Education & Skills
Funding Agency

T Level Technical Qualification in Digital: Digital Production, Design and Development

The table below maps the content overlap between the T Level Qualification in Digital Production Design and Development, the BTEC National In Computing, the BTEC Mathematics for IT Practitioners and the GCE AS and A level subject content for mathematics.

All the T Level content is mandatory. BTEC offers a mandatory and optional content structure.

T level students will need to undertake a variety of assessment types such as those that take place in Higher Education for Health related courses including examinations and controlled assessments.

T Level Core assessment is an externally set written exam(s) and an employer set project: both sets of exams assess students' knowledge, understanding and application of contexts, theories and principles relating to the core content in the specification The written exams assess route and pathway knowledge through 'unseen' examination (which samples content), meaning breadth can be assessed at appropriate level 3 depth, whilst limiting the overall duration of assessment. The written exam structure will provide students with relevant exam and revision skills for HE. The employer set project: is a more substantial project based assessment set by employers through the awarding organisation, and will develop their critical thinking and problem solving skills. The project will draw upon knowledge and understanding from across the core content synoptically, and will allow learners to effectively respond to a 'brief'. All science elements are assessed.

The occupational specialism (**Section 2** below) is also externally assessed through a synoptic project.

BTEC assessment is external, internal and synoptic. External and internal assessment is linked to a specific unit.

Published: February 2022

Mathematics/Computing Topics			
Specification content areas	Specification content by unit	Unit content	specification content by section
T Level[1]	**BTEC in Computing[2]**	**BTEC in IT (QCF)[3]** **Mathematics for IT Practitioners**	**A Level[4]**
1. Core Learning	**(M = mandatory, O = optional)**	**(M = mandatory, O = optional)**	**Sections/Overarching themes**
Problem Solving Computational Thinking Top-down, bottom-up and modularisation approaches to solve problems Problem decomposition Pattern recognition methodology Abstraction methodologies Algorithms for computer programming	**Principles of Computer Science (U1 - M)** Computational thinking Decomposition Pattern recognition Pattern generalisation and abstraction Algorithm design Methods and techniques used to develop algorithms	**Mathematics for IT Practitioners (Unit 26 – O)** Recursion: series eg Fibonacci, factorial, natural numbers; termination condition; recursive algorithms eg factorial, quicksort, binary search Calculating factorials and using search and sort programmes	**Mathematical problem solving (A Level OT 2)** Recognise mathematical structure in a situation and simplify and abstract appropriately to enable problems to be solved Concept of a mathematical problem-solving cycle Extract information from diagrams and construct mathematical diagrams to solve problems

[1] T Level Technical Qualification in Digital Production, Design and Development delivered by Pearson (603/5832/4)
[2] BTEC Nationals | Computing (2016) | Pearson qualifications
[3] BTEC Nationals | Information Technology (2010) | Pearson qualifications
[4] AS and A level maths - GOV.UK (www.gov.uk)

Definition of an algorithm Express an algorithm using flowcharts and pseudocode Write algorithms that make use of programming constructs (sequence, selection, iteration) Purposes of given algorithms Determination of correct output Identify and correct errors	Structured English (pseudocode) Flowcharts using standard symbols Common/standard algorithms Sorting Searching Other standard algorithms Stacks and queues		**Pearson Further Mathematics option: Decision 1[5]** Implementation of an algorithm given by a flow chart or text The efficiency of an algorithm The order of an algorithm i Bin packing, bubble sort and quick sort Algorithm for finding the critical path Construction of Gantt (cascade) charts.
Programming: Program data Data types and use Declare and use constants and variables that use specific data types Data structures Program variables Operators	**Programming (U1 – M)** Handling data within a program Defining and declaring constants and variables Managing variables Arithmetic operations Mathematical operators: + – / (DIV) *, %/MOD/modulo/rem Relational operators (=,<, >, <=, >=) Boolean operators (NOT, AND, OR) Date/time	**Mathematics for IT Practitioners (Unit 26 – O)** Sequences and series and probability Sequences and series: nth term of a sequence; generation of recurrence relationship; arithmetic and geometric sequences and series; sum to n terms of an arithmetic and geometric series; sum to infinity of a geometric series; Σ notation	**Mathematical Modelling (A Level OT 3)** Translate a situation in context into a mathematical model Use a mathematical model with suitable inputs to engage with and explore situations Interpret the outputs of a mathematical model in the context of the original situation Understand that a mathematical model can be refined by considering its outputs

Mathematical operators in program code and algorithms (add, subtract, divide, multiply, integer division, modulus) Relational operators Boolean operators File Handling Input and output of data using text files Program Structure Sequence, selection (branching) and iteration Write, interpret, and debug code that makes use of sequence Write, interpret and debug code that makes use of selection (branching) (IF, THEN, ELSE, ELSEIF (ELIF), CASE) Write, interpret and debug code that makes use of iteration	Built-in functions (functions provided within programming languages to perform specific tasks to process data) Validating data Control structures Procedural programming Structure Control Structures Object-orientated programming Event driven programming Coding for the web Translation **Object-oriented programming (U16 – O)** performance, safety and security Computational thinking (mathematical and logical processes that underpin the design of object-oriented programs e.g algorithms, Boolean algebra)	Probability: events e.g. union, intersection, complementary, mutually exclusive, independent; space diagrams e.g. sum of scores when two dice are thrown; visualising events using Venn diagrams; tree diagrams Number Systems Number systems: binary, octal, denary and hexadecimal; conversion between number systems; basic operations e.g. addition, division, multiplication, subtraction on number systems Applications: e.g. ASCII code (binary), MIME (hex), file permissions in Unix (octal); IP addressing v4 and v6; subnet addressing; subnet masking; class A, B and C addresses; Classless Inter Domain Routing (CIDR) **Software Design and Development (U6 - O)** Programming languages	and simplifying assumptions; evaluate whether the model is appropriate

Call functions and procedures Searching and sorting algorithms Built-in Functions Analysis of pre-written code (Python) User-written code Validation and Error Handling Validation techniques Testing Testing procedures for all system components Quality assurance methodologies Automated and functional testing tools Root cause analysis Test plan structuring	Designing object-oriented programs Developing object orientated solutions using programming languages, e.g. C++, Java®, Python™, Ruby® Constructs and techniques Arithmetic operators: [+, –, *, /, %] Logical operators: [!=,<, <=, >, >=, AND, OR, XOR, true, false] Data types, e.g. char, string, integer, real, Boolean Testing U14 Computer Games Development(O) U15 Website Development(O) U17 Mobile Apps Development (O) All require development of a product using programming language(s), tools and/or development environments	Features: sequence; selection e.g. case, if … then … else; iteration e.g. repeat – until, while … do; variables e.g. naming conventions, local and global variables, logical operators; assignment statements; input statements; output statements Data types: text; integer; floating point; byte; date; Boolean; other e.g. char, smallint	

Emerging Issues and Impact of Digital			
Legislation Understand data and risk			
Business Environment Estimate, calculate and spot errors Understand data and risk Communicate using mathematics Optimise work processes	Systems Analysis and Design (U22-O) software development models, systems analysis tools and techniques and their suitability for modelling business processes Develop a design for a computing system to meet an organisation's needs		
Data Data and information in organisations Differences and links between data, information, and knowledge	**Data structures (U1 – M)** **How data is represented by computer systems (U2 – M)** Number systems Text representation Image representation	**Mathematics for IT Practitioners (Unit 26 – O)** Matrix Methods Matrices to represent ordered data; relationship with computer program variable arrays; index notation	**Statistical sampling (A Level K)** Population and sample Sampling techniques Mathematical and statistical graphing tools and spreadsheets Large data set(s) in context

Need for data and information and how each is used How data is generated Data Formats Data types (date, integer, real, character, string, Boolean) Common forms of data format (JSON, fixed-width text file, CSV, ASCII, XML) File-based and directory-based structures Data Systems Features and functions of data systems Business information tools (analysis) Data models Data Management How data is gathered, entered and maintained Data analysis tools	How data is organised on computer systems Data structures Indices and matrices How data is transmitted by computer systems Concepts, processes and implications of data transmission in and between computer systems Error detection Error correction The use of logic and data flow in computer systems Boolean logic Flow charts and system diagrams **Relational Database Development (U18-O)** Relational database management systems (Relational algebra sets) Manipulating data structures and data in	Operations: add, subtract, scalar multiplication; multiply two matrices; inverse; transpose Techniques: solving simultaneous linear equations; vector transformation and rotation; maps and graphs Representing data: comparing data sets using back-to-back stem and leaf diagrams, mean; median; mode; interquartile ranges; histograms; variance; standard deviation Gathering data: methods of gathering quantity data e.g. measurements, questionnaires, surveys; extraction of required information from raw data; limitations of data gathered Interpreting data: e.g. analysing summary data, proving hypotheses, identifying trends and patterns	Use of spreadsheets or specialist statistical packages to explore data set(s) Analyse a subset or features of data Use data to investigate questions arising in real contexts **Data presentation and interpretation (A Level L)** Interpret diagrams/histograms Scatter diagrams and regression lines Correlation Central tendency and variation Recognise and interpret possible outliers in data sets Clean data, including dealing with missing data, errors and outliers] **Statistical Distributions (A Level N)** Probability distributions **Statistical hypothesis testing (A Level O)** Null hypothesis, alternative hypothesis, significance level, test statistic, 1-tail test, 2-tail test, critical

Metadata classification Data/access entitlements/permissions Management Platforms to access and manage data (API) Concepts of data at rest, data in use and data in motion	relational databases Normalisation Relational database design techniques and processes Design documentation Reviewing and refining designs Develop a relational database solution to meet client requirements		value, critical region, acceptance region, p-value. Correlation Coefficient Level of significance Statistical hypothesis test for mean of normal distribution
Digital Environments Physical Environments Networks Virtual Environments Cloud Environments Resilience of Environments Use rules and formulae Understand data and risk Cost a project Optimise work processes	**Fundamentals of Computer Systems (U2 -M)** Computer hardware in a computer system Computer software in a computer system Data processing by computer systems Computer architecture		
Security Risks Understand data and risk			

Core Project (applies the above core learning and is employer set) Measure with precision Estimate, calculate and spot errors Communicate using mathematics Cost a project Optimise work processes Process data Interpret and represent with mathematical diagrams Gantt Charts			
2. Occupational specialism Digital production, design, and development	**Units Linked to Projects**		
A. Digitally based Project			
1. Software development life cycle Research and familiarisation Planning and requirement analysis	Planning and Management of Computing Projects (U3 – M) Project management concepts Costs and timescales		

Perform user analysis Design a product Develop and test the product Deploy/implement the product Maintenance Roles and Responsibilities of the Digital Team in the software lifecycle Project Methodologies (Agile, Scaled Agile, Waterfall, RAD, LEAN) Impact of emerging technologies Personal training needs to boost performance 2. Ethical principles, risk, legal Requirements when developing software Legal and regulatory considerations Risk identification/software development 3. Sources of Knowledge	Quality and deliverables Risk Benefits Project lifecycle Professionalism Starting a Computer Project Interpret the business case Stakeholders Identifying assumptions and constraints Project Initiation Document (PID) Project Planning Scheduling and milestones Resources and budgeting Risk management strategy Quality management Communications Executing and monitoring a project Waterfall model		

Find new sources, evaluate reliability Select and use techniques to obtain qualitative and quantitative data to be able to evaluate software solutions 4. Design Common design approaches Data flows Test driven development Data modelling Platforms used for source code and content management Design a software solution 5. solutions in a social and collaborative environment Collaborative techniques Technologies 6. Implement a solution using at least two appropriate languages	Monitoring and tracking progress Managing issues Change management Implementation strategy Project closure and post-project review Closing a live project Review of project success Software Design and Development Project (U4 – M) Python (3.4 or a later version) or C family programming languages Software development life cycle Stages of software development Flow chart and use of standard symbol conventions Structured English (pseudocode) Test data; tests and test data to produce test plans for an identified solution		

Front and back end solutions (Python, C C# and C++, Javascript frameworks (Angular, React), Java, Go, Ruby, PHP, SQL) Interfaces that apply user experience (UX) design principles Connect code to data sources as part of a software project Select and use deployment methods for a software project 7. Test a software solution Select and apply functional, non-functional and front-end testing Select and apply testing techniques Select appropriate tests and test data to test the functionality of software 8. Change, maintain and support software	Design concepts Code readability Handling data in a program Arithmetic operations Built-in functions Validating data Control structures Data structures Evaluating a software development project; design, testing, software, Systems Methodology (U23-O) Investigate the principles of systems methodology and systems techniques used to solve computing problems Apply systems methodology tools and techniques to identify and solve a computing problem Review a solution to a computing problem.		

Changing nature of digital products and the factors that drive change Stages involved in the software change management process Maintain code as part of a larger team Support software users			
3.	**Additional Content**		**Additional Content**
	U7 IT Systems Security and Encryption (M)		
	U8 Business Applications of Social Media (M)		
	U9 The Impact of Computing(M)		
	U10 Human-computer Interaction (O)		
	U11 Digital Graphics and Animation(O)		
	U12 Digital Audio(O)		
	U13 Digital Video(O)		
	U19: Computer Networking		
	U20: Managing and Supporting Systems		

	U21: Virtualisation		
			Proof (A Level A)
			Algebra and functions (A Level B)
			Coordinate geometry in the (x,y) plane (A Level C)
			Sequences and series (A Level D)
			Trigonometry (A Level E)
			Exponentials and logarithms (A Level F)
			Differentiation (A Level G)
			Integration (A Level H)
			Numerical methods (A Level I)
			Vectors (A Level J)
			Probability (A Level M)
			Quantities and units in mechanics (A Level P)
			Kinematics (A Level Q)
			Forces and Newton’s laws (A Level R)
			Moments (A Level S)