Cambridge Assessment International Education

Cambridge International A Level

Grade Descriptions for Cambridge International A Level Computer Science 9618

What are Grade Descriptions?

Grade descriptions describe the level of performance typically demonstrated by candidates achieving the different grades awarded for a qualification. For Cambridge International A Levels, they describe performance at three levels – grades 'E', 'C' and 'A'.

Grade descriptions sit alongside other key documents that illustrate examination standards, including:

- the syllabus, which presents what students should be taught over a course of study and explains how this is assessed
- the specimen assessment materials, which illustrate the structure of the assessment and the kinds of tasks that candidates complete
- grade thresholds, which show the total mark required to achieve a grade.

Grade descriptions are produced with a wide range of audiences in mind. For teachers, they support lesson planning and curriculum development, while students may gain useful insights into what is required to achieve a high grade and what candidate performance at lower grades typically looks like. For university admissions staff and employers, and those less familiar with Cambridge, they paint a picture of typical performance at different grades.

Cambridge publishes grade descriptions for a qualification once examinations have taken place for the first time and we review them when a qualification is substantially revised. They are developed by highly experienced examiners who understand performance standards in the subject area and have studied samples of candidate work.

How do I use this resource?

Grade descriptions are presented as a grid, with content areas at the start of each row and the different grades at the top of each column.

The content areas group together various aspects of the syllabus – they reflect topics, assessment objectives, key concepts, syllabus aims and/or components. The way they are organised is specific to each subject.

For each content area there is a descriptor for each grade. Reading across the row from left to right, the descriptors represent increasing levels of performance, with each grade descriptor building on, and including, the last.

Each column represents overall performance at a particular grade. Reading down the column from top to bottom, the descriptors capture the range of knowledge, understanding and skills that a candidate comfortably achieving the grade is likely to demonstrate.

Where content areas for the first and second halves of the A Level are distinct, they are also assessed a different standards. Performance for content areas assessed at AS Level is therefore described separately in this document.

Cambridge produces grade descriptions to support teaching and learning and the interpretation of candidate scores and grades. We do not use them to set grade thresholds. As such, they cannot be used to challenge the grade awarded to any individual candidate.

Version 1, published September 2023

Grade Descriptions

Area of knowledge, understanding and skills	Typical performance at grade E	Typical performance at grade C	Typical performance at grade A
Basic principles of computer science	Key areas of knowledge: • Defining data types and storing data, file handling • Communication protocols • Types of processor, virtual machines and Boolean algebra • How operating systems and translators work *Topics assessed at AS Level standard:* • *Digital storage and compression of data, including numerical, text, image and sound* • *Computer networks, including the transmission of data and the use of IP addresses* • *Computer hardware, including construction of logic circuits and use of sensors for monitoring and control* • *Computer architecture and the use of assembly language* • *System software, including operating systems, language translators and the use of an Integrated Development Environment (IDE)*		
	• Students show basic knowledge of key areas, though there are some gaps in this knowledge. • They identify user-defined data types and methods of file organisation and access. • They identify transmission protocols and the layers of the TC/IP protocol stack. • They identify types of processor and computer architecture. • They identify process management states and stages of compilation. • They use Karnaugh maps (K-maps).	• Students show good knowledge of key areas. • They design data types and use hashing algorithms for file access. • They convert binary floating-point numbers to denary and vice versa. • They describe transmission protocols, the layers of the TC/IP protocol stack, and packet and circuit switching. • They describe massively parallel computers and virtual machines. • They complete truth tables and draw logic circuits for adders and flip-flops. • They describe process and memory management. • They use syntax diagrams, Backus-Naur Form (BNF) notation and Reverse Polish Notation (RPN).	• Students show comprehensive knowledge of key areas. • They explain the process of normalisation. • They successfully simplify Boolean expressions. • They describe the use of interrupts in low-level scheduling. • They explain how to use BNF notation and RPN.

Area of knowledge, understanding and skills	Typical performance at grade E	Typical performance at grade C	Typical performance at grade A
Data security, data management and use of AI	Key areas of knowledge: • Encryption and digital certification • Use of graphs and neural networks in AI *Topics assessed at AS Level standard:* • *Data security threats and the methods used to protect computer systems* • *Methods of protecting data integrity including validation and verification* • *How to act ethically as a computer professional* • *Copyright and software licencing* • *Artificial Intelligence (AI)* • *Relational databases and their management systems* • *Data Definition Language (DDL) and Data Manipulation Language (DML)*		
	• Students show basic knowledge of key areas, though there are some gaps in this knowledge. • They use encryption terminology, for example public and private keys. • They identify when to use the Secure Socket Layer (SSL), Transport Layer Security (TLS) and digital certificates. • They identify AI methods, for example deep learning and machine learning.	• Students demonstrate good knowledge of key areas but may lack depth in places. • They describe how data is encrypted. • They state what SSL, TLS and digital certificates are used for. • They use A* and Dijkstra's algorithms to search graphs.	• Students show good, comprehensive knowledge of key areas. • They justify the use of quantum cryptography. • They describe how SSL, TLS and digital certificates are used. • They explain the use of back propagation of errors and regression methods in machine learning.
Computational thinking skills	Key areas of knowledge: • Writing searching and sorting algorithms • Using Abstract Data Types (ADTs) and files • Comparing the efficiency of algorithms *Topics assessed at AS Level standard:* • *Abstraction, decomposition and construction of algorithms* • *Use of data types, storage, searching and sorting of data* • *Programming using pseudocode* • *The program development lifecycle and its stages*		

Area of knowledge, understanding and skills	Typical performance at grade E	Typical performance at grade C	Typical performance at grade A
	• Students demonstrate basic knowledge of key areas, though there are gaps in this knowledge. • They demonstrate basic knowledge of constructing an algorithm to solve a problem. • They write algorithms in pseudocode, though they usually contain some errors and are incomplete in places. • They identify some programming paradigms.	• Students show good knowledge of key areas. • They show good knowledge of algorithms. • They write algorithms in pseudocode that may be incomplete and/or contain minor errors. • They describe programming paradigms, including low-level, imperative and declarative. • They identify Object-Oriented Programming (OOP) terminology. • They identify the use of recursion and are familiar with the use of Big O notation to determine the efficiency of an algorithm.	• Students show comprehensive knowledge of key areas. • They show comprehensive knowledge of algorithms. • They write efficient algorithms in pseudocode, including the use of file handling and recursion, and can describe the use of recursion in compilation. • They justify the use of programming paradigms, including low-level, imperative, declarative and OOP. • They use Big O notation to determine the efficiency of an algorithm.
Programming skills	Key areas of knowledge: • Using an Integrated Development Environment (IDE) to amend, write, test and debug programs • Using high-level programming languages – Python, Visual Basic, Java		
	• Students can use an IDE to perform some basic tasks. • They attempt to write programs, though not all parts of the program will work as intended. • They declare and use variables and one-dimensional arrays. • They initialise variables. • They declare and use procedures and functions. • They write input and output statements. • They use test data and attempt to debug errors.	• Students use an IDE with increasing success to perform many tasks. • They usually write working programs. • They provide some good naming conventions and clear commenting. • They declare and use two-dimensional arrays and classes. • They initialise array elements. • They declare and use procedures and functions with parameters. • They attempt to handle files. • They write search-and-sort routines.	• Students can successfully use an IDE to perform a wide range of tasks. • They write efficient, working programs. • They can provide appropriate naming conventions and clear commenting throughout. • They declare and use ADTs. • They create and use methods and constructors. • They declare and use recursive procedures and functions. • They write file-handling routines.

Area of knowledge, understanding and skills	Typical performance at grade E	Typical performance at grade C	Typical performance at grade A
	• They write and test a simple program.	• They design and use test data to find errors and correct them. • They write and test a substantial program.	• They write different types of search-and-sort routines, for example a bubble sort and an insertion sort. • They write and test a complex program.