GCE

AS and A Level Specification

Computing

AS exams 2009 onwards
A2 exams 2010 onwards

Contents

1 Introduction

1.1 Why choose AQA?

It's a fact that AQA is the UK's favourite exam board and more students receive their academic qualifications from AQA than from any other board. But why does AQA continue to be so popular?

- **Specifications**
  Ours are designed to the highest standards, so teachers, students and their parents can be confident that an AQA award provides an accurate measure of a student's achievements. And the assessment structures have been designed to achieve a balance between rigour, reliability and demands on candidates.

- **Support**
  AQA runs the most extensive programme of support meetings; free of charge in the first years of a new specification and at a very reasonable cost thereafter. These support meetings explain the specification and suggest practical teaching strategies and approaches that really work.

- **Service**
  We are committed to providing an efficient and effective service and we are at the end of the phone when you need to speak to a person about an important issue. We will always try to resolve issues the first time you contact us but, should that not be possible, we will always come back to you (by telephone, email or letter) and keep working with you to find the solution.

- **Ethics**
  AQA is a registered charity. We have no shareholders to pay. We exist solely for the good of education in the UK. Any surplus income is ploughed back into educational research and our service to you, our customers. We don't profit from education, you do.

If you are an existing customer then we thank you for your support. If you are thinking of moving to AQA then we look forward to welcoming you.

1.2 Why choose Computing?

This specification has been designed for students who wish to go on to higher education courses or employment where knowledge of Computing would be beneficial. One can study Computing and go on to a career in medicine, law, business, politics or any type of science. Several subject areas of the previous AQA GCE Computing specification have been retained, with changes made to bring the content up to date as well as to shift the emphasis in the direction of Computer Science and Computation.

There is a clear distinction between this specification and the GCE ICT and GCE Applied ICT specifications. It has been written to avoid any overlap of subject content. Students following this specification do not need to have any prior knowledge of Computing or ICT.

The course is not about learning to use tools or just training in a programming language. Instead the emphasis is on computational thinking. Computational thinking is a kind of reasoning used by both humans and machines. Thinking computationally is an important life skill. Thinking computationally means using abstraction and decomposition. The study of computation is about what can be computed and how to compute it. Computer Science involves questions that have the potential to change how we view the world. For example, we may be computing with DNA at some stage in the future, with computer circuits made of genes. This leads to the question, does the natural world 'compute'?

Experimental Computer Science can be done with computers whereby we can learn more about the natural world by observing the emergent behaviour of a colony of interacting software agents in a simulation. Computing / Computer Science is about designing new algorithms to solve new problems. In this sense Computer Science is no more about computers than astronomy is about telescopes. Many great challenges lie in the future for Computer Scientists to solve. This course, with its emphasis on abstract thinking, general problem-solving, algorithmic and mathematical reasoning, scientific and engineering-based thinking, is a good foundation for understanding these future challenges.

In the AS specification there are two units. Unit 1 is a practical, on-screen, examination which allows candidates to demonstrate their knowledge of the fundamental principles of the subject, focusing on programming through a problem-solving scenario using pre-release material. Unit 2 focuses on the hardware and software aspects of Computing and the social and economic consequences of Computing.

The A2 specification builds on the content of AS, with Unit 3 focusing on computational thinking, what can be computed, programming and problem-solving including communication and networking. The second unit, Unit 4, is an internally assessed unit, with candidates required to complete a report on a computer-based programmed solution to a problem-solving exercise of their choice.

1.3 How do I start using this specification?

Already using the existing AQA Computing specification?

- Register to receive further information, such as mark schemes, past question papers, details of teacher support meetings, etc, at **http://www.aqa.org.uk/rn/askaqa.php** Information will be available electronically or in print, for your convenience.
- Tell us that you intend to enter candidates. Then we can make sure that you receive all the material you need for the examinations. This is particularly important where examination material is issued before the final entry deadline. You can let us know by completing the appropriate Intention to Enter and Estimated Entry forms. We will send copies to your Exams Officer and they are also available on our website **http://www.aqa.org.uk/admin/p_entries.html**

Not using the AQA specification currently?

- Almost all centres in England and Wales use AQA or have used AQA in the past and are approved AQA centres. A small minority are not. If your centre is new to AQA, please contact our centre approval team at **centreapproval@aqa.org.uk**

1.4 How can I find out more?

Ask AQA

You have 24-hour access to useful information and answers to the most commonly-asked questions at **http://www.aqa.org.uk/rn/askaqa.php**

If the answer to your question is not available, you can submit a query for our team. Our target response time is one day.

Teacher Support

Details of the full range of current Teacher Support meetings are available on our website at **http://www.aqa.org.uk/support/teachers.html**

There is also a link to our fast and convenient online booking system for Teacher Support meetings at **http://events.aqa.org.uk/ebooking**

If you need to contact the Teacher Support team, you can call us on 01483 477860 or email us at **teachersupport@aqa.org.uk**

2 Specification at a Glance

AS Examinations

AS Award 1511

Unit 1 – COMP1
Problem Solving, Programming, Data Representation and Practical Exercise

60% of AS, 30% of A Level
2 hour on-screen examination
100 marks

Skeleton program and test data released on 1 March on the AQA Website. Candidates use this to answer short questions and to write a program.

Available in June only

Unit 2 – COMP2
Computer Components, The Stored Program Concept and The Internet

40% of AS, 20% of A Level
1 hour written examination
60 marks

Compulsory short answer questions.

Available January and June

A2 Examinations

A Level Award 2511

Unit 3 – COMP3
Problem Solving, Programming, Operating Systems, Databases and Networking

30% of A Level
2 hour 30 minutes written examination
100 marks

Compulsory short and extended answer questions.

Available in June only

Unit 4 – COMP4
The Computing Practical Project

20% of A Level
Coursework, internally assessed, and moderated by AQA.
75 marks

Report written by the candidate documenting a programmed solution to a real problem associated with a user whose realistic needs should be taken into account when specifying, designing and implementing the solution.

Available in June only

AS + A2 = A Level

3 Subject Content

3.1 Unit 1 COMP1 Problem Solving, Programming, Data Representation and Practical Exercise

AQA will support the following programming languages.

- Pascal
- VB
- Java
- C
- PHP
- C#
- Python

This list will be updated as required. Centres will be asked to make known their preferences at the start of the course.

3.1.1 Fundamentals of Problem Solving

Introduction to Principles of Computation	Focus on studying the processes of computation and understanding why and where they are important in Computing. (See the *Teacher Resource Bank* for examples.)
Stages of Problem Solving	Understand the problem. Define the problem. Define boundaries. Plan solution. Check solution.
Top-down Design / Step-wise Refinement	Consider sub-problems and the use of modules.
Decision tables	Determine logical conditions and consequential actions.
Finite state machines with and without outputs • state transition diagrams • state transition tables	Draw and interpret simple state transition diagrams, transition tables.
Algorithm Design	Understand the term *algorithm*. Express the solution to a simple problem as an algorithm using flowcharts, pseudo-code or structured English and the standard constructs: • sequence • assignment • selection • repetition. Hand trace simple algorithms. Convert a simple algorithm from • structured English into pseudo-code, • pseudo-code into high level program code. Understand the standard algorithms: Bubble Sort, Linear Search.

3.1.2 Fundamentals of Programming

Features of Imperative High Level Languages	Illustrate these features for a particular imperative, third-generation language such as Pascal.
Data Types	Use the following appropriately.
• Built-in	Integer, byte, real, boolean, character, string, date/time.
• User-defined	Enumerated, subrange, sets, records, arrays.
The Role of Variables	Recognise the different roles a variable can take: fixed value, stepper, most recent holder, most wanted holder, gatherer, transformation, follower temporary.

Programming Statements • Type Definitions • Variable Declarations • Constant Definitions • Procedure/Function Declarations • Assignment • Iteration • Selection	Use these statement types.
• Procedure and Function calling	Explain the advantages of procedure/functions.
Arithmetic operators including modular arithmetic	+, –, /, x, DIV, MOD
Relational operators	=, <, >, <>, <=, >=
Boolean operators	NOT, AND, OR
Logical bitwise operators	NOT, AND, OR, XOR
Set operators	Union, difference, intersection, membership.
Built-in functions	Arithmetic functions: round, truncation. String handling functions: length, position, substring, concatenation. String conversion functions to/from integer, real, date/time.
Constants and Variables	Explain the advantages of named variables and constants.
Procedure and Function Parameters	Describe the use of parameters to pass data within programs. Understand the different mechanisms for parameter passing: by value and by reference.
Fundamentals of Structured Programming	Understand the structured approach to program design and construction. Construct and use structure tables, structure charts and hierarchy charts when designing programs. Use meaningful identifier names. Use procedures/functions with interfaces. Use procedures that execute a single task. Explain the advantages of the structured approach.
Data Structures	
• One- and Two-Dimensional Arrays	Use arrays in the design of solutions to simple problems.
• Fields, Records and Files	Read/write records from/to a file: csv file or file of records.
Validation	Understand the importance of validation of input data. Program simple validation.

3.1.3 Fundamentals of Data Representation

Bit Patterns in a Computer	Explain the different interpretations that may be associated with a pattern of bits.
Binary number system	
Pure Binary Representation of Denary Integers	Describe the representation of unsigned denary integers in binary. Perform conversion from denary to binary and vice-versa.
Binary Arithmetic	Add two binary numbers and multiply two binary numbers.
Representation of signed integers by Two's Complement	Describe the use of Two's Complement to perform subtraction. Convert a denary integer into Two's Complement and vice versa.
The Concept of Number Bases: Denary, Binary and Hexadecimal	Describe the conversion of a denary integer to hexadecimal form and vice versa. Describe the use of hexadecimal as shorthand for binary.
Integers and Numbers with a Fractional Part	Draw a distinction between integers and numbers with a fractional part in a computer context. Describe how an unsigned denary number with a fractional part is represented in fixed-point form in binary.
Information Coding Schemes • ASCII • Unicode	Describe standard coding systems for coding character data. Differentiate between the character code representation of a denary digit and its pure binary representation.
Error checking and correction	Parity bits, Hamming code.
Gray coding	Describe Gray coding. Explain why and where it is used.
Representing Images, Sound and other data	Describe how bit patterns may represent other forms of data including graphics and sound.
Bitmapped Graphics	Bitmaps: resolution, colour depth and simple bitmap file calculations.
Vector Graphics	Vector graphics: drawing list – objects and their properties. Compare bitmaps to vector graphics; advantages, disadvantages.
Sound files	The need for compression and basic techniques for compression.
Sampled Sound and Nyquist-theorem	Sampling resolution, sampling rate.
Sound Synthesis	Streaming audio.
Analogue and Digital Data Analogue and Digital Signals	Differentiate between analogue and digital data and analogue and digital signals.
Analogue to Digital Converter (ADC)	Describe the principles of operation of an analogue to digital converter.

3.1.4 Systems Development Life Cycle

Analysis	Describe the stages of development of a hardware/software system.
Design	
Implementation	
Testing	Specify the method of testing the programmed solution (dry run testing, black box testing, white box testing). Specify the selection of test data including normal (typical), boundary and erroneous data. Program the solution as per design.
Evaluation	Test the solution using selected test data.

3.2 Unit 2 COMP2 Computer Components, The Stored Program Concept and the Internet

3.2.1 Fundamentals of Computer Systems

Hardware and Software	Understand the relationship between hardware and software and be able to define both.
Classification of Software	Be aware of how software is classified. Be able to explain what is meant by system software and application software. Understand the need for and attributes of different types of software.
System Software	Understand the need for, and functions of, system software: • Operating system software • Utility programs • Library programs • Translator software (Compiler, assembler, interpreter).
Application Software	Describe the different types of application software and the criteria for selecting appropriate software for particular purposes. General purpose application software. Special purpose application software. Bespoke application software.
Generations of Programming Language	Describe machine-code language and assembly language.
First generation • Machine code Second generation • Assembly language	Awareness of the development of programming languages and the limitations of both machine-code and assembly-language programming.
Third generation • Imperative high level language	Explain the term imperative high level language and its relationship to first and second generation languages.
Fourth generation • Declarative language	Explain the term declarative programming language and where and why declarative languages are used.
Types of Program Translator • Assembler • Compiler • Interpreter	Define each type of language translator and describe situations where each would be appropriate.

3.2.2 Fundamental Hardware Elements of Computers

Logic Gates	Construct truth tables for the following gates: NOT, AND, OR, XOR, NAND, NOR. Be familiar with drawing logic diagrams involving one or more of the above gates.
Boolean Algebra	Be familiar with the use of De Morgan's laws and Boolean identities to manipulate and simplify simple Boolean expressions.

3.2.3 Machine Level Architecture

Internal and External Hardware Components of a Computer	Outline the basic internal components of a computer system. (Although questions about specific machines will not be asked it might be useful to base this section on the machines used at the centre.) Understand the need for and means of connection between components. Processor, main memory, address bus, data bus, control bus, I/O controllers and I/O ports, secondary storage, their purpose and how they relate. Know that the processor, system bus and main memory are called the CPU (central processing unit), and that components external to the CPU are called peripherals. An example of a peripheral is secondary storage.
Functional Characteristics of a Processor	Understand the concept of addressable memory. Describe the stored program concept whereby machine code instructions stored in main memory are fetched and executed serially by a processor that performs arithmetic and logical operations.
Structure and Role of the Processor • Arithmetic Logic Unit • Control Unit • Clock • General purpose and dedicated registers	Understand the characteristics of contemporary processors. Explain the role and operation of a processor and its major components. Explain the effect of clock speed, word length and bus width on performance.
Machine code and processor instruction set	The basic machine code operations of Load, Add, Store.
The Fetch–Execute cycle and the role of registers within it	Explain how the Fetch–Execute cycle is used to execute machine code programs including the stages in the cycle with details of registers used. Machine code representation in binary and hexadecimal.

3.2.4 Hardware Devices

Input and Output Devices	Know the main characteristics of contemporary devices (see the *Teacher Resource Bank*) and understand their principles of operation, including methods of error checking (check digit).
Secondary Storage Devices	Explain the need for secondary storage within a computer system, know the main characteristics and understand the principles of operation of contemporary devices (see the *Teacher Resource Bank*). Compare the capacity and speed of access of various media and make a judgement about their suitability for different applications.

3.2.5 The Structure of the Internet

The Internet and its Uses World Wide Web (WWW)	Understand the structure of the Internet, the role of packet switching and routers.
Intranet	Understand the difference between the Internet, the Web and an intranet.
Uniform Resource Locator (URL)	Describe the term URL in the context of Internet working.
Uniform Resource Identifier (URI)	Describe the role of URIs in the context of Internet working.
Domain Names and IP Addresses	Explain the terms *domain name* and *IP address*. Describe how domain names are organised.

Internet registries and Internet registrars	Explain why such services are provided.
Internet Service Providers (ISP)	Understand the role of an ISP.
Domain Name Server (DNS)	Understand the purpose of Domain Name Server.
The Client–Server Model	Be familiar with the client–server model.
Common Standard Protocols: • TCP/IP • FTP • HTTP • TELNET • POP3, SMTP • Well-known ports • Client ports • HTTPS	Describe the role of the four layers of the TCP/IP protocol stack, including sockets. Be familiar with • Telnet server for remote management of a server • Web server to retrieve web pages in text form • E-mail server to read and send e-mail • FTP client software and an FTP server to transfer files using anonymous and non-anonymous access. Understand the role of a web browser in retrieving web pages and web page resources and rendering these accordingly.

3

3.2.6 Web page design

Web page construction	Have practical experience of creating simple web pages containing hyperlinks using the tags listed in the *Teacher Resource Bank*.
HTML & style sheets	Know that HTML is used for structure only and that style sheets are used for style and layout of web pages (see the *Teacher Resource Bank* for list of style sheet type, class and ID selectors, properties and values).

3.2.7 Consequences of Uses of Computing

Legal and Ethical Issues	Discuss issues of ownership of information and programs, and the protection of data. Consider current legal controls which specifically refer to computerised data and programs, and the implications of current legislation (see the *Teacher Resource Bank*). Hacking. Consider how digital rights can be managed. Code of Conduct.
Economic and Social Issues	Discuss the social consequences of current uses of computing. Be aware of emerging technologies and appreciate their potential impact on society. Robotics: • What are machines good and bad at, in comparison to humans? • What can this tell us about the way that the human mind works? • What can we learn from machines? • What are the limitations of using machines as tools?

3.3 Unit 3 COMP3 Problem Solving, Programming, Operating Systems, Databases and Networking

3.3.1 Problem Solving

Information hiding Abstraction	Be familiar with the concept of abstraction as the modelling of a complex system, only including the essential details.
Comparing algorithms	Understand that algorithms can be compared by expressing their complexity as a function relative to the size of the problem. Understand that some algorithms are more efficient time-wise than other algorithms. Understand that some algorithms are more space-efficient than other algorithms.
Big-O notation Order of complexity	Linear time, polynomial time, exponential time.
Intractable problems	Be aware that software and hardware present limitations to solving problems.
Solvable and non-solvable	Understand that some algorithms involve too many steps to be solvable in a reasonable time by computer (The Travelling Salesman Problem) in the general case but that a heuristic approach can provide a solution for specific cases.
Halting problem	The unsolvable problem of determining whether any program will eventually stop given particular input.
Turing Machine (level 1) Universal Machine	The abstract model of the Turing Machine and the Universal Machine.
Finite state machines with and without output	Draw and interpret state transition diagrams for finite state machines with no output (automata) and with output.
Regular expressions	Form simple regular expressions for string manipulation and matching.
Backus-Naur Form (BNF) / syntax diagrams	Be able to check language syntax by referring to BNF or syntax diagrams.
Reverse Polish notation	Be able to convert simple infix notation to reverse Polish notation and vice versa.

3.3.2 Programming Concepts

Programming Paradigms

Structured programming techniques Procedural-oriented programming Event-driven programming Object-oriented programming	Understand the need for and characteristics of a variety of programming paradigms. Be familiar with the concept of an object class, an object, instantiation, encapsulation, inheritance. Practical experience of programming using objects to model a simple problem.
Recursive Techniques	Illustrate the use of recursive techniques in programming languages.
Abstract Data Types / Data Structures Lists Queues • linear, circular and priority Stacks Pointers Linked lists	Be familiar with the concept of a list, a tree, a queue, a stack, a pointer and be familiar with the data structures and methods for representing these when a programming language does not support these structures as built-in types. Distinguish between static and dynamic structures and compare their uses. Use of free memory, the heap and pointers.

Graphs Trees Rooted trees	Be aware of a graph as a data structure to represent more complex relationships. Explain the terms graph, labelled graph, unlabelled graph, vertex, edge, digraph and arc. Know how an adjacency matrix and an adjacency list may be used to represent a graph. Compare the use of each. A tree is a connected undirected graph with no cycles. A rooted tree is a tree in which one vertex has been designed as the root and every edge is directed away from the root.
Standard Algorithms Binary search Insertion sort Hashing Binary tree search Tree traversal algorithms for a binary tree Stack, queue and list operations Simple graph traversal algorithms Creating and maintaining linked lists	Describe, using algorithms or programming examples, the methods used by programmers when manipulating structured data. Discuss methods used in relation to efficiency criteria. Be aware of the link between choice of algorithms and volume of data to be processed. Describe the creation and maintenance of data within lists, trees, stacks, queues and linked lists. Simple graph traversal algorithms which traverse every edge and vertex, eg getting out of an arbitrary maze. The traversal algorithm should maintain, for each vertex, two flags: • Discovered – have we encountered this vertex before? • Completely explored – have we finished exploring this vertex yet? Explain the technique of hashing and its application.
Simulations	Know that computer simulations can represent real and imaginary situations. Know that simulations allow users to study or try things that would be difficult or impossible to do in real life. Be familiar with simple simulation techniques involving queues.

3.3.3 Real Numbers

Floating point numbers	Describe the format of floating point numbers, including the concept of mantissa and exponent and the need for normalisation.
Significant digits Precision Rounding errors (absolute errors, relative errors) Cancellation Error Underflow Overflow	Explain the limitations of representing real numbers in a computer system, and how errors occur.

3.3.4 Operating Systems

Role of an Operating System Provision of a virtual machine Resource management	Understand that the role of the operating system is to hide the complexities of the hardware from the user. In addition, it manages the hardware resources in order to provide for an orderly and controlled allocation of the processors, memories and I/O devices among the various processes competing for them.
Operating System Classification • Interactive • Real time • Network • Device • Embedded • Desktop • Server	Define interactive, real time and network operating systems and explain their operational characteristics. Compare and contrast device operating systems (smartphone, PDA), embedded operating systems (car, aircraft), desktop operating systems, server operating systems.

3.3.5 Databases

Conceptual data model Entity Relationship modelling	Produce a data model from the given data requirements for a simple scenario involving two or three entities.
Database Design and Normalisation techniques	Be able to normalise relations to Third Normal Form.
Relational Databases	Explain the concept of a relational database. Define the terms: attribute, primary key, composite key, foreign key, referential integrity.
Querying a Database • Structured Query Language (SQL)	Use SQL to retrieve, update, insert and delete data from several tables of a relational database (*see Teacher Resource Bank*).
Data Definition Language (DDL)	Explain the term DDL. Use DDL to define a database (*see Teacher Resource Bank* for commands/statements).

3.3.6 Communication and Networking

Communication Methods • Serial data transmission • Parallel data transmission	Define both serial and parallel transmission methods and describe where they are used. Consider the effect of distance on the transmission of data.
Baud, bit rate, bandwidth, latency	Define these terms. Differentiate between baud and bit rate. Consider the relationship between bit rate and bandwidth.
Asynchronous data transmission	Define asynchronous data transmission.
Start and stop bits	Describe the purpose of start and stop bits in asynchronous data transmission.
Odd and even parity	Explain the use of parity checks.
Handshaking	Explain what is handshaking in the context of data transmission.
Protocol	Explain what is meant by a protocol in this context.
Baseband Broadband	Define each mode of network operation and describe where each is appropriate.

Networks

Local Area Networks Wide Area Networks	Contrast wide area and local area networks.
Network adapter	
Topology	Define the term topology.
• Bus	Describe in general terms the operation of these networks.
• Star	Compare the advantages and disadvantages of each. Candidates should be able to compare local area networking with standalone operation.
Network Segment	Define the term and explain why local-area networks based on a bus topology are segmented.
Peer-to-peer networking Server-based networking	Explain these terms and describe situations where these might be used.
Thin client computing Web services	Compare and contrast thin client computing (software as a service, AJAX, Web 2.0, etc) vs rich client computing (client-server, peer-to-peer), web services as examples of 'systems architectures'.

Wireless networking • Wi-Fi • Bluetooth	Radio-based LAN protocols for connecting mobile/portable devices.
Inter-networking	Explain the meaning of the term inter-networking.
Routers particular, Gateways	Define these and consider where and why they are used. In consider how routing is achieved across the Internet and how local area networks are connected to the Internet via gateways.
Server-side scripting	Common Gateway Interface (CGI) Server-side scripting – the basis of dynamic web page content. Practical experience of writing simple server-side scripts. Accessing data from a DBMS using server-side scripts.
Internet Security • Firewalls • Encryption • Digital Signatures and Digital Certificates • Virus detection	Packet filtering. Proxy Server. Private/Public key encryption. Digital Certificates and digital signatures – how they are obtained and used. Discuss worms, spam, phishing, pharming as well as viruses, also vulnerabilities that these exploit and how to address them through improved code quality, monitoring, protection.
Computer Security Procedures	Authentication, Authorisation, Accounting.

3.4 Unit 4 COMP4 The Computing Practical Project

The Project provides an opportunity to test the candidates' understanding of the connections between the different areas of computing. It allows candidates to demonstrate their knowledge and understanding of the systems development life cycle. The skills to be demonstrated include analysis, design, construction/implementation, testing and evaluation of a substantial computer-based task undertaken over an extended period. The report should summarise the work carried out by the candidate.

Projects should be selected which allow candidates to demonstrate practical and problem-solving skills, as well as the techniques of documentation and system testing.

The project topic could involve a computer solution to:

- a data-processing problem of an organisation
- a scientific or mathematical problem
- a simulation of a real-life situation
- a computer-aided learning system
- a control system / robotics.

Candidates should investigate a real problem associated with a user whose realistic needs should be taken into account when designing the solution. The object of the project is to produce a complete working solution to a problem. Technical competence should be demonstrated in implementing the solution by writing a suitable program or suite of programs.

Although it is envisaged that the candidate will develop a complete working solution, the project report need only contain carefully selected samples of evidence in order to demonstrate each skill.

The Project is centre-assessed and externally moderated.

Further details about what is defined as 'complex', 'adequate', 'limited' and 'simple' problems are provided in the support and exemplar materials available for centres and the *Teacher Resource Bank*.

Systems Development	
Analysis	Evaluate the possible need for development of a computer-based solution to a problem. Judge the feasibility of a computer-based solution to a problem. Derive the user, data and processing requirements of a system including a consideration of the human aspects and physical environment. Specify and document the data and processing requirements for a computer-based solution to a problem.
Requirements Analysis • fact-finding techniques • research • produce system objectives/specification	Interview, observation, survey, examination of paperwork. Research into possible methods of solution using reference texts of published appropriate algorithms.
Data flow diagrams (to level 1)	Specify and document the data flow and the processing requirements for a system.
Entity Relationship data modelling	Establish the data requirements of the system and produce a full conceptual data model.
E-R diagrams	Document any constraints and assumptions.
Data dictionary	Produce a preliminary data dictionary.
Object-analysis diagrams	Object-analysis diagrams cover association diagrams, inheritance diagrams, aggregation diagrams, class definitions of class attributes and operations.
Volumetrics	Data volumes.

Design

	Specify and document: • the method of solving the problem including, where appropriate, evaluation of alternative proposals • the functions of the constituent parts of the system • the inter-relationships between the various parts of the system • the selection of an appropriate hardware and software configuration • the algorithms, data types, data structures and any other requirements of the solution • the effectiveness of the proposed solution in meeting the requirements of the problem.
System flowcharts Algorithm design Object-oriented design Hierarchy charts Structure charts Pseudo-code Relations	Specify and document a design that meets the requirements of a real problem in terms of hardware and software, using system flowcharts, structure charts, hierarchy charts, pseudo-code, relations, class and object diagrams as appropriate.
Prototyping	Consider the impact of prototyping on the design and development process.
Human–Computer-Interface design (HCI)	Consider: • The user – type and context (eg business or home) • User needs, usability • Input/output devices – choice of and appropriateness of • Dialogues – to be relevant, simple and clear • Colour – use of and combinations of colour • Icon usage and presentation – 3D effects and depth perception. Provide: • Feedback • Exits – clearly marked • On-line help • Shortcuts • Helpful error messages. Produce good HCI design that prevents errors occurring and minimises the amount a user has to remember.
Testing Strategies for the Development of a System	Identify suitable test strategies. Top down, Bottom up; Black-box testing, White-box testing.
Test data, Test plan	Select and document suitable test data with expected results for: • normal (typical) data • erroneous data • boundary data. Unit testing, Integration testing.

Construction / Implementation	
	Select appropriate software and hardware, and techniques for their use.
	Implement the design: make use of appropriate software tools and techniques to construct a solution to a problem.
Conversion / Rollout	Consider the problems that may arise when converting from the old to the new system.
Parallel, direct, pilot, phased	Consider the four main methods of converting from the old to the new and justify the method chosen to implement the solution.
Testing	
System testing	Test solution and document the evidence of testing.
Acceptance testing	Consider the different types of testing that may be applied to the developed system. Justify the method(s) chosen to test the solution.
Alpha and beta testing	
Training	
Installation manual, user manual, operations manual, training manual / documentation	Consider the training needs for the new system.
	Develop technical and user documentation.
Maintenance	
	Develop and document a solution for maintainability.
	Consider the factors that affect the maintainability of a solution and evaluate a solution for maintainability in terms of the ease with which a program/solution can be corrected if an error is encountered, adapted if its environment changes, or enhanced if the customer changes requirements.
Evaluation	
	Evaluate methods and solutions against the specification and on the basis of effectiveness, usability and maintainability.

Mark Scheme	
Analysis	12
Design	12
Technical Solution	20
System Testing	8
System Maintenance	7
User Manual	7
Appraisal	6
Quality of Written Communication	3
TOTAL	**75**

Assessment Criteria

Analysis

0–3 marks	4–6 marks	7–9 marks	10–12 marks
		Complex problem • evidence of a less well-structured investigation • some appreciation of system requirements • less than high-level perception of a real end user's needs • less SMART[1] objectives	**Complex problem** • evidence of extensive well-structured investigation • realistic appreciation of system requirements • high level of perception of a real end user's needs • SMART[1] objectives
OR			
	Problem of adequate complexity • poorly structured investigation • some appreciation of system requirements • less than high-level perception of a real end user's needs • less SMART[1] objectives	**Problem of adequate complexity** • evidence of a well-structured investigation • realistic appreciation of system requirements • high level of perception of a real end user's needs • SMART[1] objectives	
OR			
Limited complexity • evidence of a poorly structured investigation • some appreciation of system requirements • less than high-level perception of a real end user's needs • less SMART[1] objectives	**Limited complexity** • evidence of a well-structured investigation • realistic appreciation of system requirements • good level of perception of a real end user's needs • SMART[1] objectives		
OR			
Simple problem • evidence of a well-structured investigation • realistic appreciation of system requirements • high level of perception of a real end user's needs • SMART[1] objectives			

[1] Specific – Objectives should be specific to the problem being solved and should specify what should be achieved.
Measurable – You need to be able to measure whether you are meeting the objectives or not.
Achievable – Are the objectives you set achievable and attainable?
Realistic – Can you realistically achieve the objectives with the resources you have?
Time – When do you want to achieve the set objectives?

3

Expected contents for this section of the report are:

- background to and identification of problem
- description of the current system
- identification of the prospective user(s)
- identification of user needs and acceptable limitations
- data source(s) and destination(s)
- data volumes
- Analysis Data Dictionary (from perspective of end user)
- DfDs (existing and proposed system) to level 1
- objectives for the proposed system
- realistic appraisal of the feasibility of potential solutions
- justification of chosen solution (use of formal methods, eg observation, analysis of existing paperwork, interviews, surveys).

If appropriate:

- E-R Models for database projects
- identification of Objects and Object analysis diagrams for Object-oriented programmed solutions.

Design

0–3 marks	4–6 marks	7–9 marks	10–12 marks
Complex problem • not feasible design • poor reporting of criteria for design	**Complex problem** • less than feasible design • detailed reporting of all criteria for design	**Complex problem** • feasible design • detailed reporting of all criteria for design	**Complex problem** • effective design • detailed reporting of all criteria for design in the context of the problem being solved
OR			
Problem of adequate complexity • less than feasible design • detailed reporting of all criteria for design	**Problem of adequate complexity** • feasible design • detailed reporting of all criteria for design	**Problem of adequate complexity** • effective design • detailed reporting of all criteria for design in the context of the problem being solved	
OR			
Limited complexity • inadequate design detail • poor reporting	**Limited complexity** • effective design • detailed reporting of all criteria for design		
OR			
Simple problem • effective design • detailed reporting of all criteria for design in the context of the problem being solved			

3

Expected contents for this section of the report are:

- overall system design
- description of modular structure of system
- definition of data requirements (Design data dictionary – from the viewpoint of programmer) including:
  - description of record structure
  - validation required
  - file organisation and processing

    or
  - database design including description of normalised relations and revised E-R diagram
  - identification of appropriate storage media
- identification of processes and suitable algorithms for data transformation
- or class/object diagrams/definitions and details of object behaviours and methods
- user interface design (HCI) rationale including:
  - **sample** of planned data capture and entry designs (prototype screen dumps may be used but must be annotated with the HCI rationale)
  - **sample** of planned valid output designs
- description of measures planned for security and integrity of data
- description of measures planned for system security
- overall test strategy in relation to the problem being solved and tested.

Technical Solution

Lack of technical competence	**Limited technical competence**	**Adequate technical competence**	**High technical competence**	**Very high technical competence**
0–4 marks	**5–8 marks**	**9–12 marks**	**13–16 marks**	**17–20 marks**
For a complex problem very little evidence of code	Few processing objectives of a complex problem met	Complex problem with basic functionality. Individual components working but system not functional	Fully analysed complex problem with majority of processing objectives met	Effective use of appropriate software tools and techniques to produce a robust solution. Meets all the objectives of a fully analysed complex problem
OR				
Few processing objectives working for a problem of adequate complexity	Some processing objectives working for a problem of adequate complexity	Majority of processing objectives working for a problem of adequate complexity	All processing objectives met for a problem of adequate complexity	
OR				
Some processing objectives met for a problem of limited complexity	Majority of processing objectives working for a problem of limited complexity	All processing objectives met for a problem of limited complexity		
OR				
Majority of processing objectives working for a simple problem	All processing objectives met for a simple problem			

Candidates are expected to code routines in order to demonstrate their technical competence in programming. Much of the evidence will be contained in the appendices and/or the system maintenance section. Code should be self-documenting wherever possible, using meaningful identifiers. Layout should aid readability. Annotation can be added to clarify the meaning of code.

The types of evidence expected include the following:

- listings of the program(s)
- listings of macros coded by the candidate
- **samples** of annotated 'design views'[2] showing details of application-generated forms, reports, queries; buttons, cross-tabulations etc
- any other reported evidence showing how the implementation was achieved. Testing to prove the system works as expected
- system maintenance to enable understanding of how the system works and to enable it to be maintained
- user manual to show the system in use.

[2] Not evidence from the Design Section but screen dumps showing the structure of these elements as developed

System Testing (not related to complexity)

0–2 marks	3 or 4 marks	5 or 6 marks	7 or 8 marks
No test plan. Limited evidence of test output. Only tested using typical data.	Test plan with at least typical and erroneous data supported by annotated test output.	A nearly complete test plan showing expected results supported by selected samples of carefully annotated and cross-referenced test output. Incomplete testing, eg omitting a boundary test.	A well-designed test plan showing expected results supported by selected samples of carefully annotated and cross-referenced hard copy showing test runs that prove the reliability and robustness of the candidate's system. All significant aspects thoroughly tested using typical, boundary and erroneous data.
OR			
Limited test plan with some output (eg from user manual).	A set of annotated test output with at least typical and erroneous data, but limited test plan to support the evidence.		

3

Expected contents for this section of the report are:

A test plan that includes:

- details of individual tests using a minimal set of test data
- expected results for clearly defined typical data
- expected results for clearly defined erroneous data
- expected results for clearly defined boundary (extreme) data
- samples of annotated hard copy of actual test runs for typical, erroneous and boundary data, and samples of annotated hard copy showing the system working (system testing)
- all samples cross-referenced to the test plan.

System Maintenance

0–1 mark	2–3 marks	4–5 marks	6–7 marks
Complex problem With complete code listings which are not self-documenting or fully annotated.	**Complex problem** with self-documenting code listings or fully annotated code listings.	**Complex problem** with self-documenting code listings or fully annotated code listings and some technical aspects described.	**Complex problem** Candidate-written code/scripts listing that is self-documenting and/or well annotated, easy to follow (very easy to comprehend). Explanation of the modular structure of all the code. Reference to the design section. Reference to testing. Explanation of difficult-to-understand parts of code. List or description of system settings/ configuration.
OR			
Adequate complexity with self-documenting code listings or fully annotated code listings.	**Adequate complexity** with self-documenting code listings or fully annotated code listings and some technical aspects described.	**Adequate complexity** with self-documenting code listings or fully annotated code listings and all technical aspects fully described.	
OR			
Limited complexity with self-documenting code listings or fully annotated code listings and some technical aspects described.	**Limited complexity** system fully documented with code listings fully annotated and all technical aspects fully described.		
OR			
Simple problem, system fully documented with code listings fully annotated and all technical aspects fully described.			

Expected contents for this section of the report are:

- system overview
- a sample of the detailed algorithms as implemented by the candidate
- procedure and variable lists with descriptions for programs, and a list of package items developed with descriptions (if a package is used).

All cross-referenced to listings of **candidate-written** program code and **representative samples** of annotated 'design views' showing details of forms, reports, queries; buttons, cross-tabulations etc that have been tailored by the candidate (if a package is used).

Automatically generated code or details of items generated by using 'wizards' should NOT be included. An acknowledgement that the item has been used is all that is required.

User manual

0–1 mark	2–3 marks	4–5 marks	6–7 marks
Some of the elements below are addressed to produce an incomplete document that would not enable easy use of the system.	**Complex problem** Not all contents below are included.	**Complex problem** Well-presented documentation.	**Complex problem** Well-presented documentation, at a level appropriate for the prospective user.
OR			
Adequate complexity Not all contents below are included.	**Adequate complexity** Well-presented documentation.	**Adequate complexity** Well-presented documentation, at a level appropriate for the prospective user.	
OR			
Limited complexity Well-presented documentation.	**Limited complexity** Well-presented documentation, at a level appropriate for the prospective user.		
OR			
Simple problem Well-presented documentation, at a level appropriate for the prospective user.			

Expected contents for this section of the report are:

- contents page
- a brief introduction and installation instructions
- detailed description of how to use the system including:
  - **samples** of actual screen displays in situ
  - **samples** of error messages and error recovery procedures.

Appraisal

Maximum 6 marks to be allocated as below

0–1 mark	2 marks	3 marks
A reasoned and detailed explanation of how objectives have been met for a problem of limited complexity. **or** A reasoned and detailed explanation of how objectives have been met for a problem of adequate complexity with less SMART[1] objectives.	A reasoned and detailed explanation of how objectives have been met for a problem of adequate complexity with SMART[1] objectives. **or** A reasoned and detailed explanation of how objectives have been met for a complex problem with less SMART[1] objectives.	A reasoned and detailed explanation of how objectives have been met for a complex problem with SMART[1] objectives.
PLUS 1 mark for each of the following:		
• Clear evidence of genuine user feedback authenticated by assessor. • Thorough analysis of user feedback and/or evidence of continual interaction with user during the development of the system. • Full and realistic suggestions as to how improvements and/or extensions are related to user feedback.		

3

Quality of Written Communication

Maximum 3 marks to be allocated as below

1 mark	Set out in a clear and logical way using the sections identified in the specification.
1 mark	Good use of English grammar, punctuation and spelling. Few errors so that meanings are clear. Information presented in continuous prose that is easy to follow.
1 mark	Good use of word-processing features to include: header, footer with project title, candidate name, and automatically generated page numbers. Consistent heading styles with word-processor-generated table of contents.

[1] Specific – Objectives should be specific to the problem being solved and should specify what should be achieved.
Measurable – You need to be able to measure whether you are meeting the objectives or not.
Achievable – Are the objectives you set achievable and attainable?
Realistic – Can you realistically achieve the objectives with the resources you have?
Time – When do you want to achieve the set objectives?

4 Scheme of Assessment

4.1 Aims

AS and A Level courses based on this specification should encourage candidates to develop a broad range of skills and knowledge of computing as a basis for progression into further learning, including progression from AS to A2, and/or employment in computing-related fields.

A Level specifications in computing should encourage students to develop:

- the capacity for thinking creatively, innovatively, analytically, logically and critically
- an understanding of the organisation of computer systems including software, hardware, data, communications and people
- the ability to apply skills, knowledge and understanding of computing, including programming, in a range of contexts to solve problems
- project and time management skills
- the capacity to see relationships between different aspects of the subject and perceive their field of study in a broader perspective
- an understanding of the consequences of uses of computing, including social, legal, ethical and other issues
- an awareness of emerging technologies and an appreciation of their potential impact on society.

4.2 Assessment Objectives (AOs)

The Assessment Objectives are common to AS and A Level. The assessment units will assess the following Assessment Objectives in the context of the content and skills set out in Section 3 (Subject Content).

AO1 Knowledge and understanding

Candidates should be able to:

- describe and explain the purpose and characteristics of a range of computing applications and show an understanding of the characteristics of computer systems (hardware, software and communication) which allow effective solutions to be achieved
- describe and explain the need for, and the use of, various forms of data organisation and processing to support the requirements of a computer-based solution
- describe and explain the systematic development of high-quality solutions to problems and the techniques for implementing such solutions, including the use of a programming language
- comment critically on the consequences of current uses of computing, including economic, social, legal and ethical issues.

AO2 Skills

Candidates should be able to:

- analyse a problem and identify the parts which are appropriate for a computer-based solution
- select, justify and apply appropriate techniques and principles to develop data structures and algorithms for the solution of problems
- design, implement and document an effective solution using appropriate hardware and software, including the use of a programming language.

Quality of Written Communication (QWC)

In GCE specifications which require candidates to produce written material in English, candidates must:

- ensure that text is legible and that spelling, punctuation and grammar are accurate so that meaning is clear
- select and use a form and style of writing appropriate to purpose and to complex subject matter
- organise information clearly and coherently, using specialist vocabulary where appropriate.

IIn this specification QWC will be assessed appropriately in Units 2, 3 and 4.

Weighting of Assessment Objectives for AS

The table below shows the approximate weighting of each of the Assessment Objectives in the AS units.

Assessment Objectives	Unit Weightings (%)		Overall weighting of AOs (%)
	Unit 1	Unit 2	
AO1	25	35	60
AO2	35	5	40
Overall weighting of units (%)	60	40	100

Weighting of Assessment Objectives for A Level

The table below shows the approximate weighting of each of the Assessment Objectives in the AS and A2 units.

Assessment Objectives	Unit Weightings (%)				Overall weighting of AOs (%)
	Unit 1	Unit 2	Unit 3	Unit 4	
AO1	12.5	17.5	15	5	50
AO2	17.5	2.5	15	15	50
Overall weighting of units (%)	30	20	30	20	100

4.3 National Criteria

This specification complies with the following.

- The Subject Criteria for Computing
- The Code of Practice for GCE
- The GCE AS and A Level Qualification Criteria
- The Arrangements for the Statutory Regulation of External Qualifications in England, Wales and Northern Ireland: Common Criteria

4.4 Prior Learning

There are no prior learning requirements. Any requirements set for entry to a course following this specification are at the discretion of centres.

4.5 Synoptic Assessment and Stretch and Challenge

Synoptic assessment in Computing is assessed in the A2 units by testing the candidates' understanding of the connections between the different elements of the subject and their holistic understanding of the subject (Unit 3), and by requiring the candidates to combine their practical and problem-solving skills with knowledge and understanding of the systems development life cycle to produce a report detailing the complete working solution to a problem (Unit 4).

The requirement that Stretch and Challenge is included at A2 is met by the testing of candidates' subject knowledge in Unit 3 via the requirement for candidates to provide extended answers to some questions, as well as by the level of challenge in Unit 4 via the complexity of the chosen problem to solve.

4.6 Access to Assessment for Disabled Students

AS/A Levels often require assessment of a broader range of competences. This is because they are general qualifications and, as such, prepare candidates for a wide range of occupations and higher level courses.

The revised AS/A Level qualification and subject criteria were reviewed to identify whether any of the competences required by the subject presented a potential barrier to any disabled candidates. If this was the case, the situation was reviewed again to ensure that such competences were included only where essential to the subject. The findings of this process were discussed with disability groups and with disabled people.

Reasonable adjustments are made for disabled candidates in order to enable them to access the assessments. For this reason, very few candidates will have a complete barrier to any part of the assessment.

Candidates who are still unable to access a significant part of the assessment, even after exploring all possibilities through reasonable adjustments, may still be able to receive an award. They would be given a grade on the parts of the assessment they have taken and there would be an indication on their certificate that not all the competences had been addressed. This will be kept under review and may be amended in the future.

5 Administration

5.1 Availability of Assessment Units and Certification

Examinations and certification for this specification are available as follows:

	Availability of units		Availability of certification	
	AS	A2	AS	A Level
January 2010	2		✔	
June 2010	1, 2	3, 4	✔	✔
January 2011 onwards	2		✔	✔
June 2011 onwards	1, 2	3, 4	✔	✔

5.2 Entries

Please refer to the current version of *Entry Procedures and Codes* for up to date entry procedures. You should use the following entry codes for the units and for certification.

Unit 1 – COMP1
Unit 2 – COMP2
Unit 3 – COMP3
Unit 4 – COMP4

AS certification – 1511
A Level certification – 2511

5.3 Private Candidates

This specification is not available to private candidates.

5.4 Access Arrangements and Special Consideration

We have taken note of equality and discrimination legislation and the interests of minority groups in developing and administering this specification.

We follow the guidelines in the Joint Council for Qualifications (JCQ) document: *Access Arrangements, Reasonable Adjustments and Special Consideration: General and Vocational Qualifications.* This is published on the JCQ website (**http://www.jcq.org.uk**) or you can follow the link from our website (**http://www.aqa.org.uk**).

Access Arrangements

We can make arrangements so that candidates with disabilities can access the assessment. These arrangements must be made **before** the examination. For example, we can produce a Braille paper for a candidate with a visual impairment.

Special Consideration

We can give special consideration to candidates who have had a temporary illness, injury or indisposition at the time of the examination. Where we do this, it is given **after** the examination.

Applications for access arrangements and special consideration should be submitted to AQA by the Examinations Officer at the centre.

5.5 Language of Examinations

We will provide units in English only.

5.6 Qualification Titles

Qualifications based on this specification are:

- AQA Advanced Subsidiary GCE in Computing, and
- AQA Advanced Level GCE in Computing.

5.7 Awarding Grades and Reporting Results

The AS qualification will be graded on a five-point scale: A, B, C, D and E. The full A Level qualification will be graded on a six-point scale: A*, A, B, C, D and E. To be awarded an A*, candidates will need to achieve a grade A on the full A Level qualification and an A* on the aggregate of the A2 units.

For AS and A Level, candidates who fail to reach the minimum standard for grade E will be recorded as U (unclassified) and will not receive a qualification certificate. Individual assessment unit results will be certificated.

5.8 Re-sits and Shelf-life of Unit Results

Unit results remain available to count towards certification, whether or not they have already been used, as long as the specification is still valid.

Candidates may re-sit a unit any number of times within the shelf-life of the specification. The best result for each unit will count towards the final qualification. Candidates who wish to repeat a qualification may do so by re-taking one or more units. The appropriate subject award entry, as well as the unit entry/entries, must be submitted in order to be awarded a new subject grade.

Candidates will be graded on the basis of the work submitted for assessment.

6 Coursework Administration

The Head of Centre is responsible to AQA for ensuring that coursework/portfolio work is conducted in accordance with AQA's instructions and JCQ instructions.

6.1 Supervision and Authentication of Coursework

In order to meet the regulators' Code of Practice for GCE, AQA requires:

- **candidates** to sign the Candidate Record Form (CRF) to confirm that the work submitted is their own, and
- **teachers/assessors** to confirm on the CRF that the work assessed is solely that of the candidate concerned and was conducted under the conditions laid down by the specification.

The completed CRF for each candidate must be attached to his/her work. All teachers who have assessed the work of any candidate entered for each component must sign the declaration of authentication. Failure to sign the authentication statement may delay the processing of the candidates' results.

The teacher should be sufficiently aware of the candidate's standard and level of work to appreciate if the coursework submitted is beyond the talents of the candidate.

In most centres teachers are familiar with candidates' work through class and homework assignments. Where this is not the case, teachers should make sure that **all** coursework is completed under direct supervision.

In all cases, some direct supervision is necessary to ensure that the coursework submitted can be confidently authenticated as the candidate's own.

- If it is believed that a candidate has received additional assistance and this is acceptable within the guidelines for the relevant specification, the teacher/assessor should award a mark which represents the candidate's unaided achievement. The authentication statement should be signed and information given on the relevant form.
- If the teacher/assessor is unable to sign the authentication statement for a particular candidate, then the candidate's work cannot be accepted for assessment.

6.2 Malpractice

Teachers should inform candidates of the AQA Regulations concerning malpractice.

Candidates must **not**:

- submit work which is not their own
- lend work to other candidates
- allow other candidates access to, or the use of, their own independently-sourced source material (this does not mean that candidates may not lend their books to another candidate, but candidates should be prevented from plagiarising other candidates' research)
- include work copied directly from books, the internet or other sources without acknowledgement or an attribution
- submit work typed or word-processed by a third person without acknowledgement.

These actions constitute malpractice, for which a penalty (eg disqualification from the examination) will be applied.

If malpractice is suspected, the Examinations Officer should be consulted about the procedure to be followed.

Where suspected malpractice in coursework/portfolios is identified by a centre after the candidate has signed the declaration of authentication, the Head of Centre must submit full details of the case to AQA at the earliest opportunity. The form JCQ/M1 should be used. Copies of the form can be found on the JCQ website **(http://www.jcq.org.uk/)**.

Malpractice in coursework/portfolios discovered prior to the candidate signing the declaration of authentication need not be reported to AQA, but should be dealt with in accordance with the centre's internal procedures. AQA would expect centres to treat such cases very seriously. Details of any work which is not the candidate's own must be recorded on the coursework/portfolio cover sheet or other appropriate place.

6.3 Teacher Standardisation

We will hold annual standardising meetings for teachers, usually in the autumn term, for the coursework units. At these meetings we will provide support in developing appropriate coursework tasks and using the marking criteria.

If your centre is new to this specification, you must send a representative to one of the meetings. If you have told us you are a new centre, either by submitting an estimate of entry or by contacting the subject team, we will contact you to invite you to a meeting.

We will also contact centres if

- the moderation of coursework from the previous year has identified a serious misinterpretation of the coursework requirements,
- inappropriate tasks have been set, or
- a significant adjustment has been made to a centre's marks.

In these cases, centres will be expected to send a representative to one of the meetings. For all other centres, attendance is optional. If you are unable to attend and would like a copy of the materials used at the meeting, please contact the subject team at **computing@aqa.org.uk**.

6.4 Internal Standardisation of Marking

Centres must standardise marking within the centre to make sure that all candidates at the centre have been marked to the same standard. One person must be responsible for internal standardisation. This person should sign the Centre Declaration Sheet to confirm that internal standardisation has taken place.

Internal standardisation may involve:

- all teachers marking some trial pieces of work and identifying differences in marking standards
- discussing any differences in marking at a training meeting for all teachers involved in the assessment
- referring to reference and archive material such as previous work or examples from AQA's teacher standardising meetings

but other valid approaches are permissible.

6.5 Annotation of Coursework

The Code of Practice for GCE states that the awarding body must require internal assessors to show clearly how the marks have been awarded in relation to the marking criteria defined in the specification and that the awarding body must provide guidance on how this is to be done.

The annotation will help the moderator to see as precisely as possible where the teacher considers that the candidates have met the criteria in the specification.

Work could be annotated by either of the following methods:

- key pieces of evidence flagged throughout the work by annotation either in the margin or in the text
- summative comments on the work, referencing precise sections in the work.

6.6 Submitting Marks and Sample Work for Moderation

The total mark for each candidate must be submitted to AQA and the moderator on the mark forms provided or by Electronic Data Interchange (EDI) by the specified date. Centres will be informed which candidates' work is required in the samples to be submitted to the moderator.

6.7 Factors Affecting Individual Candidates

Teachers should be able to accommodate the occasional absence of candidates by ensuring that the opportunity is given for them to make up missed assessments.

If work is lost, AQA should be notified immediately of the date of the loss, how it occurred, and who was responsible for the loss. Centres should use the JCQ form JCQ/LCW to inform AQA Candidate Services of the circumstances. Where special help which goes beyond normal learning support is given, AQA must be informed through comments on the CRF so that such help can be taken into account when moderation takes place (see Section 6.1).

Candidates who move from one centre to another during the course sometimes present a problem for a scheme of internal assessment. Possible courses of action depend on the stage at which the move takes place. If the move occurs early in the course the new centre should take responsibility for assessment. If it occurs late in the course it may be possible to arrange for the moderator to assess the work through the 'Educated Elsewhere' procedure. Centres should contact AQA at the earliest possible stage for advice about appropriate arrangements in individual cases.

6.8 Retaining Evidence and Re-using Marks

The centre must retain the work of all candidates, with CRFs attached, under secure conditions, from the time it is assessed, to allow for the possibility of an enquiry about results. The work may be returned to candidates after the deadline for enquiries about results. If an enquiry about a result has been made, the work must remain under secure conditions in case it is required by AQA.

7 Moderation

7.1 Moderation Procedures

Moderation of the coursework is by inspection of a sample of candidates' work, sent by post from the centre to a moderator appointed by AQA. The centre marks must be submitted to AQA and to the moderator by the specified deadline (see **http://www.aqa.org.uk/deadlines.php**). We will let centres know which candidates' work will be required in the sample to be submitted for moderation.

Following the re-marking of the sample work, the moderator's marks are compared with the centre marks to determine whether any adjustment is needed in order to bring the centre's assessments into line with standards generally. In some cases it may be necessary for the moderator to call for the work of other candidates in the centre. In order to meet this possible request, centres must retain under secure conditions and have available the coursework and the CRF of every candidate entered for the examination and be prepared to submit it on demand. Mark adjustments will normally preserve the centre's order of merit, but where major discrepancies are found, we reserve the right to alter the order of merit.

7.2 Post-moderation Procedures

On publication of the AS/A Level results, we will provide centres with details of the final marks for the coursework unit.

The candidates' work will be returned to the centre after moderation has taken place. The centre will receive a report with, or soon after, the despatch of published results giving feedback on the appropriateness of the tasks set, the accuracy of the assessments made, and the reasons for any adjustments to the marks.

We reserve the right to retain some candidates' work for archive or standardising purposes.

Appendices

A Performance Descriptions

These performance descriptions show the level of attainment characteristic of the grades at A Level. They give a general indication of the required learning outcomes at the A/B and E/U boundaries at AS and A2. The descriptions should be interpreted in relation to the content outlined in the specification; they are not designed to define that content.

The grade awarded will depend in practice upon the extent to which the candidate has met the Assessment Objectives (see Section 4) overall. Shortcomings in some aspects of the examination may be balanced by better performances in others.

AS Performance Descriptions

	Assessment Objective 1	Assessment Objective 2
Assessment Objectives	**Knowledge and understanding** Candidates should be able to: • describe and explain the purpose and characteristics of a range of computing applications and show an understanding of the characteristics of computer systems (hardware, software and communication) which allow effective solutions to be achieved • describe and explain the need for, and the use of, various forms of data organisation and processing to support the requirements of a computer-based solution • describe and explain the systematic development of high-quality solutions to problems and the techniques for implementing such solutions, including the use of a programming language • comment critically on the consequences of current uses of computing, including economic, social, legal and ethical issues.	**Skills** Candidates should be able to: • analyse a problem and identify the parts which are appropriate for a computer-based solution • select, justify and apply appropriate techniques and principles to develop data structures and algorithms for the solution of problems • design, implement and document an effective solution using appropriate hardware and software, including the use of a programming language.

AS Performance Descriptions *continued*

	Assessment Objective 1	Assessment Objective 2
A/B boundary performance descriptions	Candidates characteristically: a) understand the purpose and characteristics of a range of computing applications b) demonstrate knowledge of the characteristics of the main hardware, software and communication components of computer systems and how they allow effective solutions to be achieved c) understand the need to organise data appropriately and process it efficiently in order to solve problems using computers d) understand the need to adopt a systematic approach when developing high-quality solutions to problems e) show knowledge of appropriate techniques to implement solutions, including the use of a programming language f) demonstrate a critical understanding of the consequences of current uses of computing, including economic, social, legal and ethical issues.	Candidates characteristically: a) use subject-specific terminology appropriately and accurately b) analyse a complex problem and identify the parts that are appropriate for a computer-based solution c) derive most of the user and information requirements of a system to solve a problem d) select and use appropriate techniques to develop a solution with suitable data structures and algorithms e) choose and justify appropriate hardware and software with which to solve a problem, including the use of a programming language f) design an effective solution and document it appropriately g) implement a workable solution, testing and documenting it appropriately.

AS Performance Descriptions *continued*

	Assessment Objective 1	Assessment Objective 2
E/U boundary performance descriptions	Candidates characteristically: a) demonstrate some understanding of the purpose and characteristics of a limited range of computing applications b) show a limited knowledge of the characteristics of the main hardware, software and communication components of computer systems c) have some understanding of the need to organise data appropriately and process it efficiently in order to solve problems using computers d) demonstrate some understanding of the need to adopt a systematic approach when developing high-quality solutions to problems e) show a limited knowledge of appropriate techniques to implement solutions, including the use of a programming language f) have a limited understanding of the consequences of current uses of computing, including some economic, social, legal and ethical issues.	Candidates characteristically: a) use subject-specific terminology b) analyse a problem and identify parts that are appropriate for a computer-based solution c) derive some of the user and information requirements of a system to solve a problem d) select and use some appropriate techniques to develop a solution with generally suitable data structures and algorithms e) choose hardware and software with which to solve a problem, including the use of a programming language f) design a simple solution, and document it to a limited extent g) produce a solution, with limited testing and documentation.

A2 Performance Descriptions

	Assessment Objective 1	Assessment Objective 2
Assessment Objectives	**Knowledge and understanding** Candidates should be able to: • describe and explain the purpose and characteristics of a range of computing applications and show an understanding of the characteristics of computer systems (hardware, software and communication) which allow effective solutions to be achieved • describe and explain the need for, and the use of, various forms of data organisation and processing to support the requirements of a computer-based solution • describe and explain the systematic development of high-quality solutions to problems and the techniques for implementing such solutions, including the use of a programming language • comment critically on the consequences of current uses of computing, including economic, social, legal and ethical issues.	**Skills** Candidates should be able to: • analyse a problem and identify the parts which of are appropriate for a computer-based solution • select, justify and apply appropriate techniques and principles to develop data structures and algorithms for the solution of problems • design, implement and document an effective solution using appropriate hardware and software, including the use of a programming language.

A2 Performance Descriptions *continued*

	Assessment Objective 1	Assessment Objective 2
A/B boundary performance descriptions	Candidates characteristically: a) demonstrate a thorough understanding of the purpose and characteristics of a wide range of computing applications b) show an extensive knowledge of the characteristics of a wide range of hardware, software and communication components of computer systems c) have a thorough understanding of the need to organise data appropriately and process it efficiently in order to solve problems using computers d) demonstrate a thorough understanding of the need to adopt a systematic approach when developing high-quality solutions to problems e) show an extensive knowledge of appropriate techniques to implement solutions, including the advanced use of a programming language f) have an in-depth understanding of the consequences of current uses of computing, including a wide range of economic, social, legal and ethical issues.	Candidates characteristically: a) use subject-specific terminology appropriately and accurately b) analyse a complex problem and identify the parts that are appropriate for a computer-based solution c) derive the user and information requirements of a system to solve a problem d) select and use appropriate techniques to develop an effective solution with suitable data structures and algorithms e) choose and justify the most appropriate hardware and software with which to solve a problem, including the use of a programming language f) design an effective and efficient solution and document it thoroughly g) implement an efficient solution, testing and documenting it thoroughly.

A2 Performance Descriptions *continued*

	Assessment Objective 1	**Assessment Objective 2**
E/U boundary performance descriptions	Candidates characteristically: a) demonstrate a basic understanding of the purpose and characteristics of some computing applications b) show a basic knowledge of the characteristics of a range of hardware, software and communication components of computer systems c) understand the need to organise data appropriately and process it efficiently in order to solve problems using computers d) understand the need to adopt a systematic approach when developing solutions to problems e) demonstrate a basic knowledge of appropriate techniques to implement solutions, including the advanced use of a programming language f) show some understanding of the consequences of current uses of computing, including a range of economic, social, legal and ethical issues.	Candidates characteristically: a) use a basic range of subject-specific terminology b) analyse a fairly straightforward problem and identify the parts that are appropriate for a computer-based solution c) derive some of the user and information requirements of a system to solve a problem d) select and use appropriate techniques to develop a solution with suitable data structures and algorithms e) choose and justify some appropriate hardware and software with which to solve a problem, including the use of a programming language f) design a workable solution and document it to some extent g) implement a workable solution, testing and documenting it to some extent.

B Spiritual, Moral, Ethical, Social and Other Issues

European Dimension

AQA has taken account of the 1988 Resolution of the Council of the European Community in preparing this specification and associated specimen units.

Environmental Education

AQA has taken account of the 1988 Resolution of the Council of the European Community and the Report "Environmental Responsibility: An Agenda for Further and Higher Education" 1993 in preparing this specification and associated specimen units.

Avoidance of Bias

AQA has taken great care in the preparation of this specification and specimen units to avoid bias of any kind.

C Overlaps with other Qualifications

There are no overlaps with any other qualifications.

D Key Skills – Teaching, Developing and Providing Opportunities for Generating Evidence

Introduction

The Key Skills Qualification requires candidates to demonstrate levels of achievement in the Key Skills of Communication, Application of Number and Information Technology.

The units for the 'wider' Key Skills of Improving own Learning and Performance, Working with Others and Problem Solving are also available. The acquisition and demonstration of ability in these 'wider' Key Skills is deemed highly desirable for all candidates, but they do not form part of the Key Skills Qualification.

Copies of the Key Skills Units may be downloaded from QCA's website:
www.qca.org.uk/qca_6455.aspx

The units for each Key Skill comprise three sections:

- What you need to know.
- What you must do.
- Guidance.

Candidates following a course of study based on this specification for Computing can be offered opportunities to develop and generate evidence of attainment in aspects of the Key Skills of:

- Communication
- Application of Number
- Information Technology
- Working with Others
- Improving own Learning and Performance
- Problem Solving.

Areas of study and learning that can be used to encourage the acquisition and use of Key Skills, and to provide opportunities to generate evidence for Part B of the units, are signposted on the facing page.

Key Skills Opportunities in Computing

	Unit 1	Unit 2	Unit 3	Unit 4
Communication				
C3.1a	✔	✔	✔	✔
C3.1b	✔	✔	✔	✔
C3.2	✔	✔	✔	✔
C3.3	✔	✔	✔	✔
Application of Number				
N3.1	✔	✔	✔	✔
N3.2	✔	✔	✔	✔
N3.3	✔	✔	✔	✔
Information Technology				
ICT3.1	✔	✔	✔	✔
ICT3.2	✔	✔	✔	✔
ICT3.3	✔	✔	✔	✔
Working With Others				
WO3.1	✔	✔	✔	✔
WO3.2	✔	✔	✔	✔
WO3.3	✔	✔	✔	✔
Improving Own Learning and Performance				
LP3.1	✔	✔	✔	✔
LP3.2	✔	✔	✔	✔
LP3.3	✔	✔	✔	✔
Problem Solving				
PS3.1	✔	✔	✔	✔
PS3.2	✔	✔	✔	✔
PS3.3	✔	✔	✔	✔

GCE Computing (2510) 2009 onwards

Qualification Accreditation Number: AS 500/2574/0 - A Level 500/2573/9

To obtain free specification updates and support material or to ask us a question register with Ask AQA:
www.aqa.org.uk/ask-aqa/register

Free launch meetings are available in 2007/8 followed by further support meetings through the life of the specification. Further information is available at:
http://events.aqa.org.uk/ebooking