Pearson Level 3

Alternative Academic Qualification BTEC National in

Computing (Certificate)

L3

Specification

First teaching from September 2026

First certification from 2027

Issue 2

Qualification Number: 610/6194/3

Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Certificate)

Specification

First teaching September 2026
First certification from 2027
Issue 2

About Pearson

This specification is Issue 2. Key changes are summarised on the next page. We will inform centres of any changes to this issue. The latest issue can be found on our website.

VQ000487

Welcome

BTEC Nationals are widely recognised by higher education and industry as the vocational qualification of choice at Level 3. They provide students with meaningful and practical learning experiences across a range of career sectors to prepare them to progress to higher education as a route to graduate-level employment.

Recent data has shown that one in five adults of working age in the UK has a BTEC qualification. What's more, well over 90,000 BTEC students apply to UK universities every year and their BTEC Nationals are accepted by over 150 UK universities and other higher education institutions for relevant degree programmes either on their own or in combination with A Levels.

Why are BTECs so successful?

BTECs embody a fundamentally student-centred approach to the curriculum, with a flexible, unit-based structure and knowledge applied through a balanced combination of assignments and examinations. They enable the holistic development of the practical, interpersonal and thinking skills required to succeed in higher education and employment.

When creating these BTEC Nationals we focused on the skills and personal attributes needed to navigate the future, and have worked with many higher education providers, professional bodies, colleges and schools to ensure that their needs are met. 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.

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. This Transferable Skills framework has been used to embed transferable skills in the qualifications where they naturally occur and also to signpost opportunities for delivery and development as a part of the wider BTEC learning experience. See page 6 for further information.
- Supporting the delivery of Sustainability Education and Digital Skills development naturally through the content design of the qualifications. Mapping is provided for each qualification to identify where the opportunities for teaching and learning exist.
- Updating sector-specific content to ensure it is 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 ensure that students and their teachers have the best possible experience during their course. See Section 5 for details of the support we offer.

This specification document should be used in conjunction *with the Pearson Level 3 Alternative Academic Qualification BTEC National Specification Supplementary Information* document which is available on our website.

A word to students

Today's BTEC Nationals will require commitment and hard work, as you would expect of the most respected applied learning qualification in the UK. You will have to complete a range of units, be organised, take some assessments that we will set and mark and undertake practical tasks and assignments. But you can feel proud to achieve a BTEC because, whatever your plans in life – whether you decide to study further, go on to work or an apprenticeship – your BTEC National will be your passport to success in the next stage of your life.

Good luck, and we hope you enjoy your course.

Summary of changes to Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Certificate) specification Issue 2

Summary of changes made between previous issue and this issue	**Page number**
Grading information updated to remove requirement for students to achieve a Near Pass (N) or above in external units to achieve the qualification	64

If you need further information on these changes or what they mean, please contact us via our website at: qualifications.pearson.com/en/support/contact-us.html.

Contents

1 Introduction

Why choose Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Certificate)?

We've listened to feedback from all parts of the digital subject community, including higher education. We've used this opportunity of curriculum change to redesign qualifications so that they reflect the demands of a truly modern and evolving computing environment – qualifications that enable your students to apply themselves and give them the skills to succeed in their chosen pathway.

The Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Certificate) gives students opportunities to develop the core knowledge, skills and understanding that underpin computer programming and to develop computational thinking and programming skills that will enable them to solve problems. Students will explore the critical components that safeguard information systems and data and will acquire knowledge and skills to comprehend and apply security and encryption principles and practices to computer networks in various situations. Students will also have the opportunity to examine the underlying principles of human-computer interaction (HCI) and develop a HCI solution to meet the requirements of a given brief. Students will draw synoptically on their programming learning and computational thinking skills to manage the development of a software solution to a problem.

The qualification consists of 1 mandatory unit and 2 optional units. The mandatory unit is externally assessed by examination, and the optional units are internally assessed by a Pearson Set Assignment Brief (PSAB). Unit 3 and 4 give students opportunities to put their knowledge, understanding and skills into practice in HCI design and software development.

The qualification is designed to be taken alongside A Levels as part of a study programme and can link to learning in A Levels in Mathematics, Physics and Business. It is intended for students that wish to progress into higher education as a pathway to employment.

Total Qualification Time

For all regulated qualifications, Pearson specifies a total number of hours that it is estimated students will require to complete and show achievement for the qualification: this is the Total Qualification Time (TQT). Within TQT, Pearson identifies the number of Guided Learning Hours (GLH) that we estimate a centre delivering the qualification might provide. Guided learning means activities, such as lessons, tutorials, online instruction, supervised study and giving feedback on performance, that directly involve teachers and assessors in teaching, supervising and invigilating students. Guided learning includes the time required for students to complete external assessment under examination or supervised conditions.

In addition to guided learning, other required learning directed by teachers or assessors will include private study, preparation for assessment and undertaking assessment when not under supervision, such as preparatory reading, revision and independent research.

BTEC Nationals have been designed around the number of hours of guided learning expected. Each unit in the qualification has a GLH value of 60, 90 or 120. There is then a total GLH value for the qualification.

Each qualification has a TQT value. This may vary within sectors and across the suite depending on the nature of the units in each qualification and the expected time for other required learning.

The following tables show the qualifications in this sector and their GLH and TQT values.

Qualification title	Size and structure	Summary purpose
Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Certificate)	180 GLH (227 TQT) Equivalent in size to half an A Level. 2 Units of which 1 is mandatory and externally assessed. External assessment – 66%	The Certificate is for students looking to develop their applied knowledge and skills in computing as part of a study programme alongside three A Levels. This qualification provides students with the opportunity to develop an understanding of computing and software development which will facilitate progression into multiple disciplines in HE that have digital aspects in their courses.
Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Extended Certificate)	360 GLH (462 TQT) Equivalent in size to one A Level. 4 mandatory units of which two are externally assessed. External assessment – 66%	The Extended Certificate is for students who are interested in learning about the computing sector alongside other fields of study, with a view to progressing to a wide range of higher education courses, not necessarily in computing-related subjects. It is designed to be taken as part of a programme of study that includes A Levels.

Structures of the qualifications at a glance

This table shows all the units and the qualifications to which they contribute. The full structure for this Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Certificate) is shown in *Section 3 Structure*. **You must refer to the full structure to select units and plan your programme.**

Key

Externally assessed units are shown in bold	M	**Mandatory units**	O	**Optional units**

Unit (number and title)	Unit size (GLH)	Certificate (180 GLH)	Extended Certificate (360 GLH)
1 Programming Fundamentals	**120**	**M**	**M**
2 Computer Network Security and Encryption	**120**		**M**
3 Human-Computer Interaction	**60**	**O**	**M**
4 Practical Programming	**60**	**O**	**M**

Qualification and unit content

Pearson has developed the content of the new BTEC Nationals in collaboration with representatives from higher education and relevant professional bodies. In this way, we have ensured that content is up to date and that it includes the knowledge, understanding, skills and attributes required in the sector.

Centres should ensure that delivery of content is kept up to date. Some of the units within the specification may contain references to legislation, policies, regulations and organisations, which may not be applicable in the country you deliver this qualification in (if teaching outside of England), or which may have gone out-of-date during the lifespan of the specification. In these instances, it is possible to substitute such references with ones that are current and applicable in the country you deliver subject to confirmation by your Standards Verifier.

Assessment

Assessment is specifically designed to fit the purpose and objective of the qualification. It includes a range of assessment types and styles suited to vocational qualifications in the sector. There are three main forms of assessment that you need to be aware of: external, internal and synoptic.

Externally-assessed units

Each external assessment for a BTEC National is linked to a specific unit. All of the units developed for external assessment are of 60, 90 or 120 GLH to allow students to demonstrate breadth and depth of achievement. Each assessment is taken under specified conditions, then marked by Pearson and a grade awarded. Students are permitted to resit the examination twice. This equates to three attempts in total: one inclusive of registration, the remaining two attempts as resits. If students resit an examined unit, the best grade achieved will count towards their overall qualification grade, not necessarily the most recent sitting. External assessments are available twice a year. For detailed information on the external assessments, please see the table in *Section 3*. For further information on preparing for external assessment, see the *Pearson Level 3 Alternative Academic Qualification BTEC National Specification Supplementary Information* document, which is available on our website.

Internally-assessed units

Internally assessed units are assessed by a Pearson Set Assignment Brief (PSAB), which is set by Pearson, marked by you and subject to external standards verification. Before you assess you will need to become an approved centre, if you are not one already. You will need to prepare to assess using the guidance in the *Pearson Level 3 Alternative Academic Qualification BTEC National Specification Supplementary Information* document, which is available on our website.You will make grading decisions based on the requirements and supporting guidance given in the units. Where a student has not achieved their expected level of performance for an assignment, they may be eligible for one resubmission of improved evidence for each assignment submitted if authorised by the Lead Internal Verifier.

To ensure any resubmissions are fairly and consistently implemented for all students, the Lead Internal Verifier can only authorise a resubmission if certain conditions are met. If the Lead Internal Verifier does authorise a resubmission, it must be completed within 15 working days of the student receiving the results of the assessment.

Feedback to students can only be given to clarify areas where they have not achieved expected levels of performance. Students cannot receive any specific guidance or instruction about how to improve work to meet assessment criteria or be given solutions to questions or problems in the tasks.

If a student has still not achieved the targeted pass criteria following the resubmission of improved evidence for an assignment, the Lead Internal Verifier may authorise, under exceptional circumstances, one retake opportunity to meet the required pass criteria. The retake can be of a task or subset of the Pearson Set Assignment Brief that is of evidence in a new or revised form. The deadline for submission of the retake must fall within the same academic year.

Synoptic assessment

Synoptic assessment requires students to demonstrate that they can identify and use effectively, in an integrated way, an appropriate selection of skills, techniques, concepts, theories and knowledge from across the whole sector as relevant to a key task. Synoptic links between units are flagged within the units. Please refer to *Unit 4: Practical Programming* for further details.

Language of assessment

Assessment of the internal and external units for these qualifications will be available in English. All student work must be in English. A student taking the qualifications may be assessed in British or Irish Sign Language where it is permitted for the purpose of reasonable adjustment.

For information on reasonable adjustments see the *Pearson Level 3 Alternative Academic Qualification BTEC National Specification Supplementary Information* document, which is available on our website.

Grading for units and qualifications

Achievement in the qualification requires a demonstration of depth of study in each unit, assured acquisition of a range of practical skills required for progression to higher education, and successful development of transferable skills. Students achieving a qualification will have completed all units.

Units are assessed using a grading scale of Distinction (D), Merit (M), Pass (P), Near Pass (N) and Unclassified (U). The grade of Near Pass is used for externally-assessed units only. All mandatory and optional units contribute proportionately to the overall qualification grade, for example a unit of 120 GLH will contribute double that of a 60 GLH unit.

BTEC National qualifications are graded using a scale of P to D*, **or** PP to D*D*, **or** PPP to D*D*D* depending on the size of the qualification. Please see *Section 6* for more details. The relationship between qualification grading scales and unit grades will be subject to regular review as part of Pearson's standards monitoring processes on the basis of student performance and in consultation with key users of the qualification.

UCAS tariff points

The BTEC Nationals attract UCAS points. Please go to the UCAS website for full details of the points allocated.

Preparing students for the future

Transferable skills

Recent future skills reports have highlighted the growing importance of transferable skills for students to succeed in their careers and lives in this fast-changing world.

Following research and consultation with FE educators and higher education institutions, Pearson has developed a Transferable Skills Framework to facilitate and guide the development of transferable skills through this qualification. The Framework has four broad skill areas, each with a cluster of transferable skills as shown below:

1. **Managing Yourself**: (1) Taking personal responsibility; (2) Personal strengths and resilience; (3) Career orientation planning; (4) Personal goal setting
2. **Effective Learning**: (1) Managing own learning; (2) Continuous learning; (3) Secondary research skills (4) Primary research skills
3. **Interpersonal Skills**: (1) Written communications; (2) Verbal and non-verbal communications; (3) Teamwork; (4) Cultural and social intelligence
4. **Solving Problems**: (1) Critical thinking (2) Problem solving; (3) Creativity and innovation

Each transferable skill has a set of descriptors that outline what achievement of the skill looks like in practice. Each unit in the qualification will show whether a transferable skill has been:

1. fully embedded through the design of the teaching and learning content and assessment of the unit. Skills that are embedded are 'naturally occurring' in that they are inherent to the unit content and don't require extension activities to deliver.
2. signposted as an opportunity for delivery and development and would require extension activities to deliver.

Units will show a summary of the transferable skills that have been embedded or signposted and *Appendix 2* shows the descriptors for each skill across all the skill clusters.

More information on the framework, its design and relevance for student progression is available in the *BTEC Transferable Skills Guide for Teachers*. Resources and guidance to support teachers in the delivery and development of these skills will be available in the Planning and Teaching Guide for this qualification and through our training offer.

Digital skills

Digital skills are required in every industry as well as in everyday life and with the acceleration of automation and AI in industry it is critical for students to understand how digital technologies are relevant and applied in the context of the sector they are studying.

With this in mind, we have used the Digital Skills Framework published by IFATE as a frame of reference to identify opportunities for the delivery and development of digital skills in this qualification.

This Digital Skills framework has five categories with specific digital characteristics that apply in varying extent across sectors:

1. **Problem Solving** – The use of digital tools to analyse and solve problems
2. **Digital Collaboration and Communication** – Using digital tools to communicate and share information with stakeholders
3. **Transacting Digitally** – Using digital tools to set up accounts and pay for goods/services
4. **Digital Security** – Identify threats and keep digital tools safe
5. **Handling Data Safely and Securely** – Follow correct procedures when handling personal and organisational data

Opportunities to develop these digital skills are identified where they are relevant and appropriate to a sector, meaning:

- where they naturally occur
- where they add no assessment burden
- they will enhance a student's skills and knowledge in the sector.

Appendix 3 shows a mapping of the teaching and learning content to the five categories of the framework to show where opportunities to develop these digital skills exist in this qualification.

Sustainability skills

To help students develop sustainability skills, practices and mindset, we have designed content in this qualification, aligned to the UNESCO Sustainable Development Goals (17 SDGs), that are relevant and appropriate to the sector. The SDGs are the most common point of reference for content that addresses sustainability education and provides a useful and pragmatic way of presenting this content.

Sustainability knowledge and understanding may be included in the teaching and learning content but not directly assessed. Alternatively, it could be assessed – the approach chosen for each unit is based on the relevance of the sustainability skills, knowledge and understanding to the purpose and scope of the unit.

Appendix 4 shows a mapping of the teaching and learning content to the relevant SDGs to show where sustainability concepts have been included in this qualification.

2 Qualification purpose

Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Certificate)

In this section you will find information on the purpose of this qualification and how its design meets that purpose through the qualification objective and structure. We publish a full 'Statement of Purpose' for each qualification on our website. These statements are designed to guide you and potential students to make the most appropriate choice of qualification at recruitment.

Who is this qualification for?

The Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Certificate) is an Alternative Academic Qualification (AAQ) designed for post-16 students with an interest in the Digital sector.

Equivalent to half an A level in size, it is suitable for students looking to develop their applied knowledge and skills in computing as part of a study programme alongside 3 A levels. This qualification provides students with the opportunity to develop an understanding of Computing and Software Development which will facilitate progression into multiple disciplines in HE that have digital aspects in their courses.

What will the student study as part of this qualification?

The qualification has been developed in consultation with higher education representatives and sector experts to ensure students have the knowledge, understanding and skills needed to progress to, and thrive in, higher education.

The qualification has one mandatory unit and a choice from two optional units covering the following topic areas:

- **Programming Fundamentals** – Computing concepts and their application through programming and design methodologies
- **Human-Computer Interaction** – User experience (UX) and user interface (UI) design principles and their application in creating interfaces; key principles of HCI design, including meeting diverse needs of users
- **Practical Programming** – Principles of computer science related to software development and their application in developing and managing a software solution.

What knowledge and skills will the student develop as part of this qualification and how might these be of use and value in further studies?

Students will develop the following knowledge and skills:

- Applied knowledge and understanding of programming design and methodologies and principles of software development in relation to solving specific problems
- Technical skills to:
  - Create a design proposal for an accessible HCI solution to meet the requirements of a brief and develop and evaluate the solution
  - Manage the development of a software solution to meet the requirements of a brief.

Students will develop critical thinking skills which will be beneficial when encountering analytical tasks in higher education. Taking personal responsibility for their learning on this qualification will help students to manage their work and balance the demands of many degree programmes which require independent study.

Which subjects will complement this qualification?

The following subjects would be suitable to combine with this qualification:

- Mathematics
- Physics.
- Design and Technology
- Business.

What further learning will this qualification lead to?

This qualification can lead to progression to the following degrees:

- BSc Computer Science
- BSc Electrical Engineering
- BA Artificial Intelligence.

This qualification is part of a larger suite. This size provides students with the opportunity to develop an understanding of the digital sector to progress into the sector through degree or degree apprenticeship pathways. It also allows for progression onto degree programmes with significant Computing content, which is the case in many sectors.

The Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Extended Certificate), equivalent to one A Level, is for those students who require additional Computing knowledge and skills to progress into Higher Education to study degrees that have programming or computing elements within them.

3 Structure

Qualification structure

Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Certificate)

Students must complete 1 mandatory unit and 1 optional unit.

See *Section 6* for rules on qualification awarding.

Mandatory units – students complete all units

Unit number	Unit title	GLH	Type	How assessed
1	Programming Fundamentals	**120**	**Mandatory**	External

Optional units – students complete 1 unit

Unit number	Unit title	GLH	Type	How assessed
3	Human-Computer Interaction	**60**	**Optional**	Internal
4	Practical Programming	**60**	**Optional**	Internal

External assessment

66% of the total qualification GLH is made up of external assessment. A summary is given below. See the unit content and sample assessment materials for more information.

Unit	Type	Availability
Unit 1: Programming Fundamentals	• **An external examination set and marked by Pearson** • **90 marks**	**January and May/June** **First assessment January 2027**

Synoptic assessment

The assessment of synoptic knowledge requires students to apply learning from one unit to the assessment in another unit. Within the assessments for *Unit 3: Human-Computer Interaction* and *Unit 4: Practical Programming*, students will be assessed on underpinning knowledge, ideas and concepts from *Unit 1: Programming Fundamentals*. Synoptic links are flagged within the units.

There might be some further naturally occurring synoptic opportunities across the qualification where students can synthesise their learning. These will be outlined in the Delivery Guide to help with planning for your teaching.

4 Units

Understanding your units

The units in this specification set out our expectations of assessment in a way that helps you to prepare your students for assessment. The units help you to undertake assessment and quality assurance effectively.

Each unit in the specification is set out in a similar way. There are two types of unit format:

- Internally assessed units
- Externally assessed units.

This section explains how the units work. It is important that all teachers, assessors, internal verifiers and other staff responsible for the programme review this section.

Internally assessed units

Section	Explanation
Unit number	The number is in a sequence in the sector. Numbers may not be sequential for an individual qualification.
Unit title	This is the formal title that we always use and it appears on certificates.
Unit level	All units are Level 3 on the national framework.
Unit type	This confirms that the unit is internally assessed. See structure information in *Section 3* for full details.
GLH	Units may have a Guided Learning Hours (GLH) value of 120, 90 or 60. This indicates the numbers of hours of teaching, directed activity and assessment expected. It also shows the weighting of the unit in the final qualification grade.
Unit in brief	A brief formal statement on the content of the unit that is helpful in understanding its role in the qualification. You can use this in summary documents, brochures etc.
Unit introduction	This is designed with students in mind. It indicates why the unit is important, how learning is structured and how learning might be applied when progressing to employment or higher education.
Learning aims	These help to define the scope, style and depth of learning of the unit. You can see where students should be learning standard requirements ('understand') or where they should be actively researching ('investigate'). You can find out more about the verbs we use in learning aims in *Appendix 1*.

Section	Explanation
Summary of unit	This helps teachers to see the main content areas against the learning aims and the structure of the assessment at a glance.
Content	This sets out the required teaching content of the unit. Content is compulsory except where shown as 'e.g.'. Students should be asked to complete summative assessment only after the teaching content for the unit or learning aim(s) has been covered.
Assessment criteria	Each learning aim has Pass and Merit criteria. Each assignment has at least one Distinction criterion. A full glossary of terms used is given in *Appendix 1*. Distinction criteria represent outstanding performance in the unit. Some criteria require students to draw together learning from across the learning aims.
Transferable skills	This summarises the transferable skills present within this unit. The key helps to identify whether they are signposted but require additional assessment, embedded and achieved on completion or not present in this unit.
Essential information for Pearson Set Assignment Brief (PSAB)	This shows a brief summary of the activities required for the mandatory Pearson Set Assignment Brief. Centres must download and use the mandatory PSAB without alteration or contextualisation.
Further information for teachers and assessors	This gives you information to support the implementation of assessment. It is important that this is used carefully alongside the assessment criteria and PSAB.
Resource requirements	Any specific resource requirements that you need to be able to teach and assess are listed in this section.
Essential information for assessment decisions	This information gives guidance for each learning aim or assignment of the expectations for Pass, Merit and Distinction standard. This section contains examples and essential clarification.
Links to other units	This shows you the main relationship between units. This can help you to structure your programme and make best use of materials and resources.

Externally assessed units

Section	Explanation
Unit number	The number is in a sequence in the sector. Numbers may not be sequential for an individual qualification.
Unit title	This is the formal title that we always use and it appears on certificates.
Unit level	All units are Level 3 on the national framework.
Unit type	This confirms that the unit is externally assessed. See structure information in *Section 3* for full details.
GLH	Units may have a Guided Learning Hours (GLH) value of 120, 90 or 60. This indicates the numbers of hours of teaching, directed activity and assessment expected. It also shows the weighting of the unit in the final qualification grade.
Unit in brief	A brief formal statement on the content of the unit that is helpful ~in understanding its role in the qualification. You can use this in summary documents, brochures etc.
Unit introduction	This is designed with students in mind. It indicates why the unit is important, how learning is structured and how learning might be applied when progressing to employment or higher education.
Summary of assessment	This sets out the type of external assessment used and the way in which it is used to assess achievement.
Assessment outcomes	These show the hierarchy of knowledge, understanding, skills and behaviours that are assessed. Includes information on how this hierarchy relates to command terms in sample assessment materials (SAMs).
Content	For external units all content is obligatory. The depth of content is indicated in the assessment outcomes and sample assessment materials (SAMs). The content will be sampled through the external assessment over time, using the variety of questions shown.
Transferable skills	This summarises the transferable skills present within this unit. The key helps to identify whether they are signposted but require additional assessment, embedded and achieved on completion or not present in this unit.
Key terms typically used in assessment	These definitions will help you analyse requirements and prepare students for assessment.
Resources	Any specific resource requirements that you need to be able to teach and assess are listed in this section.

Index of units

Unit 1: Programming Fundamentals

Level: 3

Unit type: External

Guided learning hours: 120

Unit in brief

Students will explore the core knowledge, skills and understanding that underpin computer programming. The unit will support the development of computational thinking and programming skills, as well as developing an understanding of how these skills are used to solve problems.

Unit introduction

Problem solving is an essential skill in many areas of life. To be successful, professionals in many disciplines including computing need to be able to analyse the needs of individuals and organisations and to evaluate the suitability and effectiveness of current ways of working, in order to develop solutions that utilise computer systems to improve or enhance processes and/or outcomes.

In this unit, you will explore logical and structured ways to develop programs and process data to solve specific problems. You will examine the features of effective computer programming. You will also develop the skills to effectively analyse a problem, break it down into its component parts, and design and evaluate solutions.

The knowledge, skills and understanding developed in this unit will support progression to computing-related higher education courses or to the workplace as a computing professional.

Summary of assessment

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.

Assessment outcomes

AO1 Demonstrate knowledge and understanding of computing facts, terms, standards, concepts, technologies and processes

AO2 Demonstrate application of knowledge and understanding of computing facts, terms, standards, concepts, technologies and processes

AO3 Demonstrate analysis of data and information related to computing to communicate solutions/understanding

AO4 Demonstrate evaluation of technologies, procedures, outcomes and solutions to make reasoned judgements.

[SP-PS]

Content

The essential content is set out under content areas. Students must cover all specified content before the assessment.

Students will be expected to demonstrate their understanding of Topics A, B and C using the programming language Python 3 version 3.10 (or higher), flowcharts and through written responses.

Where program concepts listed are not available in the Python 3 programming language, students would be expected to demonstrate understanding of the underpinning knowledge and understanding.

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 provides a list of key functions, methods and libraries that students should be able to use.

Students will be expected to write and interpret algorithms and design solutions to given problems using flowcharts. Students will be expected to use the flowchart symbols listed in Appendix 2.

A: Introduction to programming, logic and number

A1 Number systems used in computers

Students will need to know, understand and apply the mathematical operations performed on denary and binary number systems.

A1.1 Conversion of numbers between number systems.

A1.2 Use of binary to represent negative and floating-point numbers.

A1.3 Performing addition, subtraction, division and multiplication.

A1.4 Binary shifts.

A1.5 Overflow errors including:

- **A1.5.1** what they are
- **A1.5.2** problems that they can cause
- **A1.5.3** how to avoid and/or deal with overflow errors.

A2 Fundamentals of data and logic

Students will need to know, understand and be able to demonstrate the fundamentals of mathematical and data operations used to create programming code.

A2.1 Mathematical operators and their application (add, subtract, divide, multiply, integer division, modulus).

A2.2 Relational operators and their application (==, <, >, <>, <=, >=).

A2.3 Boolean operators and their application (NOT, AND, OR).

A2.4 Data types used in programming (string, integer, float/real, Boolean).

A2.5 Declaration, initialisation, and use of constants and variables.

A3 Program structure

Students will need to be able to structure program code to perform tasks and solve problems effectively.

A3.1 The use of appropriate sequencing in program structure, including:

- **A3.1.1** most efficient and logical order of actions
- **A3.1.2** the correct order of actions to ensure accurate outputs and avoid errors.

A3.2 The structure and application of selection (branching), including:

- **A3.2.1** IF statements (IF, ELSE, ELSEIF/ELIF)
- **A3.2.2** switch/case (match).

A3.3 The structure and application of iteration, including:

- **A3.3.1** count-controlled loops (FOR loops)
- **A3.3.2** condition-controlled loop (WHILE loops).

A3.4 User defined functions:

- **A3.4.1** defining and calling functions
- **A3.4.2** defining parameters
- **A3.4.3** passing arguments
- **A3.4.4** returning values.

A3.5 Local and global variables:

- **A3.5.1** using local and global variables
- **A3.5.2** benefits of each
- **A3.5.3** potential issues and how to mitigate/reduce problems.

A3.6 Recursion:

- **A3.6.1** the concept and uses of recursion
- **A3.6.2** benefits and drawbacks of using recursion.

B: Extending program functionality

Students will need to demonstrate the skills, knowledge and understanding required to create computer programs that make use of a range of functionality to solve problems effectively.

B1 Data structures

Students will need to demonstrate an understanding of common data structures used within a computer program and how they can be used to store and process data.

B1.1 Single- and two-dimensional data structures:

- **B1.1.1** indexing
- **B1.1.2** fixed length and dynamic

- **B1.1.3** single and mixed data types.

B1.2 Features of different data structures and how these impact their use:

- **B1.2.1** mutability:
  - mutable structure (lists, arrays, dictionaries)
  - immutable structures (tuples)
- **B1.2.2** key-value pairs
- **B1.2.3** indexing.

B1.3 Stacks and queues:

- **B1.3.1** how data is stored and accessed (last in first out (LIFO), first in first out (FIFO))
- **B1.3.2** the use of stacks and queues to solve problems
- **B1.3.3** implement stacks and queues using native Python data structures and associated built-in functions.

B2 Built-in functions

Students will need to demonstrate an understanding of common functions and how these are used to perform tasks and solve problems.

B2.1 Handing basic input and output.

B2.2 Numerical functions:

- **B2.2.1** generating and using random numbers
- **B2.2.2** defining and using a range
- **B2.2.3** rounding and truncation.

B2.3 String handling functions:

- **B2.3.1** concatenating and splitting strings and lists
- **B2.3.2** formatting inputs and outputs
- **B2.3.3** length
- **B2.3.4** position
- **B2.3.5** string conversion:
  - integer/float to string
  - string to integer/float.

B2.4 Using data structures:

- **B2.4.1** locating, adding and removing data
- **B2.4.2** ordering items
- **B2.4.3** numerical and statistical operations (highest and lowest values, average, counting occurrences, total number of data items).

B2.5 Working with external text files:

- **B2.5.1** common delimiters (comma, whitespace)
- **B2.5.2** reading data from and writing data to .txt files
- **B2.5.3** benefits and drawbacks of using .txt files to store program data.

C: Developing programs to solve problems and specific requirements

C1 Problem solving

Students will need to demonstrate an understanding of how to use mathematics, logic and processes of computer systems to solve problems and communicate solutions using flowcharts and the Python 3 programming language.

C1.1 The use of computational thinking to explore problems and apply solutions (decomposition, pattern recognition, abstraction, algorithmic design).

C1.2 The concept of an algorithm as a set of step-by-step instructions to solve a problem.

C1.3 Why developers use different methods for expressing algorithms/solutions:

- **C1.3.1** pseudocode
- **C1.3.2** flowcharts
- **C1.3.3** program code
- **C1.3.4** benefits and drawbacks of these methods.

C1.4 Produce algorithms that demonstrate understanding of programming logic and constructs (sequence, selection and iteration) using code/flowcharts.

C1.5 Interpreting given algorithms including:

- **C1.5.1** determining their purpose.
- **C1.5.2** determining the outputs generated:
  - dry-running
  - trace tables
- **C1.5.3** identifying and correcting errors.

C1.6 Common algorithms:

- **C1.6.1** searching algorithms (binary search, linear search):
  - describe how they work
  - interpret and debug code that uses searching algorithms
  - benefits and drawbacks.
- **C1.6.2** sorting algorithms (bubble sort, quick sort):
  - describe how they work
  - interpret and debug code that uses sorting algorithms
  - benefits and drawbacks.

C2 Developing high-quality computer programs

Students will need to demonstrate an understanding of how to produce high-quality program code, ensuring computer programs meet user expectations and produce reliable outcomes.

C2.1 Techniques to ensure readable and maintainable code:

- **C2.1.1** use of style and layout guides (tabs, spacing, indents, line length)
- **C2.1.2** use of case (snake_case, camelCase, UPPER CASE, PascalCase)
- **C2.1.3** code annotations/comments
- **C2.1.4** modularisation.

C2.2 Validation techniques to improve accuracy of data and reliability of programs including:

- **C2.2.1** validation check techniques used in computer programs:
  - type check
  - range check
  - presence check
  - format check
  - constraints check (value check/lookup, uniqueness).
- **C2.2.2** program actions following a validation check including:
  - feedback to user
  - confirming validity and moving to next instruction/process
  - looping until validity is confirmed.
- **C2.2.3** how validation check techniques can be implemented using the Python 3 programming language.

C2.3 Providing meaningful user interaction to a user:

- **C2.3.1** clarity and formatting of output messages
- **C2.3.2** formatting numeric data with consideration of:
  - purpose
  - accuracy
  - ease of use
- **C2.3.3** handling user input:
  - simplification of user input (coding, menu options)
  - reformatting to allow processing (changing case, type casting).

C2.4 Testing programs to ensure that they are functional and produce the correct outcomes:

- **C2.4.1** description of the purpose of the test

C2.4.2 identification of test data to be used:

- testing using typical values that would be accepted and or rejected (valid and invalid data)
- testing the boundaries of logic (extreme data)
- testing using atypical data or data of incorrect type (erroneous data)

C2.4.3 describing the expected outcomes

C2.4.4 using the outcomes of testing to identify and fix issues.

D: Issues relating to developing computer programs

Students will need to demonstrate an understanding of the wider issues that computer programmers should consider when developing computer software, and possible impacts of these issues on development.

D1.1 Use of code written by others (third-party code):

D1.1.1 Open-source code and Application Protocol Interfaces (APIs)

D1.1.2 Closed source/proprietary APIs

D1.1.3 Compatibility

D1.1.4 Costs

D1.1.5 Licencing issues

D1.1.6 Legal and contractual issues.

D1.1.7 Use of AI generated code:

- security concerns (e.g. bugs in the code, sharing existing code/data with the AI model, developer not fully aware of what the generated code does)
- compatibility with/integration into larger code base
- checking and testing generated code

D1.2 The concepts of diversity and inclusion when developing computer software:

D1.2.1 Web Content Accessibility Guidelines (WCAG) (perceivable, operable, understandable, robust) and how they can be applied

D1.2.2 Potential bias in data sets and how to minimise them:

- types of bias – sampling, measurement, response, analysis, reporting
- mitigating bias – diversifying collection team, use of multiple sources, transparency of data and process, gather feedback

D1.3 Common problems associated with developing computer programs:

D1.3.1 Compatibility issues and how they can be addressed:

- web/cloud deployment
- code translation to develop different versions of the same software
- software versions

D1.3.2 Addressing the digital divide in relation to access to and use of software:

- ensuring ease of use
- optimisation of programs for use on a range of devices

D1.3.3 Mitigating security issues (threats to data)

- user access
- secure coding practices
- outdated tools and libraries

D1.3.4 Impacts of development time of new programs:

- unrealistic timescales from clients
- feature creep
- rapid response to changes in user expectations and industry trends.

Transferable skills

Managing Yourself	Effective Learning	Interpersonal Skills	Solving Problems
MY – TPR	EL – MOL	IS – WC	SP – CT
MY – PS&R	EL – CL	IS – V&NC	SP – PS *
MY – COP	EL – SRS	IS – T	SP – C&I
MY – PGS	EL – PRS	IS– C&SI	

Table key

*	Signposted to indicate opportunities for development as part of wider teaching and learning.
√	Embedded in teaching, learning and assessment
Blank	TS not embedded or signposted in unit

Key terms typically used in assessment

The following table shows the key terms that will be used consistently by Pearson in our assessments to ensure students are rewarded for demonstrating the necessary skills.

Please note: the list below will not necessarily be used in every paper/session and is provided for guidance only.

Command or term	Definition
Assess	Give careful consideration to all the factors or events that apply. Make a judgement on the extent to which a set of given requirements have been met.
Calculate	Students apply some form of mathematical or computational process, showing relevant working if necessary.
Complete	Provide the missing information for a table, diagram, line of code, or process.
Describe	Students provide an account of something, or to highlight a number of key features of a given topic. May also be used in relation to the stages of a process.
Develop	Students produce a section of programming code to meet a set of given requirements.
Discuss	Consider the different aspects in detail of an issue, situation, problem or argument and how they interrelate. Does not require a conclusion.
Draw	Students produce a flowchart to show understanding of a system or process.
Evaluate	Consider various aspects of a subject's qualities in relation to its context such as: strengths or weaknesses, advantages or disadvantages, pros or cons. Come to a judgement supported by evidence which will often be in the form of a conclusion.
Explain	Requires identification of a point and linked justification or expansion of that point.

Command or term	Definition
Give/Name/State	All of these command words are synonyms. They generally all require recall of one or more pieces of information.
Identify	Requires some key information to be selected from a given stimulus/resource. Requires a single word or short sentence answer.
Write	Students produce a short section/line of code.

Appendix 1: Python commands, functions and methods

Students should be given the opportunity to explore the programming and problem-solving concepts listed in Topic A using the Python 3 programming language (version 3.10 or higher).

This appendix provides additional information as to the scope of Python 3 that students should be familiar with. This appendix does not replace the specification content but should be used to support the teaching of the content listed.

Coding questions and examples used in the live assessment will draw upon this list.

Handling basic input and output

- input()
- print()
- int()
- str()
- float()
- try:
- except:

Functions and variables

- def
- global
- return

Selection

- if
- elif
- else
- match
- case

Iteration

- while
- for

Built-in functions and standard library commands

- import
- from
- as
- *

Built-in functions and standard libraries

- math
- random
- statistics

Numerical

- random( )
- randint( )
- uniform( )
- sample( )
- range( )
- round( )
- math.trunc( )
- math.floor( )
- math.ceil( )
- max( )
- min( )
- mean()
- count( )

String handling

- isupper( )
- islower( )
- upper( )
- lower( )
- isalpha( )
- isdigit( )
- split( )

- len( )
- format( )
- join( )

Using data structures (lists and arrays)

- index( )
- append( )
- insert( )
- remove( )
- count( )
- pop( )
- sort( )
- in
- not in
- len( )

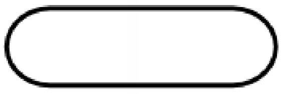
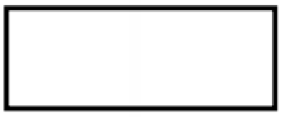
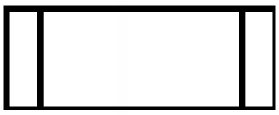
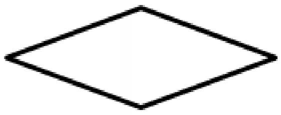
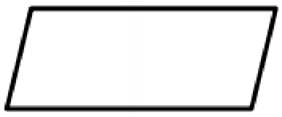
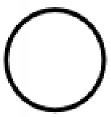
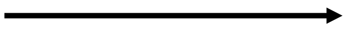
Working with external text files

- with
- open( )
- write( )
- close( )
- read( )
- readline( )
- readlines( )
- line.split( )

Appendix 2: flowchart symbols

This appendix provides additional information about the flowchart symbols that will be used in the core examinations. Students are expected to respond to flowchart questions using these symbols.

This appendix does not replace the specification content but should be used to support the teaching of the content listed.

	Denotes the start and end of an algorithm
	Denotes a process to be carried out
	Denotes a sub-process
	Denotes a decision to be made
	Denotes input or output
	Denotes a connection to part of a flowchart that cannot easily be linked using an unbroken flow arrow
	Shows the logical flow of the program

Unit 3: Human-Computer Interaction

Level: 3

Unit type: Internal

Guided learning hours: 60

Unit in brief

Students will examine the underlying principles of human-computer interaction (HCI) and develop a HCI solution to meet the requirements of a given brief.

Unit introduction

As technology develops, so do the ways in which we interact with computer systems. From voice-controlled digital assistants to touchscreen-controlled smartphones, and even prosthetic limbs that can be more precisely controlled by the wearer, the ways in which we interact with electronic devices are becoming increasingly sophisticated.

Computing professionals need to understand how to make the interaction between the user and the computer as natural and efficient as possible. Significant research in the field of HCI continues to inform how systems can achieve this goal.

In this unit, you will consider how the principles of user experience (UX) and user interface (UI) design are used to develop methods of interaction between humans and computers that allow a range of needs to be met. You will explore how the user and the purpose of the digital system influence the design choices for HCI solutions, including how systems can be made so that a range of users with potentially diverse needs can be accommodated.

You will apply problem-solving skills and your understanding of key principles of HCI design to design and develop a solution to a HCI-based scenario.

The knowledge and skills you develop in this unit will help prepare you for a number of roles in STEM industries and support progression to further study.

Learning aims

In this unit you will:

A Explore the factors affecting human-computer interaction

B Develop a proposal and designs for a human-computer interaction solution in response to a brief

C Develop your planned human-computer interaction solution in response to the brief.

Summary of unit

Learning aim	Key content areas	Assessment approach
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	Research the application of HCI principles in different use cases.
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	Design, prepare and evaluate a prototype HCI solution in response to a given brief (covers learning aims B and C).
C Develop your planned human-computer interaction solution in response to a 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	Design, prepare and evaluate a prototype HCI solution in response to a given brief (covers learning aims B and C).

Content

The essential content is set out under content areas. Students must cover all specified content before the assessment.

Learning aim A: Explore the factors affecting human-computer interaction

A1 Fundamental concepts of human-computer interaction

Students explore the core concepts of human-computer interaction and how they are applied to ensure effective digital systems.

- The differences and links between user experience (UX) and user interface (UI) design.
- Basic concepts of good UX including:
  - Usability
  - Accessibility
  - Efficiency
  - Engagement
  - Enjoyment.
- Basic concepts of good UI including:
  - Simplicity
  - Clarity
  - Consistency
  - Feedback
  - Accuracy
  - Efficiency
  - Flexibility
  - Accessibility.
- Features and uses of different types of user interaction including:
  - Command Line Interface (CLI)
  - Menu-driven interface
  - Graphic user interface (GUI)
  - Voice-activated interface
  - Gesture-based interface.
- Benefits and drawbacks of different types of user interaction.

A2 Use and purpose considerations

How the requirements for different devices and the needs of different users can be met and how these impact on HCI design.

- Target platforms and devices including:
  - traditional workstations (e.g. laptop/desktop PC, screen, keyboard, mouse)
  - smartphones/tablets
  - wearables (e.g. fitness trackers, smartwatches)
  - digital assistants (e.g. Amazon Alexa, Google Nest, Siri)
  - gaming consoles
  - virtual reality, augmented reality
  - new technologies.
- Characteristics of target users:
  - Types of user (expert, regular, occasional, beginner/novice)
  - Demographics (e.g. age, education, cultural factors, location/globalisation)
  - Accessibility requirements.
- The industry/sector of the intended system/service:
  - Education
  - Entertainment/leisure
  - Manufacturing
  - Retail
  - Hospitality.
- Purpose of intended system/service and required outputs to meet user requirements including:
  - Provide data and information (textual, graphical, tabulated)
  - Entertainment (sound/music, video, text, images)
  - Support a task (e.g. driving data, activity tracking, smart meter, medical monitoring).
- Use and purpose requirements in different sectors (education, entertainment/leisure, manufacturing, retail, hospitality).
- Uses and requirements of haptic feedback.
- Use and integration of assistive technologies, e.g. eye gaze system, braille, screen magnifiers, avatars for sign language.

A3 The principles of HCI design

The principles of how humans perceive and interact with digital systems and how this affects the design of HCI solutions.

- The difference between recognition and recall.
- Screen design for intuitive data entry.
- Menu selection.

- Perception:
    - gestalt principles.
    - colour, to include trichromatic system and luminance.
    - gross 3D shapes.
- Shneiderman's 8 golden rules of interface design:
    - consistency
    - use of shortcuts
    - informative feedback
    - design dialog to yield closure
    - error handling
    - reversal of actions
    - support internal locus of control
    - reduce short-term memory load.
- Behavioural models for UX/UI design:
    - Attention model (novel, familiar, meaningful, proximity, movement)
    - Intention model (goals, expectations, perceptions, motivations)
    - Emotion model (happy, sad, angry, surprised, fearful)
    - Memory model (novel, meaningful, proximity, repetition).

Learning aim B: Develop a proposal and designs for a human-computer interaction solution in response to a brief [IS-WC]

B1 Defining requirements for a HCI solution

Defining the problem and preparing initial documentation to provide a proposal for a HCI solution.

- The problem solving process; stages and activities in the problem solving process
- Exploring the scope and impact of the problem:
    - Tasks to be performed
    - Input required, e.g. mouse, touchscreen, voice
    - Output required, e.g. graphics, animations, audio feedback, physical feedback
    - User needs, e.g. accessibility considerations, purpose of system, environmental factors.

B2 Design documentation for a HCI solution

Documentation needed to plan the development of a HCI solution and communicate ideas to others.

- Research into similar systems and common practices to generate ideas to identify potential solutions.

- Selecting and presenting the solution:
    - overview of user requirements
    - task flows
    - style guides (e.g. colour palettes, typography, media components)
    - visualisation/interface design, (e.g. sketches, wireframes, storyboards)
    - technical specifications including:
        - functional requirements specification
        - non-functional requirements specification
    - user stories and personas
    - technical designs (e.g. algorithms, example code, wiring diagrams)
    - user stories and personas.

Learning aim C: Develop your planned human-computer interaction solution in response to the brief

C1 Content preparation for a human-computer interface

Selection and application of appropriate processing and editing techniques to prepare resources to meet identified requirements.

- Creating unique content, e.g. sounds, images, control code.
- Use of content created by others:
    - permissions
    - acknowledging sources
    - legal and ethical considerations e.g. using content created by others, privacy, political bias, representation of individuals and groups.
- Optimisation, e.g. file size, image size.
- Alternate formats for screen orientation, e.g. landscape, portrait.
- File formats, i.e. compatibility, performance, quality.
- Compression requirements for items such as images, possible constraints, file size and image quality.

C2 Developing a HCI solution

Application of HCI design principles to develop a solution in response to a brief

- Primary interface implementation, e.g. standard icons, menus, window layout.
- Implementing alternative interfaces, e.g. mobile version, adaptive for user needs.
- Software integration, e.g. event handling, coding to add functionality, applying interface to intended program.
- Hardware integration, e.g. bespoke controllers, recognising keystrokes, adaptive technologies, coding to control connected hardware.

C3 Testing a HCI solution

Methods of testing and refining a HCI solution during development to ensure it is of high-quality and meets identified needs.

- Identifying how and what to test, e.g. producing a test plan, choosing test data, test user identification.
- Types of testing, e.g. effectiveness, functionality, performance, user acceptance.
- Making improvements and/or refinements to solutions in response to the outcomes of testing

C4 Reviewing the development process and outcomes

Review of the success of development of the HCI solution developed.

- Producing a review of the extent to which the solution meets a given brief including consideration of:
    - suitability for audience and purpose
    - ease of use
    - quality of the solution, e.g. reliability, usability, efficiency/performance, maintainability, portability
    - constraints, e.g. time, sourcing hardware components, platform, compatibility
    - legal and ethical considerations
    - strengths and weaknesses of the solution
- how the implemented solutions could be improved to better meet the needs of the user and fulfil the identified needs.

Assessment criteria

Learning aim A: Explore the factors affecting human-computer interaction

Pass	Merit	Distinction
A.P1 Explain the UX and UI considerations that impact on HCI design. **A.P2** Describe how the use and purpose of digital systems impact on HCI design.	**A.M1** Analyse how HCI design principles are applied to ensure effective UX and UI.	**A.D1** Evaluate how HCI design principles are applied to ensure effective UX and UI.

Learning aim B: Develop a proposal and designs for a human-computer interaction solution in response to a brief

Pass	Merit	Distinction
B.P3 Produce an adequate proposal for a solution that meets most of the requirements of the brief **B.P4** Produce adequate designs that show some awareness of principles of UX/UI design. **[SP-C&I]**	**B.M2** Produce a good proposal for a solution that meets the requirements of the brief **[SP-PS]** **B.M3** Produce detailed designs that show good awareness of principles of UX/UI design	**B.D2** Proposal and designs are comprehensive and effective, showing excellent awareness of the requirements of the brief.

Learning aim C: Develop your planned human-computer interaction solution in response to the brief

Pass	Merit	Distinction
C.P5 Produce an adequate HCI solution to meet most of the requirements of the brief. **[SP-C&I]** **C.P6** Produce an adequate partially supported review of the extent to which the HCI solution meets the requirements of the brief.	**C.M4** Produce an effective HCI solution to meet the requirements of the brief. **C.M5** Produce a mostly supported review of the extent to which the HCI solution meets the requirements of the brief. **[SP-PS]**	**C.D3** Produce a highly effective HCI solution to fully meet the requirements of the brief. **C.D4** Produce a fully supported review of the extent to which the HCI solution meets the requirements of the brief.

Transferable skills

Managing Yourself	Effective Learning	Interpersonal Skills	Solving Problems
MY – TPR	EL – MOL	IS – WC *	SP – CT
MY – PS&R	EL – CL	IS – V&NC	SP – PS √
MY – COP	EL – SRS	IS – T	SP – C&I √
MY – PGS	EL-PRS	IS – C&SI	

Table key

*	Signposted to indicate opportunities for development as part of wider teaching and learning.
√	Embedded in teaching, learning and assessment
Blank	TS not embedded or signposted in unit

Essential information for Pearson Set Assignment Brief (PSAB)

Pearson sets the assignment for the assessment of this unit.

The PSAB will take 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.

Assessing the PSAB

You will make assessment decisions for the PSAB using the assessment criteria provided.

Section 1 gives information on PSABs and there is further information on our website.

Further information for teachers and assessors

Resource requirements

For this unit, students must have access to technological resources that will allow them to apply the practical principles of HCI. These may include:

- graphics software
- prototyping tools, for example Figma, InVision, Framer, JustInMind
- appropriate development/coding environment for producing interactive functionality, for example Visual basic®, Python®
- prototyping boards, for example Raspberry Pi®, Arduino®
- specialised input/output devices.

Essential information for assessment decisions

Learning aim A

While there is no requirement for a specific number of digital solutions that the students should consider, it is recommended that students examine at least three different digital systems, within a similar context or industry as that of the PSAB. Students may select the systems themselves or tutors may direct students to suitable systems. The systems selected must provide suitable opportunity for students to consider a range of intended user needs and purposes and how these are addressed (or not) in the identified systems.

Students should present their evidence in the form of a detailed report. Where students have specific needs which may prevent them from producing a detailed written report, sensible adaptations can be made to the form of the evidence, for example video presentation, spoken report. Where adaptations are made, the evidence would still be expected to meet the requirements of the assessment criteria and the essential information outlined in this section.

For distinction standard, students must evaluate the effectiveness of UX and UI design in a number of different example systems that are relevant to the context of the PSAB.

The evaluation must consider how different HCI concepts are applied (e.g. usability, clarity, flexibility etc) and how principles of HCI design are utilised in the identified systems. The students should apply their knowledge of basic principles of HCI design within the given context and use this to make value judgments as to how well the identified systems meet the use and purpose considerations of the identified systems. Students must provide supported justifications for the value judgments that they make with clear reference to the needs of the users and the purpose of the system.

At this level students should make appropriate use of technical vocabulary to effectively support the points they make in their report.

For merit standard, students must analyse the use of UX and UI design in a number of different example systems that are relevant to the context of the PSAB.

The analysis must consider how different HCI concepts are applied (e.g. usability, clarity, flexibility etc) and how principles of HCI design are utilised in the identified systems. The students should apply their knowledge of basic principles of HCI design within the given context and provide supported statements that explore in detail how design principles are applied to ensure effective consideration of the intended use and purpose of the identified system. Students must provide clear and consistent support for the points they make.

At this level students should make mostly appropriate use of technical vocabulary to effectively support the points they make in their report.

For pass standard, students will explain the ways in which UX and UI considerations have impacted on the design of the identified systems. They will identify a number of different UI/UX concepts and provide an explanation as to why they have been used in the system. They will include descriptions of how the use and purpose of the identified system has impacted on the way it has been designed and implemented and how these relate to the intended audience.

At this level the students should apply their understanding through the selection of appropriate examples from the identified systems. The student's descriptions and explanations will show an adequate understanding, but they may not always expand the points made and may not always effectively link their points to the chosen examples. They may at times rely on more generic assertions.

Student responses will make some accurate observations but will not demonstrate multiple chains of reasons. For example, while they might identify that a lack of responsive design may prevent some users from accessing the system effectively, they are unlikely to expand on this further.

At this level students will make appropriate use of technical vocabulary in places, but its use and accuracy may be inconsistent.

Learning aims B and C

Students will provide evidence of planning, developing, implementing and testing a HCI solution to meet the requirements of a brief. The focus of the HCI solution may vary, depending on the student’s interpretation of the identified brief. For example, students may choose to build and design a hardware solution that is designed to provide bespoke or assistive interaction with a computer system. Alternatively, students may choose to design and create a software-based HCI solution.

For distinction standard, students will draw on, and show synthesis of, knowledge across the learning aims to produce comprehensive designs for a HCI solution. The designs will include a comprehensive proposal, that effectively communicates a solution to the given brief. The proposed solution will clearly and effectively meet the specific requirements of the brief, demonstrating an excellent awareness of the context.

The student will expand on their proposal to produce comprehensive design documentation that effectively communicates how their proposed solution would be implemented.

At this level, the proposal and design documentation should be of high quality and provide high levels of clarity and detail so that, if required, a third party could easily use them to create the proposed solution.

Students will demonstrate implementation of their design through an iterative approach to development of a prototype HCI solution. The student will demonstrate effective use of UX/UI tools to produce a high-quality solution that effectively meets the requirements of the brief.

The student will provide detailed logs for the continued testing and developments to the solution carried out throughout the assessment, which will include how any issues were discovered and resolved.

The student will provide an evaluation of the HCI solution against the identified requirements and quality criteria. The student's review will effectively evaluate the extent to which it meets the requirements in the brief and will be supported by evidence from the development and testing stages, and will provide supported conclusions and suggest future developments.

For merit standard, students will demonstrate good knowledge and skills in relation to HCI concepts through the production of good designs for a HCI solution that mostly meets the requirements of the brief. The designs will include a detailed proposal, that communicates a solution to the given brief. The proposed solution will meet the requirements of the brief, demonstrating an awareness of the context although sometimes the link may not be clear or may be implied.

The student will expand on their proposal to produce detailed design documentation that communicates how their proposed solution would be implemented.

At this level, the proposal and design documentation should be of sufficient quality, clarity and detail so that, if required, a third party could, with minimal difficulty, use them to create the proposed solution.

Students will demonstrate implementation of their design through a mostly iterative approach to development of a prototype HCI solution. The student will demonstrate good and appropriate use of UX/UI tools to produce a solution that effectively meets most requirements of the brief.

The student will provide logs for the continued testing and developments to the solution carried out throughout the assessment which will include how any issues were discovered and resolved. However some minor errors may still persist.

The student will review the extent to which their the HCI solution meets the identified requirements of the brief. The student's review will provide evidence from their work to support most of the points they make. Their review may be slightly unbalanced and the conclusions they reach, and suggested future developments, while appropriate may be a little generic.

For pass standard, students will demonstrate adequate knowledge and skills in relation to HCI concepts through the production of acceptable designs for a HCI solution that meets some of the requirements of the brief. The designs will include a proposal that communicates a solution to the given brief. The proposed solution will meet most of the requirements of the brief, demonstrating a satisfactory awareness of the context.

The student will expand on their proposal to produce design documentation that sufficiently communicates how their proposed solution would be implemented.

At this level, the design documentation will be of a reasonable quality, so that a third party could mostly create the proposed solution, but there may be some difficulties due to lack of detail or clarity in places.

Students will demonstrate implementation and testing of their HCI solution. The student will demonstrate adequate use of UX/UI tools to produce a solution that meets most requirements of the brief.

The student will provide logs of how any issues were discovered and resolved, but the process may be quite linear and the scope of testing quite narrow, focusing on only common/obvious issues. A number of errors and issues may still persist.

The student will review the extent to which their the HCI solution meets the identified requirements of the brief. The student’s review will consider what they have done well and what they could improve but support and justification will often be generic or implied.

Unit 4: Practical Programming

Level: 3

Unit type: Internal

Guided learning hours: 60

Unit in brief

Students explore the principles of computer science in connection with the concepts of software development. Students use a systematic and methodical approach to manage the development of a software solution to a problem.

Unit introduction

Solutions to real-life, complex problems require an effective and efficient application of computational thinking skills. They also require a fuller understanding of fundamental data structures and algorithms that can be extended or customised for different problems. Solutions, by definition, should be functional and meet requirements. Programmers, in real life, use a variety of techniques to ensure that development is focused on delivering one functional part of a solution at a time.

In this unit, you will explore principles of computer science that will allow you to tackle more complex problems requiring a programmed solution. You will use computational thinking skills to effectively analyse a problem, decompose it into its component parts, design data structures, and algorithms. You will then implement the design, using a systematic approach to ensure functional outcomes that meet requirements. You will produce documentation detailing the development process and maintain a repository of program code versions.

Learning aims

In this unit you will:

A Explore principles of computing related to software development

B Manage the development of a software solution.

Summary of unit

Learning aim	Key content areas	Assessment approach
A Explore principles of computing related to software development	**A1** Input and output **A2** Data structures **A3** Searching **A4** Sorting **A5** Good practice in programming	Develop a software solution to a problem, incorporating the principles of computer science.
B Manage the development of a software solution	**B1** Develop a computer program to solve a problem	Manage the development of a software solution to a problem. Produce a document describing the methodical approach to managing the development of a software solution.

Content

The essential content is set out under content areas. Students must cover all specified content before the assessment.

Learning aim A: Explore principles of computing related to software development [MY-TPR]

As students move on to develop solutions for complex problems, they will need a wider range of tools and techniques to draw upon. They will develop skills in importing and exporting persistent data. They will develop skills in searching and sorting algorithms to maintain one-dimensional and two-dimensional data structures. They will develop skills in designing solutions that are efficient, robust, and demonstrate good practice in programming.

Good practice in programming means those things software developers do to ensure a methodical approach to solution development. These include focusing on a single feature at a time and ensuring code is delivered in functional units. Students will employ a systematic approach to software development.

A1 Input and output

- Validation
  - Presence
  - Range
  - Length
  - Pattern/format
  - Lookup
- Text files (character delimited)
  - Reading
  - Writing

A2 Data structures

- Dimensionality
  - One-dimensional arrays
  - Two-dimensional arrays
- Operations
  - Append
  - Insert
  - Update
  - Delete

A3 Searching

- Linear search
  - Sorted array
    - Efficiency (early exit on found or passed over)

  - Unsorted array
    - Efficiency (early exit on found)
- Binary search
  - Sorted array
- Efficiency (compare to linear to binary)

A4 Sorting

- Bubble sort
- Quick sort (with recursion)
- Efficiency (compare to bubble to quick)

A5 Good practice in programming

- Aids to understanding
  - Comments
  - Layout
  - White space
  - Indentation
- Separation of concerns
  - Subprograms
  - Functions
  - Procedures
  - Subprogram interfaces
  - Parameter passing
- Handling errors
  - Exception handling
- Systematic approach to software development
  - Features (functional tasks)
  - User stories expressed in the given, when, then format
  - Unit tests
  - Linting

Learning aim B: Manage the development of a software solution [MY – TPR]

Students will manage the development of a software solution using a systematic methodical approach. In the context of a problem statement, including requirements, students will express the requirements as features, each of which consists of one or more user stories. They will then implement each feature serially. They will complete one feature before moving on to another. Once a feature is complete, the program code will be put into a repository or file structure to preserve it. This point in the development is a milestone.

For each milestone, students will write documentation detailing the user stories, how the student knows the code is working, the location of the preserved code, and any issues, problems, or discoveries that are important.

B1 Develop a computer program to solve a problem

- Use software development management
  - Express tasks as features
  - Express features as user stories in the given, when, then format
  - Implement user stories sequentially, one at a time, to completion before starting on another
  - Write unit tests where appropriate
  - Run unit tests during development
- Document functional milestones
  - Demonstrate a functional completed feature
  - Reflect on relevant issues or problems in completing the feature
- Keep a repository of code and documents representing functional milestones
  - Archive written documents and code
  - Name archived materials so that they are identifiable
- Use decomposition and abstraction to provide separation of concerns
  - Make effective use of subprograms, with well-defined interfaces, to isolate logic
- Use techniques to make complex code readable and maintainable
  - Limit global variables
  - Lint the code

Assessment criteria

Learning aim A: Explore principles of computing related to software development

Pass	Merit	Distinction
A.P1 Develop software to meet some requirements that incorporates some appropriate: • inputs and outputs • data structures • sorting and searching algorithms **[SP-C&I]** **A.P2** Employ some good practices in programming during software development **[SP-C&I]** **A.P3** Apply a systematic approach to software development	**A.M1** Employ a range of good practices and effective systematic approaches to develop software that meets most requirements and incorporates mostly appropriate: • inputs and outputs • data structures • sorting and searching algorithms	**A.D1** Consistently employ a range of good practices effectively and use efficient systemic approaches to develop software that fully meets requirements and incorporates • well-chosen inputs and outputs • a range of data structures • a range of sorting algorithms and searching algorithms

Learning aim B: Manage the development of a software solution

Pass	Merit	Distinction
B.P4 Employ software development management strategies, document functional milestones, and maintain a repository of files containing computer program code.	**B.M2** Employ effective software development management strategies, fully document functional milestones, and maintain a repository of files demonstrating functional milestones.	**B.D2** Employ fully appropriate and effective software development management strategies, fully and effectively document functional milestones, and maintain a repository of files effectively demonstrating functional milestones.

Transferable skills

Managing Yourself	Effective Learning	Interpersonal Skills	Solving Problems
MY – TPR *	EL – MOL	IS – WC	SP – CT
MY – PS&R	EL – CL	IS – V&NC	SP – PS
MY – COP	EL – SRS	IS – T	SP – C&I √
MY – PGS	EL-PRS	IS – C&SI	

Table key

*	Signposted to indicate opportunities for development as part of wider teaching and learning.
√	Embedded in teaching, learning and assessment
Blank	TS not embedded or signposted in unit

Essential information for Pearson Set Assignment Brief (PSAB)

Pearson sets the assignment for the assessment of this unit.

The PSAB will take 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.

Assessing the PSAB

You will make assessment decisions for the PSAB using the assessment criteria provided.

Section 1 gives information on PSABs and there is further information on our website.

Further information for teachers and assessors

Resource requirements

For this unit, students must have access to:

- a programming language translator capable of producing executable code
- an integrated development environment, including an editor, which can highlight syntax errors, and a debugger, which provides breakpoints, stepping, and variable inspection
- a linter for the programming language source code files
- the internet for research and tutorials
- productivity software, including a word-processor, a spreadsheet, presentation, diagramming, and PDF reader applications.

Essential information for assessment decisions

Learning aim A

For distinction standard, students must demonstrate appropriate and comprehensive application of a wide range of techniques in developing a software solution.

Students must demonstrate manipulation of a range of data structures. They must select and use appropriate operations to affect individual items in the data structures to meet the requirements of the solution.

Students must demonstrate understanding of a range of searching algorithms, including efficient linear searches and binary search. They must implement these algorithms, with basic language constructs and without the use of language constructs specifically designed for that purpose, for example, <list>.find().

Students must demonstrate understanding of a range of sorting algorithms, including bubble sort and quick sort. They must implement these algorithms, with basic language constructs and without the use of language constructs specifically designed for that purpose, for example, sort().

Students must compare the efficiency of searching algorithms programmatically, for example by timing search algorithm execution on a variety of sorted and unsorted data, where the target is present and where the target is absent.

Students must compare the efficiency of sorting algorithms programmatically, for example by timing sort algorithm execution on the same data set or data sets.

Students must demonstrate, programmatically, the manipulation of persistent data. This could be done, for example, by importing data from a persistent store and exporting data to a persistent store. The student must choose the format of the persistent store.

Students' software solutions must fully meet requirements, including provision of accurate and useful information to the user, under a variety of circumstances, including reporting on current status and errors.

Students must ensure that their software solution is robust. They must implement strategies to ensure only valid input is allowed to enter their program, from the user and from the import process. They must design and implement validation rules to ensure the robustness of their solution.

Students must produce software solutions that handle predictable errors, using advanced techniques, such as exception handling. They must demonstrate advanced testing techniques, for example writing unit tests for a search and a sort algorithm.

Students must design and produce program code that demonstrates decomposition and separation of concerns, for example, by the use of separate code files and subprogram interfaces. They must use the concept of scope, via parameters and arguments, to good effect so that the use of global variables is limited.

Students must produce program code that is easy to read, understand, and maintain. They must demonstrate appropriate and consistent use of techniques such as file naming conventions, commenting, meaningful identifiers, layout, and adherence to programming language standards, enforced by linting.

For merit standard, students must demonstrate appropriate application of a range of techniques in developing a software solution.

Students must demonstrate manipulation of strings, one-dimensional structures, and two-dimensional structures. They must use appropriate operations to affect individual items in the data structures, for example, by inserting, appending, changing, and deleting.

Students must demonstrate understanding of searching algorithms, including linear search and binary search. They must implement these algorithms, with basic language constructs and without the use of language constructs specifically designed for that purpose, for example, <list>.find().

Students must demonstrate understanding of a sorting algorithm, for example bubble sort. They must implement a sorting algorithm, with basic language constructs and without the use of language constructs specifically designed for that purpose, for example, sort().

Students must demonstrate a way to collect data about the efficiency of algorithms, for example by using a timing function to collect data about searching and sorting algorithms.

Students must demonstrate, programmatically, the manipulation of persistent data. This could be done, for example, by importing data from a text file and exporting data to a text file.

Students must provide accurate and useful information to the user, under a variety of circumstances, including reporting on current status and errors.

Students must implement strategies to ensure only valid input is allowed to enter their program, from the user and from the import process. They must design and implement validation rules to ensure their solution handles predictable input errors.

Students must design and produce program code that demonstrates decomposition and separation of concerns, for example, by the use of subprogram interfaces. They must use the concept of scope, via parameters and arguments, so that the use of global variables is limited.

Students must produce program code that is easy to read, understand, and maintain. They must demonstrate appropriate use of techniques such as file naming conventions, commenting, meaningful identifiers, and layout.

For pass standard, students must apply some software development techniques while producing a software solution.

Students must demonstrate manipulation of strings, one-dimensional structures, and two-dimensional structures. They must use appropriate operations to affect individual items in the data structures, for example by appending and changing.

Students must demonstrate understanding of searching algorithms, for example, a linear search. They must develop program code to locate a target in both sorted and unsorted data.

Students must demonstrate understanding of a sorting algorithm, for example bubble sort. They must implement a sorting algorithm, with basic language constructs and without the use of language constructs specifically designed for that purpose, for example, sort().

Students must demonstrate a way to collect data about the efficiency of an algorithm, for example by using a timing function to collect data about a sorting algorithm.

Students must demonstrate, programmatically, the manipulation of persistent data. This could be done, for example, by importing data from a file and exporting data to a file.

Students must provide accurate and useful information to the user, under a variety of circumstances, including errors.

Students must ensure that their software solution handles predictable situations.

Students must design and produce program code that demonstrates decomposition and separation of concerns, for example, by the use of subprogram interfaces.

Students must produce program code that is easy to read and understand. They must demonstrate appropriate use of techniques such as file naming conventions, commenting, meaningful identifiers, and layout.

Learning aim B

For distinction standard, students must provide comprehensive evidence of a systematic and methodical approach employed during the development of the software solution.

Students must produce comprehensive evidence of the accurate and consistent interpretation of requirements, that they have effectively implemented in program code.

They must produce evidence of fully functional milestones, based on requirements, reached during the development. Students must establish and maintain a logically organised and structured repository of code and documentation, reflecting these milestones.

Students must provide details that exactly align with the repository to allow changes to be reversed quickly and efficiently. Students must demonstrate the ability to retrieve, build, and execute code to produce a functional version of the software solution at any given milestone.

The students must recognise and highlight interesting, non-trivial, or troublesome issues encountered during each milestone. The students must concisely, but effectively, describe how these were resolved. Students must acknowledge lessons learned that can be taken forward to the development of their next software solution.

Students must discuss the methods used to ensure the program code is robust, functions as designed, and meets the requirements. Students must demonstrate a wide range of quality assurance methods, including unit testing.

Overall, the evidence must be logically structured with correct technical terms and written at a high standard including the consistent use of correct grammar and spelling.

For merit standard, students must provide evidence of a systematic approach employed during the development of the software solution.

The students must produce evidence of the consistent interpretation of requirements, that they have implemented in program code.

They must produce evidence of functional milestones, based on requirements, reached during the development. Students must maintain an organised repository of code for each milestone.

Students must provide details that align with the repository to allow different versions of the code to be rebuilt and executed.

The students must highlight interesting or troublesome issues encountered during each milestone. The students must clearly discuss how these were resolved.

Students must clearly discuss the methods used to ensure the program code is robust and functions. Students must demonstrate different quality assurance methods, for example, printing to screen, or tracing under debug.

Overall, the evidence must be logically structured, technically accurate, and easy to understand.

For pass standard, students must provide evidence of the approach to software development employed during the development of the software solution.

The students must produce evidence of interpreting some requirements.

They must produce evidence of functional milestones, based on requirements, reached during the development. Students must keep versions of their code in a logical structure.

Students must provide details that reflect the structure of the saved code, so that the state of the code can be found at different times across the development.

The students must discuss issues encountered during development. They must explain how these issues were resolved.

Students must describe ways they used to ensure that the code works, such as testing.

Overall, the evidence must be logically structured. It may be basic in parts, for example, there may be a limited range of code versions. The evidence may contain minor inaccuracies or omissions but will still generally convey the information.

Links to other units

The assessment for this unit allows students to draw on knowledge and understanding developed from *Unit 1: Programming Fundamentals*.

5 Planning your programme

Supporting you in planning and implementing your programme

There will be lots of free teaching and learning support to help you deliver the new qualifications, including:

- Our Planning and Teaching Guide will help you to plan how to deliver the content and assessments that make up the Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Certificate) qualification. It also highlights opportunities to develop the transferable skills identified within the units in this specification.
- Sample Assessment materials are available for each external unit to help you to plan and prepare for assessments.
- Our mapping document highlights key differences between the new qualification and Pearson BTEC Level 3 National Certificate in Computing (603/0446/7), which this qualification replaces.

Is there a student entry requirement?

As a centre it is your responsibility to ensure that students who are recruited have a reasonable expectation of success on the programme. There are no formal entry requirements but we expect students to have qualifications at or equivalent to Level 2.

Students are most likely to succeed if they have:

- five GCSEs at good grades, and/or
- BTEC qualification(s) at Level 2
- achievement in English and mathematics through GCSE or Functional Skills.

Students may demonstrate ability to succeed in various ways. For example, students may have relevant work experience or specific aptitude shown through diagnostic tests or non-educational experience.

6 Understanding the qualification grade

Awarding and reporting for the qualification

This section explains the rules that we apply in awarding a qualification and in providing an overall qualification grade for each student. It shows how all the qualifications in this sector are graded.

The awarding and certification of these qualifications will comply with regulatory requirements.

Eligibility for an award

In order to be awarded a qualification, a student must:

- complete and **have an outcome** (D, M, P, N or U) for all units within a valid combination
- achieve the **minimum number of points** at a grade threshold.

Award of the qualification grade

The final grade awarded for a qualification represents an aggregation of a student's performance across the qualification. As the qualification grade is an aggregate of the total performance, there is some element of compensation in that a higher performance in some units may be balanced by a lower outcome in others.

The Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Certificate) is a Level 3 qualification and is awarded at the grade range shown in the table below.

Qualification	Available grade range
Certificate	P to D*

The *Award of qualification grade* table, shown further on in this section, shows the minimum thresholds for calculating these grades. The table will be kept under review over the lifetime of the qualification. The most up-to-date table will be issued on our website.

Pearson will monitor the qualification standard and reserves the right to make appropriate adjustments.

Students who do not meet the minimum requirements for a qualification grade to be awarded will be recorded as Unclassified (U) and will not be certificated. They may receive a Notification of Performance for individual units. The *Information Manual* gives full information.

Points available for internal units

The table below shows the number of **points** available for internal units. For each internal unit, points are allocated depending on the grade awarded.

Grade	Unit size (60 GLH)
U	0
Pass	6
Merit	10
Distinction	16

Points available for external units

Raw marks from the external units will be awarded **points** based on performance in the assessment. The table below shows the **minimum number of points** available for each grade in the external units.

Grade	Unit size (120 GLH)
U	0
Near Pass	8
Pass	12
Merit	20
Distinction	32

Pearson will automatically calculate the points for each external unit once the external assessment has been marked and grade boundaries have been set. For more details about how we set grade boundaries in the external assessment please go to our website.

Claiming the qualification grade

Subject to eligibility, Pearson will automatically calculate the qualification grade for your students when the internal unit grades are submitted and the qualification claim is made. Students will be awarded qualification grades for achieving the sufficient number of points (with valid combinations) within the ranges shown in the relevant *Award of qualification grade* table for the cohort.

Award of qualification grade

Applicable for registration from August 2026.

Certificate (180 GLH)

Grade	Points threshold
U	0
Pass	18
Merit	26
Distinction	37
Distinction*	45

The table is subject to review over the lifetime of the qualification. The most up-to-date version will be issued on our website.

Example grading table for Pearson Level 3 Alternative Academic Qualification BTEC National in Computing (Certificate)

Example of a grading table and how a qualification grade is awarded

Unit number	GLH	Type (Int/Ext)	Grade	Unit points
1	120	Ext	Merit	23
2	60	Int	U	0
TOTAL	**180**	–	**Pass**	**23**

Appendix 1 Glossary of terms used for internally-assessed units

Term	Definition
Adequate	Student work is satisfactory or acceptable in quality and quantity.
Analyse	Students break the issue/situation down into the key elements and show their understanding of the issues/situation applied to the scenario/context. Responses would be significantly beyond generic.
Apply/use/employ	Students implement a method, technique, process or approach in an activity.
Assess	Students give careful consideration to all the factors or events that apply, identify which are the most important or relevant and make a judgement on the importance of the factors.
Carry out	Students demonstrate skills through practical activities, in line with certain requirements.
Clear/ly	The qualities required are well demonstrated, unambiguous and beyond a basic level.
Coherent	Student intentions are clear, logically structured and can be interpreted by others.
Compare	Students show knowledge and understanding by identifying the main factors relating to two or more items/situations or aspects of a subject that is extended with the required explanations, e.g, similarities/differences, advantages/disadvantages, impacts.
Comprehensive	Used to describe either scope or depth, e.g. • Student work is well developed and thorough covering all aspects/information in terms of both depth and breadth Or: • Students demonstrate in-depth and accurate understanding of the aspects being assessed.
Confident	Student work demonstrates well-developed and secure application of skills or processes that are significantly beyond a basic level.
Consistent	Students demonstrate reliable and constant practice that maintains a set standard.
Create/produce	Students generate an idea/outcome to specific criteria.
Demonstrate	Students carry out and apply knowledge, understanding and/or skills in a practical situation.
Describe	Students provide an account of something, or highlight a number of key features of a given topic or process that shows a level of understanding.
Detailed	Students cover most if not all of the expected requirements and demonstrate a high level of understanding.

Term	Definition
Develop	Students apply a process of improving/progressing skills, concepts or work in order to produce outcomes.
Discuss	An issue, situation, process will be presented and the student will need to break the issue/situation/process down into the key elements, show their understanding of the issues/situation/process applied to the scenario/context (so generic answers are not acceptable), and show interrelationship in their answers.
Effective	Students demonstrate skills or provide outcomes that are well developed with a range of proficient qualities and that achieves objectives
Evaluate	Students consider various aspects of a subject's qualities in relation to its context such as: strengths or weaknesses, advantages or disadvantages, pros or cons. They will come to a judgement supported by evidence which will often be in the form of a conclusion.
Examine	Students demonstrate an ability to thoroughly inspect something in order to determine its qualities beyond a basic exploration.
Explain	Students can give an insight into the topic showing some level of understanding by providing reasons or examples.
Explore	Students undertake practical research or investigation to develop their skills or understanding of the topic/activity.
Implement	Students take actions or measures to put something into effect.
Investigate	Students perform a systematic inquiry into a topic using research skills, usually to demonstrate their understanding of a topic.
Justify	Students give relevant and logical reasons or evidence to support their actions or opinions.
Partial/some	To an extent, but not completely. Students do not include all of the requirements.
Perform	Students demonstrate a range of skills required to complete a given activity.
Prepare	Students organise a task/equipment/individuals/activities in advance of carrying it out.
Realistic/feasible	Students demonstrate insight into the logistics and manageability of proposals/plans/objectives/ideas and show consideration of the potential to achieve the outcomes.
Refine/optimise	Students make considered improvements to outcomes.
Review	Students consider evidence in order to make judgements about the qualities.
Understand	Students demonstrate insight or ability to interpret a subject.
Undertake	Students demonstrate skills through practical activities, often referring to given processes or techniques.

Appendix 2 Transferable Skills framework

Code = transferable skill initials-skill cluster initials

Managing yourself

Code	Skill cluster	Performance Descriptor
MY-TPR	Taking personal responsibility	• Demonstrates understanding of their role and responsibilities and the expected standards of behaviour. • Demonstrates compliance with codes of conduct and ways of working. • Makes use of available resources to complete tasks. • Manages their time to meet deadlines and the required standards. • Demonstrates accountability for their decisions or actions.
MY-PS&R	Personal strengths and resilience	• Identifies own personal strengths and demonstrates the ability to utilise/ these in relevant areas. • Demonstrates the ability to adapt own mindset and actions to changing situations or factors. • Uses challenges as learning opportunities.

Code	Skill cluster	Performance Descriptor
MY-COP	Career orientation planning	• Undertakes research to understand the types of roles in the sector in which they could work. • Reviews own career plans against personal strengths and identifies areas for development to support progression into selected careers. • Takes part in sector-related experiences to support career planning.
MY-PGS	Personal goal setting	• Sets SMART goals using relevant evidence and information. • Reviews progress against goals and identifies realistic areas for improvement. • Seeks feedback from others to improve own performance.

Effective learning

Code	Skill cluster	Performance Descriptor
EL-MOL	Managing own learning	• Maintains a focus on own learning objectives when completing a task. • Demonstrates the ability to work independently to complete tasks. • Reviews and applies learning from successful and unsuccessful outcomes to be effective in subsequent tasks.
EL-CL	Continuous learning	• Engages with others to obtain feedback about own learning progress. • Responds positively to feedback on learning progress from others. • Monitors own learning and performance over the short and medium term.
EL-SRS	Secondary research skills	• Define the research topic or question • Uses valid and reliable sources to collate secondary data. • Interprets secondary data and draws valid conclusions. • Produces a reference list and cites sources appropriately.
EL-PRS	Primary research skills	• Define the research topic or question • Carries out primary data collection using appropriate and ethical research methodology. • Interprets primary data to draw valid conclusions

Interpersonal skills

Code	Skill cluster	Performance Descriptor
IS-WC	Written communications	• Produces clear formal written communication using appropriate language and tone to suit purpose.
IS-V&NC	Verbal and non-verbal communications	• Uses verbal communication skills confidently to suit audience and purpose. • Uses body language and non-verbal cues effectively • Uses active listening skills and checks understanding when interacting with others.
IS-T	Teamwork	• Engages positively with team members to understand shared goals and own roles and responsibilities. • Respectfully consider the views of team members and consistently shows courtesy and fairness. • Completes activities in line with agreed role and responsibilities. • Provide support to team members to achieve shared goals.
IS-C&SI	Cultural and social intelligence	• Demonstrates awareness of own cultural and social biases • Demonstrates diversity, tolerance and inclusivity values in their approach to working with others.

Solving problems

Code	Skill cluster	Performance Descriptor
SP-CT	Critical thinking	• Demonstrates understanding of the problem or issue to be addressed • Make use of relevant information to build ideas and arguments • Assesses the importance, relevance and/or credibility of information and ideas • Analyses, interprets and evaluates information to present reasoned conclusions
SP-PS	Problem solving	• Presents a clear definition of the problem • Gathers relevant information to formulate proposed solutions • Selects relevant and significant information to formulate proposed solutions. • Identifies negative and positive implications of proposed solutions. • Presents and justifies selected solutions to problems.
SP-C&I	Creativity and innovation	• Identifies new and relevant ideas to help solve a problem. • Refines ideas into workable solutions based on test results and/or feedback.

Appendix 3 Digital Skills framework

Problem solving

Using digital tools to analyse and solve problems:

Performance descriptor	Unit mapping
Use digital tools and techniques for research, collaboration and resolution of problems.	Unit 1, content areas A, B, C Unit 3, content areas A, B, C Unit 4, content areas A, B
Have up-to-date knowledge of ways that technology is used within a sector.	Unit 3, content area A
Present ideas and finding using digital tools.	Unit 3, content areas B, C Unit 4, content areas A, B
Use digital tools to manipulate data.	Unit 4, content area A

Digital collaboration and communication

Using digital tools to communicate and share information with stakeholders:

Performance descriptor	Unit mapping
Understand and use digital collaboration and communication platforms.	Unit 3, content areas A, B, C
Use collaboration tools to meet with, share and collaborate with customers and colleagues.	n/a

Transacting digitally

Using digital tools to set up accounts and pay for goods/services:

Performance descriptor	Unit mapping
Use online systems to access and update digital records.	n/a
Set-up accounts to complete transactions.	n/a

Digital security

Identify threats and keep digital tools safe:

Performance descriptor	Unit mapping
Understand the types of malware.	n/a
Understand the threats involved in carrying out online activities.	Unit 1, content area D
Protect personal and organisation information and data.	Unit 1, content area D
Keeping systems secure.	n/a

Handling data safely and securely

Follow correct procedures when handling personal and organisational data:

Performance descriptor	Unit mapping
Manage passwords and keep them secure.	n/a
Identify website and services that are secure and insecure.	n/a
Understand the digital policy for a sector.	n/a
Understand the impact of online data.	Unit 1, content area D
Understand copyright and intellectual property.	n/a

Appendix 4 Sustainability framework

Sustainable development goal	Unit mapping
SDG 1: No poverty	
SDG 2: Zero hunger	
SDG 3: Good health and wellbeing	
SDG 4: Quality education	
SDG 5: Gender equality	
SDG 6: Clean water and sanitation	
SDG 7: Affordable and clean energy	
SDG 8: Decent work and economic growth	Unit 1, content area D Unit 3, learning aim A
SDG 9: Industry, innovation and infrastructure	Unit 3, learning aims A, B
SDG 10: Reduced inequalities	Unit 1, content area D Unit 3, learning aim A
SDG 11: Sustainable cities and communities	
SDG 12: Responsible consumption and production	
SDG 13: Climate action	
SDG 14: Life below water	
SDG15: Life on land	
SDG 16: Peace, justice and strong institutions	Unit 1, content area D
SDG 17: Partnerships for the goals	

Pearson
BTEC

Publication code:
VQ000487