secure communication process: One student encrypts a message using a symmetric key. The other simulates key exchange using asymmetric encryption. Reflect on how the two methods complement each other in real-world applications. - **Individual activity – Encryption method infographic** o Students will design an infographic summarising: How symmetric encryption works (single key for encryption/decryption) How asymmetric encryption works (public/private key pairs) Examples of encryption algorithms for each type Advantages and disadvantages of each method.	Symmetric vs. Asymmetric Encryption: Understanding the Differences: This article provides a detailed comparison between symmetric and asymmetric encryption, discussing their key differences, advantages, and typical use cases. https://www.commandlink.com/symmetric-vs-asymmetric-encryption-understanding-the-differences/

C1.5 Encryption methods C1.5.1 simple/pre-computer cyphers	- **Whole class instruction teaching and learning – Introduction** o Introduce simple/pre-computer ciphers and their historical significance. Use examples like Caesar cipher (substitution) and rail fence cipher (transposition) to demonstrate how they function. o Show how XOR operates at the binary level and explain how steganographic techniques hide data within other media. o Provide real-world applications, such as using one-time pads in secure communications. - **Small group activity – Cipher creation challenge** o Groups will create their own simple cipher based on one of the methods: Choose a cipher type (e.g., substitution, transposition). Develop a rule for encoding and decoding a message. Share their cipher with another group to decode the message. o Groups will summarise the strengths and weaknesses of their cipher in a brief presentation. - **Pair activity – Steganographic puzzle** o Pairs will embed and retrieve hidden messages using steganography: Use physical or digital media to embed a short message (e.g., writing within images or text). Exchange media with another pair and attempt to retrieve the hidden message. Discuss how this technique could be applied or detected in modern cybersecurity.	Cryptography – One-Time Pad Cipher: This tutorial provides an overview of the one-time pad cipher, its working principles, and its advantages and drawbacks. https://www.tutorialspoint.com/cryptography/cryptography_one_time_pad_cipher.htm

C1.5 Encryption methods C1.5.2 Modern, computer encryption	• **Whole class instruction teaching and learning – Introduction** o Explain the fundamental principles of modern encryption methods. o Use diagrams to illustrate the symmetric nature of AES and the asymmetric nature of RSA. o Highlight their common applications, such as AES for encrypting files and RSA for secure data transmission. o Discuss key aspects like block size, key length, and the importance of computational efficiency and security. • **Small group activity – Encryption algorithm comparison** o Groups will analyse AES and RSA, focusing on: How they work (step-by-step breakdown of their encryption processes). Strengths and weaknesses (e.g., speed, security level). Real-world applications. o Groups will summarise findings using a comparison infographic or table. • **Pair activity – Encryption in action** o Pairs will simulate the use of AES and RSA: Encrypt and decrypt a short text using a simplified AES model (e.g., substitution and permutations). Exchange a message using RSA, where one student encrypts with the public key, and the other decrypts with the private key. Reflect on how the encryption methods ensure confidentiality and integrity in communication.	Difference Between AES and RSA Encryption: This article provides a detailed comparison between AES and RSA encryption, discussing their operational mechanisms, strengths, and typical use cases. https://www.geeksforgeeks.org/difference-between-aes-and-rsa-encryption/

C1.6 Digital signatures C1.7 Digital certificates C1.8 Public Key Infrastructure (PKI)	• **Whole class instruction teaching and learning – Introduction** o Introduce the concepts of digital signatures, digital certificates, and PKI. o Use a hierarchical diagram to show how these components interrelate to ensure secure communication. Explain: Digital signatures: how they verify data integrity and authenticity. Digital certificates: their role in verifying public key ownership. PKI: the infrastructure that supports secure certificate management. o Use examples like online banking and e-commerce to illustrate practical applications. • **Individual activity – Visual explanation** o Students will create a visual representation (e.g., infographic or poster) to explain: How digital signatures provide data integrity and authenticity. The purpose and structure of digital certificates. The components and operation of PKI. o Include practical examples such as HTTPS or email encryption.	Digital Signatures and Certificates: This article provides an overview of digital signatures and certificates, explaining their components and roles in secure communications. https://www.geeksforgeeks.org/digital-signatures-certificates/
C1.9 Hashing algorithms C1.9.1 message-digest algorithm (MD5) C1.9.2 Secure Hash Algorithms (SHA-2 and SHA-3)	• **Whole class instruction teaching and learning – Introduction** o Explain the concept of hashing and its uses in data integrity, password storage, and digital signatures. o Use diagrams to show how input data is transformed into a fixed-size hash value. o Introduce MD5, SHA-2, and SHA-3, highlighting their structures, strengths, and vulnerabilities. o Discuss how collision resistance and computational efficiency are critical in secure hashing.	Hash Algorithm Comparison: MD5, SHA-1, SHA-2 & SHA-3: This article provides a comparative analysis of various hash algorithms, detailing their differences, strengths, and weaknesses. https://codesigningstore.com/hash-algorithm-comparison

	• **Small group activity – Hashing experiment** o Groups will perform a hashing activity using a hashing tool or manual simulation: Input different strings into a hashing function (or provided mock hashing tool) to observe unique hash outputs. Test slight changes to the input and note the drastic changes in output (avalanche effect). Discuss the implications of weak hashes like MD5 in modern security contexts and present findings using a cause-and-effect diagram. • **Pair activity – Compare and evaluate** o Pairs will research and create a Venn diagram comparing MD5, SHA-2, and SHA-3 based on: Strengths and weaknesses (e.g., collision resistance, speed). Common applications (e.g., MD5 for checksums, SHA for blockchain). Vulnerabilities and modern relevance. o Pairs will present their findings in a brief class discussion. • **Individual activity – hashing analysis report** o Students will write a short report addressing: Why MD5 is considered insecure for cryptographic purposes. How SHA-2 and SHA-3 improve upon earlier algorithms. Practical applications for hashing algorithms in cybersecurity (e.g., verifying file integrity). o Include examples and real-world cases where weak hashing algorithms led to security breaches.	

D: Evaluating cyber security and encryption solutions		
D1 Appropriate cyber security and encryption methods and techniques to secure data transmission and storage on a network D1.1 Use of Transport Layer Security (TLS)	• **Whole class instruction teaching and learning – Introduction** o Introduce Transport Layer Security (TLS) by explaining its role in securing data transmission over networks. o Use a diagram to illustrate how TLS encrypts data between client and server, emphasising its components: handshake, encryption, and authentication. o Discuss its advantages, such as data confidentiality and integrity, and potential limitations. • **Pair activity – Encryption key management** o Students will pair up to investigate encryption key management practices. Each pair will: Research how key management contributes to secure TLS operations. Identify real-world examples of failures in key management and their impacts. Create a checklist for best practices in key management. o Pairs will share their checklists with the class for discussion. • **Individual activity – Research on secure communication protocols** o Students will individually research secure communication protocols related to TLS, such as SSL and VPNs. They will: Write a short analysis comparing these protocols. Evaluate their effectiveness and limitations in securing network communications. Suggest specific use cases for each protocol. o The analyses will be compiled into a class resource document.	Key Management Cheat Sheet: This resource offers guidelines on effective encryption key management, crucial for secure TLS operations. https://cheatsheetseries.owasp.org/cheatsheets/Key_Management_Cheat_Sheet.html SSL/TLS Best Practices: This article discusses best practices for SSL/TLS implementation, focusing on encryption algorithms and protocol versions. https://www.ssl.com/guide/ssl-best-practices/

D1.2 Use of VPN and secure communication protocols	• **Whole class instruction teaching and learning – Introduction** o Explain the role of Virtual Private Networks (VPNs) and secure communication protocols (IPsec, SSL, TLS) in ensuring data security over networks. o Use examples to show how these protocols protect data confidentiality and integrity in different environments (e.g., remote work, online banking). o Visualise their layers of operation within the OSI model to highlight differences and overlaps. • **Small group activity – Protocol comparison workshop** o In groups, students will: Compare IPsec, SSL, and TLS in terms of security features, performance, and use cases. Use a decision matrix to analyse which protocol is most suitable for specific scenarios, such as a corporate VPN or e-commerce website. Summarise their findings on a poster or infographic. o Groups will display their work and participate in a gallery walk to review others' findings. • **Pair activity – VPN implementation analysis** o Each pair will: Investigate a real-world VPN implementation (e.g., corporate, consumer-grade, or cloud-based VPNs). Discuss how IPsec or SSL/TLS is used in the VPN. Create a concept map showing how the VPN architecture ensures secure communication. o Pairs will present their concept maps to the class.	IPsec VPNs: Operating at the network layer (Layer 3) of the OSI model, IPsec secures all data transmitted across the network by creating secure tunnels that encapsulate data packets. It's commonly used for site-to-site connections, effectively linking two segments of a private network over the internet. https://www.paloaltonetworks.com/cyberpedia/ipsec-vs-ssl-vpn Configuration Complexity: IPsec VPNs can be more complex to set up and manage compared to SSL/TLS VPNs, which are often simpler to implement and use. https://www.ninjaone.com/blog/ssl-vpn-vs-ipsec/

D1.3 Encryption and key management	• **Whole class instruction teaching and learning – Introduction** o Explain the importance of encryption and key management in securing data transmission and storage. o Use examples such as symmetric and asymmetric encryption to illustrate how keys are generated, distributed, and stored. o Highlight the risks of poor key management, including unauthorised access and data breaches. Use a flowchart to demonstrate the lifecycle of encryption keys. • **Small group activity – Encryption key lifecycle exercise** o In groups, students will: Examine the stages of the encryption key lifecycle (generation, distribution, use, rotation, and destruction). Identify risks and potential failures at each stage and suggest mitigation strategies. Create a process diagram to visually represent best practices for key management. o Groups will present their process diagrams and answer questions from peers. • **Pair activity – Case study analysis on key management failures** o Each pair will: Research a real-world example where poor key management led to a data breach (e.g., hardcoded keys, weak key storage). Analyse what went wrong and how proper key management could have prevented the issue. Write a short report summarising their findings and recommendations. o Pairs will share their case studies in a class discussion.	Centralise Key Management: Utilise a unified platform to oversee all key-related operations, which enhances control and reduces complexity. https://phoenixnap.com/blog/encryption-key-management-best-practices Equifax Data Breach (2017): A failure to renew an expired certificate delayed the detection of a breach, exposing sensitive information of approximately 147 million individuals. https://www.csoonline.com/article/567833/equifax-data-breach-faq-what-happened-who-was-affected-what-was-the-impact.html

	- **Individual activity – Encryption strategy proposal** - Students will: Design an encryption strategy for a hypothetical company, specifying key management policies, such as secure storage (e.g., hardware security modules), rotation schedules, and user roles. Include a risk assessment identifying potential vulnerabilities in the proposed strategy. Submit the strategy for review.	
D1.4 Data encryption at rest	- **Whole class instruction teaching and learning – Introduction** - Introduce the concept of data encryption at rest by explaining the difference between full disk encryption and file encryption. - Use examples such as BitLocker (for disk encryption) and AxCrypt (for file encryption) to illustrate practical applications. - Discuss how encryption at rest ensures data security even if physical devices are compromised. - **Individual activity – Designing encryption policies** - Students will: Draft a brief encryption policy for an organisation of their choice, specifying the use of full disk encryption and file encryption. Include guidelines for key management, user training, and device management. Submit their policies for peer feedback.	Understanding Full Disk and File Encryption: This article explains the differences between full disk encryption and file encryption, using BitLocker and AxCrypt as examples. https://axcrypt.net/blog/the-ultimate-guide-to-file-encryption-vs-disk-encryption-which-one-is-best-for-you/

D2.1 Security level D2.2 Performance D2.3 Cost D2.4 Usability D2.5 Scalability	• **Whole class instruction teaching and learning – Introduction** o Introduce the concept of selecting cyber security and encryption solutions based on network requirements. o Explain key evaluation criteria: security level, performance, cost, usability, and scalability. o Use examples such as comparing a small business network to a large enterprise network to illustrate how requirements shape solution choices. • **Small group activity – Network solution evaluation** o In groups, students will: Be given different network scenarios (e.g., a school network, a multinational corporate network, a home IoT setup). Analyse the given scenario to prioritise requirements (e.g., security level over cost for a corporate network). Recommend suitable encryption and cyber security solutions, justifying their choices. o Groups will present their recommendations using comparison tables or matrices. • **Individual activity – Network solution proposal** o Students will: Select a type of network (e.g., VPN, IoT network, cloud network). Write a proposal recommending cyber security and encryption solutions tailored to that network's requirements, addressing security level, performance, cost, usability, and scalability. Submit the proposal for review and peer feedback.	Evaluating Information Security Solutions: Swapping the Cost of Failure for Success: This article discusses methodologies like SWOT analysis to evaluate information security solutions, considering strengths, weaknesses, opportunities, and threats. https://www.isaca.org/resources/isaca-journal/issues/2015/volume-2/evaluating-information-security-solutions-swapping-the-cost-of-failure-for-success

D2.6 Compatibility D2.7 Constraints D2.8 Risks D2.9 Benefits	• **Whole class instruction teaching and learning – Introduction** o Explain the importance of evaluating compatibility, constraints, and risks when selecting cyber security and encryption solutions for networks. o Use examples such as integrating encryption tools into legacy systems or balancing security with operational limitations. o Highlight potential risks such as misconfigurations or vulnerabilities introduced by third-party tools. • **Small group activity – Compatibility assessment** o In groups, students will: Be given different network setups (e.g., a legacy corporate network, a hybrid cloud system, an IoT-based smart home). Analyse the compatibility of proposed encryption solutions with existing systems and infrastructure. Create a Venn diagram showing overlapping features, compatibility gaps, and areas requiring modification. o Groups will present their findings to the class for discussion. • **Pair activity – Identifying constraints and risks** o Pairs will: Investigate a specific encryption solution (e.g., VPNs, TLS, or IPsec). Identify potential constraints, such as hardware requirements, implementation complexity, or regulatory limitations. List associated risks, such as incorrect configuration, performance degradation, or insufficient user training. o Create a cause-and-effect diagram showing how constraints might lead to risks. o Pairs will share their diagrams with the class.	Addressing Integration Barriers with Legacy Data Security Systems: This article discusses the challenges of integrating encryption tools into legacy systems, highlighting potential security risks such as outdated software and lack of security updates. https://www.datasecurityintegrations.com/challenges/addressing-integration-barriers-legacy-data/

	- **Individual activity – Risk mitigation plan** o Students will: Select a network type and propose a cyber security or encryption solution. Identify compatibility challenges, constraints, and associated risks. Develop a risk mitigation plan, including actions to address each identified issue (e.g., upgrading hardware, training users, or implementing layered security measures).	

Delivering signposted transferable skills

Signposted transferable skills are not mandatory for the delivery of the unit, and it is therefore your decision to deliver these skills as a part of the qualification. Below we have provided some ideas of teaching and learning activities that you could use to deliver these skills if you chose to.

Transferable skills	Ideas for delivery
SP – CT Critical thinking	Case study analysis: • Summarise the key issues in their own words • Create mind maps identifying core problems and related factors • Write problem statements that demonstrate understanding of complex situations Research projects: • Create annotated bibliographies justifying their source selections • Compare contrasting viewpoints from different sources • Build evidence tables showing which information supports different arguments • Develop research portfolios documenting their information gathering process Evaluation exercises: • Fact-check news articles and identify potential bias • Create credibility scorecards for different sources • Design evaluation matrices comparing different solutions • Lead peer discussions analysing the strength of different arguments Real-world problem-solving: • Create decision trees showing their analytical process • Develop presentations explaining how they reached their conclusions • Participate in structured debates defending their reasoning

Resources

This section has been created to provide a range of links and resources that are publicly available that you might find helpful in supporting your teaching and delivery of this unit in the qualification. We leave it to you, as a professional educator, to decide if any of these resources are right for you and your students, and how best to use them.

Pearson is not responsible for the content of any external internet sites. It is essential that you preview each website before using it to ensure the URL is still accurate, relevant, and appropriate. We'd also suggest that you bookmark useful websites and consider enabling students to access them through the school/college intranet.

Websites

Action Fraud – UK's national fraud and cybercrime reporting centre.
www.actionfraud.police.uk

BCS Security Group – Professional body for IT practitioners with a security focus.
www.bcs.org/security

Centre for the Protection of National Infrastructure (CPNI) – UK government authority for protective security advice.
www.cpni.gov.uk

Cyber Aware – UK government's advice on how to stay secure online.
www.cyberaware.gov.uk

Cyber Essentials – UK government-backed certification scheme.
www.cyberessentials.ncsc.gov.uk

Cyber Security Challenge UK – Organisation running cybersecurity competitions and learning programs.
www.cybersecuritychallenge.org.uk

Department for Digital, Culture, Media & Sport (DCMS) – UK government department overseeing cyber policy.
www.dcms.gov.uk/cyber-security

Government Communications Headquarters (GCHQ) – UK's intelligence and security organisation.
www.gchq.gov.uk

Information Commissioner's Office (ICO) – UK's independent authority for data protection and information rights.
www.ico.org.uk

Institute of Information Security Professionals (IISP) – UK's professional body for cybersecurity practitioners.
www.iisp.org

ISO 27001:2013 – International Organization for Standardization's information security standard.
www.iso.org/isoiec-27001-information-security.html

Kaspersky Live Cyber Attack Map
https://cybermap.kaspersky.com/

National Cyber Security Centre (NCSC) – UK's technical authority for cyber security incidents and guidance.
www.ncsc.gov.uk

National Institute of Standards and Technology (NIST) – US authority for cybersecurity standards and guidelines.
www.nist.gov

Open Web Application Security Project (OWASP) – International web security community and standards.
www.owasp.org

Radware Live Threat Map
https://livethreatmap.radware.com/

Textbooks

Bhardwaj, D.A., Kaushik, K., Practical Digital Forensics: Forensic Lab Setup, Evidence Analysis, and Structured Investigation Across Windows, Mobile, Browser, HDD, and Memory, BPB Publications, 2023, (978-93-5551-145-4).

Kiser, Q., Computer Networking and Cybersecurity: A Guide to Understanding Communications Systems, Internet Connections, and Network Security Along with Protection from Hacking and Cyber Security Threats, 2020, (9798682990887).

Lammle, T., Buhagiar, J., CompTIA Network+ Study Guide: Exam N10-009, Sybex, 2024, (978-1-394-23560-5).

Neil, I., CompTIA Security+ SY0-701 Certification Guide: Master cybersecurity fundamentals and pass the SY0-701 exam on your first attempt;, Packt Publishing, 2024, (978-1-83546-153-2).

Walker, S., Cybersecurity Bible: The Complete Guide to Detect, Prevent and Manage Cyber Threats | Includes Practical Tests & Hacking Tips for IT Security Specialists, 2024, (9798336422184).

Pearson paid resources also available

- Pearson Student book
- ActiveBook (a digital version of the Student Book, via ActiveLearn Digital Service)
- Digital Teacher Pack (via ActiveLearn Digital Service)

Unit 3: Human-Computer Interaction

Unit overview

Unit 3: Human-Computer Interaction	
Assessment type: Internal	
Learning Aim	**Topics**
A Explore the factors affecting human-computer interaction	A1 Fundamental concepts of human-computer interaction A2 Use and purpose considerations A3 The principles of HCI design
B Develop a proposal and designs for a human-computer interaction solution in response to a brief	B1 Defining requirements for a HCI solution B2 Design documentation for a HCI solution
C Develop your planned human-computer interaction solution in response to the brief.	C1 Content preparation for a human-computer interface C2 Developing a HCI solution C3 Testing a HCI solution C4 Reviewing the development process and outcomes
Assessment overview This unit is Internal assessed through a Pearson-Set Assignment Brief (PASB). Pearson sets the assignment for the assessment of this unit. The PSAB will take approximately 20 hours to complete. The PSAB will be marked by centres and verified by Pearson. The PSAB will be valid for the lifetime of this qualification.	

Common student misconceptions

Below are some common misconceptions related to the content of this unit by students and ideas for how you can help your students to avoid and overcome these.

What is the misconception?	How to help students overcome it
UX and UI are the same thing.	Explain that UX (User Experience) is about the user's overall experience and satisfaction, while UI (User Interface) focuses on the design of interactive elements. Use examples to show how they contribute differently to user interaction.
Accessibility only concerns physical disabilities.	Illustrate that accessibility also covers cognitive, sensory, and technological limitations, such as low vision or limited access to high-speed internet.
Visual aesthetics are more important than functionality.	Use case studies to show that while aesthetics are significant, functionality ensures user retention and satisfaction.
Users will adapt to any design.	Use examples of poor design leading to user frustration to show the need for user-centred design principles.
Designing for mobile is just about shrinking desktop designs.	Discuss mobile-specific considerations like touch targets, limited screen space, and gesture-based interactions.

Learning Activities and Resources

This section offers a starting point for delivering the unit by outlining a logical sequence through the unit topics and suggesting practical activities and teacher guidance for covering the main areas of content during guided learning time. Transferable skills are integrated into various activities, with those embedded in a unit indicated by an acronym in square brackets. The acronym combines the letters from the broad skill area and the specific transferable skill, e.g., **[IS-WC]**.

Please note that the activities provided below are suggestions and not mandatory.

Learning Topic	Activities and guidance for unit content delivery	Resources
Learning aim A: Explore the factors affecting human-computer interaction		
A1 Fundamental concepts of human-computer interaction	• **Whole class teaching and learning – Introduction** o Introduce the fundamental concepts of human-computer interaction (HCI) by discussing the differences and connections between user experience (UX) and user interface (UI) design. Illustrate with examples that emphasise how UX and UI contribute to the effectiveness of digital systems. • **Whole class teaching and small group activity – UX Design** o Introduce the fundamental concepts of UX by discussing the impact of usability, accessibility, efficiency, engagement, and enjoyment on effective digital systems. o Use a class mind map on the board to connect these concepts with everyday digital experiences, encouraging students to provide examples of positive and negative user interactions. o Organise students in groups. Each group should rotate between stations, each focused on one UX concept (usability, accessibility, efficiency, engagement, enjoyment). At each station, groups brainstorm real-world applications, document challenges, and suggest improvements. Groups share insights with the class to highlight how each factor contributes to an effective UX.	Comprehensive UK Government design principles and patterns: https://www.gov.uk/service-manual/design BBC Global Experience Language (GEL) design system: https://www.bbc.co.uk/gel HCI course https://youtu.be/CUTnU7y3s0U?si=oAY1InCE7lCBfr0l

	• **Whole class teaching and small group activity – UI Design** o Introduce the essential UI principles by explaining each concept—simplicity, clarity, consistency, feedback, accuracy, efficiency, flexibility, and accessibility. o Use visual examples of well-designed and poorly designed interfaces to illustrate each principle, encouraging students to identify which concepts are effectively or ineffectively applied. o Organise students in groups. Ask each group to analyse the UI of a chosen application or website. Each group assess the interface based on the key UI principles, specifically noting elements like clarity, consistency, and feedback. Groups present their critiques, highlighting improvements to enhance user experience. • **Whole class teaching and small group activity – Types of interaction** o Introduce the concept of human-computer interaction (HCI), focusing on different types of user interfaces (e.g., CLI, GUI, menu-driven, voice-activated, and gesture-based). o Explain the characteristics, features and use cases for each interface type. o Lead a class discussion about where students have encountered each type of interface in their daily lives, encouraging them to share their thoughts on the advantages and limitations of each type. o Organise students in groups. Assign each group one type of user interface (CLI, GUI, menu-driven, voice-activated, gesture-based). In their groups, students will: Analyse how their assigned interface type impacts user experience. Create a list of situations where their interface type might be most effective. Identify key drawbacks of their interface type. o Each group presents their findings to the class.	5 Key Principles for Effective User Experience Design: Essential UX design principles, including user needs, accessibility, simplicity, and continuous improvement, complemented by real-world examples. https://ideatheorem.com/insights/blog/ui-ux-design/5-key-principles-for-effective-user-experience-design

A2 Use and purpose considerations	• **Whole class teaching and small group activity – Target platforms** o Introduce the idea of target platforms and devices by examining how different devices serve specific user needs and purposes. o Discuss examples such as how smartphones support quick access to information, while VR systems offer immersive experiences. o Organise students in groups. Each group is assigned a target platform or device (e.g., smartwatches, gaming consoles, VR headsets). Their challenge is to design a basic interface tailored to their assigned platform, considering the platform's limitations, typical users, and unique interaction methods. Groups present their designs to the class, explaining their design choices. • **Whole class teaching and small group activity – Characteristics of target users** o Introduce the concept of user characteristics in HCI, covering types of users (e.g., experts, regular users, beginners/novices), demographics, and accessibility requirements. o Use examples such as educational apps for young children versus productivity tools for professionals to illustrate how user characteristics shape interface design. o In small groups, students create user personas that represent different target user types (e.g., a beginner user with accessibility needs, a tech-savvy professional, an elderly user with limited digital experience). Each group outline key characteristics, goals, and potential challenges for their persona, then present their personas to the class. • **Whole class teaching and paired activity – Industry/Sector needs** o Introduce the importance of industry-specific considerations in HCI design. o Discuss examples from different sectors (e.g., retail, education, healthcare) and how these industries have unique user needs and design requirements, such as the simplicity needed in retail point-of-sale systems versus the precision required in medical monitoring interfaces.	Challenges in Human-Computer Interaction Design for Mobile Devices: This paper discusses the unique challenges of designing HCI for mobile devices, including considerations for different platforms and devices. https://tinyurl.com/mrkufn92 Personas and Empathy Mapping for Understanding Customers and Users: This article discusses the creation and application of personas and empathy maps to gain a deeper understanding of user needs and behaviours. https://persona.qcri.org/blog/personas-and-empathy-mapping-for-understanding-customers-and-users/ 12 Examples of Haptic Technology Used Today: This article discusses various applications of haptic technology, providing real-world examples of how haptic feedback is used in different devices and industries. https://www.builtinsf.com/articles/haptic-technology

	○ Assign each pair an industry (e.g., education, hospitality, manufacturing, retail). Pairs should analyse the primary user needs, required outputs, and design priorities for HCI solutions in their assigned sector. Each pair presents their findings, highlighting the unique challenges and considerations that impact HCI design in their sector. • **Whole class teaching and small group activity – Purpose of intended systems** ○ Introduce the concept of purpose-driven design in HCI, focusing on how different systems serve specific functions (e.g., information delivery, entertainment, task support) and require tailored outputs to meet user needs. ○ Discuss examples like a medical app's data visualisation needs compared to a gaming console's focus on immersive audio-visual feedback. ○ Organise students in groups. Assign each group a system type with a clear purpose (e.g., data visualisation for business, task tracking for fitness, entertainment for gaming). Each group identifies the primary outputs required to meet user needs (such as visual graphs, interactive elements, or real-time feedback). Groups present their findings, focusing on how these outputs align with the system's purpose. ○ Next, each pair selects a system (such as an educational platform or a productivity tool) and considers how it might be redesigned if its purpose changed (e.g., shifting a productivity app to support collaborative project management). Pairs outline the new outputs that would be necessary to meet user needs and briefly present their redesign. • **Whole class teaching and activity – Haptic feedback** ○ Introduce the concept of haptic feedback by explaining how it provides tactile responses to users, enhancing interaction and usability in systems like mobile devices, gaming controllers, and medical tools. ○ Show short video examples or demonstrate devices with haptic feedback (if available) to illustrate its impact on user experience.	Types of Assistive Technologies and Who Uses Them: This resource provides an overview of various assistive technologies, including screen magnifiers, screen readers, and braille displays, detailing their functionalities and the user groups they benefit. https://govtns.github.io/web-a11y-guidance/ka/how-disabled-people-use-the-web/at/types-of-at-and-who-uses-them.html

	○ Lead a brainstorming session with students to generate ideas for where haptic feedback might be useful across different industries, such as in automotive controls, wearable devices, or virtual reality. ○ List each industry on the board, capturing specific ways haptic feedback could enhance user interaction, safety, or realism in each context. • **Whole class instruction teaching and learning – Assistive technologies** ○ Introduce the role of assistive technologies in HCI, discussing various examples such as eye gaze systems, braille displays, screen magnifiers, and sign language avatars. ○ Explain how these technologies help different user groups overcome barriers and provide accessibility in digital interactions. ○ Use examples or short videos to illustrate how these technologies function in real-world applications. ○ Set up stations around the room with information and visuals on different assistive technologies (e.g., a station for screen magnifiers, another for eye gaze technology). Students move between stations, taking notes on the purpose of each technology, the specific needs it addresses, and the ways it integrates into various interfaces.	
A3 The principles of HCI design	• **Whole class activity – Recognition vs. recall demonstration** ○ Introduce the principles of recognition versus recall, intuitive screen design, and menu selection in HCI. ○ Display examples of interfaces that rely on recognition (e.g., icon-based toolbars) versus those that require recall (e.g., text-only command inputs). ○ Ask students to perform simple tasks using each example and discuss how each interface type impacts ease of use and user memory load. ○ Summarise findings on the board to illustrate how design can reduce recall needs by emphasising recognition.	Recognition vs. Recall in UX Design: This article discusses the differences between recognition and recall in user experience design, emphasising the importance of designing interfaces that facilitate recognition to reduce cognitive load. https://www.geeksforgeeks.org/what-is-recognition-vs-recall-in-ux-design/

	• **Whole class teaching and small group activity – Perception** o Conduct a visual experiment using optical illusions or colour contrast images to illustrate how perception affects interpretation. Discuss how elements like contrast, grouping, and proximity can influence user focus and comprehension in digital systems. Record insights on the board to summarise how perception guides effective interface design. o Organise students in groups. Assign each group an existing digital interface (e.g., an app's homepage, a website menu, or a product dashboard). Each group analyses the design and identifies ways to apply Gestalt principles to improve clarity and user flow. Groups redesign the interface with these principles in mind and present their revised designs, explaining how their adjustments enhance user perception. o Next, each pair reviews a specific interface and evaluates how colour and shape are used to draw attention, create hierarchy, and aid recognition. Pairs create a list of elements in the interface that could be adjusted (e.g., improving contrast for readability or using colour coding for categorisation) and briefly share their suggestions with the class. • **Whole class teaching and paired activity – Schneiderman's 8 Golden Rules** o Introduce Schneiderman's 8 Golden Rules of Interface Design, explaining each rule and its importance in creating effective, user-friendly interfaces. o Provide real-life examples of digital systems that follow these principles well, such as consistent layouts in social media platforms, informative feedback in e-commerce sites, and reversible actions in productivity software. o Divide the class into pairs or small groups and assign each group one of Schneiderman's rules (e.g., "Consistency," "Offer informative feedback," "Permit easy reversal of actions"). o Each group finds a digital example that demonstrates their assigned rule effectively. o Groups share their examples with the class, explaining how the rule is applied and how it enhances user experience. • **Whole class teaching and case study activity – Behaviour models**	Gestalt Principles in UI Design: This article explores how Gestalt principles, such as proximity and similarity, influence user interface design, providing practical examples of their application to enhance clarity and user flow. https://www.toptal.com/designers/ui/gestalt-principles-of-design Colour Theory for Designers, Part 1: The Meaning of Colour: This resource delves into colour theory, discussing how different colours can affect user perception and behaviour, and offering guidance on effectively using colour in interface design. https://www.superside.com/blog/gestalt-principles-of-design Applying the 8 Golden Rules of User-Interface Design: This article examines how Shneiderman's rules can be implemented in modern interface design, offering practical advice and case studies to illustrate their effectiveness. https://www.uxmatters.com/mt/archives/2022/10/applying-the-8-golden-rules-of-user-interface-design.php

	○ Introduce behavioural models in UX/UI design, focusing on how understanding user behaviour can enhance interface design. ○ Cover key models such as the Attention model (how users focus on novel or familiar elements), the Intention model (how users' goals shape their interactions), the Emotion model (how users' emotions influence their experience), and the Memory model (how familiarity aids usability). ○ Use examples to show how these models are applied in common digital interfaces. ○ Present a case study of a well-known app or website (e.g., a social media platform or e-commerce site) and analyse it with the class based on the four behavioural models. ○ For instance, discuss how the app grabs user attention with notifications, aligns with user intentions for ease of use, supports memory through familiar icons, and evokes positive emotions through aesthetics. ○ Record observations on the board to highlight how each model shapes user experience.	What is Behavioural Design?: This article provides an overview of behavioural design, explaining how shaping behaviour predictably can encourage desired actions, such as regular donations or environmentally friendly practices. https://uxmag.com/articles/what-is-behavioral-design Behavioural Design: An Overview: This resource discusses the integration of behavioural science principles into UX design, highlighting the overlap between behavioural design and UX design, and how understanding user behaviour can inform design decisions. https://www.thebehavioralscientist.com/behavioral-design-an-overview

Learning aim B: Develop a proposal and designs for a human-computer interaction solution in response to a brief		
B1 Defining requirements for a HCI solution	• **Whole class teaching and learning – Introduction** o Introduce the problem-solving process in the context of HCI by outlining key stages (e.g., identifying the problem, defining requirements, and assessing user needs). o Use a class mind map on the board to collaboratively identify various aspects of a problem and its potential impacts. o Encourage students to contribute examples of problems and discuss the input, output, and accessibility considerations for HCI. • **Small group activity – Problem scope exploration** o Using group investigation, have each group explore a given HCI problem scenario. Their task is to: Identify and prioritise tasks the system should perform. Outline required inputs (e.g., touchscreen, voice command) and outputs (e.g., visual, audio feedback). Consider user needs, including accessibility and environmental factors. o Groups create a process map summarising their findings, which they present to the class. • **Paired activity – User needs analysis** o Students work in pairs for a compare and contrast activity, where they analyse user needs for two different HCI solutions. o Using a Venn diagram, pairs identify shared and unique user needs, with attention to accessibility and purpose. o Students present their findings to the class.	Design Process & Task Analysis in Human-Computer Interaction (HCI): This article provides an overview of the HCI design process, emphasizing goal-directed problem-solving activities informed by intended use, target domain, materials, cost, and feasibility. https://www.geeksforgeeks.org/design-process-task-analysis-hci/ Introduction to HCI Design Process: This presentation outlines the stages of the HCI process, including identifying stakeholders and deploying evaluation techniques at different stages, providing a roadmap for effective HCI design. https://groups.cs.umass.edu/nmahyar/wp-content/uploads/sites/8/2019/01/690A-02-introduction-to-HCI-Design-process.pdf

B2 Design documentation for a HCI solution	• **Small group activity – Research similar systems** o Using a design challenge format, each group chooses a specific type of HCI system (e.g., wearable device, VR interface, or voice-controlled assistant). o Groups research this system type, identify common design practices, and brainstorm ideas on how to enhance it. o They create a flow diagram mapping the user interaction flow they would adopt, incorporating improvements they suggest based on their research. o Groups present their flow diagrams to the class. • **Whole class teaching and small group activity – Style guide development** o Discuss the importance of understanding user requirements in HCI solution design. o In small groups, students create a basic style guide for a mock HCI project, selecting colour palettes, typography, and media elements that reflect a cohesive brand or user experience theme. o They compile their choices and rationale into a reference document and share with the class. • **Whole class teaching and paired activity – Task flows** o Lead an interactive Think-Pair-Share-Square activity to explore user requirements and task flows in HCI design. o Begin with individual brainstorming on the significance of user-cantered design, then move to paired discussions o Bring pairs together to share insights with the whole class. o Conclude with a brief synthesis on the board, highlighting key points. • **Small group activity – Design challenge** o Assign groups a specific HCI scenario for which they create a solution. o Each group draft an overview of user requirements and sketch task flows that illustrate the main actions a user take within the interface. o Groups present their task flows and initial design concepts to the class for feedback.	Interaction Design Patterns: This resource provides an overview of interaction design patterns, which are general repeatable solutions to commonly occurring usability problems in interface design. https://www.interaction-design.org/literature/book/the-glossary-of-human-computer-interaction/interaction-design-patterns Canva Style Guide Generator: Canva provides templates and tools for creating professional style guides, allowing students to experiment with colour palettes, typography, and branding elements. https://www.canva.com/ Adobe Color: This tool helps in selecting cohesive colour palettes and experimenting with colour harmony, making it useful for developing the visual elements of a style guide. https://color.adobe.com/

	• **Individual activity – Wireframe and storyboard creation** o Students produce a wireframe and accompanying storyboard for a selected interface, illustrating user interaction steps and interface features. o These be compiled into a visual document that includes annotations to highlight design decisions. • **Individual activity – Technical requirements outline** o Students draft an outline of the technical specifications for their HCI solution, including functional and non-functional requirements. o They provide a rationale for each requirement, explaining its relevance to the user experience and the project's goals.	Introduction to Sketches, Storyboards, and Prototypes: This resource provides an overview of various design tools, including sketches, storyboards, and prototypes, explaining their roles in the design process and how they contribute to effective HCI solutions. https://lecture-notes.tiu.edu.iq/wp-content/uploads/2024/04/HCI-Lec34_2023-2024.pdf UX Design: Wireframe vs Storyboard vs Wireflow vs Mockup vs Prototyping: This article discusses the differences between various design documentation tools, such as wireframes, storyboards, wireflows, mockups, and prototypes, highlighting their unique purposes and applications in UX design. https://www.visual-paradigm.com/guide/ux-design/wireframe-vs-storyboard-vs-wireflow-vs-mockup-vs-prototyping/

Learning aim C: Develop your planned human-computer interaction solution in response to the brief		
C1 Content preparation for a human-computer interface	• **Whole class teaching and group activity – Selecting appropriate techniques** o Introduce the importance of selecting and applying appropriate processing techniques in HCI design. o Present different content types such as sounds, images, and control code, discussing how each can be optimised for performance, compatibility, and file size. o Use examples to illustrate the effects of compression and format choices on usability and system performance. o Assign each group a specific content type (e.g., sounds, images, or control code) and have them research optimal processing and editing techniques for that type. o Afterward, groups form mixed teams to share their findings, explaining how their techniques apply to different interface requirements and usability needs. • **Whole class teaching and learning – Using third-party content** o Introduce the concept of using third-party content in HCI projects, emphasising the importance of permissions, proper source acknowledgment, and legal and ethical considerations. o Discuss real-world examples where failure to comply with these standards led to legal consequences or public backlash. o Highlight issues like privacy, political bias, and fair representation of individuals or groups. • **Whole class teaching and small group activity – Optimising content** o Introduce the principles of optimising content for HCI. Use examples to explain how file size, image quality, and format affect performance and usability across different devices. o Demonstrate the impact of orientation (landscape vs. portrait) and the importance of selecting compatible file formats to balance quality and performance.	Fundamentals of Image Compression: This resource from Imperial College London offers an in-depth exploration of image compression techniques, discussing methods to reduce image size while maintaining quality. It covers both lossless and lossy compression methods, providing a solid foundation for understanding image optimisation in HCI. https://www.commsp.ee.ic.ac.uk/~tania/teaching/DIP%202014/Image%20Compression%20Fundamentals.pdf Legal and Ethical Considerations in Using Third-Party Content: This article discusses the legal implications of using third-party content, including copyright infringement and the necessity of obtaining proper permissions. It also explores ethical concerns such as privacy and fair representation. https://lawandpixels.com/legal-use-third-party-content/

	o Each group be assigned a different digital content type (e.g., images, audio, or video). o Their task is to create two versions of their content optimised for different requirements: high quality low file size/performance efficiency. o Groups present their optimised versions, discussing choices for file formats, compression, and orientation adjustments. • **Paired activity – back-to-back drawing** o One learner should describe an optimised digital asset they created (such as an image resized or compressed for performance), while the other recreates it based on the description alone. o They then compare their work and discuss how different optimisations impact quality, file size, and usability across devices.	Understanding Responsible and Ethical Use of Technology: This guide from the National Council for Voluntary Organisations provides insights into ethical technology use, including considerations for using third-party content responsibly. https://www.ncvo.org.uk/help-and-guidance/digital-technology/technology-tools-and-software/responsible-and-ethical-use-technology/understanding-ethical-and-responsible-use-technology/
C2 Developing a HCI solution	• **Whole class teaching and learning – Introduction** o Lead a guided discussion on essential HCI design principles, covering primary interface elements like icons, menus, and layouts. o Explain the importance of creating alternative interfaces (e.g., mobile versions). o Discuss real-life examples of software and hardware integrations that enhance user experience, such as adaptive technologies for accessibility or bespoke controllers for specific applications. • **Small group activity – Feature comparison workshop** o Each group be assigned a popular digital interface (e.g., a social media app, an e-commerce platform, or a game). Groups explore how primary interface components are adapted across devices (e.g., desktop vs. mobile) and evaluate software and hardware integrations (e.g., touch controls, voice commands). o Groups present their findings, highlighting HCI principles that make the interface effective and proposing potential improvements.	File Size and Image Quality: Balancing file size and image quality is essential. Large files can slow down performance, while overly compressed files may degrade visual quality. Techniques such as image compression and resizing help reduce file size while maintaining acceptable quality. https://imagekit.io/guides/image-optimization/

	• **Pair activity – Rapid prototyping** o Each pair create a low-fidelity paper prototype for a specific HCI interface, incorporating primary elements such as icons, menus, and layout. o After designing the prototype, pairs swap with another pair and provide feedback on clarity, usability, and any potential for adaptive or mobile-friendly adjustments. o Pairs iterate on their designs based on peer feedback. • **Individual activity – Interface integration log** o Students document an “Interface Integration Log” detailing steps they would take to implement a primary interface component (such as a menu or button) with integrated software events (e.g., hover effects, button clicks) and a planned hardware feature (e.g., gesture control, touchscreen adaptation). o The log include explanations of coding approaches and challenges they might anticipate in integrating these elements into a cohesive HCI design.	
C3 Testing a HCI solution	• **Whole class teaching and learning – Introduction** o Explain the importance of testing in the HCI development process and how it ensures that the solution meets user needs. o Introduce types of testing such as effectiveness, functionality, performance, and user acceptance. o Engage students with a class mind map on the board, illustrating connections between testing stages and types. • **Small group activity – creating a test plan** o In small groups, students produce a test plan for an HCI solution. Their tasks are to: Identify what elements need testing (e.g., interface usability, performance). Choose appropriate test data. Outline criteria for selecting test users. Each group share their test plan and justify their decisions with the class.	Effectiveness Testing: Assesses whether users can achieve their goals using the interface. This involves evaluating task completion rates and identifying obstacles that hinder user success. https://www.interaction-design.org/literature/book/the-encyclopedia-of-human-compute`r-interaction-2nd-ed/usability-evaluation

	• **Pair activity – test data creation** o Pairs design test data sets that cover a variety of scenarios, including normal, boundary, and erroneous data. They analyse the test data sets to ensure comprehensive testing coverage for functionality and user experience. Pairs swap their test data with another pair to review for completeness and accuracy.	Select Appropriate Test Data: Design test data that encompasses a range of scenarios, including typical use cases, boundary conditions, and potential error situations. https://maze.co/guides/usability-testing/methods/
C4 Reviewing the development process and outcomes	• **Whole class teaching and learning – Introduction** o Discuss the purpose and importance of reviewing the development process in HCI, focusing on assessing how well the solution meets the given brief. o Use a class mind map to outline key review elements, such as audience suitability, ease of use, and solution quality. o Encourage students to think critically about factors that may impact the review, like constraints and legal/ethical considerations. • **Small group activity – Solution review and critique** o Each group receive a sample HCI solution (or description of one). They will: Analyse it according to review criteria (e.g., audience suitability, usability, quality). Identify the solution's strengths and weaknesses. List possible improvements. o Groups present their findings, and other groups provide feedback on their review approach and suggestions. • **Pair activity – Strengths and weaknesses analysis** o Pairs choose one element of a completed HCI solution (such as usability or performance) and conduct an in-depth analysis of its strengths and weaknesses. o Each pair record and share how this element impacts the overall quality and user experience, then suggest practical improvements that could enhance this aspect of the solution.	Solution Review and Critique: Analysing a sample HCI solution against established criteria to identify strengths, weaknesses, and areas for improvement. https://www.cl.cam.ac.uk/teaching/1213/HCI/HCI2013-lecture8.pdf Evaluation Methods in HCI: The University of Edinburgh's open course materials provide an overview of evaluation methods, including Cognitive Walkthrough, Think Aloud, and Questionnaire techniques, with practical examples. https://opencourse.inf.ed.ac.uk/hci/week4

	• **Individual activity – Review report** o Each learner individually write a review report on a developed HCI solution, covering: how well it meets the audience’s needs and purpose, the solution’s overall quality and any limitations due to constraints, potential legal or ethical issues, a reflective analysis of possible improvements. This report should be thorough, demonstrating critical evaluation and clear recommendations. • **Individual activity – Self-evaluation checklist** o Using a prepared checklist, students independently assess their own HCI solution, rating each criterion (e.g., usability, efficiency, ethical considerations) and providing explanations for their ratings.	

Delivering signposted transferable skills

Signposted transferable skills are not mandatory for the delivery of the unit, and it is therefore your decision to deliver these skills as a part of the qualification. Below we have provided some ideas of teaching and learning activities that you could use to deliver these skills if you chose to.

Transferable skills	Ideas for delivery
IS – WC Written communications	Create formal documents with executive summaries and recommendations. Write clear step-by-step instructions for processes. Write and review professional emails.
SP – PS Problem solving	Present real-world business problems or community issues. Students must: • Define the core problem in their own words • Create an information gathering plan • List potential solutions with pros/cons • Present and defend their recommended solution. Present complex scenarios. Students must: • State the problem clearly • List criteria for evaluation • Gather relevant data • Create weighted scoring system • Justify final recommendation.

Transferable skills	Ideas for delivery
SP – C&I Creativity and innovation	Present everyday problems. Students must: • Generate multiple solution ideas quickly • Create simple prototypes using basic materials or design software • Test ideas and prototypes with users and gather feedback • Iterate and improve based on results. Design sprint projects. Week-long focused innovation projects. Students must: • Day 1: Understanding problem and ideation • Day 2: Sketching solutions • Day 3: Decision making and storyboarding • Day 4: Prototyping • Day 5: Testing with users and refining

Resources

This section has been created to provide a range of links and resources that are publicly available that you might find helpful in supporting your teaching and delivery of this unit in the qualification. We leave it to you, as a professional educator, to decide if any of these resources are right for you and your students, and how best to use them.

Pearson is not responsible for the content of any external internet sites. It is essential that you preview each website before using it to ensure the URL is still accurate, relevant, and appropriate. We'd also suggest that you bookmark useful websites and consider enabling students to access them through the school/college intranet.

Websites

BIMA (British Interactive Media Association) – UK's largest digital and tech community, offering industry insights and networking opportunities.
https://www.bima.co.uk

Creative Review – British publication covering design, advertising, and visual culture with a UK industry focus.
https://www.creativereview.co.uk

Design Council UK – Government's strategic advisor on design, offering research, case studies, and UK-specific design standards.
https://www.designcouncil.org.uk

Design Week – Leading UK design news, featuring British agencies, trends, and industry developments.
https://www.designweek.co.uk

Digital Arts – UK-focused digital design news, tutorials, and reviews tailored to British creative professionals.
https://www.digitalartsonline.co.uk

Figma Community – Free design resources, UI kits, and collaborative design tools shared by the global design community.
https://www.figma.com/community

Figma Education – An industry design platform offering free accounts for education.
https://www.figma.com/education

GOV.UK Design System – The UK government's acclaimed design system, patterns, and accessibility guidelines that set industry standards.
https://www.gov.uk/service-manual

Interaction Design Foundation – Educational platform offering structured UX/UI courses, comprehensive guides, and literature reviews on design principles.
https://www.interaction-design.org

Miro Education Plan – Free enterprise-level collaboration platform for verified students and educators, offering unlimited team members, boards, and templates for visual learning and teaching.
https://help.miro.com/hc/en-us/articles/360017730473-Education-Plan

Nielsen Norman Group (NNG) – Research-based UX insights, detailed articles, and comprehensive reports from industry pioneers. Known for their scientific approach to user experience.
https://www.nngroup.com

Smashing Magazine – In-depth tutorials, design patterns, case studies, and practical solutions for both beginners and advanced designers.
https://www.smashingmagasine.com

UK UX Professionals' Association – Professional body for UX practitioners in the UK, offering events and resources.
https://www.ukuxpa.org

UX Collective UK – British perspective on UX design with articles and case studies from UK-based designers.
https://www.uxdesign.cc/uk

Textbooks

Park, U., Introduction to Design Thinking for UX Beginners: 5 Steps to Creating a Digital Experience That Engages Users with UX Design, UI Design, and User Research. Start Building Your UX Career, 2023, (9798386967475).

Pereyra, I., Universal Principles of UX: 100 Timeless Strategies to Create Positive Interactions between People and Technology, Rockport Publishers, 2023, (978-0-7603-7804-5).

Press, S.L., UX/UI Designer Notebook (White): UX/UI Design for Mobile, Tablet, and Desktop – Sketchpad – User Interface – Experience App Development – Sketchbook – App MockUps, 2021, (9798482054680).

Staiano, F., Designing and Prototyping Interfaces with Figma – Second Edition: Elevate your design craft with UX/UI principles and create interactive prototypes, Packt Publishing, 2023, (978-1-83546-460-1).

Yablonski, J., Laws of UX: Using Psychology to Design Better Products & Services, O'Reilly, 2020, (978-1-4920-5531-0).

Pearson paid resources also available

- Pearson Student book
- ActiveBook (a digital version of the Student Book, via ActiveLearn Digital Service)
- Digital Teacher Pack (via ActiveLearn Digital Service)

Unit 4: Practical Programming

Unit overview

Unit 4: Practical Programming	
Assessment type: Internal	
Learning Aim	**Topics**
A Explore principles of computing related to software	A1 Input and output A2 Data structures A3 Searching A4 Sorting A5 Good practice in programming
B Manage the development of a software solution	B1 Develop a computer program to solve a problem
Assessment overview This unit is Internal assessed through a Pearson-Set Assignment Brief (PASB). Pearson sets the assignment for the assessment of this unit. The PSAB will take approximately 36 hours to complete. The PSAB will be marked by centres and verified by Pearson. The PSAB will be valid for the lifetime of this qualification	

Common student misconceptions

Below are some common misconceptions related to the content of this unit by students and ideas for how you can help your students to avoid and overcome these.

What is the misconception?	How to help students overcome it
Students often think that the first element in an array is at index 1, rather than index 0.	Explain that in most programming languages (like Python, Java, and C++), array indexing starts at 0, not 1. For example, if an array has 5 elements, the first element is at index 0, the second at index 1, and so on up to index 4.
Students may think that linear search is always the fastest method to find data in a list, without considering the size of the data.	Show that while linear search works well for small datasets, its performance decreases as the size of the dataset increases because it checks each element one by one. Explain that there are more efficient searching methods, such as binary search, for larger datasets.
Students may think that quicksort always works fast, regardless of the data.	Explain that quicksort is generally fast, but its performance can worsen if the pivot is poorly chosen, especially with already sorted or nearly sorted data. Discuss strategies like choosing a random pivot or using a median-of-three approach to improve performance.
Students often think that data validation is a check performed on data to ensure that it is correct, rather than a check to ensure that the data is reasonable, sensible, and within set boundaries.	Model validation rules on data to demonstrate how data can be incorrect while still meeting the validation criteria. For example, a presence check will verify that data has been entered, but it does not ensure that the data is actually correct. A format check can confirm that a date of birth is structured correctly but does not validate its accuracy.

Learning Activities and Resources

This section offers a starting point for delivering the unit by outlining a logical sequence through the unit topics and suggesting practical activities and teacher guidance for covering the main areas of content during guided learning time. Transferable skills are integrated into various activities, with those embedded in a unit indicated by an acronym in square brackets. The acronym combines the letters from the broad skill area and the specific transferable skill, e.g., **[IS-WC]**.

Please note that the activities provided below are suggestions and not mandatory.

Learning Topic	Activities and guidance for unit content delivery	Resources
A1 Input and output	**Whole Class Teaching and Learning – Introduction to Data Validation** • Explain the difference between input and output to students and then introduce the term data. Discuss why data validation is performed at the input stage rather than the output stage. • Display the different data validation methods listed in the specification (presence, range, length, pattern/format, lookup) and discuss their purposes. • Show examples of user input forms, which could be electronic or paper-based. Discuss how data validation has been, or could be, used to ensure data is sensible and reasonable. • Next, show students an example of an electronic form with data validation set up. Model how the use of validation does not ensure data accuracy. For example, demonstrate how a presence check ensures that data is entered but does not verify the correctness of the data entered. **Paired Activity – Data Validation Programming Code**	Notes of data validation, e.g. Geeks For Geeks - geeksforgeeks.org Notes on text file handling, e.g. Geeks For Geeks - geeksforgeeks.org W3Schools - w3schools.com

	• Ask students to write code for different validation rules. Initially, students can use built-in functions within the programming language. Then they could implement their own custom validation logic for the validation rules. • Students could be given specific data, such as a postcode, and determine which validation rules would be appropriate. They can then create the code for the validation rules (e.g. presence checks, type checks, and format checks for fields like postcodes). • Students could then present their validation rules and logic to the rest of the class. **Whole Class Teaching and Learning – Introduction to Text File Handling** • Discuss what happens to all data stored within the program once it ends. Consider scenarios where data needs to be saved beyond the program's execution. For example, a computer game may need to save the player’s progress, such as their score, so it can be retrieved when the game is played again. • Explain that one common method of storing data permanently is by using a text file. Discuss the benefits of using text files, such as simplicity and cross-platform compatibility. • Introduce the term delimiter and provide common examples of characters that can be used, such as commas and whitespace. • Outline the stages involved in working with text files, including opening the file, reading/writing data to/from the file, and then closing the file. **Individual Activity – Using Text File Handling in Python**	

	• Provide students with practical exercises to read from and write to a text file. For example, students could create a program that asks the user for their name and age, then saves this information to a text file. Another exercise could involve reading a list of items from a text file and displaying them to the user, or saving user-entered notes into a file and reading them back when the program runs again.	
A2 Data structures	**Whole Class Teaching and Learning – Introduction to Arrays** • Show students some code containing various variables, each storing an item. For example, this could be a list of items that a user can collect during a game. Ask students to identify which variables could be grouped together to simplify the code and make it more manageable. • Introduce students to the concept of a data structure and how data within it can be accessed and manipulated using indexes. Then demonstrate what the list of items would look like when stored in an array. • Discuss the key features of arrays, such as their scope and data type. **Paired Activity – One-Dimensional and Two-Dimensional Arrays** • Demonstrate how to create a one-dimensional array, then show how to create a two-dimensional array and discuss the similarities and differences between them. Discuss the different uses of arrays and the types of problems they can solve. • Give students practical exercises to practise implementing one-dimensional and two-dimensional arrays, including adding, updating, and removing data using relevant commands or by	Notes on arrays, e.g. W3Schools - w3schools.com Geeks For Geeks - geeksforgeeks.org

	referencing array index numbers. For example, students could create a program that manages a simple to-do list. They should start with an empty list and implement functions to: o Append tasks o Insert tasks at specific positions (e.g., urgent tasks at the beginning) o Update a task if its description changes o Delete a completed task	
A3 Searching	**Whole Class and Individual Activity – Introduction to Search Algorithms** • Discuss practical examples of when a program may need to search for data. For example, in a customer service system, a program may need to search through tickets or support requests to locate particular customer queries or issues, enabling a support team to respond quickly and accurately. **Whole Class and Individual Activity – Linear Search** • Explain to students that one search method is the linear search and discuss how a linear search works. • Provide students with a list of unsorted numbers (e.g., 12, 8, 3, 7, 4, 9, 15, 6, 11, 10) and demonstrate the steps a linear search would take to find the value 9. Then model the use of a linear search on lists containing strings. • Discuss potential benefits and drawbacks of the linear search, particularly its efficiency on long lists of data. • Discuss what would happen if a list contained 10,000 items and the search item was found after the third iteration. Explain how an inefficient linear search might continue unnecessarily, in	Notes on linear and binary searches, e.g. Geeks For Geeks - geeksforgeeks.org BBC Bitesize – bbc.co.uk

	contrast to an efficient linear search that would exit early once the item is found. **Whole Class and Individual Activity – Binary Search** - Explain to students that another search method is the binary search and discuss how a binary search works on sorted arrays. - Demonstrate how the binary search works with both integers and strings. Model what happens if a binary search is attempted on a list of unsorted values. - Discuss the potential benefits and drawbacks of the binary search, especially its efficiency on long lists of data when compared to the linear search. **Paired Activity – Implementing Search Algorithms** - Provide students with questions containing lists of data and ask them to trace both the linear and binary search methods to find given items. - Ask students to implement the programming code for both the linear search and binary search in a programming language. If students find this challenging, they could be provided with written descriptions of the steps or a flowchart to help them write the code. Encourage students to add comments to their code to demonstrate their understanding of each line. - Ask students to share their solutions with others in the class and discuss the different approaches they have taken.	
A4 Sorting	**Whole Class and Individual Activity – Introduction to Sorting Algorithms**	Notes on bubble and quick sorts, e.g. Medium - medium.com

	• Discuss practical examples of when a program may need to sort data in a specific order. For example, a computer game may need to sort users in ascending order based on their scores, or an e-commerce website may need to sort products by price. **Whole Class and Individual Activity – Bubble Sort** • Explain to students that one sorting method is the bubble sort and discuss how a bubble sort works. • Provide students with a list of unsorted numbers (e.g., 10, 5, 3, 1, 11, 2) and demonstrate the steps a bubble sort would take to arrange the values in ascending order. Then demonstrate how to sort the items in descending order. • Model how to use the bubble sort to sort a list of string values in both ascending and descending order. • Discuss the potential benefits and drawbacks of the bubble sort, particularly its inefficiency on long lists of unsorted data. **Whole Class and Individual Activity – Quick Sort** • Explain that another sorting method is the quick sort and discuss how a recursive quick sort works and how to choose a suitable pivot. Provide students with the same list of unsorted numbers (e.g., 10, 5, 3, 1, 11, 2) and demonstrate the steps a quick sort would take to arrange the values in ascending order. Then demonstrate how to sort the items in descending order. • Next demonstrate the impact of a poorly chosen pivot and discuss how this impacts the performance of the sort. • Discuss the differences between the bubble sort and the quick sort, particularly in terms of efficiency on long lists of unsorted	Geeks For Geeks (bubble sort) - geeksforgeeks.org Geeks For Geeks (quick sort) - geeksforgeeks.org

	data. Explain why the quick sort is generally preferred over the bubble sort. **Paired Activity – Implementing the Sorting Algorithms** • Ask students to implement the programming code for both the bubble sort and the recursive quick sort within a programming language. If students find this challenging, they could be given written descriptions of the steps or a flowchart to help them write the code. Encourage students to add comments to their code to demonstrate their understanding of each line. • Ask students to share their solutions with others in the class and discuss the different approaches they have taken.	
A5 Good practice in programming	**Whole Class and Individual Activity – Good Programming Aids** • Discuss practical examples of when program code may need to be modified or maintained, such as complying with new legislation, fixing security vulnerabilities, or meeting new user needs. • Next, discuss aids that programmers can use to ensure another programmer will be able to maintain the program in the future (e.g., comments, layout, whitespace, and indentation). Explain what these features are and how they help improve code quality. • Show students some code that does not follow these good practices and discuss the implications of maintaining such code. • Allow students to apply the techniques discussed to improve code quality and maintainability. **Whole Class and Individual Activity – Separation of Concerns**	Notes on code maintainability, e.g. Coveros - coveros.com

- Show students an example of a program containing repeated code, such as a simple program that calculates the area of different shapes (e.g., rectangles, circles, triangles) where the logic for calculating the area is repeated for each shape. Discuss the issues this can cause and possible solutions.
- Discuss the concept of subprograms, including the difference between functions and procedures, and cover key concepts such as defining subprograms, calling subprograms, parameters, parameter passing, and return values.
- Model how to implement the code using a subprogram such as a function and show how to call the function with different parameters for different shapes to demonstrate reuse.

Paired Activity – Applying Separation of Concerns

- Give students exercises that allow them to apply the principle of separation of concerns and implement the programming code using subprograms. Encourage them to use the programming aids covered earlier. For example, students could create a basic calculator that performs addition, subtraction, multiplication, and division, where each mathematical operation is in a separate subprogram, and the main program handles input/output. Alternatively, students could create a simple to-do list program, separating the logic for adding, removing, and displaying tasks into different subprograms. The main program could focus on taking user input and calling the subprograms.

Whole Class and Individual Activity – Handling Errors

- Explain to students that testing can help correct errors before a program is used by end-users. However, some errors may still

	occur during program execution and cannot always be anticipated, even after testing is completed. Discuss examples of these, such as user input errors, file handling errors, division by zero, and out-of-memory errors. • Explain how exception handling can be used to catch and manage these errors to prevent the program from crashing. Introduce the common stages of exception handling: try, except, else, and finally, and ask students to discuss their purposes. • Demonstrate how to implement exception handling in code. For instance, show how a program can catch and handle errors like invalid input or division by zero to avoid crashes and provide useful feedback to the user. • Give students practical activities that allow them to implement exception handling, which they could apply to the program they created for the separation of concerns activity.	
B1 Develop a computer program to solve a problem	**Whole Class and Individual Activity – Software Development Management** • Introduce a program concept, such as a budgeting app that helps users manage their money and discuss the stages involved in its development. • Break down these stages to include feature specification, writing user stories, implementation, creating unit tests, and running unit tests. Explain each stage, what it covers and why these steps need to be completed in this order. **Individual Activity – Abstraction and Decomposition** • Explain the terms abstraction and decomposition and their importance in simplifying complex programs.	Code repository software, e.g. GitHub – github.com Code repository tutorials, e.g. GitHub Docs– docs.github.com

- Provide students with a scenario, such as developing an online shopping platform where customers can browse products, add items to a cart and make payments. Ask students to apply abstraction by identifying the key elements needed for the program, excluding unnecessary details. Then, students should apply decomposition by breaking the shopping platform into smaller, manageable components.

Paired Activity – Code Repositories

- Introduce the concept of a code repository, its purpose and its main features, including version control, collaboration, commit history, remote access, and code review. Ask students if they have used one before.
- Provide students with a program they can complete in pairs, however, independently, like a basic To-Do List app. One student could develop the functionality to add tasks, while the other focuses on displaying tasks in the list. Each student should create a branch for their respective tasks and then merge their work into the main project.

Project-Based Learning – Document Functional Milestones

- Give students an outline of a program to develop, including a couple of features and user stories. For example, a book inventory management program might have the following user stories:
    - As a library worker, I want to add, remove and update book details to keep the library's collection current.
    - As a library user, I want to check out and return books so I can borrow and return them conveniently.

	- Ask students to work in pairs to create the program. As they work, they should: o Follow the software development management cycle o Use abstraction and decomposition to separate concerns o Document functional milestones as they progress o Use a code repository to track code and documents for each functional milestone o Ensure their code is readable and maintainable by minimising global variables and using linting tools	

Delivering signposted transferable skills

Signposted transferable skills are not mandatory for the delivery of the unit, and it is therefore your decision to deliver these skills as a part of the qualification. Below we have provided some ideas of teaching and learning activities that you could use to deliver these skills if you chose to.

Transferable skills	Ideas for delivery
MY – TPR Taking personal responsibility	There are lots of opportunities for students to demonstrate that they have taken personal responsibility. For example: Students can set personal milestones for progress, monitoring their own work to stay on track. Students can create a clear plan for their software solution, breaking down the project into manageable tasks with deadlines. Students can write good quality, readable, and well-documented code, ensuring that each area is easily understood by others (or by themselves in the future). Students can regularly test their code to identify and fix bugs early, preventing issues that may occur during a later stage of their project. Students can seek feedback from peers to improve the design, functionality, and performance of their solution. Students can prioritise the most important features and ensure they are working before moving on to additional features or enhancements. Students can actively use version control (e.g., Git) to manage changes, ensuring their code is always backed up and easy to revert to previous versions if necessary. Students can research and implement best practices for coding, security, and performance to ensure their software solution is reliable and efficient.

Resources

This section has been created to provide a range of links and resources that are publicly available that you might find helpful in supporting your teaching and delivery of this unit in the qualification. We leave it to you, as a professional educator, to decide if any of these resources are right for you and your students, and how best to use them.

Pearson is not responsible for the content of any external internet sites. It is essential that you preview each website before using it to ensure the URL is still accurate, relevant, and appropriate. We'd also suggest that you bookmark useful websites and consider enabling students to access them through the school/college intranet.

Websites

PythonByteSize – A range of Python demonstration tutorial videos.
https://www.pythonbytesize.com/detailed-videos.html

The Python Tutorial – An overview of the basic concepts and features of the Python language.
https://docs.python.org/3/tutorial/index.html

W3Schools – A Python code editor and a range of Python tutorials and exercises.
https://www.w3schools.com/python/default.asp

Textbooks

Matthes, E., Python Crash Course, 3rd Edition: A Hands-On, Project-Based Introduction to Programming, 2023 (ISBN 1718502702)

Shovic, J., Simpson, A., Python All-in-One For Dummies (For Dummies: Learning Made Easy), 2024 (ISBN 1394236158)

Pearson paid resources also available

- Pearson Student book
- ActiveBook (a digital version of the Student Book, via ActiveLearn Digital Service)
- Digital Teacher Pack (via ActiveLearn Digital Service)

5. Pearson Qualification Support and Resources

This section provides information on support and resources that are available on the Pearson website for this qualification.

Exam Wizard

A free online resource containing a bank of past paper questions and support materials to help you create your own mock exams and tests.

Pearson Set Assignments (PSABs)

These internal assessments are set by Pearson and marked internally by the centre. They should be used for all internal assessments on the course. There are specific PSABs for each internally assessed unit on the course.

Results plus

a free online results analysis tool for teachers that gives you a detailed breakdown of your students' performance in BTEC external assessments.

Specification

This document contains an overview of the qualification, qualification purpose and structure, units including content and assessment, planning and implementing the qualification, qualification grade, glossary of terms used for internally assessed units, Transferable skills framework, digital skills framework, sustainability framework.

Sample Assessment Material (SAMs)

These resources illustrate the format and style of questions for the external assessment for this qualification. A mark scheme is also provided which shows how credit is awarded for these questions. The resources can be used to help prepare students for their external assessment.

Training

Getting Started and Preparing to Assess training events and recorded sessions will be available from July 2024 onwards.

Transferable Skills Guide for Teachers

This guide provides and overview of the BTEC Transferable Skills Framework and how it has been used to integrate the delivery of these skills in the new suite of BTEC Level 3 and Level 2 qualifications starting in 2025.

Transition Guides

This guide provides an overview of what's new in the qualification, a comparison of the previous qualification to this new qualification, an overview of the assessment approach, a mapping guide to show where content is the same, updated or new

Statement of Purpose

This provides an overview of the qualification's key details. It outlines what students will study, the knowledge and skills they will develop, and any related subjects that complement the qualification. It also highlights potential progression routes for further learning and lists the Higher Education Institutes that have formally expressed their support and recognition for the qualification.

Subject Adviser

A dedicated subject adviser available throughout the year so please do get in touch if you would like any support or guidance with:

- Planning your courses
- Overview of BTEC quality assurance processes
- Suggested resources
- Teaching and Assessment of internal units and components
- Teaching external units and components
- The training and support materials we have available.

Annexe

Curriculum Planning

The models in this section are intended to support your delivery planning and provide suggestions for the types and subjects of qualifications that might be delivered with this qualification.

Suggested combinations with other qualifications

This qualification can be combined in the following ways depending on the destination of students.

For students intending to progress to higher education to study Computer Science

Option 1	Option 2	Option 3
AAQ BTEC in Computing	A Level Mathematics	A Level Physics

For students intending to progress to higher education to study Artificial Intelligence.

Option 1	Option 2	Option 3
AAQ BTEC in Computing	A Level Design and Technology	A Level Business

BTEC Key Terms

GLH – Guided Learning Hours, time the students have supervised teaching and learning

IV – Internal Verification, for internal quality assurance

Lead IV – the person responsible for the internal quality assurance across a qualification or programme subject area.

PSAB – Pearson Set Assignment Brief, used for summative internal assessments

SV – Standards Verification, for external quality assurance

Transferable Skills

Managing Yourself

Acronym	
MY-TPR	Taking Personal Responsibility
MY-PS&R	Personal Strengths and Resilience
MY-COP	Career Orientation Planning
MY-PGS	Personal Goal Setting

Effective Learning

Acronym	
EL-MOL	Managing Own Learning
EL-CL	Continuous Learning
EL-SRS	Secondary Research Skills
EL-PRS	Primary Research Skills

Interpersonal Skills

Acronym	
IS-WC	Written Communications
IS-V&NC	Verbal and Non-verbal Communications
IS-T	Teamwork
IS-C&SI	Cultural and Social Intelligence

Solving Problems

Acronym	
SP-CT	Critical Thinking
SP-PS	Problem Solving
SP-C&I	Creativity and Innovation

January 2025

For information about Pearson Qualifications, including Pearson Edexcel
and BTEC qualifications visit qualifications.pearson.com

Edexcel and BTEC are registered trademarks of Pearson Education Limited

Pearson Education Limited. Registered in England and Wales No. 872828
Registered Office: 80 Strand, London WC2R 0RL.

VAT Reg No GB 278 537121